1.3 Durchgeführte Tätigkeiten

Tätigkeit	Beschreibung	von – bis
VR-Recherche	Aufgaben: • Verschiedene Interaktions- und Bewegungsmöglichkeiten für VR finden, in Unity implementieren und präsentieren	02.10.17 – 06.10.17
PAL-App	Abschluss der IOS/Android App Aufgaben: • Bugs Fixen • Visualisierung starker schmerzen • Neue Charaktere erstellt • UI für den Erste Hilfe Koffer entwerfen • App UI bearbeiten	09.10.17 – 11.10.17
MANV VR	Erstellen eines neuen VR Projekts auf Basis der PAL-App Aufgaben: • Projekt und Repository erstellen • Game Master UI erstellen • Sound Manager schreiben • Wetter Manager schreiben und Partikeleffekte erstellen(Regen, Schnee, Blitze) • Szene aus 3d Assets bauen • Launcher erstellen • Recherche zur Performance Optimierung und deren Anwendung • Charaktere aus der App kompatibel machen • Konzept und Präsentation einer Ingame UI und deren Implementierung • Bewegung und Interaktion mit Gegenständen/UIs ermöglichen • Patienten Hotspots interaktiv machen	12.10.17 – 16.02.17

	• Unity Editor Tools schreiben • Recherche für einen Multiplayer Modus • Multiplayer Modus implementiert und älteren Code neu schreiben um ihn kompatibel zu machen • Avatar für die Mitspieler in Blender erstellen und einfügen • Distraction Manager schreiben • Verschiedene Ablenkungen hinzufügen (Autoalarm, Helikopter, Schreie, Blaulicht) • Animation des Helikopters • Game Audio bearbeiten • Performante Minimap erstellen • Diverse Anleitung schreiben • Software Engineering • Erstellen eines Klassendiagramms zur Übertragung der Simulationslogik von der App in VR • Refactoring des Codes um ihn Fehlerresistent zu machen • Erstellen einer Dokumentation • Bugs Fixen • Fehlerfreien Build erstellen um ihn möglichen Kunden zu präsentieren	
Photogrammmetrie	Recherche zu Fotogrammmetrie in Unity 3D Aufgaben • Präsentation der Ergebnisse • Schreiben einer gut verständlichen Anleitung	18.12.17 – 20.12.17
Architektur in VR	Aufgaben • Anleitung schreiben: 3ds Max to Unity VR	12.01.18 – 15.01.18

2 Projekt

2.1 MANV-VR

2.1.1 Aufgabenstellung

Die Aufgabe bestand darin, die wesentlichen Funktionen der fast fertigen PAL-App(MANV Simulation) in eine Virtual Reality-Anwendung zu übertragen. Da sich die Bedienung und auch die Möglichkeiten fundamental zur App unterscheiden, ist es notwendig, sich mit den grundsätzlichen Eigenschaften von „Virtuellen Welten“ zu beschäftigen. So ändert sich nicht nur die Darstellung von einem Display auf eine spezielle Brille und zusätzlich einem Bildschirm, sondern auch das grundsätzliche Bedienkonzept muss von Grund auf neu überdacht werden. Durch die erhöhten Anforderungen ändern sich auch die Anforderungen an die Hardware. So wird zum Beispiel neben der speziellen VR-Brille ein Desktop-PC mit einer leistungsstarken Grafikkarte benötigt. Aber auch Softwareseitig muss auf den Detailgrad und die Optimierung der Skripte geachtet werden, um die Performance nicht zu gefährden. Ziel ist es, durch einen erhöhten Realismus in einer VR-Anwendung die Simulationsteilnehmer noch mehr in die Lage zu versetzen, wie in einem Ernstfall zu handeln, möglicher weise sogar gezielt an Stresssituationen zu gewöhnen, um so den Trainingserfolg zu maximieren.

2.1.2 Methoden und Werkzeuge

In diesem Projekt wurde bei der Entwicklungsumgebung auf Unity und Visual Studio gesetzt. Der Grund dafür ist die schon vorhandene Schnittstelle für Entwickler zur VR-Hardware. Und auch GIT und Redmine wurden für die erfolgreiche Umsetzung des Projektes verwendet.
Gängige VR-Brillen wie die HTC vive oder Oculus Rift verwenden die Software SteamVR. Sie stellt die Treiber für die Brille zur Verfügung und übernimmt die Kalibrierung des Geräts.
Um die VR-Brille und die Controller in eine Unity-Anwendung einzubinden wurde auf das Toolkit „VRTK“ (Virtual Reality ToolKit) zurückgegriffen. Das Virtual Reality Toolkit bietet die Möglichkeit, durch erweiterbare Skripte mit verschiedener VR-Hardware die Interaktion mit Spielobjekten erheblich zu erleichtern, sowie die Möglichkeiten die Steuerung, Fortbewegung und den Entwicklungskomfort zu erweitern.

Da für die Anwendung auch ein Multiplayer-Modus entwickelt werden sollte wurde auf die „Photon Network“ Bibliothek zurückgegriffen. Sie regelt die Netzwerkkommunikation und stellt einen gemeinsamen Raum auf einem Server zur Verfügung.
Um 3D-Objekte auch selbst erstellen zu können wurden im Entwicklungsprozess immer wieder auf Blender zurückgegriffen, um Objekte zu modellieren.
Als Vorgehensmodell entschieden wir uns für Scrum. So gehörte ein Standup-Meeting zu jedem Tag und der Entwicklungsprozess der einzelnen User-Storys wurde auf einem Srum-Board festgehalten. Neu entstehende Tasks wurden in Redmine festgehalten und dann nach Prioritäten abgearbeitet. In einem Sprint-Review informierten wir unsere Vorgesetzten regelmäßig über den Fortschritt des Projektes und ließen neue Anforderung in die Planung mit einfließen.

2.1.3 Durchführung

Das Projekt lässt sich in einzelne Phasen gliedern.

Erstellung der Szene

Zunächst einmal war es notwendig, eine Szene in Unity zu erstellen, in der man sich später mit der VR-Brille bewegen konnte. Eine leere Szene besteht in Unity zunächst nur aus einer Kamera und einer Lichtquelle. Da das modellieren jedes einzelnen Spielinhaltes (so genannte Assets) zu aufwändig wäre, wurde auf den von Unity bereitgestellten Asset-Store zurückgegriffen. Hier befinden sich sowohl kostenpflichtige als auch frei herunterladbare Inhalte. Da nicht jedes Objekt den Ansprüchen entspricht, ist hier ein gewisser Rechercheaufwand notwendig, um für die eigene Szene verwendbare Assets zu finden. So war es die Aufgabe, zunächst eine Szene in einer Stadt mit verschiedenen Fahrzeugen aufzubauen. Durch einen Trick konnte ein Häuserblock-Asset jeweils vier Mal verwendet werden, um so eine Straßenkreuzung zu inszenieren. Durch einzelne Fahrzeuge und einige Rettungsfahrzeuge war so ein erster Grundaufbau der Szene gegeben.

Abbildung 1

Nun mussten einige Details wie 3D-Cubes hinzugefügt werden, um das hindurchscheinen des Lichtes durch die Gebäude zu verhindern oder um einzelne Objekte in der Szene ein Collider hinzugefügt werden, da sich der User später sonst durch die verschiedenen Objekte bewegen hätte können. Da sich der Spieler nur in einem gewissen Bereich in der Szene bewegen dürfte, musste die Szene mit Barrieren beschränkt werden. Diese wurden in Blender modelliert und gezielt am Ende der jeweiligen Straße angebracht. Die Patienten für die Simulation konnten aus der PAL-App (siehe Abbildung 1) importiert werden. So war ein erster Prototyp der Szene gegeben.

VRTK Einbindung

Nach längerer Recherche stellte sich heraus, dass das Virtual Reality Toolkit einen gewissen Umfang an Funktionen bieten konnte, der für unser Projekt benötigt wurde. So ist es zum Beispiel notwendig, das Headset der VR-Brille mit einer SDK einzubinden. Um nicht nur eine VR-Brille nutzen zu können bietet VRTK zum Beispiel die Möglichkeit, verschiedene SDKs benutzen zu können. Des Weiteren steht mit VRTK eine „Simulator“-SDK zur Verfügung, die es ermöglicht auch ohne VR-Brille die Grundfunktionen zu testen und sich in der Szene mit den „WASD“-Tasten bewegen zu können. Und auch die Interaktion zwischen den Controllern des VR-Headsets und Objekten aus der Szene lässt sich mit VRTK deutlich einfacher realisieren. Dennoch ist eine gewisse Einarbeitungszeit für die einzelnen Funktionen notwendig. Dies sind nur zwei Beispiele für die Funktionen, die VRTK liefert. Im Laufe des Praktikums wurde immer wieder auf dieses Toolkit zurückgegriffen.

(Fort)-Bewegung

Auch Fortbewegung des eigenen Charakters in der Szene oder die Bewegung im realen Raum muss in VR komplett neu überdacht werden. So

steht dem User durch das mit Kabel verbundene Headset nur ein gewisses Interaktionsfeld zur Verfügung, in dem er sich bewegen kann. Auch durch die Sensoren, die das Headset und die Controller tracken wird der Raum begrenzt. So muss für die Fortbewegung ein neues Bedienkonzept integriert werden. Im Praktikum wurden mit verschiedenen Bewegungskonzepten die einzelnen Vor- und Nachteile ausprobiert. In der Gruppe einigten wir uns dann auf ein Konzept aus drei Möglichkeiten.
Zum einen implementierten wir die Möglichkeit, mit dem Druck auf die „Grab“-Taste und dem gleichzeitigen auf- und ab bewegen der Controller sich in kleinen Schritten fortzubewegen (siehe Abbildung 2). Dies bietet die Möglichkeit, auch kleinere Distanzen in der Szene kontrolliert zurück zu legen. Der Nachteil dieser Variante ist es, dass das Gehirn getäuscht wird. So findet zwar Visuell eine Fortbewegung statt, der Körper selbst kann diese aber nicht feststellen. So treten bei vereinselten Usern nebenerscheinungen wie die sogenannte „Motion sickness“ auf.

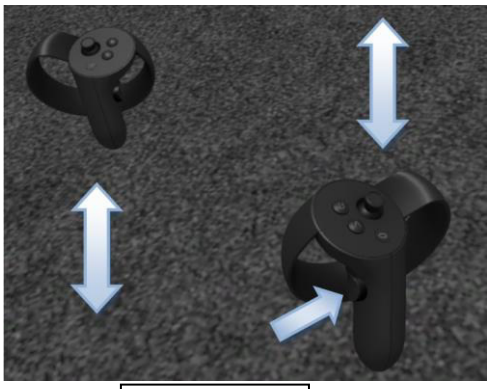


Abbildung 2

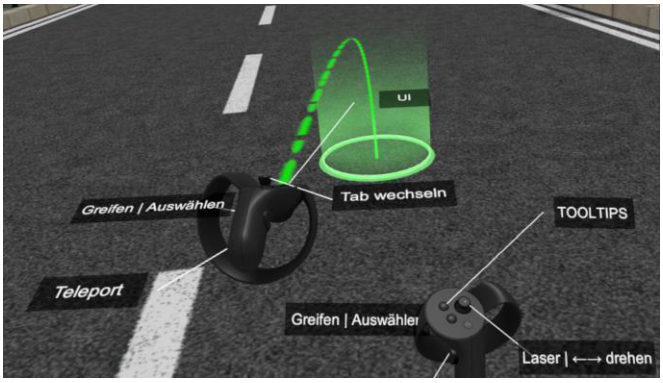


Abbildung 3

Die zweite Fortbewegungsmöglichkeit, die implementiert wurde ist das so genannte Teleportieren (Abbildung 3). Hier kann durch einen Tastendruck am Controller eine Parabel aktiviert werden, die den Zielort der Teleportation aufzeigt. Nach dem „los lassen“ des Buttons befindet sich der Charakter an der gewünschten Position. Hierbei tritt das Problem der Motion Sickness nicht so häufig auf.
Da durch manche Headsets (wie der Oculus) das Drehen des Kopfes nur eingeschränkt möglich ist, da man sich sonst aus dem Erkennungsbereich der Sensoren bewegt musste noch eine Möglichkeit implementiert werden, um den gesamten Charakter zu drehen, ohne sich jedes mal tatsächlich vom Monitor weg drehen zu müssen. Dies wurde durch das nach rechts und links Bewegen des Joysticks am rechten Controller realisiert.

GameMaster

Zusätzlich zu der Bedienung über die Brille wurde die Möglichkeit geschaffen, verschiedene Einflüsse in der Szene zu ändern. So kann über den Monitor, der an dem Desktop-PC angeschlossen ist (an dem auch die Brille angeschlossen ist) ein Bedienelement angezeigt werden. Das so genannte GameMaster-UI. Dies dient zu Steuerung der Simulation sowie der Überwachung der Vitalparameter der einzelnen Patienten in der Szene. So kann zum Beispiel der Simulationsleiter über eine Minimap (siehe Abbildung 4) das Geschehen in der Szene überwachen. Hierzu wurde eine zusätzliche Kamera in der Szene integriert, die nur Objekte auf bestimmten Layern war nimmt. So wird der Rechenaufwand minimiert, um die Performance zu schonen und der Simulationsleiter hat einen guten Überblick über die Szene. Ein weiteres Element, auf das der Simulationsleiter (der so genannte GameMaster) Einfluss hat ist die Steuerung der Sounds. Hier besteht die Auswahl zwischen zuschaltbaren Geräuschen oder Ambienten Sounds. Sie lassen sich nach belieben aktivieren und deaktivieren, beziehungsweise die Lautstärke ändern.

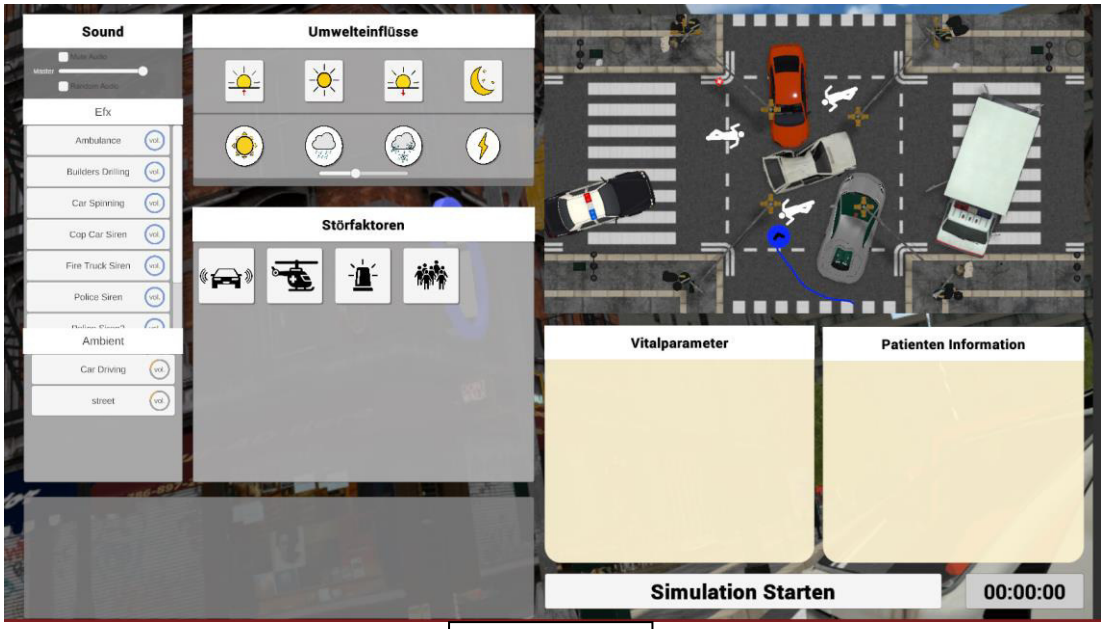


Abbildung 4

Zwei weitere Bestandteile des GameMaster-UIs sind die Umwelteinflüsse und Störfaktoren.

Umwelteinflüsse

Abbildung 5

Über die verschiedenen Buttons im Umwelteinflüsse-Fenster lässt sich unter anderem die Tageszeit und Wetterlage ändern, sowie die Intensität von Regen und Schnee, die für den Simulationsteilnehmer wahrgenommen wird. Die Tageszeit verändert den Einfallwinkel der Lichtquelle und lässt die Farben je nach „Sonnenstand“ greller oder blasser wirken. Für den Niederschlag wird ein Partikel-System verwendet, dass an der Position des jeweiligen Spielers kleineste Grafiken instanziiert (siehe Abbildung 5).

Störfaktoren

Einen weiteren Einfluss, den der Game Master auf die Szene hat, sind die so genannten Störfaktoren. Hier wurde im Entwicklungsprozess immer wieder neue Elemente hinzugefügt. Sie nehmen direkt auf die Szene Einfluss und können so den Simulationsteilnehmer in eine Stresslage versetzen. Auch hierdurch wird der Teilnehmer noch einmal speziell dafür geschult, in Stresssituationen Ruhe zu bewahren und sein erlerntes Wissen weiterhin anzuwenden.

So ist es unter anderem möglich, Für ein bestimmtes Fahrzeug in der Szene die Alarmanlage zu starten, was sich mit blinkenden Lichtern und einem schrillen Geräusch bemerkbar macht.

Abbildung 6

Ein zusätzlicher Störfaktor ist das Aktivieren eines Helikopters. Dieser fliegt dann über einen festgelegten Pfad über die Szene, versucht zu landen und verlässt das Geschehen wieder, dies wird mit einem passenden Soundunterstrichen.

Abbildung 7

Auch hat der Game Master die Möglichkeit, von den Einsatzwagen das Blaulicht zu aktivieren, um so für den Simulationsteilnehmer das Gefühl hervorzuheben, an einem Einsatzort zu sein.

Abbildung 8

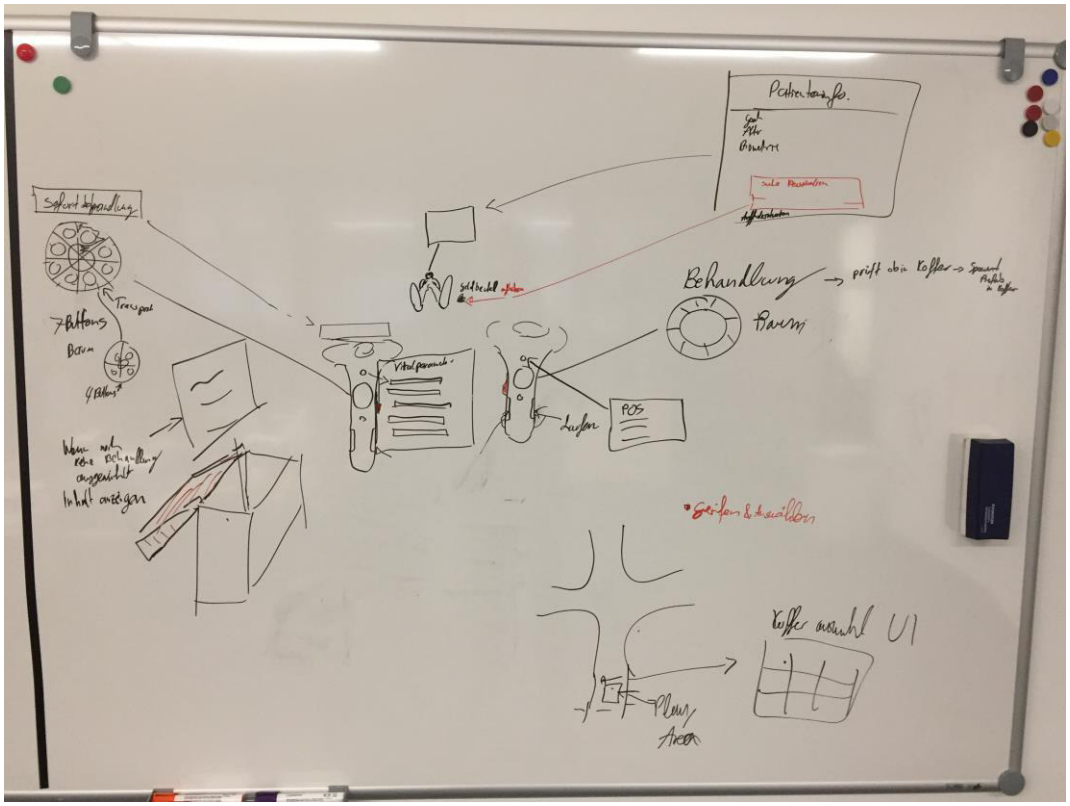
Abbildung 9

Um den Realismus der Szene zu erhöhen wurden einzelne Schaulustige der Szene hinzugefügt. Sie stehen um die jeweiligen Patienten herum und versuchen Kontakt aufzunehmen, einzelne so genannter „Bystander“ laufen auf einem festgelegten Pfad durch die Szene und rufen nach Hilfe.

User Interface-Gestaltung

Da das Bedienkonzept der PAL-App nicht übernommen werden konnte, musste für die VR-Anwendung ein komplett neues User Interface-Konzept entworfen werden, was den gleichen Umfang an Informationen und Interaktionsmöglichkeiten bietet wie es die App zur Verfügung stellt. Zunächst setzten wir uns mit den verschiedenen Funktionen der App auseinander und testeten, wie dies in einer VR-Anwendung umzusetzen sei. So war es eine Aufgabe, einfache Informationsfenster der App neu zu entwerfen und für eine VR-Anwendung umzugestalten. Aber auch ganz neue Bedienkonzepte konnten dank der Interaktion zwischen den Kontrollern und Objekten entworfen werden (dazu mehr im folgenden Kapitel „Interaktion“). In der Folgenden Abbildung ist ein erster Entwurf des grundsätzlichen Bedienkonzepts zu sehen. So wurde mit verschiedenen Entwürfen ausprobiert, welcher Weg die nutzerfreundlichste Bedienung bietet und zugleich alle Vorteile der Virtual Reality ausnutzt.

Abbildung 10



Zunächst wurden die einzelnen Funktionen der PAL-App aufgelistet und dann versucht mit den Möglichkeiten der VR die Funktionen möglichst nutzerfreundlich und intuitiv auf die Controller (Abbildung 10 Mitte) und andere Elemente unter zu bringen. Das runde Menü links in der Abbildung 10 beispielsweise war ein erster Entwurf eines Radialmenüs. Dieses wurde dann aber aufgrund der Unübersichtlichkeit wieder verworfen.

Abbildung 11

Stattdessen entschieden wir uns für ein UI-Board am linken Kontroller (siehe Abbildung 12), das sich über die Menü-Taste ein und ausblenden lässt. Die einzelnen Funktionen am Board lassen sich mit einem Pointer am rechten Controller auswählen – ähnlich einer Maus als Zeigegerät.
So wurden unter anderem die Vitalparameteranzeige für die einzelnen Patienten in diesem Board untergebracht. So können mit dem Pointer die einzelnen Vitalparameterfunktionen abgerufen und so die Gesundheitliche Lage des Patienten gut überwacht werden. Die Anzeige über dem Board zeigt noch einmal eine komplette Erklärung des Parameters.

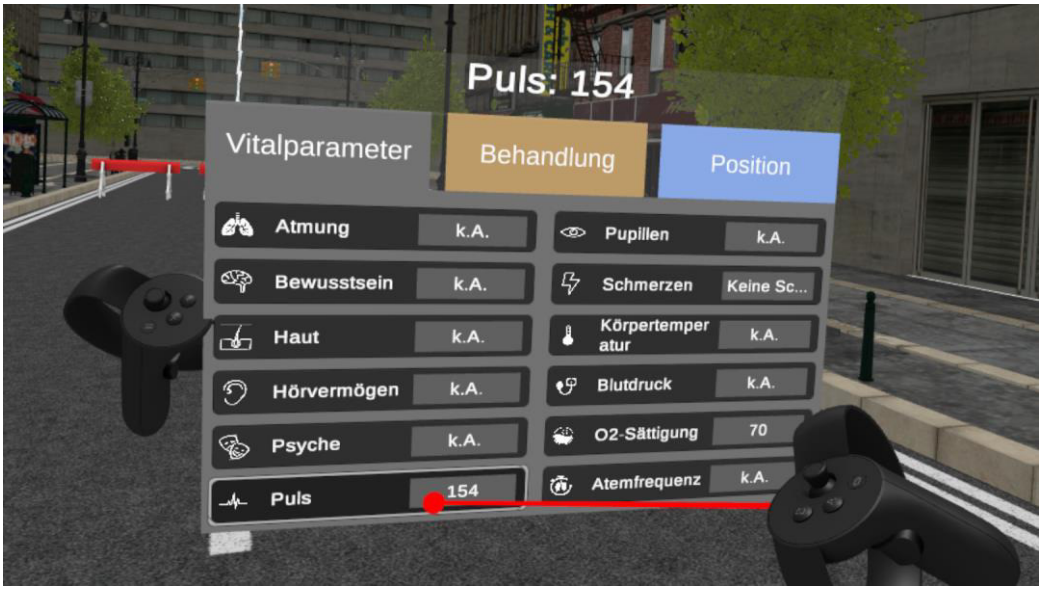


Abbildung 12

Als weiteres UI-Element wurden die Patienteninformationen in eine Anzeige über die jeweiligen Patienten hinzugefügt. Sie stellt grundsätzliche Informationen wie „Geschlecht“, „Alter“, „Personalien“ und „Auffindesituation“ da. Die Patienteninformation ist zunächst deaktiviert und lässt sich mit dem Pointer und der durch einen Klick auf das 3D-Info Icon über dem Patienten aktivieren. Zunächst sind keinerlei Informationen über die Personalien des Charakters eingeblendet. Ein Feld in der Mitte des Fensters weist darauf hin, dass nach einem Portemonnaie (Wallet) gesucht werden muss, um die Daten zu erhalten (dazu Mehr im folgenden Kapitel „Interaktion“).

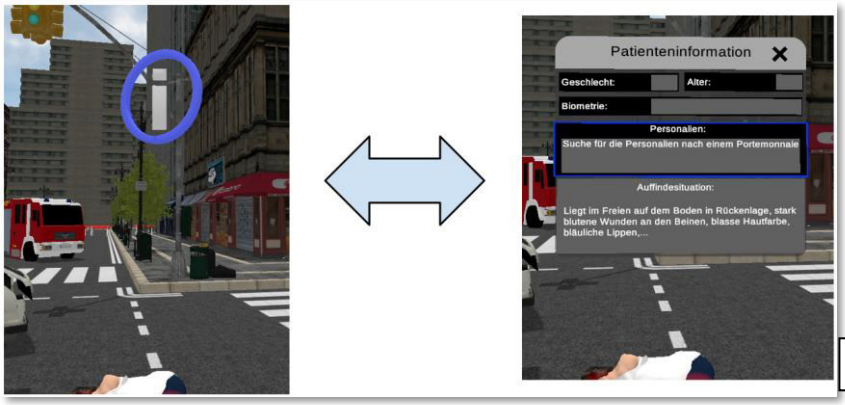


Abbildung 13

Interaktion (UIs, Wallet, Koffer)

Aber auch die Interaktionsmöglichkeiten ändern sich von Grund auf in Virtual Reality. So können Lösungen, die in einer Standard-Anwendung mit Buttons realisiert wurden, nun durch Interaktionen mit dem Controller deutlich realitätsnäher umgesetzt werden.

Ein Beispiel für diese Interaktion ist die implementierte Wallet. Sie wurde zunächst mit Blender modelliert und dann in Unity eingefügt. Um den Nutzer die Funktion zu erklären befindet sich in der Patienteninformation ein Hinweis, dass er nach einem Portemonnaie Ausschau halten soll. Dieses befindet sich meist wenige Meter vom Patienten entfernt. Berührt der Simulationsteilnehmer nun mit einem Controller die Geldbörse, „fliegt“ diese in Richtung der Patienteninformation. Anschließend werden die aktuellen Patienteninformationen in den dafür vorgesehenen Felder angezeigt. In der Abbildung 14 ist ein Beispiel für einen Patienten mit einer Geldbörse zu sehen.

Abbildung 14

Des Weiteren ist auf der Abbildung ein Notfallkoffer zu erkennen. Dieser wurde zu Testzwecken entworfen und bietet im Moment nur eine demonstrative Funktion. Für den User ist es möglich, wie in der realen Welt den Koffer am Griff zu öffnen. Daraufhin öffnet sich ein User Interface, das verschiedene Buttons zur Auswahl gibt. In Zukunft sollen hier die verschiedenen Medikamente und Behandlungen zur Auswahl stehen, die einer Rettungskraft auch im Real Fall zur Verfügung stehen. Um den Realitätsgrad noch weiter zu erhöhen, sollte sich an den zurzeit verwendeten Koffern und Rucksäcken für Rettungssanitäter orientiert werden. Nach einigen Versuchen kristallisierte sich aber heraus, dass der Modellierungsaufwand für jeden einzelnen Koffer den Aufwand übersteigen würde und so wurde nach neuen Möglichkeiten gesucht, Objekte aus dem realen Umfeld in die Szene zu integrieren. Nach längerer Recherche wurden einige Versuche mit der Photogrammmetrie durchgeführt.

Photogrammmetrie

Die Photogrammmetrie ist ein Verfahren, um aus einer gewissen Anzahl an Fotografien eines Objektes eine dreidimensionale Form zu erstellen. So können unter anderem „.obj“-Dateien erzeugt und in beliebige Unity-Szenen importiert werden. Zu Beginn werden aus verschiedenen Blickrichtungen und Winkeln Fotos des zu digitalisierenden Objektes angefertigt. Diese werden nun in eine spezielle Software wie „PhotoScan“ importiert, die nun die einzelnen Zusammenhänge zwischen den Bildern errechnet und daraus ein Modell formt. Für unser Projekt wurden mehrere Testdurchläufe unternommen, um heraus zu finden welche Kamera, Kameraeinstellungen und Software sich am besten für das Verfahren eignen. Einen Überblick über die Entwicklungsstufen der Objekte sind in der Abbildung 15 bis 17 dargestellt.

Abbildung 15

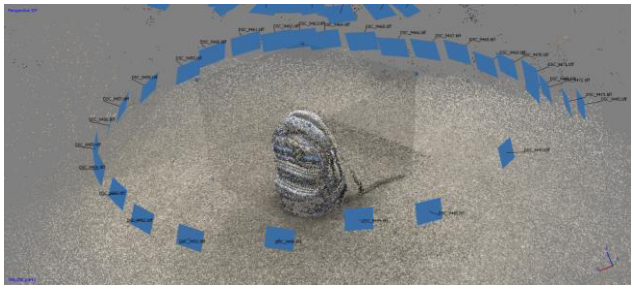


Abbildung 16

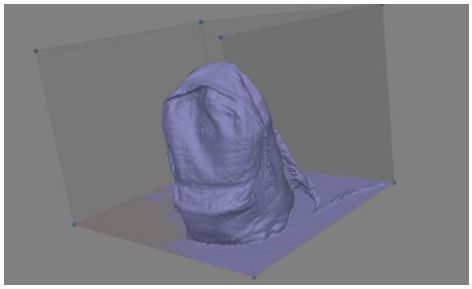


Abbildung 17

Multiplayer-Modus

Die grundsätzliche Idee der Simulation ist den Trainingserfolg der Rettungskräfte nachhaltig zu erhöhen. Zu dieser Aufgabe gehört auch die Zusammenarbeit mit anderen Helfern. Um diesen Aspekt in die Simulation mit einfließen zu lassen, wurde beschlossen einen Multiplayer-Modus mit einzuführen, um mehrere Rettungskräfte gleichzeitig in einer Szene zusammen arbeiten lassen zu können. Nach dem Vergleich von mehreren Netzwerklösungen wurde sich für die Lösung von „Photon Network“ entschieden. Diese stellt die Netzwerkkommunikation und einen gemeinsamen Raum auf einem Server zur Verfügung. Um die Funktionen und Anwendung von Photon Network implementieren zu können mussten zunächst das grundsätzliche Vorgehen mit der Netzwerkschnittstelle erlernt werden. Aus diesem Grund bestand zunächst die Aufgabe darin das Einstiegs-Tutorial von Photon-Network umzusetzen. Hier wurden die wichtigsten Funktionen anhand von Beispielen nähergebracht. Grundsätzlich gibt es mehrere Möglichkeiten, die Funktionen und Aktivitäten zwischen den verschiedenen Simulationsteilnehmern zu synchronisieren. Um die Daten einem eindeutigen Spieler zuzuordnen, wird jedem Spieler eine „Photon View“-Komponente hinzugefügt. Diese wird von Photon Network zur Verfügung gestellt und weist den zu synchronisierenden Elementen eine eindeutige ID zu. Für die Synchronisation von zum Beispiel Bewegungen der einzelnen Simulationsteilnehmer muss zunächst eine „Transform View“ auf den Avatar, der den Spieler darstellt, hinzugefügt werden. In den Einstellungen der „Transform View“ kann dann die Synchronisation von Position (Translation), Rotation und Skalierung aktiviert, beziehungsweise deaktiviert werden.

Auch Funktionen wie die Störfaktoren oder Umwelteinflüsse werden synchronisiert, um bei jedem Simulationsteilnehmer im gleichen Raum vergleichbare Simulationsbedingungen zu schaffen. Dies wird erzielt, indem Funktionen mit einer so genannten „RPC“ versehen werden. „RPC“ steht für Remote Procedure Call und bedeutet, dass wenn bei einem Spieler einzelne Funktionen aufgerufen werden, diese auch bei den anderen Speilern im selben Raum aufgerufen werden. Tritt ein Spieler erst im Laufe der Simulation bei, können diese Funktionen auch noch im Nachhinein aufgerufen werden. So ist es möglich, für jeden beigetretenen Simulationsteilnehmer die gleichen Bedingungen zu schaffen.

Um in den selben Raum beitreten zu können muss von jedem neuen User zunächst eine Verbindung zum Server und daraufhin dem gewünschten Raum aufgebaut werden. Um dies zu ermöglichen wurde die „Launcher“-Szene erstellt.

Launcher

Die Launcher-Szene wird zu Beginn der Simulation angezeigt und bietet mehrere Einstellungsmöglichkeiten. So ist es zum einen möglich, die verschiedenen Patienten auszuwählen, die sich in der Szene befinden sollen. Des Weiteren wird nach wie vor die Möglichkeit geboten, die Szene Offline zu erstellen und auch ohne Internetverbindung eine Simulation durchführen zu können. Wurde bereits von einem anderen Simulationsteilnehmer eine Szene erstellt, ist es möglich dieser durch den selben Raumnahmen beizutreten um so in einer gemeinsamen Szene eine Simulation durchzuführen. In der Folgenden Abbildung 18 ist die Launcher-Szene mit ihren Einstellungsmöglichkeiten zu erkennen.

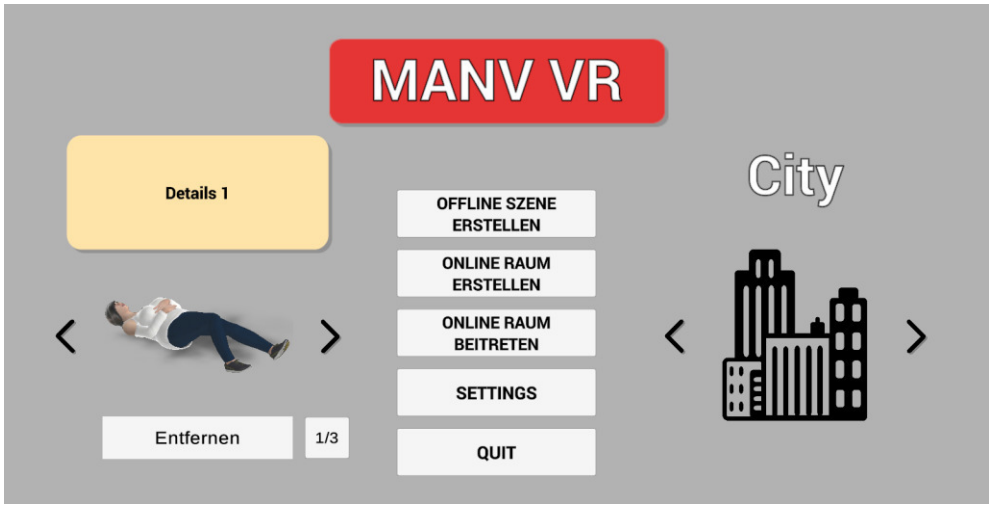


Abbildung 18

2.1.4 Ausblick

Die Anwendung wurde an einen Werksstudenten von nVista technologies übergeben. Dieser wird in der nächsten Zeit weitere Funktionen implementieren und das Projekt weiterhin betreuen. Die Simulation selbst bietet einen guten Überblick, was mit dem heutigen Stand der Technik im Bereich der Rettungssimulationen umsetzbar ist und wird möglichen Kunden als Demonstration vorgeführt.