

UNIVERSITÄT BASEL

Imersiv Augmented Reality

Bachelorarbeit

Departement Mathematik und Informatik
High-Performance and Web Computing
<http://informatik.unibas.ch/>

Betreuer Dr. Martin Guggisberg
Prüfer Prof. Helmar Burkhart

Autor Maurus Dähler
E-Mail m.daehler@stud.unibas.ch
Matr.-Nr. 09-053-547

25.08.2014



Inhaltsverzeichnis

Abbildungsverzeichnis	V
Abkürzungen	VII
1 Einleitung	1
2 Hintergründe zu Virtual und Augmented Reality?	3
3 Übersicht ausgewählter VR/AR Projekte	7
3.1 LifeClipper2	7
3.2 Schweißmaske der Zukunft	8
3.3 Panzer aus Glas	9
3.4 AR für Blinde	9
4 State of the art VR/AR-Technologie	11
4.1 Formen von AR	11
4.2 Head Mounted Displays	16
4.3 Weitere Ausgabegeräte für VR/AR	19
4.4 Positional Tracking	20
5 Virtual and Augmented Reality Town	23
5.1 Einführung	23

Inhaltsverzeichnis

5.2	Kreieren des virtuellen Terrains	23
5.3	Hinzufügen des Spielerobjekts	25
5.4	AR Town	25
6	Einblicke in Programmiertechniken	27
6.1	Kurzer Überblick in Unity	27
6.2	Use-Case: Hinzufügen einer Brücke	28
7	Schlusswort	31
8	Anhang	33
8.1	TileGrabber	33
8.2	Flugmodus	37
	Literaturverzeichnis	39

Abbildungsverzeichnis

2.1	Die sechs erkennbaren Lageänderungen eins 6-DOF Sensors	5
3.1	LifeClipper2 im Einsatz	7
3.2	Schweissmaske der Zukunft	8
3.3	Norwegischer Panzersoldat mit <i>Oculus Rift</i>	9
3.4	<i>Le Chal</i> : Der sehende Schuh	9
4.1	Magic-Lens-Effekt	11
4.2	Beispielaufbau Video See-Through-AR	12
4.3	Beispiel eines optischen See-Through HMD	13
4.4	Bauweisen für See-Through Systeme.	14
4.5	Beispiel einer projektionsbasierten AR	15
4.6	<i>Vuzix Wrap 1200DXAR</i>	16
4.7	<i>Epson Moverio BT-200</i>	17
4.8	<i>castAR</i> mit optionalem mobil AR Clip-On	17
4.9	<i>Oculus Rift DK2</i> mit externer IR-Kamera	18
4.10	Beispiel-marken für Marken-Tracking	20
4.11	Tracking eines Targets mit mehreren Kameras	21
4.12	Zuordnung von Features des Kamerabildes zu einer Featuremap	21
5.1	VR St. Johann in Unity	23
5.2	Sicht des Spielerobjekts	25

Abbildungsverzeichnis

6.1	Maske der Unity Engin	27
6.2	Modell der Brücke in SketchUp	28
6.3	Modell der Brücke im VR St. Johann	29
8.1	TileGrabber Beispiele.	36

Abkürzungen

Abkürzung	Bedeutung
AR	Augmented Reality: Einblenden digitaler Inhalte in die reale Welt
DGPS	Differential Global Positioning System: Verfeinertes GPS Systems, welches eine Abweichung von wenigen Zentimetern bietet
DOF	Degrees of Freedom: Anzahl Freiheitsgrade bzw. Bewegungsrichtungen, welche ein Sensor erkennen kann
FOV	Field of View: Das Sichtfeld der Kamera bzw. des Auges
HMD	Head Mounted Display: Bildschirmsystem, welches direkt vor den Augen auf dem Kopf montiert wird
SBAS	Satellite Based Augmentation System: Positionserkennung mithilfe geostationärer Satelliten
VR	Virtual Reality: gänzlich von einem Computer generierte Welt, in welcher man sich bewegen kann

1 Einleitung

Diese Arbeit soll einerseits einen Überblick über virtuelle und augmentierte Realität bieten und anderseits das erstellte Softwareprojekt vorstellen. Die Resultate sollen in das Projekt HUVIS (Handheld Urban Visualization) einfließen.

Nach einer kurzen Einführung in die Grundlagen von VR/AR und ausgewählter Projekte, wie z.B. das in Zusammenarbeit mit der *Fachhochschule Nordwestschweiz, i-art interactive AG* und der *Universität Basel* entwickelte *LifeClipper2*, wird auf verschiedene Hardwaresysteme eingegangen. Es werden die diverse Bildschirmtechnologien mit einem verstärkten Fokus auf Head Mounted Displays (HMD) vorgestellt, da diese die höchste immersive Erfahrung bieten. Hier soll der *Oculus Rift* besonders beachtet werden, da dieser für das Softwareprojekt eingesetzt wurde. Des Weiteren werden in diesem Kapitel akustische und haptische Ausgabegeräte und diverse Positionsbestimmungsmethoden, welche für AR Anwendungen wichtig sein werden, betrachtet.

Im zweiten Teil der Arbeit wird das Softwareprojekt, in welchem eine VR Erfahrung des St. Johann Quartiers in Basel entwickelt wurde, beschrieben. Für das erstellen der Bodentexturierung konnte mithilfe des selbstentworfenen Programms *TileGrabber* eine hochauflösende Karte des Gebiets erstellt werden. Anschliessend folgen Überlegungen, wie das Projekt in einer nächsten Phase zu einer augmented Reality Erfahrung weiterentwickelt werden kann. Im letzten Kapitel wird schliesslich anhand eines Use-Cases ein Teil der Möglichkeiten der *Unity Engine* vorgestellt.

2 Hintergründe zu Virtual und Augmented Reality?

Ziel eines VR Systems ist es den Betrachter vollständig in die vom Computer erschaffen Welt einzubringen. Sie wird bereits seit mehreren Jahren für Produktentwicklungen sowie vielen Industriebranchen, wie Automobilhersteller, Architekten oder das Militär, als Werkzeug genutzt. Durch die vorangeschrittene Entwicklung in der Displaytechnologie konnten die Systeme immer günstiger produziert und verkauft werden. Somit sind sie nun für Privatverbraucher erschwinglich. Doch ob es sich nun um einfache Head Mounted Displays handelt oder visionäre Systeme wie das Holodeck aus Star Trek, wird VR durch folgende Definitionen äusserst treffend beschrieben.

"Virtual Reality (VR) refers to the use of three-dimensional displays and interaction devices to explore real-time computer generated environments."

(Steve Bryson, Call for Participation 1993 IEEE Symposium on Research Frontiers in Virtual Reality)

"Virtual Reality refers to immersiv, interactive, multi-sensory, viewer-centered, three-dimensional computer generated environments and the combination of technologies required to build these environments."

(Carolina Cruz-Neira, SIGGRAPH '93 Course Notes "Virtual Reality Overview")

Ziel von VR ist es somit 3D-Inhalte des Computers nicht über den zweidimensionalen Display wahrzunehmen, sondern mithilfe von stereoskopischen Displays die Inhalte räumlich

2 Hintergründe zu Virtual und Augmented Reality?

darzustellen. Hinzu kommt das Ansprechen weiterer Sinne wie Hör- und Tastsinn. Bei der Umsetzung eines solchen Systems möchte man einen hohen Grad an Immersion erreichen. Nach Slater und Wilbur [SW97, S. 5 ff.] müssen Ausgabegeräte für Immersion folgende vier Eigenschaften besitzen

1. Sinneseindrücke des Menschen sollen möglichst ausschliesslich durch den Computer generiert werden, d.h. der Nutzer soll weitestgehend von der realen Umgebung isoliert werden
2. möglichst viele Sinne sollen angesprochen werden
3. die Ausgabegeräte sollen den Nutzer vollständig umgeben, anstatt nur ein enges Sichtfeld zu bieten
4. zudem sollen die Ausgabegeräte eine "lebendige" Darstellung bieten, z.B. durch hohe Auflösung und Qualität der Farbdarstellung

Somit stellen geschlossene HMDs, welche das Sichtfeld des Nutzers umschließen und es ihm nur noch ermöglichen die virtuellen Inhalte zu sehen, höchst immersive Displays dar.[DBGJ13, S. 12 ff.]

In manchen Bereichen ist es jedoch nicht nötig oder erwünscht eine komplette virtuelle Welt zu betrachten, sondern man möchte ein virtuelles Objekt in die reale Welt einfügen. Mithilfe von AR ist dies möglich. Im Grunde lässt sich AR in fünf Schritte aufteilen: Videoaufnahme, Tracking, Registrierung, Darstellung und Ausgabe. Die Videoaufnahme entfällt, falls es sich um ein optisches See-Through-AR System handelt, siehe dazu Kapitel 4.1. Ansonsten wird das Videobild genutzt, um später die digitalen Inhalte darüberzulegen. Beim Tracking wird die Position und Orientierung des Betrachters erfasst, damit eine perspektivische korrekte Darstellung der Inhalte erfolgt.

Hauptsächlich werden 3-DOF Sensoren, welche aus Inertialsensoren, Gyrosensoren und Magnetometern bestehen verwendet, um die Lage zuverlässig zu bestimmen. DOF (Degrees of Freedom) bedeutet Freiheitsgrade. In diesem Fall beschreibt es die Anzahl Orientierungen, welche der Sensor aufnehmen kann. Ein 3-DOF Sensor kann somit drei unterschiedliche Lageänderungen erkennen. Bei 6-DOF sind bereits sechs Lageänderungen erkennbar und somit eine relativ genaue Positionsbestimmung im Raum möglich.[Wik14]

Die genaue Position in einem grösseren Massstab zu bestimmen ist hingegen schwierig, in Kapitel 4.4 wird genauer darauf eingegangen. Bei der Registrierung werden die gewonnenen Informationen aus dem Tracking so genutzt, dass die digitalen Inhalte in der Realität verankert erscheinen. Dies führt dazu, dass das Objekt an seinem zuvor definierten Punkt in der realen Welt bleibt, auch wenn die Position des Betrachters verändert wird. Im Darstellungsschritt werden schliesslich die digitalen Inhalte der Videoaufnahme hinzugefügt und gegebenenfalls angepasst um perspektivische Korrektheit zu gewährleisten. Schlussendlich erfolgt die Ausgabe des augmentierten Bildes. Dies kann auf jeglicher Art Display erfolgen, für höchste Immersion sind jedoch HMDs zu bevorzugen.[DBGJ13, S. 241 ff.]

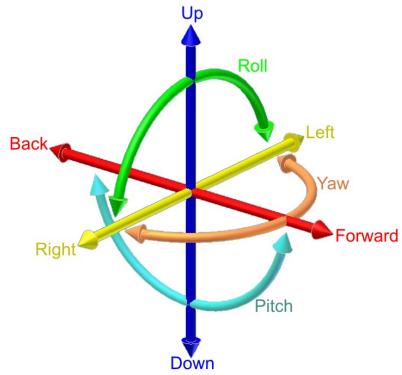


Abbildung 2.1: Die sechs erkennbaren Lageänderungen eines 6-DOF Sensors

3 Übersicht ausgewählter VR/AR Projekte

3.1 LifeClipper2

LifeClipper2 ist ein in Zusammenarbeit mit der *Fachhochschule Nordwestschweiz, i-art interaktiv AG* und der *Universität Basel* entwickeltes AR Forschungsprojekt. Es beinhaltet vier Szenarien, mit welchen es dem Anwender möglich ist die Entwicklung des St. Johann Quartiers als AR Erfahrung zu erleben. Mithilfe eines virtuellen Lifts ist es möglich das Gebiet aus grosser Höhe zu betrachten. Da der Nutzer selbst, und somit sein Blickfeld, am Boden bleibt findet ein Wechsel von augmentierter zur virtuellen Realität statt. Zudem ist es möglich die Resultate grösserer Bauprojekte bereits als Augmentierung zu betrachten. Im Gegensatz zu vorgefertigten Videosimulationen oder Bildern ist die Erfahrung um einiges realer [Mü12, S. 172 ff.]



Abbildung 3.1: LifeClipper2 im Einsatz: Rechts das augmentierte Bild des Systems, links ein Bild des Anwenders

3 Übersicht ausgewählter VR/AR Projekte

3.2 Schweißmaske der Zukunft

Moderne Schweißmasken erkennen, wann das Schweißen beginnt und verdunkeln das Visier um die Augen des Anwenders zu schützen. Es ist jedoch äusserst schwierig die Spitze des Schweißgerätes mit verdunkeltem Visier zu sehen, was nötig wäre um präzise schweissen zu können. Mithilfe von High Dynamic Range Kameras kann die Szene aufgenommen und korrigiert werden. Dunkle Stellen werden aufgehellt, helle hingegen abgedunkelt. Dies geschieht mit einem Kontrastverhältnis von 100'000'000:1. Ein Wert, welcher weit über den menschlichen Wahrnehmungsfähigkeiten liegt.

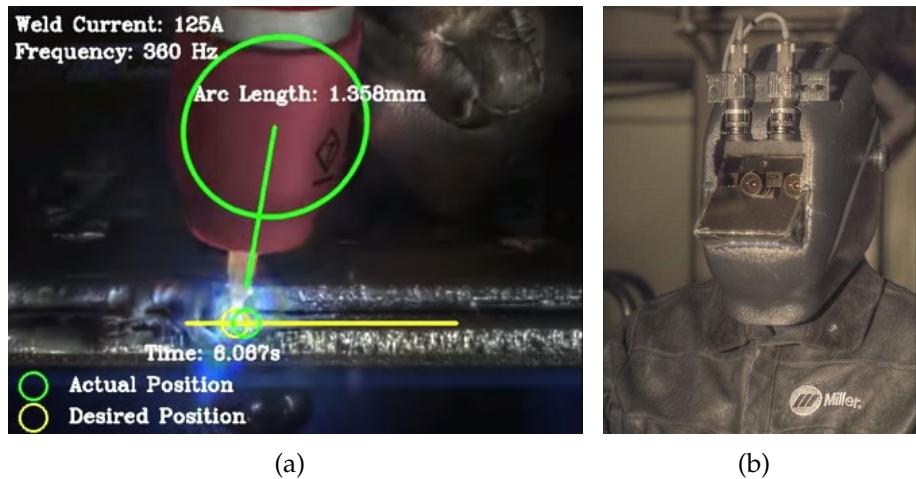


Abbildung 3.2: Schweißmaske der Zukunft: (a) Augmentierungen während des Schweißens, (b) Aussehen der Schweißmaske

Resultat dieser Aufnahme- und Bearbeitungsart ist, dass beim Schweißen lediglich ein weisser Punkt an der Spitze der Schweißnadel zu sehen ist und der Rest der Szenerie völlig normal beleuchtet wird. Daneben ist es auch möglich weitere Augmentierungen, welche nützliche Informationen für den Anwender liefern, einzublenden. So zum Beispiel eine gelbe Linie, nach der man sich richten kann, welche sich rot färbt sobald man davon abweicht oder der Abstand zwischen der Schweißnadel und dem Metallstück zu klein oder zu gross ist.[Lan12]

3.3 Panzer aus Glas

In Norwegen wird beim Militär zur Zeit ein AR System für Panzerfahrer mithilfe des *Oculus Rift* getestet. Das System besteht aus vier Kameras, welche mit einem FOV von 185° eine 360°-Sicht um den Panzer herum ermöglichen. Diese Bilder werden anschliessend genutzt um der Besatzung des Panzers durch den *Oculus* eine freie Sicht auf das geschehen Ausserhalb zu ermöglichen. Da jedoch Ausrüstung, welche ausserhalb des Panzers angebracht wird, sich in einer Gefahrenzone befindet (entweder durch Direktfeuer oder Schrapnell) kann das System noch nicht vollständig integriert werden.[Sta14]



Abbildung 3.3: Norwegischer Panzersoldat mit *Oculus Rift*

3.4 AR für Blinde

Der in Indien entwickelte haptische Schuh kann mit einem beliebigen Smartphone über Bluetooth gekoppelt werden. Nach der Routeneingabe im Navigationsapp des Smartphones teilt der Schuh mithilfe von Vibrationen mit, ob der Nutzer nach links, rechts oder geradeaus zu laufen hat. Zudem befindet sich an der Spitze des Schuhs ein Annäherungssensor, welcher den Anwender vor Hindernissen warnt. Für Menschen mit eingeschränkter Sicht ist es so möglich, sich frei und ohne andere Hilfsmittel wie Blindenstock oder Blindenhund zu bewegen.[Hea14]



Abbildung 3.4: Le Chal: Der sehende Schuh

4 State of the art VR/AR-Technologie

4.1 Formen von AR

Im Bereich der AR unterscheidet man zwischen drei grundlegenden Formen der Umsetzung. Je nach Anforderung an das System haben diese Formen Vor- und Nachteile vorzuweisen. Es ist ihnen jedoch gemein, dass sie die digitalen Inhalte in perspektivisch korrekter Form darstellen sollen. Somit muss eine Übereinstimmung der digitalen und realen Blickrichtung gegeben sein und das virtuelle Blickfeld muss dem tatsächlichen entsprechen.[DBGJ13, S. 247]

Video See-Through-AR

Bei der Video See-Through-AR Methode wird über eine oder mehrere Kameras die reale Welt aufgenommen. Anschliessend werden die digitalen Inhalte hinzugefügt und an den Display zurückgesendet. Um eine Übereinstimmung der realen und virtuellen Blickrichtung sowie des Blickfelds zu erreichen, müssen die Positionen der Kameras in der Realität mit der Position der digitalen Kamera in der Simulation übereinstimmen, siehe Kapitel 4.4. Ansonsten führt dies zu einer Fehlplatzierung des digitalen Objekts. Bei der korrekten Ausrichtung beider Kameras tritt der sogenannte Magic-Lens-Effekt auf,



Abbildung 4.1: Magic-Lens-Effekt

4 State of the art VR/AR-Technologie

bei welchem es scheint als ob man durch eine Scheibe die veränderte Realität beobachtet, siehe Abbildung 4.1.[DBGJ13, S. 248].

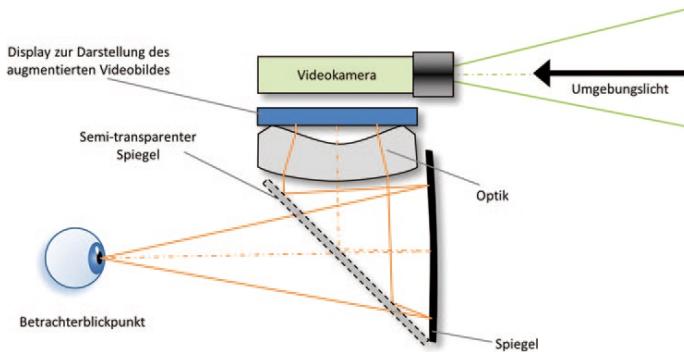


Abbildung 4.2: Beispielaufbau Video See-Through-AR

Video-See-Through-Displays sind grundsätzlich Datenbrillen mit einer oder mehreren Kameras, welche auch für VR Anwendungen genutzt werden können. Es können auch Smartphones und Tablets dafür verwendet werden, in dieser Arbeit wird jedoch nicht weiter darauf eingegangen da sie

keinen hohen Grad an Immersion ermöglichen. Die Kameras sollten über ein grösseres Blickfeld als die Datenbrille verfügen, da das Anbringen der Kameras im Strahlengang, welcher direkt vor den Augen liegt, meist nicht möglich ist. Deshalb wird eine perspektivische Korrektur der Videoinhalte nötig. Sollte es möglich sein die Linse der Kamera direkt vor dem Auge anzubringen bzw. mithilfe von Spiegeln das eingehende Licht umzuleiten, ist diese Korrektur nicht mehr nötig.[DBGJ13, S. 271 ff.]

Vorteil eines solchen Systems ist wie schon erwähnt die Möglichkeit es als VR Lösung einzusetzen. Die Nachteile liegen insbesondere bei dem höheren Rechenaufwand, da neben dem digitalen Inhalt auch das Videobild der Kamera verarbeitet werden muss. Ausserdem wird durch das Anbringen von Kameras an dem HMD das System, im Gegensatz zu optischen See-Through-AR, relativ gross und schwer.

Optisches See-Through-AR

Mithilfe semitransparenter Displays ist bei der optischen See-Through-AR Methode die Aufnahme der Realität mithilfe von Kameras nicht nötig, kann jedoch zur Positionsbestimmung vorhanden sein. Wie schon bei der Video See-Through-AR Methode muss die Perspektive des realen Umfelds und der virtuellen Inhalte übereinstimmen. Dazu muss der Blickpunkt relativ zum Display bekannt sein. Es ist daher nötig, dass für jedes Auge ein separater Display verwenden werden muss. Mit einem stereoskopischen Display ist es jedoch möglich, dass beide Augen denselben Display betrachten.[DBGJ13, S. 248 f.]

Im Aufbau gibt es verschiedene Ansätze. Allen ist jedoch gemein, dass sie die reale Umgebung lediglich mit den virtuellen Inhalten optisch überlagern. Dadurch findet, im Gegensatz zur Video-See-Through Methode, kein Qualitätsverlust bei der Wahrnehmung der Realität statt. Es ist nicht mehr möglich Objekte der Realität vollständig auszublenden oder zu überdecken. Durch die Überlagerung findet des weiteren eine Reduktion der einfallenden Lichtmenge statt und wegen der geringen Lichtstärke der meisten HMD dieser Art ist eine Anwendung bei Sonnenlicht kaum möglich.

Die gängigste Bauweise verwendet semi-transparente Spiegel. Die digitalen Inhalte werden über zwei Spiegel zum Auge umgeleitet. Diese lassen jeweils einen Teil des Lichts passieren und reflektieren einen anderen Teil. Dadurch wird die Helligkeit des Umgebungslichts als auch des Displays reduziert. Mithilfe von Prismen versucht man diesen Helligkeitsverlust zu reduzieren. Da ein einzelnes Prisma, welches die digitalen Inhalte zum Auge leitet, auch die einfallenden Lichtstrahlen der Umgebung beeinflussen würde, muss ein zweites Prisma verwendet werden um diese Krümmung zu korrigieren. Retinale Datenbrillen hingegen projizieren die digitalen Inhalte direkt auf die Retina. Dadurch sind sehr kompakte Bauweisen mit grossen Blickfeldern möglich und es entfällt eine



Abbildung 4.3: Beispiel eines optischen See-Through HMD

4 State of the art VR/AR-Technologie

aufwändige Optik, durch die sich das Auge auf die digitalen Inhalte fokussieren kann. Es wird zumeist moduliertes Laserlicht, welches über semi-transparente Spiegel oder ein Prisma zum Auge geleitet wird, verwendet.

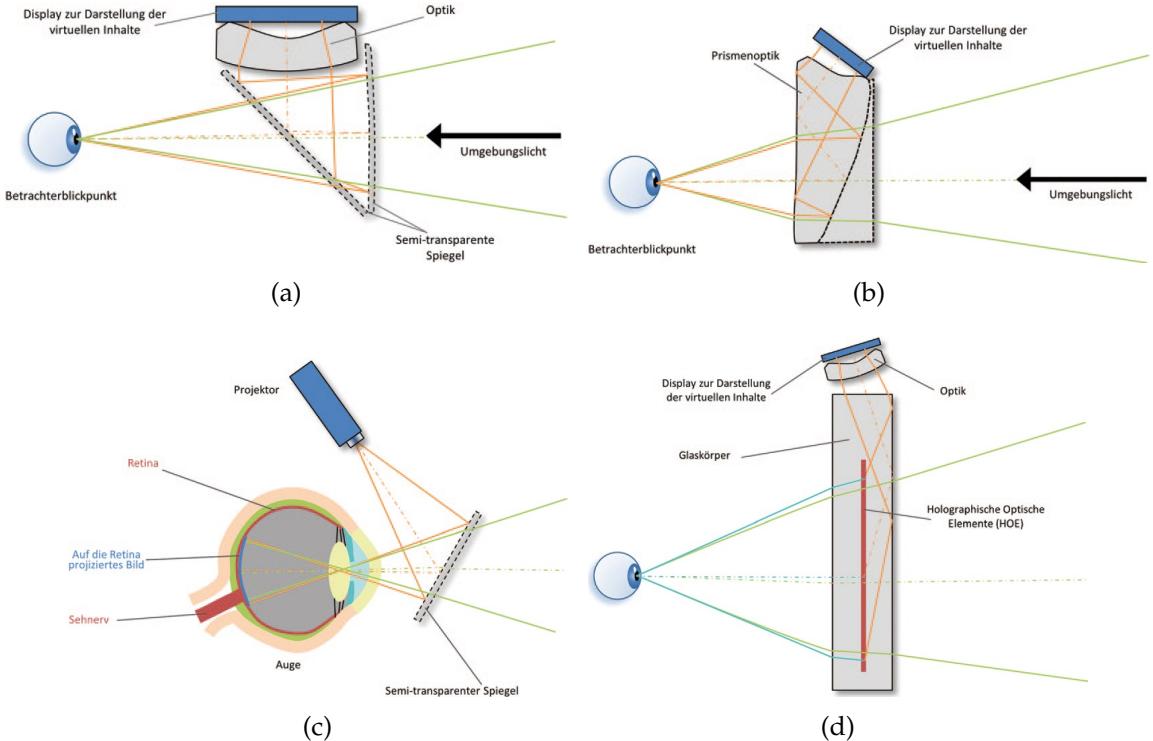


Abbildung 4.4: Bauweisen für optische See-Through Systeme: (a) semi-transparente Spiegel, (b) prismenbasiertes Display, (c) Retinale Datenbrille, und (d) optische Elemente.

Eine weitere Möglichkeit ist das integrierte optische Elemente an. Hierbei wird Licht im Inneren eines planaren Glaskörpers so umgeleitet, dass es ins Auge reflektiert wird. D.h. das digitale Bild wird seitlich oder von oben in die Glasscheibe eingespielen und tritt vor dem Auge wieder aus. Dies erlaubt eine relativ flache Bauweise des Displays und so konstruierte Datenbrillen können stark an normale Sonnenbrillen erinnern.[DBGJ13, S. 273 ff.]

Projektionsbasierte AR

Die projektionsbasierte AR Methode verwendet im Gegensatz zu den zuvor vorgestellten Methoden keinen Display zur Darstellung der Augmentierung. Hier werden mit Hilfe von Projektoren Oberflächeneigenschaften manipuliert oder zusätzliche Informationen dargestellt.[DBGJ13, S. 249]

Mobile Anwendungen sind wegen der benötigten Oberflächen zur Darstellung der Augmentierung daher kaum zu realisieren. Außerdem ist durch die geringe Helligkeit aktueller Projektoren eine Anwendung im Freien bei Sonnenlicht nicht vorstellbar. Deshalb wird im Rahmen dieser Arbeit, bis auf die Vorstellung eines projektionsbasierten HMD im Kapitel 4.2, auf diese Methode nicht weiter eingegangen.



Abbildung 4.5: Beispiel einer projektionsbasierten AR

4.2 Head Mounted Displays

Allgemein

Durch das erhöhte Interesse an VR/AR wurden in den letzten Monaten mehrere neue Systeme entwickelt, welche bereits im Handel verfügbar oder als Developer Version für Tests erhältlich sind. In diesem Abschnitt wird eine Auswahl dieser Systeme vorgestellt, mit welchen sich Outdoor-AR-Anwendungen realisieren lassen.

Die wichtigste Kenngrösse von HMDs ist ihr Field of View. Dieser beschreibt den horizontalen und vertikalen Winkel ausgehend vom Auge, von welchem die virtuellen Informationen wahrgenommen werden. Je grösser das FOV umso immersiver ist die VR Erfahrung. Für Video See-Through-AR Systeme ist es deshalb wichtig Kameras mit Weitwinkelobjektiven zu verwenden, da ansonsten der FOV bereits durch die Kameras eingeschränkt wird. Bei optischen See-Through-AR Systemen wird mithilfe des FOV hingegen die Einschränkung der Sicht durch das System beschrieben (z.B. vorhandene Ränder der Brille, welche die Sicht verdecken).[DBGJ13, S. 142 ff.]

Vuzix Wrap 1200DXAR

Die Firma Vuzix stellt schon seit einigen Jahren Videobrillen zum Betrachten von Multimediainhalten her. Bei der *Wrap 1200DXAR* handelt es sich um ein Video See-Through-AR System, siehe Kapitel 4.1. Die Brille verfügt über zwei Kameras, welche das Bild in VGA Qualität aufnehmen, einen 3-DOF Tracker um die Lage der Brille zu bestimmen, In-Ear Headphones und zwei LCD Displays. Des weiteren ist es möglich den Fokus der beiden Displays für Brillenträger zwischen +2 und -3 Dioptrien einzustellen, was das Tragen einer korrigierten Brille oder Kontaktlinsen erübriggt. AR Anwendungen können mithilfe des zur Verfügung gestellten



Abbildung 4.6: Vuzix Wrap 1200DXAR

4.2 Head Mounted Displays

SDK für Windows programmiert werden. Das AR System unterstützt neben Windows keine weiteren Betriebssysteme.[Vuz14]

Epson Moverio BT-200



Abbildung 4.7: Epson Moverio BT-200

Bei der Moverio handelt es sich um ein optisches See-Through-AR System, welches Android als Betriebssystem verwendet. Anders als die zuvor erwähnte Vuzix ist somit ein externes System in Form eines Computers oder Laptops, welcher die AR Applikation ausführt und das Bild zu der Brille sendet, nicht nötig. Es ist jedoch zu beachten, dass durch die geringe Leistung der Hardware die graphische Qualität der virtuellen Inhalte eher bescheiden ist. Sie besitzt wie auch die Vuzix über einen 3-DOF Tracker zur Bestimmung der Kopflage. Ausserdem ist ein GPS Sensor und ein Kompass in der Brille integriert. Zudem besitzt sie über ein Mikrofon, mit welchem Applikationen per Sprachbefehl gesteuert werden können.[Eps14]

castAR

Ein gänzlich anderer Ansatz kommt von der Firma Technical Illusions. Bei der von Technical Illusions entwickelten Brille handelt es sich um ein projektionsbasiertes AR System. Die Brille verfügt über zwei Projektoren, welche das Bild auf die Oberfläche vor dem Nutzer projizieren. Die polarisierten Gläser der Brille fügen die beiden Teilbilder anschliessend, ähnlich wie bei 3D-Kino Brillen, zu einem einzigen dreidimensionalen Bild zusammen. Des Weiteren verfügt



Abbildung 4.8: castAR mit optionalem mobil AR Clip-On

4 State of the art VR/AR-Technologie

sie über eine hochauflösende Kamera, um die absolute Kopfposition zu bestimmen und einen 3-DOF Tracker zur Bestimmung der Kopflage. Ausserdem wird in naher Zukunft ein Zusatz angeboten werden, welcher auf die vorhandene Brille geklebt werden kann. Dieser reflektiert die Ausgabe der Projektoren direkt vor den Augen, womit auch eine mobile Anwendung für das System in Frage kommt.[cas14]

Oculus Rift



Abbildung 4.9: *Oculus Rift DK2 mit externer IR-Kamera*

Bei dem *Oculus Rift* handelt es sich um ein VR System. Den Entwicklern war es wichtig, dass sie mit dem *Oculus* eine günstige VR-Lösung für jedermann anbieten können. Die erste Version des *Oculus*, besitzt ein einfaches Display, den standartmässigen 3-DOF Tracker zur Lagebestimmung und eine vollständige Integration in die Gameengine Unity, mit welcher Applikationen für die gängigen Betriebssysteme Windows, Mac und Linux, geschrieben werden können.[Ocu14a]

Da das Projekt selbst als Kickstarter finanziert wurde, waren die Prototypen sehr schnell verbreitet und in kurzen Zeit wurden die ersten Abänderungen der Hardware für eine Video-See-Through-AR Erfahrung in den Foren diskutiert.[Ste13]

Die nächste Version, unter dem Projektnamen "Crystal Cove" bekannt, verfügt über ein hochauflösendes Display und ein verbessertes Headtrackingsystem. Mithilfe einer IR-Kamera ist es nun möglich nicht nur die Lage der Brille sondern die genaue Position im Raum zu bestimmen. Kameras zur offiziellen AR Unterstützung sind jedoch immer noch nicht vorhanden und durch das neue Trackingsystem, welches eine externe Kamera benötigt, rückt die Vorstellung für eine mobile Anwendung in die Ferne.[Ocu14b]

4.3 Weitere Ausgabegeräte für VR/AR

Akustische Ausgabegeräte

Wie bereits in Kapitel 2 beschrieben müssen bei einer möglichst immersiven VR/AR Erfahrung mehrere Sinne angesprochen werden. Zusätzlich zum visuellen Sinn ist es relativ einfach Geräusche zu generieren. In VR Umgebungen lassen Umgebungsgeräusche den Anwender die Welt realer erscheinen. Das Fehlen auditiver Feedbacks, z.B. beim Fallenlassen einer Kiste in der virtuellen Welt, würde zu einem Bruch in der angeblichen Realität führen.

Mithilfe von Mehrkanalsystemen kann ein räumlicher Effekt entstehen (wie man es aus Dolby Surround Sound kennt). Eine Weiterentwicklung dieses Systems ist die Wellenfeldsynthese. Dabei wird das Wellenfeld realer Ereignisse aufgenommen und kann anschließend als synthetisches Wellenfeld über mehrere Lautsprecher ausgegeben werden. Es ist somit möglich verschiedene Schallquellen an beliebigen Positionen in einem begrenzten Gebiet zu erzeugen.[DBGJ13, S. 154]

Haptische Ausgabegeräte

Haptische Ausgabegeräte sprechen den Tastsinn des Anwenders an (z.B. die haptischen Schuhe in Kapitel 3.4). Somit kann der Nutzer die virtuelle Welt nicht nur sehen und hören, sondern auch fühlen. Mittels Vibrationen wird schon seit längerer Zeit in Videospielen der Rückstoss von Gewehren simuliert. Subwoofer können außerdem verwendet werden, um den Boden vibrieren zu lassen.

Für ein haptisches Feedback sind die geometrischen Modelle, welche zur optischen Darstellung verwendet werden, wegen des hohen Detailgrades nicht gut geeignet. Deshalb sollte zur Verkürzung der Rechenzeit ein betreffendes Haptikmodell mit in die virtuelle Umgebung aufgenommen werden. Dies ist ebenfalls nötig, da der Mensch viel mehr haptische als visuelle Eindrücke pro Sekunde aufnehmen kann.[DBGJ13, S. 154 f.]

4.4 Positional Tracking

Damit Augmentierungen perspektivisch korrekt dargestellt werden, ist es essentiell die Position und Blickrichtung des Betrachters zu kennen. Bei den hier vorgestellten Methoden wurde darauf geachtet, dass sie einen relativ geringen Rechenaufwand besitzen und somit für mobile Anwendungen genutzt werden können.

Marken-Tracking

Das markenbasierte Verfahren des optischen Trackings verwendet zur Vermeidung der Fehleranfälligkeit bei schlechter Beleuchtung, sowie Verringerung der Berechnungskomplexität spezifizierte Marker. Diese Marker können passiv Licht reflektieren oder aktiv emittieren. Schwarzweissmarken haben sich in der Praxis bewährt, da sie leicht herzustellen sind und selbst von einfachen Kameras bei schlechten Lichtverhältnissen gut erkannt werden. Trotz ihrer Ähnlichkeit, haben sie jedoch nichts mit QR-Tags zu tun. Diese Marken reichen vollkommen aus, falls das Tracking nur die Position des Objekts erkennen soll. Soll jedoch auch die Orientierung bestimmt werden müssen sogenannte Targets verwendet werden.[DBGJ13, S. 104 ff.]

Die grössere der Marker ist entscheidend, da sie entweder nicht komplett erkannt werden können oder bei zu kleinen Marken die Anzahl der Markerpixel zu gering ist und es deshalb zu Fehlern in der Mustererkennung kommt. Durch einen zu flachen Betrachtungswinkel, können Transformationswerte stark schwanken und es kommt ebenfalls zu einer fehlerhaften Mustererkennung. Der grösste Nachteil solcher Marken ist der, dass sie direkt an dem zu augmentierenden Objekt angebracht werden müssen. Manchmal ist dies nicht möglich, sie würden vom Benutzer selbst verdeckt, wenn er das Objekt in die Hand nehmen müsste oder störend auffallen, da die Marker visuell nicht sehr ansprechend sind.[DBGJ13, S. 256 ff.]

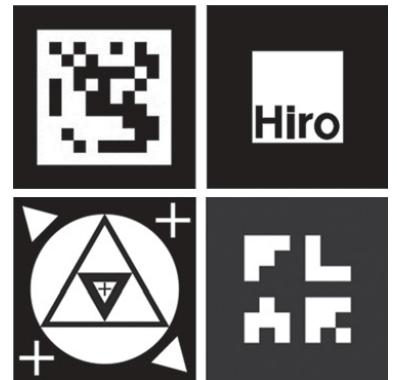


Abbildung 4.10: Beispielmarken für Marken-Tracking

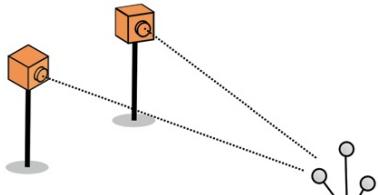


Abbildung 4.11: Tracking eines Targets mit mehreren Kameras

Ein Target besteht aus mehreren Einzelmarken, von welchen die relative Position zueinander bekannt ist. Durch verschiedene Bauweisen von mehreren Targets kann das System diverse Objekte differenzieren. Wird mit solchen Targets gearbeitet, müssen mindestens zwei Kameras, je nach Grösse des zu überwachenden Gebiets auch mehr, benutzt werden. Die Bilddaten der einzelnen Kamera

liefern bei einer Vorverarbeitung eine genaue Position in der zweidimensionalen Ebene. Mithilfe der zweiten Kamera ist es anschliessend möglich die genaue Position im Raum zu bestimmen.[DBGJ13, S. 106 ff.]

PTAM

Wie im vorhergehenden Abschnitt bereits erwähnt, ist es nicht immer möglich Marken an zu augmentierende Objekte anzubringen. Eine Lösung zu diesem Problem bietet PTAM. Hierbei handelt es sich um ein markerloses Trackingverfahren. Es werden dazu Features (Kanten und Eckpunkte) aus dem Bild, welches in verschiedenen Auflösungen zur Verfügung steht, extrahiert und mit dem zuvor aufgenommenen Frame verglichen. Somit kann die Änderung des Blickwinkels berechnet werden.[DBGJ13, S. 106 ff.]

PTAM verwendet im Gegensatz zu üblichen markerlosen Trackingverfahren eine geringere Anzahl von Features. Dies resultiert in einer geringeren CPU-Auslastung und verringert ausserdem Messfehler aufgrund von Bewegungsunschärfe, falls der Blickwinkel rasch verändert wird.[KM08]

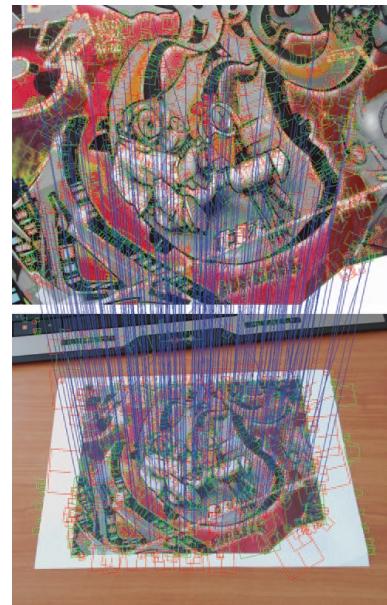


Abbildung 4.12: Zuordnung von Features des Kamerabildes zu einer Featuremap

4 State of the art VR/AR-Technologie

DGPS und SBAS

Für Outdoor-AR-Anwendung bieteet sich am ehesten DGPS und SBAS zur Positionsbestimmung an. Bei GPS können Abweichungen von 10 Metern oder mehr auftreten. Besonders in Wäldern oder Häuserschluchten kann die Position mit GPS nicht mehr genau bestimmt werden. Bei DGPS wird mithilfe eines lokalen Referenzempfängers ein Korrektursignal berechnet. Nachdem das Signal über Funkt oder Internet an den GPS-Empfänger gesendet und mit dessen Signal verrechnet wurde, ist eine Genauigkeit von bis zu wenigen Zentimetern möglich.

SBAS nutzt mehrere geostationäre Satelliten als Referenzsystem und ist deshalb nur in bestimmten Gebieten verfügbar (zurzeit Europa, Nordamerika und Japan). Dieses System verfügt über eine Abweichung von ca. einem Meter, kann jedoch bei eingeschränkter Sicht nach Süden fehleranfällig sein, da die Umlaufbahn geostationärer Satelliten über dem Äquator liegt. Bei der Augmentierung von Gebäuden und Gegenständen im Freien reicht die Genauigkeit jedoch aus, falls sich der Betrachter nicht allzu nah am Objekt befindet.[DBGJ13, S. 253 ff.]

5 Virtual and Augmented Reality Town

5.1 Einführung

Im Rahmen der Arbeit wurde Mithilfe der *Unity Engine* und dem *Oculus Rift* ein VR Prototyp erstellt. In der Anwendung ist es dem Anwender möglich sich im Stadtquartier St. Johann virtuell zu bewegen. Mithilfe des "Flugmodus" können Gebäude aus unterschiedlichsten Perspektiven betrachtet oder, dank der erhöhten Bewegungsgeschwindigkeit in der Luft, in kurzer Zeit grössere Distanzen zurückgelegt werden.

Für die Implementierung konnte ein 3D Modell des Gebietes, welches von der Stadt Basel zur Verfügung gestellt wurde, verwendet werden. Mithilfe der Unity Engine, welche ein offizielles SDK für den Oculus Rift besitzt, konnte relativ einfach ein VR Prototyp realisiert werden. Das folgende Architekturbild beschreibt das Zusammenspiel der einzelnen Komponenten.

5.2 Kreieren des virtuellen Terrains

Da das zur Verfügung gestellte Stadtmodell als DXF-Datei vorlag musste dieses zuerst in ein für Unity kompatibles Format konvertiert werden. Dafür wurde das Modell in Google SketchUp geladen um anschliessend auf die reale grössse zu skalieren und als OBJ-Datei exportiert. Mihilfe von SketchUp ist es auch möglich weitere Objekte für die virtuelle Welt zu erstellen und später hinzuzufügen.



Abbildung 5.1: VR St. Johann in Unity

5 Virtual and Augmented Reality Town

In *Unity* konnte die OBJ-Datei geladen werden. Innerhalb der Datei wurden die einzelnen Modelle der Häuser und das Terrain in die Kategorien "Wände", "Dächer" und "Terrain" unterteilt und konnten einzeln ausgewählt und eingefärbt werden. Das Terrain musste zuerst noch mithilfe eines Scripts[Hai12] in ein Terrain Object umgewandelt werden, damit es als Untergrund korrekt erkannt wurde.

Anschliessend wurde mittels einer Java-Applikation, im folgenden *TileGrabber* genannt, die Karte des Gebiets im Massstab 1:500 zusammengestellt und als Textur des Terrains verwendet. Der Quellcode kann im Anhang unter 8.1 gefunden werden. Im Prinzip werden beim Abruf eines Kartenausschnitts auf der Webseite nur die benötigten Quadrate (Tiles) mit einer Auflösung von 256x256 Pixel dargestellt. Je nach Zoomstufe werden andere, dem Massstab entsprechende, Tiles geladen. Mithilfe des *TileGrabbers* ist es nun möglich einen sehr grossen Kartenausschnitt mit Tiles eines geringen Massstabs zusammenzustellen. Das Ergebnis ist eine Karte mit einer Auflösung von 9216x9216 Pixel.

Um der virtuellen Welt etwas mehr Tiefe zu geben wurden an den in der Karte gekennzeichneten Positionen Bäume hinzugefügt. Zusätzlich sorgen Tonaufnahmen einer stark befahrenen Strasse als Hintergrundgeräusche in der Nähe von Strassen sowie Aufnahmen von Singvögeln bei Baumgruppen für eine immersive Erfahrung. Der Himmel wird mit einer Skybox statisch dargestellt. Es gibt deshalb keinen Tag-Nacht-Zyklus oder Wetterwechsel.

In Abbildung 5.1 sieht man die von der Skybox umschlossene virtuelle Welt. Wie erwähnt konnten die verschiedenen Objekte je nach Typ mit unterschiedlichen Texturen belegt werden. So sind Wände grau, Dächer rot und der Boden verfügt über den Stadtplan als Textur. Bei genauerer Betrachtung ist es möglich die Bäume der verschiedenen Alleen und Parks des Gebiets zu erkennen. Der Schattenwurf der verschiedenen Objekte wird zur Steigerung der Performance im Vorschaufenster von *Unity* ausgeblendet und ist deshalb nicht erkennbar.

5.3 Hinzufügen des Spielerobjekts

Mithilfe des zur Verfügung gestellten SDK von *Oculus* kann ein für den HMD konzipiertes Spielerobjekt in die Welt eingefügt werden. Es verfügt bereits über grundlegende Bewegungen wie Springen, Laufen oder Rennen und verwendet die Informationen des integrierten 3-DOF Sensors um die Bewegungen des Kopfes in der Realität ebenfalls in der virtuellen Welt umzusetzen. Außerdem werden die Inhalte bereits für den *Oculus Rift* korrekt ausgegeben, also als Doppelbild für stereoskopische Sicht.

Um dem Benutzer die Möglichkeit sich schnell von Punkt A nach B zu bewegen und einen besseren Überblick über die Szene zu geben, wurde ein "Flugmodus" implementiert¹. Dazu wurde in dem für die Charakterbewegung verantwortlichen Skript das Springen verändert. Solange die Taste für das Springen gedrückt gehalten wird, ist die Gravitation deaktiviert und das Spielerobjekt schwebt in der Luft. Lässt man los, sinkt man zu Boden.

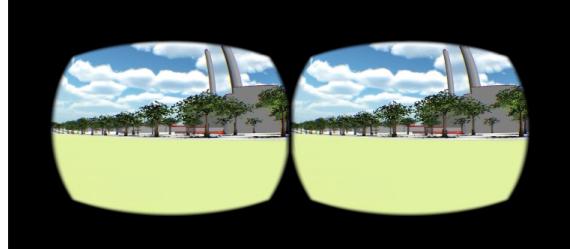


Abbildung 5.2: Sicht des Spielerobjekts

5.4 AR Town

Hier werden die theoretischen Überlegungen, wie man die erstellte VR Anwendung in eine AR Applikation weiterentwickeln kann, ausgeführt. Um dies zu bewerkstelligen müssen einerseits Änderungen an der Hardware, sprich anbringen einer Stereokamera und gegeben falls Sensoren für die Positionsbestimmung, und andererseits der Programmcode umgeschrieben werden.

Da es sich beim *Oculus Rift* um einen geschlossenen HMD handelt, müssen Videokameras angebracht werden um aus dem VR System ein Video See-Through-AR System zu erstellen. Bei den Kameras ist darauf zu achten, dass sie mindestens 60 Bilder pro Sekunde

¹ siehe 8.2

5 Virtual and Augmented Reality Town

aufnehmen, um ein flüssiges Bild garantieren zu können. Außerdem ist ein möglichst grosser Aufnahmewinkel nötig, da der Oculus selbst über ein relativ breites FOV verfügt und ansonsten schwarze Ränder entstehen würden.[Ste13]

Zur Positionsbestimmung kommen mehrere Alternativen in Frage (siehe 4.4). Da die Applikation mehrheitlich im Freien genutzt wird, bietet sich am ehesten DGPS an. Eine andere Möglichkeit wäre eine Kombination aus dem vorgestellten PTAM, dem Multiple Maps System der selben Arbeitsgruppe[KM09] und GPS. Mithilfe von GPS wird die Position grob bestimmt und anschliessend mithilfe des Multiple Maps System die verschiedenen Featuremaps der Umgebung geladen. Anschliessend kann mithilfe von PTAM die Augmentierung richtig positioniert werden.

Im Programmcode müssen dementsprechend Änderung angebracht werden. So muss das gewonnenen Doppelbild als steter Hintergrund der Szene verwendet werden. Möchte man dass Bäume, Baugerüste oder andere Gebäude das virtuelle Objekt verdecken können, muss eine Methode gefunden werden, welche die vorhandenen realen Objekte erkennt und das 3D-Modell an den entsprechenden Stellen zuschneidet.

6 Einblicke in Programmiertechniken

6.1 Kurzer Überblick in Unity

Mithilfe von *Unity* kann ein weiteres Objekt in wenigen Schritten der virtuellen Welt hinzugefügt werden. Die Engine ist übersichtlich und dank "Drag and Drop" Funktionen leicht zu bedienen.

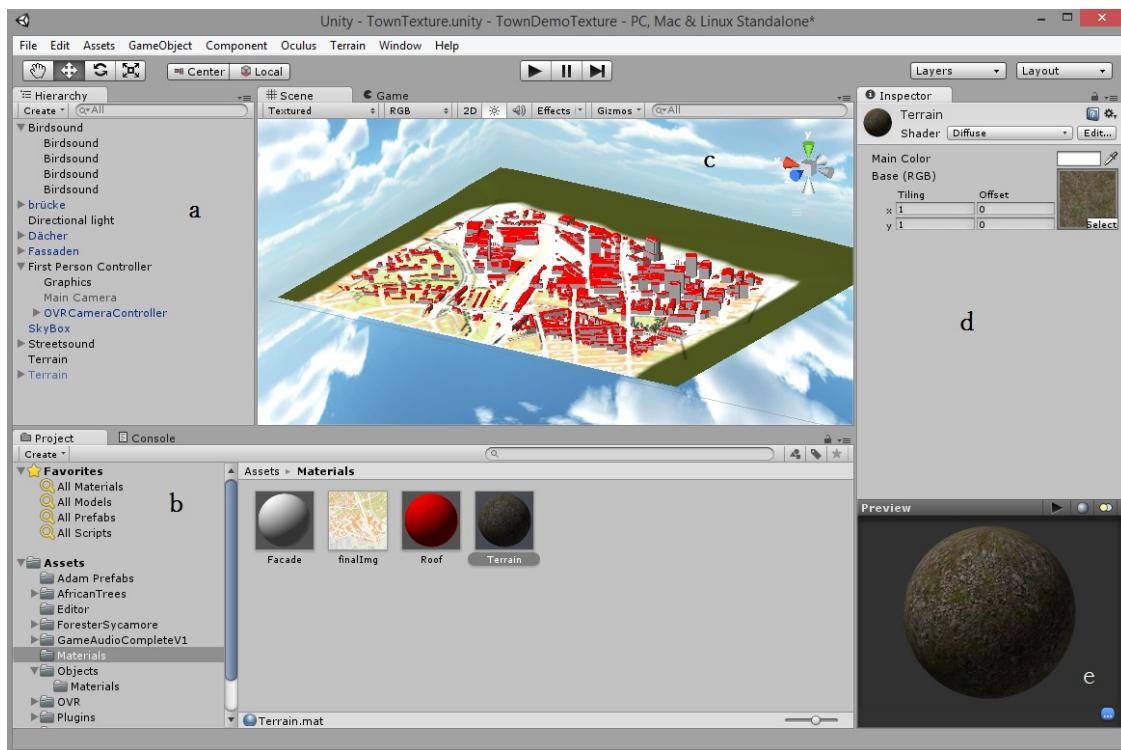


Abbildung 6.1: Maske der Unity Engin: (a) Objektexplorer, (b) Dateiexplorer, (c) Scene Preview, (d) Inspector, (e) Object Preview

In Abbildung 6.1 wird die Maske von *Unity* gezeigt. Alle Elemente, welche in (a) enthalten sind, sind in die Welt integriert. Erscheint der Name des Objekts grau, ist es deaktiviert

6 Einblicke in Programmiertechniken

(in diesem Fall die Hauptkamera, da der Oculus Rift verwendet wird und somit die OVRCameraController aktiviert sein müssen). Hier ist es auch möglich einzelne Elemente zu einer Gruppe zusammenzufassen. Neben einer besseren Übersicht bietet dies die Möglichkeit Vater-Kind-Beziehungen herzustellen. Bewegt sich nun der "First Person Controller" so wird auch der "OVRCameraController" in die selbe Richtung bewegt. In (b) sind hingegen die Dateien, welche bereits in *Unity* importiert jedoch noch nicht unbedingt verwendet wurden, angezeigt. Durch klicken der rechten Maustaste in diesem Feld können weitere Elemente importiert werden. Mithilfe einer "Drag and Drop" Funktion können Elemente von (b) sowohl in (a) oder gleich in das Vorschaufenster (c) an die richtige Position gezogen werden. In (d) sind weitere Informationen zum ausgewählten Objekt zu finden. Bei Spielerobjekten wäre dies z.B. die Höhe der Figur, seine Laufgeschwindigkeit, die Sprunghöhe und im Falle des Oculus auch sein Augenabstand. Das letzte Feld (e) ist lediglich eine Vorschau des ausgewählten Objekts, unter der Voraussetzung, dass es sich dabei um eine Textur oder ein 3D-Modell handelt.

6.2 Use-Case: Hinzufügen einer Brücke

In dem zur Verfügung gestellten 3D-Modell des St. Johann Quartiers hat die Brücke über die Bahngleise in der Nähe des Lothringerplatzes gefehlt. Mithilfe von *SketchUp* können einfach Strukturen relativ schnell als 3D-Modell umgesetzt werden. Durch das Hinzufügen des Geostandorts konnten die Informationen von Googlemaps genutzt werden um den Verlauf der Brücke realitätsnah nachzubilden. Außerdem ist damit garantiert, dass das Objekt bereits über die korrekte Grösse verfügt und nicht mehr skaliert werden muss.

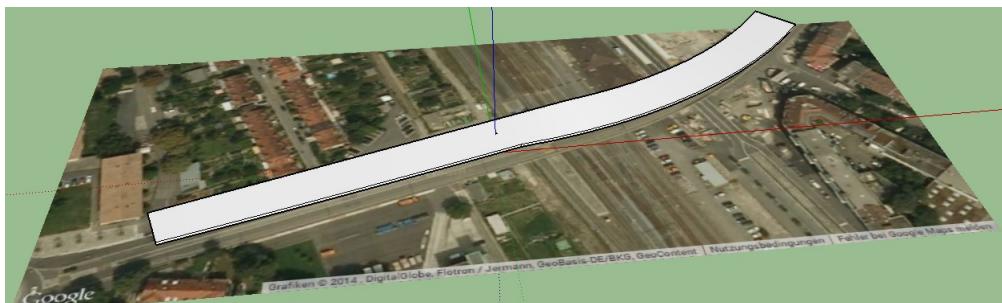


Abbildung 6.2: Modell der Brücke in SketchUp

6.2 Use-Case: Hinzufügen einer Brücke

Nachdem das Objekt als OBJ-Datei exportiert wird kann es in Unity importiert und per "Drag and Drop" in dies Szene gezogen werden. Anschliessend können im Inspector noch Änderungen am Objekt vorgenommen werden, in diesem Fall musste die Brücke um 2° auf der Y-Achse gedreht werden, damit sie mit dem Strassenverlauf übereinstimmt. Des weiteren wurde ein "Mesh Collider" hinzugefügt, damit der Anwender nicht durch die Brücke hindurch fällt, falls er versucht sie zu überqueren.



Abbildung 6.3: Modell der Brücke im VR St. Johann

7 Schlusswort

Das Schreiben dieser Arbeit weckte in mir den Pioniergeist. Beinahe wöchentlich wurden Informationen zu neuer Hardware oder VR/AR Projekten veröffentlicht. Die Begeisterung, welche AR momentan erfährt treibt die Entwicklung immer weiter voran und selbst der in diesem Projekt verwendete *Oculus Rift*, welcher zu Beginn relativ neu war, kann bereits als veraltete Technologie bezeichnet werden. Doch selbst Systeme wie die erst kürzlich erschienene *castAR* werden in wenigen Jahren belächelt.

Diese Arbeit soll ein erster Schritt in die Welt der VR/AR sein und einen groben Überblick in das Themengebiet geben. Das im Rahmen der Arbeit entwickelte Softwareprojekt läuft stabil und dank *Unity* ist es möglich in wenigen Schritten neue Gebäude in die virtuelle Welt zu integrieren und das veränderte Stadtbild mithilfe des *Oculus Rift* zu betrachten.

8 Anhang

8.1 TileGrabber

Mithilfe des TileGrabber Programms können Karten aus der Region Basel im Massstab 1:500 zusammengestellt werden. Da es keine Relation zwischen der Indexierung der einzelnen Teilbilder und geographischer Koordinaten (z.B. LV95) gibt, müssen Start und Endpunkt aus dem HTML Code ausgelesen werden. Das Programm sammelt anschliessend die Teilbilder von West nach Ost und von Nord nach Süd um sie schlussendlich in einem grossen Bild zusammenzufügen. Dieses kann anschliessend als Textur des Terrains benutzt werden.

```
1 package main;
2
3 import java.awt.Color;
4 import java.awt.Graphics;
5 import java.awt.Graphics2D;
6 import java.awt.image.BufferedImage;
7 import java.io.File;
8 import java.io.IOException;
9 import java.net.MalformedURLException;
10 import java.net.URL;
11
12 import javax.imageio.ImageIO;
13
14 public class Main {
15
16     public final static String PAGE = "http://www.stadtplan.bs.ch/tmp"
17         + "/kacache/default/500/"
18         + "LAYEROBJ_PARZPLAN_VEKTOR_FARB_500/def/t-@estart/l@nstart/t@"
19         - @ecoordl@ncoord.png";
```

8 Anhang

```
19 //http://www.stadtplan.bs.ch/tmp/kacache/default/500/,
20 LAYEROBJ_PARZPLAN_VEKTOR_FARB_500/def/t-9592320/119725312/t,
21 -9591808119726336.png
22
23
24 public final static int HIGHESTE = 9598464; //First east tile
25 public final static int HIGHESTN = 19725312; //First north tile
26
27 public final static int LOWESTE = 9590784-1536; //Last east tile
28 public final static int LOWESTN = 19732992+1536; //Last north
29 tile
30
31 public final static int TILESIZE = 256; //Width of a tile in
32 pixel
33 public final static int FOLDERSIZE = 1536; //Width of all tiles
34 in a folder
35
36 public static int startEast;
37 public static int endEast;
38 public static int startNorth;
39 public static int endNorth;
40
41 public static void main(String[] args) {
42     startEast = HIGHESTE;
43     startNorth = HIGHESTN;
44     endEast = LOWESTE;
45     endNorth = LOWESTN;
46
47     int rows = (LOWESTN-HIGHESTN)/TILESIZE; //we assume the no.
48         of rows and cols are known and each chunk has equal width
49         and height
50     int cols = (HIGHESTE-LOWESTE)/TILESIZE;
51     int chunks = rows * cols;
52
53     System.out.println("rows:" + rows + "cols:" + cols + "chunks:" + chunks);
54
55     //Initializing the final image
56     BufferedImage finalImg = new BufferedImage(TILESIZE*cols,
57             TILESIZE*rows, BufferedImage.TYPE_INT_RGB);
58     Graphics2D g = (Graphics2D) finalImg.getGraphics();
59     int num = 0;
60     for (int i = 0; i < rows; i++) {
61         for (int j = 0; j < cols; j++) {
62             try {
63                 //get the tile
```

8.1 TileGrabber

```
55     String img = (((PAGE.replace("@estart", ""+findFolder(,
56         startEast-(i*TILESIZE),false))).  
57         replace("@nstart", ""+findFolder(startNorth+(j*),
58             TILESIZE),true))).  
59         replace("@ecoord", ""+(startEast-(i*TILESIZE)))).  
60         replace("@ncoord", ""+(startNorth+(j*TILESIZE))));  
61     BufferedImage buffImages = ImageIO.read(new URL(img));
62     //add it at the right position
63     g.drawImage(buffImages, TILESIZE * j, TILESIZE * i,,
64         null);
65 } catch (IOException e) {
66     e.printStackTrace();
67 }
68 num++;
69 System.out.println("Chunk" + num + " added..." + (chunks-
70     num) + " to go");
71 }
72 g.dispose();
73
74 System.out.println("Image concatenated.....");
75 try {
76     //save the image
77     File outimg = new File("C:/temp/finalImg.png");
78     ImageIO.write(finalImg, "png", outimg);
79 } catch (IOException e) {
80     e.printStackTrace();
81 }
82
83 /**
84 *get the next folder
85 */
86 private static int findFolder(int coord, boolean isNorth){
87     int start = HIGHESTE;
88     if(isNorth){
89         start = HIGHESTN;
90     }
91     return start+((coord-start)/(1536))*(1536);
92 }
93 }
94 }
```

8 Anhang



Abbildung 8.1: TileGrabber Beispiele: (a) Gesamte Karte aus TileGrabber (Lothringenplatz markiert), (b) Detailansicht auf Lothringenplatz, (c) Lothringenplatz in VR.

8.2 Flugmodus

Aus Platzgründen werden hier nur Teile des Codes aufgeführt, welche für den "Flugmodus" abgeändert wurden. Alle Funktionen sind Bestandteil des Standartsskripts zur Charaktersteuerung in Unity. Um nach dem Springen in der Luft zu bleiben wurde in der Funktion *ApplyGravityAndJumping* ein Else hinzugefügt, welches, solange die Sprungtaste gedrückt und die maximale Höhe noch nicht erreicht wurde, die Bewegung in y-Richtung ausser Kraft setzt.

```

1 //we can't jump in the air
2 if (grounded)
3     velocity.y = Mathf.Min(0, velocity.y) - movement.gravity * Time.
    .deltaTime;
4 else {
5     velocity.y = movement.velocity.y - movement.gravity * Time.δ
        deltaTime;
6
7     // When jumping up we don't apply gravity when the user is δ
        holding the jump button.
8     if (jumping.jumping && jumping.holdingJumpButton) {
9         // Fly upwards until we have the max height
10        if (Time.time < jumping.lastStartTime + jumping.extraHeight / δ
            CalculateJumpVerticalSpeed(jumping.baseHeight)) {
11            // Negate the gravity we just applied, except we push in δ
                jumpDir rather than jump upwards.
12            velocity += jumping.jumpDir * movement.gravity * Time.δ
                deltaTime;
13        }else{
14            //disable gravity as long as the jump button is hold
15            velocity.y = 0.0;
16        }
17    }
18
19    // Make sure we don't fall any faster than maxFallSpeed. This δ
        gives our character a terminal velocity.
20    velocity.y = Mathf.Max (velocity.y, -movement.maxFallSpeed);
21 }
```

8 Anhang

Für die erhöhte Bewegungsgeschwindigkeit in der Luft wurde in *GetDesiredHorizontalVelocity* ein Else hinzugefügt, welches die maximale Geschwindigkeit mit einem Wert multipliziert, solange man in der Luft ist.

```
1 private function GetDesiredHorizontalVelocity () {
2     // Find desired velocity
3     var desiredLocalDirection : Vector3 = tr.›
4         InverseTransformDirection(inputMoveDirection);
5     var maxSpeed : float = MaxSpeedInDirection(desiredLocalDirection)›
6         ;
7     if (grounded) {
8         // Modify max speed on slopes based on slope speed multiplier ›
9             curve
10        var movementSlopeAngle = Mathf.Asin(movement.velocity.›
11            normalized.y) * Mathf.Rad2Deg;
12        maxSpeed *= movement.slopeSpeedMultiplier.Evaluate(›
13            movementSlopeAngle);
14    } else{
15        // Modify max speed in the air
16        maxSpeed *= 15.0;
17    }
18    return tr.TransformDirection(desiredLocalDirection * maxSpeed);
19 }
```

Literaturverzeichnis

- [cas14] castAR. castAR, Juli 2014. <http://technicalillusions.com/>.
- [DBGJ13] Ralf Dörner, Wolfgang Broll, Paul Grimm, and Bernhard Jung. *Virtual und Augmented Reality*. Springer-Verlag Berlin, Heidelberg, 2013.
- [Eps14] Epson. Moverio BT-200 Product Page, Juli 2014. <http://www.epson.de/moverio>.
- [Hai12] Eric Haines. Object2Terrainl, 2012. <http://wiki.unity3d.com/index.php?title=Object2Terrain>.
- [Hea14] Chris Heathcote. Shoes that see: Le Chal, 2014. <http://postscapes.com/shoes-that-see-le-chal>.
- [KM08] Georg Klein and David Murray. Parallel Tracking and Mapping on a Camera Phone, 2008. <http://www.robots.ox.ac.uk/~gk/publications/KleinMurray2009ISMAR.pdf>.
- [KM09] Georg Klein and David Murray. Video-rate Localization in Multiple Maps for Wearable Augmented Reality, 2009. http://www.robots.ox.ac.uk/~bob/publications/castle_etal_iswc2008/castle_etal_iswc2008.pdf.
- [Lan12] Ben Lang. Augmented Reality Welding Mask Provides 100 Million to 1 Contrast Ratio in 3D for Precision Welding, 2012. <http://www.roadtovr.com/augmented-reality-welding-mask/>.

Literaturverzeichnis

- [Mü12] Florian Müller. Remembering in the Metaverse: Preservation, Evaluation, and Perception, 2012. http://edoc.unibas.ch/16452/1/diss_mueller_pdfa_final.pdf.
- [Ocu14a] Oculus. Oculus Rift DK1, Februar 2014. <http://www.oculusvr.com/>.
- [Ocu14b] Oculus. Oculus Rift DK2, Juli 2014. <http://www.oculusvr.com/>.
- [Sta14] Michelle Star. Norwegian army trials Oculus Rift to drive tanks, 2014. <http://www.cnet.com/news/norwegian-army-trials-oculus-rift-to-drive-tanks/>.
- [Ste13] William Steptoe. AR Rift, 2013. <http://willsteptoe.com/post/66968953089/ar-rift-part-1>.
- [SW97] Mel Slater and Sylvia Wilbur. *A framework for immersive virtual environments(FIVE): specultion on the role of presence in virtual environments.* 1997. <http://publicationslist.org/data/melslater/ref-232/pres5.pdf>.
- [Vuz14] Vuzix. Wrap 1200DX AR Product Page, Juli 2014. http://store.vuzix.co.uk/euro/acatalog/Wrap_1200DX_AR.html.
- [Wik14] Wiki. Six Degrees of Freedom, 2014. http://en.wikipedia.org/wiki/Six_degrees_of_freedom.

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, dass alle Stellen der Arbeit, die wörtlich oder sinngemäß aus anderen Quellen übernommen wurden, als solche kenntlich gemacht und dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegt wurde.

Ort, Datum

Unterschrift