Hochschule für Angewandte Wissenschaften Hamburg
*Hamburg University of Applied Sciences*

# Bachelorarbeit

**Eric Salomon**

## Analyse des WebVR-Standards am Beispiel eines virtuellen interaktiven Museumserlebnisses

*Fakultät Technik und Informatik*
*Studiendepartment Informatik*

*Faculty of Engineering and Computer Science*
*Department of Computer Science*

Eric Salomon

# Analyse des WebVR-Standards am Beispiel eines virtuellen interaktiven Museumserlebnisses

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Philipp Jenke
Zweitgutachter: Prof. Dr. Stefan Sarstedt

Eingereicht am: 20. März 2018

**Eric Salomon**

**Thema der Arbeit**
Analyse des WebVR-Standards am Beispiel eines virtuellen interaktiven Museumserlebnisses

**Stichworte**
WebVR, Virtual Reality, HTC Vive, Oculus Rift, Virtuelles Museum

**Kurzzusammenfassung**
Diese Ausarbeitung erörtert den aktuellen Zustand der Technologie von Virtual Reality im Web. Hierfür wurde prototypisch ein virtuelles Museum entwickelt, welches auf Grund einer Anforderungsanalyse mehrere Funktionen testet, die von einer eigenständigen VR-Anwendung erwartet werden können. Hierzu zählt die Möglichkeit des Umherbewegens durch den Raum sowie das Aufheben von Objekten. Ebenfalls wurde demonstriert, wie eine Navigation durch das Web innerhalb einer solchen VR-Anwendung möglich ist.

**Eric Salomon**

**Title of the paper**
Analysis of the WebVR standard using the example of a virtual interactive museum experience

**Keywords**
WebVR, Virtual Reality, HTC Vive, Oculus Rift, Virtual Museum

**Abstract**
This thesis discusses the current state of the technology of virtual reality on the web. For this purpose, a prototypical virtual museum was developed, using a requirement analysis, with requirements which could be expected of a standalone VR application. This includes the ability to move around the room or picking up objects. Additionally, it was demonstrated how navigation through the web may be possible, while being in such a VR application.

# Inhaltsverzeichnis

# 1 Einleitung

## 1.1 Motivation

Die Entwicklung des 'Sword of Damocles' [Sutherland (1968)], dem womöglich ersten Head-Mounted Display, im Jahre 1968 kann als Startpunkt der modernen Virtual Reality in der Computergraphik gesehen werden. Ein Head-Mounted Display ist ein Gerät, welches vor das Gesicht des Benutzers geschnallt wird und mit Hilfe von einem oder mehreren Bildschirmen Bilder präsentiert. Moderne Vertreter von Head-Mounted Displays sind zum Beispiel die Oculus Rift oder die HTC Vive.
Seit 1968 kam das Thema VR zwischenzeitlich immer wieder auf. Jedoch war keine Technologie fortgeschritten genug, um für den Privatgebrauch ein Gerät herauszubringen. Lange Zeit war es nicht möglich, ein preiswertes und zugleich qualitativ hochwertiges Gerät zu entwickeln.

Man kann nun argumentieren, dass mit der Akquirierung von Oculus VR von Facebook 2014 [Bounds (2014)] neues Interesse in der Industrie entfacht wurde, neue Head-Mounted Displays und Virtual Reality Anwendungen zu entwickeln, um den VR-Markt zu erobern. Ob dies der wirkliche Grund ist lässt sich nicht klar sagen, jedoch ist es eindeutig, dass in der Öffentlichkeit das Thema VR wieder in aller Munde ist.
Mit Preisen von 400 bis 700 Euro liegen diese Brillen finanziell im höheren Bereich. Jedoch sind diese sogenannten Virtual-Reality Brillen nicht mehr der einzige Weg, um in die Virtuelle Realität einzutauchen. 2014 hat Google die erste Version der Google Cardboard veröffentlicht. Dies ist mehr eine Halterung für das Smartphone mit speziellen Linsen, welche es möglich macht, mit dem Smartphone VR-Anwendungen zu nutzen. Da heutzutage nahezu jede Person ein Smartphone besitzt, sind solche Halterungen eine günstige Alternative für eine 'echte' Virtual-Reality Brille. Der größte Unterschied zwischen 'echten' VR-Brillen und den Brillen für Smartphones ist die Möglichkeit der Bewegung im dreidimensionalen Raum. Dies wird in 'Degrees of Freedom' (DOF) gemessen, wo Smartphones in der Regel weniger Freiheiten haben als VR-Brillen. Sie unterstützen höchstens ein Rotieren auf den Achsen (3-DOF), jedoch keine Translation, also Vor- und Zurückbewegen, auf den Achsen (6-DOF). Es existieren jedoch auch

Smartphones, welche Motion-Tracking Sensoren eingebaut haben, weshalb sie eine völlig freie Bewegung unterstützen. Diese sind Geräte, welche ARCore oder ARKit unterstützen, Augmented Reality Plattformen, um AR-Anwendungen auf dem Smartphone zu ermöglichen.[Google (23.02.2018); Elliott (2017)]

Des Weiteren muss, zum jetzigen Zeitpunkt, immer eine extra Anwendung heruntergeladen und installiert werden, die das VR-Erlebnis enthält. Diese Anwendungen sind jedoch meist sehr Speicherplatzintensiv, was einen eher langwierigen Aufwand bedeutet, bis die VR-Anwendung erlebt werden kann. Eine Änderung verspricht hier WebVR, was eine Technologie für den Browser ist. Hier übernimmt der Browser selbst die Darstellung der Virtuellen Realität, während die Webseite nur noch die Informationen über die Szene liefert.

## 1.2 Zielsetzung

Im Rahmen dieser Ausarbeitung wird das Thema WebVR genauer betrachtet und die Fähigkeiten der Technologie analysiert. Es soll erläutert werden, wie man als Anwender WebVR nutzen kann, aber auch wie ein Entwickler Anwendungen umsetzen kann. Hierfür soll prototypisch eine Anwendung entworfen und umgesetzt werden, welche auf WebVR basiert. Als Thema der Anwendung wurde ein Museum gewählt. Dies ist darauf zurückzuführen, dass zum Einen derzeit hauptsächlich Anwendungen für die Unterhaltung für VR erschaffen werden. Ein virtueller Museumsrundgang eignet sich hervorragend für dieses Thema, weil es zum Einen eine Erweiterung der Nutzung weg von einfacher Unterhaltung, hin zu Bildungszwecken aufzeigt, und zum Anderen, weil die Darstellung der Ausstellungsstücke einen hohen Qualitätsanspruch erfordert. Gemeint ist, dass Details vom Modell gut dargestellt werden, sodass eine möglichst guter Vergleich zur Realität existiert. Der hohe Detailgrad ist hier allerdings ein Widerspruch zur Web-Technologie, wo Daten am Besten klein gehalten werden sollten, um Übertragungen schnell zu halten. Außerdem bedeutet ein hoher Detailgrad von Modellen, dass es mehr Rechenaufwand für die Grafikengine, in diesem Fall der Browser, nach sich zieht. Dies ist ebenfalls eine gute Möglichkeit, die Grenzen von WebVR auszutesten.

## 1.3 Aufbau der Arbeit

Diese Ausarbeitung ist so strukturiert, dass zunächst eine Einführung in die Grundlagen zu WebVR erfolgt. Hier wird die Kompatibilität analysiert, welche Browser WebVR unterstützen, sowie welche VR-Brillen nutzbar sind. Ebenfalls in diesem Kapitel wird auf das Funktionsprinzip von WebVR, sowie auf die Eingabemethoden für ein VR-Erlebnis eingegangen. Außerdem wird behandelt, wie die Darstellung auf nicht-VR-Brillen umgesetzt ist. Zuletzt wird in dem Kapitel auch auf andere Arbeiten eingegangen, welche Bezug zu WebVR und den Thema virtuelles Museum besitzen.
Daraufhin wird das Konzept für den Prototypen vorgestellt. Hierfür wurden mehrere funktionale und nicht-funktionale Anforderungen in einer Art Lastenheft aufgestellt, welche zur Messbarkeit der Lösung beitragen sollen. Danach wird die Umsetzung der Anwendung beschrieben, welche gefolgt wird von der Auswertung der Technologie. Zuletzt wird thematisiert, wie der Prototyp weiterentwickelt werden könnte, und ein Ausblick in die Zukunft von WebVR wird gewagt.

# 2 Grundlagen

Dieses Kapitel soll den aktuellen Stand der Technik erörtern, wie die Technologie WebVR zum Zeitpunkt dieser Arbeit nutzbar ist.
Abschnitt 2.1 gibt eine kurze Einführung in die Begriffe Degree of Freedom, sowie Immersion und Presence. In 2.2 werden die derzeit unterstützenden Browser vorgestellt, sowie die Hardware im Sinne der VR-Brillen, welche zum Erleben von WebVR benutzbar sind.
In Abschnitt 2.3 wird auf den Aufbau und Ablauf einer WebVR-Session eingegangen. Außerdem werden andere Darstellungsmethoden erklärt, die keine Virtual Reality Brille erforden. Ebenfalls wird auf das Thema Eingabemethoden eingegangen, welche für die Kontrolle in WebVR Anwendungen genutzt werden können. In Abschnitt 2.4 werden ausgewählte, bestehende WebVR Anwendungen und ihre Besonderheiten betrachtet. Hierbei wird auch ein Augenmerk auf die VR Browser JanosVR und Supermedium gelegt. Ebenfalls wird Thematisiert, welches Interesse die Forschung an WebVR hat. Außerdem wird ein Blick auf ein anderes Projekt gerichtet, welches sich mit dem Thema virtuelles Museum auseinandergesetzt hat.

Bei diesem Kapitel ist es wichtig zu erinnern, dass dies eine Momentaufnahme des aktuellen Zustands von WebVR ist. Es ist zu erwarten, dass in naher und ferner Zukunft die Kompatibilität zwischen den Browsern und den VR-Headsets sich noch verbessern kann, da sich immer mehr Unternehmen an dem Projekt WebVR beteiligen. [w3c] Außerdem existiert bereits ein neues Draft für die WebVR API Spezifikation, welche zwar nicht implementiert werden soll, an welcher jedoch aktiv gearbeitet wird. [Vukicevic u. a. (2017)]

## 2.1 Begriffe

### 2.1.1 Degrees of Freedom

Unter 'Degrees of Freedom' (DOF) versteht man die Möglichkeit, wie ein Objekt im dreidimensionalen Raum bewegbar ist. Für jede Achse gibt es hier zwei Grade, einen für die Bewegung auf der Achse, vor und zurück, und das Rotieren um die Achse selbst. Demnach ist das Maximum ein 6-DOF, um ein Objekt zu bewegen. In diesem Format werden auch Anzeigegeräte für VR kategorisiert, so haben die HTC Vive sowie die Oculus Rift six degrees of freedom (6-DOF), aufgrund ihrer Sensoren. Diese werden in einem Raum installiert und nehmen die Position des Nutzers auf und setzt diese in einer Anwendung um, sogenanntes Positionstracking. Währenddessen haben die meisten Smartphones bisher kein Positionstracking, sondern besitzen nur die Möglichkeit, die Rotation auf den Achsen aufzunehmen. Dies entspricht three degrees of freedom (3-DOF). Eine Ausnahme bilden hier Geräte, die bereits ARCore und ARKit fähig sind. Das sind Plattformen für Android und iOS-Geräte, welche Augmented Reality (AR) Anwendungen ermöglichen. Diese Technologie bringt, zusammen mit zusätzlich im Gerät verbauten Sensoren, die Möglichkeit von Positionstracking, und somit 6-DOF. [Google (23.02.2018); Apple Inc (2018)] DOF muss beachtet werden, da es mit WebVR auch möglich ist, die selben VR-Inhalte auf dem Smartphone darzustellen, welche ebenfalls von Computern dargestellt werden können. Eine Problematik besteht darin, dass die Steuerung, die am PC genutzt wird, offensichtlich nicht am Smartphone genutzt werden kann. Stattdessen muss auf andere Möglichkeiten zurückgegriffen werden, wie zum Beispiel das Teleportieren mit Hilfe von visuellen Markierungen in der Anwendung.

### 2.1.2 Immersion & Presence

Beim Bau von virtuellen Welten ist oft die Rede von Immersion, welches fachsprachlich auch mit 'eintauchen' beschrieben werden kann. Nach Ernest W. Adams lässt sich Immersion in virtuellen Welten in 3 Teile zerlegen: taktische Immersion, strategische Immersion und narrative Immersion. Während sich die taktische und strategische Immersion sich eher auf Computerspiele bezieht, kann die narrative Immersion auch für andere Medien wie Filme oder Bücher benutzt werden. [Adams (2004)] Sie beschreibt das Gefühl, welches man bekommt, sobald man eine Verbindung mit der Geschichte oder mit Charakteren eingeht, und man quasi von dieser Welt 'eingezogen' wird. Entsprechend ist die Frage, ob man diese Welt als 'richtig' oder 'korrekt' empfindet. Hierbei ist weniger wichtig, ob diese Welt nach unserem Wissen, wie von der Physik, korrekt ist, sondern eher, ob sie in sich logisch ist. Deshalb können Welten mit Themen wie Magie trotzdem als immersiv empfunden werden.

Für VR exisitiert parallel das Konzept der Presence. Presence beschreibt das Erfahren von Umgebungen, als existieren sie in der echten Welt. Es ist das völlige Eintauchen in eine virtuelle Welt, welche nicht nur mental als korrekt empfunden wird, sondern auch körperlich. [Steuer (1992)]
Presence ist von normalen Displays nicht zu erreichen, jedoch von Head-Mounted Displays wie VR-Brillen schon. Als Beweis hierfür kann man zum Beispiel Aufnahmen von Menschen nehmen, die mit einer VR-Anwendung eine Achterbahn fahren. Während der Betrachter still auf einem Stuhl sitzt, bewegt er seinen Körper entsprechend zu den Kurven der Bahn, die er sieht und der Körper empfindet echten Stress und Angst. Ebenfalls steigt der Puls oder der Betrachter bekommt schweißige Hände, dieses Erlebnis, als wäre das, das gesehen wird real, ist Presence.[1]

## 2.2 Wiedergabe von Virtual Reality im Web

Damit WebVR für viele Menschen zur Verfügung steht und sich damit ein Standard durchsetzen kann, müssen möglichst viele Browser die API auch unterstützen. Ebenfalls muss die API fähig sein, mit VR-Brillen zu kommunizieren, um die Fähigkeiten von Virtual Reality voll auszunutzen. Einige Brillen wie die HTC Vive oder die Gear VR von Samsung sind zusätzlich mit Controllern verbunden, welche Interaktionen innerhalb der virtuellen Welt ermöglichen sollen.

### 2.2.1 Mozilla

Als Vorreiter der Technologie hat Mozilla in seinem Desktop-Browser seit Firefox 55 WebVR nativ auf Windows unterstützt. Die Android-Version des Browsers bietet allerdings keine Unterstützung von VR, ebenso ist es für iOS Geräte wie dem iPhone. Für MacOS exisitert eine Nightly-Version, womit WebVR auch auf Mac Geräten funktioniert. VR-Headsets, die von Firefox 55 und den Nightly-Versionen unterstützt werden sind die HTC Vive und die Oculus Rift. [Dan Callahan]
Mozillas Embedded Browser Servo unterstützt die HTC Vive für Windows. [mozilla (17.08.2017)]

[1] Beispiel für ein solches Video https://youtu.be/GoQ0OXJCbaE

### 2.2.2 Google

Googles Open-Source Browser Chromium unterstützt WebVR in experimentellen Versionen auf Windows und die HTC Vive und die Oculus Rift können genutzt werden.
Für das Android-Smartphone stellt Google ebenfalls Chromium als auch die Chrome Canary App zur Verfügung, welche beide WebVR unterstützen. Beide Versionen sind mit den Brillen Google Daydream und Google Cardboard nutzbar.
Für beide Browser gilt, dass sie derzeit als instabil und experimentell deklariert sind, und in den Einstellungen müssen manuell bestimmte Flags aktiviert werden, um die Nutzung von WebVR zu ermöglichen. Für iOS Geräte ist kein Google-Browser vorhanden, der WebVR unterstützt. [mozilla (17.08.2017)]

### 2.2.3 Microsoft

Microsoft unterstützt mit Edge für Windows 10 WebVR, allerdings nur für ihr eigenes Mixed Reality Headset. Außerdem wird das Windows 10 Fall Creators Update benötigt. [Microsoft]

### 2.2.4 Andere

Apple hat sich im Juli 2017 sich der Community Group angeschlossen [Schwan (06.07.2017)] jedoch gab es bisher keine Ankündigung, ob und wann der Browser Safari die WebVR API integriert. Dennoch kann man mit Mozilla Firefox auf Mac-Geräten WebVR erleben, wenn man eine entsprechende Brille besitzt. Für iOS-Geräte gibt es keine offizielle Unterstützung von Googles Daydream und Cardboard Brillen.

Samsung unterstützt mit dem Browser Samsung Internet für Android Google Cardboard und ihre eigene Brille Gear VR. Auch hier muss ein Flag in den Einstellungen aktiviert werden, bevor WebVR nutzbar ist. [samsung (17.08.2017)]
Oculus Browser Carmel hat eine Developer Preview Version, welche WebVR für die GearVR nutzbar macht. [mozilla (17.08.2017)]
Andere Browser, welche keine integrierte Unterstützung für WebVR besitzen, müssen auf sogenannte Polyfills zurückgreifen. Dies ist eine Art von Workaround oder 'Hack', um WebVR zu ermöglichen. [mozilla]

## 2.3 Softwareentwicklung

WebVR ist eine JavaScript API und nutzt WebGL als Basis für alle Berechnungen. Während bereits verschiedene Frameworks wie 'AFrame' oder 'ReactVR' existieren, welche Arbeit abnehmen, ist es wichtig, den Grundablauf einer VR-Session zu verstehen. Eine Session ist nach Spezifikation in 6 Schritte gegliedert, diese werden hier erläutert. Ebenfalls das Wiedergeben des VR Inhalts auf nicht-VR Bildschirmen wird behandelt.

### 2.3.1 Funktionsprinzip

Im Folgenden wird der Lebenszyklus einer VR-Anwendung beschrieben, wie sie im API-Explainer definiert ist. [Brandon Jones u. a.]

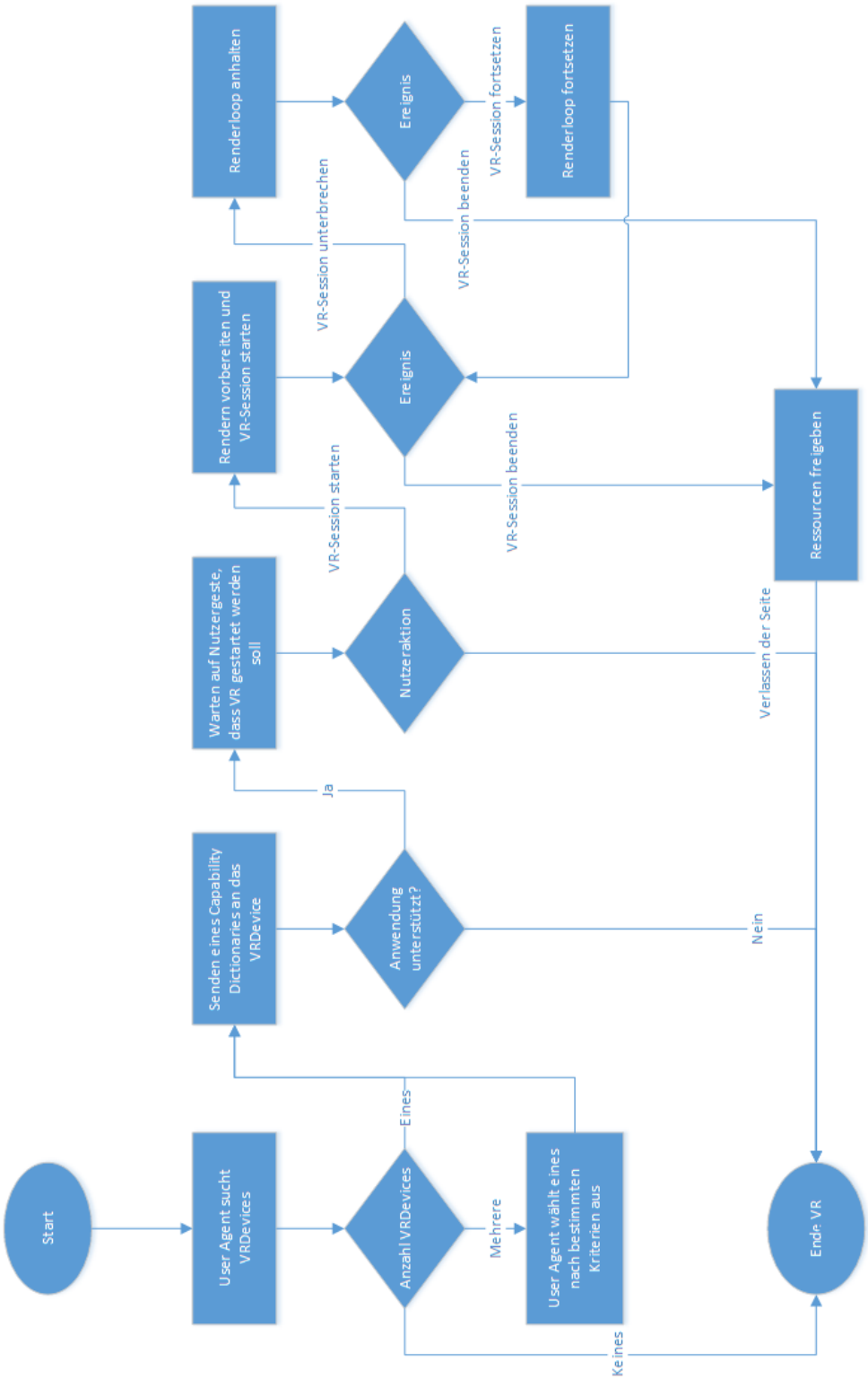


Abbildung 2.1: Ablauf einer VR-Session

### Ein VRDevice finden

Der User Agent (UA), welcher in der Regel der Webbrowser ist, fragt nach einem VRDevice. Auf Desktop-Clients ist dies meist ein VR Headset. Allerdings wird auch ein Mobilgerät, beispielsweise in Form eines Smartphones, das den Browser ausführt, als VRDevice gezählt. Als VRDevice kann auch ein Gerät zählen, welches keine stereoskopische Präsentation besitzt, aber erweiterte tracking-Eigenschaften besitzt, beispielsweise ARCore und ARKit kompatible Geräte. Mehr hierzu unter Abschnitt 2.3.1.
Für den Fall, dass mehrere VRDevices vorhanden sind, entscheidet der UA, welches zur Präsentation genutzt wird. Dieser kann frei Kriterien festlegen welches Gerät gewählt wird, so könnte er per User Interface eine Prioritätenliste in den Einstellungen zur Verfügung stellen. Allerdings darf die Anfrage nach einem VRDevice keine Geräteauswahl-UI öffnen, da Seiten ohne Benutzeraktivierung keine VR-spezifische Dialoge zeigen sollen.

An Desktop-Geräten besteht die Möglichkeit, externe Hardware wie ein VR Headset jederzeit an- und abzustecken. In diesem Fall wird ein 'devicechange' Event ausgelöst, womit die Seite die Geräteverfügbarkeit anpassen kann.

### VR-Kompatibilität feststellen

Wenn ein VRDevice gefunden wurde, wird dieses mit einem 'supportsSession' Aufruf angefragt, ob es die VR Anwendung unterstützt. Hierfür wird mit dem Aufruf ein 'Capability Dictionary' mitgesendet, welches eine Auflistung der Anforderungen der Anwendung ist. Wenn das Gerät diese alle erfüllt, bestätigt sie den Aufruf, andernfalls wird sie abgelehnt. Der Vorteil dieses Verfahrens ist, dass die Anwendung nicht selbst starten und Sensoren ansteuern, oder die Präsentation beginnen muss.
Wenn das VRDevice den Aufruf bestätigt, kann dem User ein 'Enter VR'-Button zur Verfügung gestellt werden ähnlich wie 'Vollbild' bei Videowiedergabe.

### VRSession starten

Wenn der Benutzer VR aktivieren möchte und dies mit einer Geste, wie dem Klicken eines 'Enter-VR' Buttons bestätigt, wird von dem VRDevice eine VRSession angefragt. In diesem Aufruf ist erneut das Capability Dictionary enthalten. Da es schon im supportsSession Aufruf enthalten war, kann angenommen werden, dass es erneut bestätigt wird.
Pro User Agent und VRDevice darf nur eine aktive VRSession exisitieren. Wird die Session gestartet, müssen einige Vorbereitungen für das Rendern getroffen werden. Zum einen muss

ein Koordinatensystem definiert werden, welches für das Tracking verwendet wird. In diesem sogenannten Frame of Reference wird ebenfalls bestimmt, welche Bewegungen in VR möglich sind. So soll für eine 360° Fotogalerie möglicherweise nur rotieren, nicken und gieren möglich sein (3-DOF).

#### Renderloop starten

Sobald die Vorbereitungen abgeschlossen sind, kann begonnen werden, Frames zu rendern. Hierfür müssen Frames von der VRSession angefordert werden, und in den Framebuffer des User Agents geschrieben werden.
Für den Fall, dass das Tracking fehlschlägt, muss die Webseite entscheiden, wie damit umgegangen wird. Beispielsweise könnte die Szene schwarz ausblenden, um Desorientierung zu vermeiden, oder der Frame of Reference wird geändert.

#### Unterbrechen der Session

Nun werden Frames dargestellt, bis der Benutzer die Session beenden möchte. Der User Agent kann allerdings jederzeit die Session temporär anhalten, wobei angenommen werden muss, dass diese wieder fortgesetzt wird. Als Beispiel könnte angenommen werden, dass das weitere Darstellen von Frames ein Sicherheitsrisiko darstellt oder die Privatssphäre verletzen kann, wie bei einer Passworteingabe. Diese Unterbrechung sowie die Fortsetzung geschieht per Event. Auf dem VRDevice wird dann nur das letzte Frame dargestellt, die Renderloop läuft jedoch weiter.

#### VRSession beenden

Soll die VRSession beendet werden, geschieht dies entweder durch das Aufrufen einer end()-Methode oder der UA beendet die Session. Grund hierfür kann sein, dass das VRDevice abgesteckt wird, oder eine andere Anwendung exklusiven Zugriff benötigt.
Sobald eine Session beendet wurde, ist sie nicht mehr nutzbar und kann auch nicht mehr wiederhergestellt werden.

#### Andere Darstellungsmöglichkeiten

Die WebVR API unterscheidet bei der Darstellung von VR Inhalten in zwei Kategorien, Exclusive und Non-Exclusive Access. Exclusive Access stellt VR Inhalte stereoskopisch, direkt im VRDevice dar, so wie eine 'normale', nicht-Browser VR Anwendung. Dieser Modus bietet

die größte Immersion und ermöglicht das erfahren von Presence. Um Exclusicve Access zu aktivieren, wird eine explizite Bestätigung des Nutzers benötigt, z.B. in Form von klicken eines Buttons.
Non-Exclusive Access bezeichnet die Darstellung von VR Inhalten von nur einem Auge, also wie eine normale 3D Anwendung, allerdings gibt es dennoch Zugriff auf Trackingfunktionen. Bei dieser Darstellung wird zwischen 'Mirroring' und 'Magic Windows' unterschieden. Das exakte Spiegeln des Bildes, welches auf einem VR Headset angezeigt wird, auf einem Standardmonitor ist Mirroring.
Magic Window ist das darstellen einer Szene, wie es in jeder anderen 3D-Anwendung der Fall wäre, ohne stereoskopische Präsentation. Es ist besonders interessant für mobile Geräte, für die keine VR Darstellung zur Verfügung steht, aber dennoch Trackingfunktionen genutzt werden sollen. Dies ist könnte sein, weil keine VR-Brille für das Gerät vorhanden ist. Ein Beispiel wäre das Betrachten von Panoramas. [Brandon Jones u. a.]

Abbildung 2.2: Beispiel für ein Magic Window am PC (Bildquelle: Screenshot von https://cecropia.github.io/thehallaframe, Cecropia Solutions, zuletzt abgerufen am 17.03.18)

### 2.3.2 Eingabemethoden

Für die Steuerung in einer WebVR Anwendung stehen einige Möglichkeiten zur Verfügung. Als erstes die Bedienung mittels Positionstracking, welches allerdings nur den VR-Headsets mit entsprechenden Sensoren vorbehalten ist, mit der Ausnahme von Smartphones, welche ARKit und ARCore unterstützen. Die Steuerung durch Tastatur und Maus ist in der WebVR API enthalten. Für Controller ist eine Controller API vorhanden, welche das Drücken von Buttons und die Achsen der Analog-Sticks abfragt und nutzbar macht. Für komplexere Controller, wie die von der HTC Vive oder der Oculus Rift existiert für Firefox eine experimentelle API Extension, welche auch die Position und Orientation dieser aufnehmen kann. [Mills u. a. (02.08.2017)]

## 2.4 Related Work

Das Thema WebVR erregt immer mehr Interesse in der Öffentlichkeit. Sowohl an Universitäten, als auch bei Unternehmen und Privatpersonen steigt die Beschäftigung mit dieser Technologie. Zum Beispiel hat Mozilla 2016 einen Blog eingerichtet, welcher seit dem wöchentlich Highlights präsentiert, was mit dem Framework entwickelt wird. [mozilla (28.01.2018)]
Vorwiegend liegt das Interesse an VR in Unterhaltungs- und Sozialtechnologien, wie Spiele oder virtuelle Treffräume, z.B. 'Facebook Spaces' oder 'VR Chat', welche in diesem Kapitel vorgestellt werden.

### 2.4.1 Dateiübertragung

Da VR hochqualitative Bildinformationen benötigt, wie hochauflösende Texturen und gute Modelle, werden Szenen schnell groß im Bezug auf Dateigrößen. Die Bandbreite der Internetanbindung zur Übertragung von solchen Szenen ist daher am schnellsten der Flaschenhals, um angenehme VR Erlebnisse zu gewähren. Als Lösung hierfür könnte man zum Einen bessere, kleinere Dateiformate entwickeln, die schneller übertragen werden können.
Eine andere Möglichkeit sind sogenannte Low-Poly Modelle, welche gezielt so modelliert werden, dass sie eine geringe Anzahl von Polygonen, Beziehungsweise Dreiecken besitzen. Dies zieht jedoch ein distinktives, eher 'unrealistisches' Aussehen nach sich, welches gegebenenfalls unerwünscht ist.
Die letzte Möglichkeit ist, dass man neue Downloadmethoden entwickelt. Ein Team der Tongji University in China hat ein Konzept mit dem Namen SMLAOI entwickelt, welches für 'Scalable Multi-Layer Area of Interest' steht, und beschreibt das priorisierte Laden von Informationen, welche näher am aktuellen Viewpoint des Nutzers sind. [Wang und Jia (2009)]

### 2.4.2 Social

Das Web zeichnet sich gerade in den letzten Jahren durch Social Media und Social Platforms, wie Facebook, Twitter, Instagram u.A. aus, in denen Menschen sich anderen über das Internet mitteilen können. Außerdem existieren seit Anfang des Internets Chaträume, in denen Menschen Kontakt zueinander haben können, ohne physisch beieinander zu sein. Daher ist es naheliegend, dass gerade VR- Erlebnisse, welche Aktivitäten in Gruppen ermöglichen, relativ schnell ihren Weg auf den Markt finden werden. Seit Februar 2017 existiert die externe Anwendung 'VRChat', in denen Besitzer einer VR Brille einen virtuellen Chatraum betreten können und sich via Sprachchat unterhalten können.[VRChat Inc]

Etwas später, im April hat Facebook eine App mit dem Namen 'Facebook Spaces' veröffentlicht, welche Facebook Usern ermöglicht, sich online mit der HTC Vive oder Occulus Rift zu treffen und zu unterhalten.[Franklin (2017)]
Alternativ existiert auch die Anwendung 'AltspaceVR', welches das selbe Prinzip wie Facebook Spaces besitzt, jedoch auch diverse Spiele und einen eigenen Browser integriert hat.
Sowohl VRChat als auch Facebook Spaces und AltspaceVR sind jedoch keine WebVR Anwendungen, welche die Technologie WebVR nutzen. Stattdessen sind es eigenständige Anwendungen, welche heruntergeladen werden müssen. Allerdings ist es sehr naheliegend, dass ebenfalls solche VR-Erlebnisse über WebVR sich in den Browser finden werden. Es existiert sogar schon eine Erweiterung für das WebVR Framework A-Frame mit dem Namen 'networked-aframe', welches bereits Multi-User VR-Erfahrungen ermöglicht.

### 2.4.3 Virtual Museum

#### Semantic Web und Virtuelle Museen

Die Idee eines virtuellen Museums ist keine neue. In 'Semiotik der Kultur Bd. 6' sind mehrere Arbeiten enthalten, welche sich mit den Themen Semantic Web und virtuelles Museum beschäftigen. Semantic Web ist ein Konzept, welches das Web erweitern soll. So soll eine Maschine aufgrund von gegebenen Informationen zu einem Thema weiterführende Informationen zu diesem Thema finden [Berners-Lee u. a. (2001)]. Als Beispiel würde eine Aussage in einem Text wie 'Hamburg feiert im Mai Hafengeburtstag' dazu führen, dass zu der Information 'Hamburg' die Information 'Hafengeburtstag' sowie 'Mai' verknüpft werden. Die Idee hierbei ist, dass ein Mensch, der mit der Maschine interagiert, so mit ihr umgehen soll, als hätte die Maschine das Wissen selbst verstanden.
Nun ist es naheliegend, dass so eine Art von Informationsverknüpfung für ein virtuelles Museum sehr nützlich ist. In 'An Information Model for Virtual Museums - The Case of an Eco-Museum' wird ein Modell erarbeitet, wie Informationen repräsentiert und gespeichert werden können, um sie im Semantic Web nutzbar machen zu können.[Robering (2008)]
Ein anderes Projekt ist ein deutsch-dänisches virtuelles Museum, welches allerdings kein 'begehbares' 3D Museum ist, wie es in dieser Arbeit ist, sondern eine einfache Webseite. Idee des Projekts ist aber, Informationen in unterschiedlichen Formaten anzubieten, je nachdem, wer der Besucher ist. Die Webseite kategorisiert in 5 verschiedene Gruppen: Touristen, Schüler/Studenten, Lehrer, Wissenschaftler und Kinder. Je nachdem, wer die Seite besucht, werden unterschiedliche Informationen angeboten oder diese anders dargestellt. So bekommt ein Besucher, welcher sich als Tourist anmeldet, eher Informationen zu Orten und deren Sehens-

würdigkeiten, während Wissenschaftler eher längere Texte und Bibliographien angeboten bekommen. Für Besucher, welche sich in keine der Kategorien einordnen, existiert eine sechste Kategorie 'Andere'. Diese wählt eine Präsentation, welche auf die Allgemeinheit zugeschnitten ist, sprich, eine erwachsene Person mit eher geringem Wissen über die Geschichte der Region. Das Wissen zu diesem Projekt ist auf Basis einer Ontologie eingeteilt. Ontologie in diesem Kontext bedeutet, dass Informationen in einer Art so gegliedert sind, dass ein Vokabular entsteht, welches von einem Computer verarbeitet werden kann. Denkbar ist dies z.B. mit Hilfe vom Metadaten. Diese Informationen können dann in einem Graph dargestellt werden, welcher ein Computer nutzt, um Verknüpfungen zu erstellen. Das Projekt hat hierfür eine Topic Map genutzt, um Informationen zu speichern. Eine Topic Map ist ein abstraktes Modell, welches Themen, sogenannte 'Topics' mit Assoziationen verbindet. Topics besitzen zudem noch sogenannte 'Occurrences', welches Informationen zu diesem Topic selbst beinhaltet. Dies ist beispielsweise ein Verweis auf einen wissenschaftlichen Text. Dieses Museum ist mittlerweile leider nicht mehr einsehbar, jedoch wurde parallel dazu das Projekt 'The Virtual Natural Historical Museum' *virtnatmus* gestartet, welches ebenfalls auf Basis einer Ontologie Wissen anbietet. Dieses Projekt, sowie virtnatmus sollen in Zukunft zusammengefasst werden. [Robering (2008)]

#### Cecropia Solutions

Das Software-Unternehmen Cecropia Solutions hat ein Projekt mit dem Namen 'The Hall', welches ein Three.js VR-Museum ist, erarbeitet. Allerdings liegt bei diesem Projekt weniger der Aspekt auf Wissensübermittlung, stattdessen war es rein zum Testen der Möglichkeiten von WebVR. Später wurde dieses Projekt mit dem Framework A-Frame neu geschrieben, wo auch mehr Möglichkeiten der Steuerung hinzugefügt wurden. Zuerst war nur ein Teleportieren zwischen festen Punkten innerhalb des Raumes möglich war. Als Dateiformat für Modelle haben sie .ctm gewählt, welches nicht nativ von A-Frame geladen werden kann, weshalb extra ein Dateiloader entwickelt wurde. [Crecropia Solutions (2017)]

### 2.4.4 VR Browser

#### JanusVR

JanusVR ist ein spezieller Browser, welcher mit einer eigenen Sprache, JML, läuft. Dieser Browser stellt nicht wie ein Standardbrowser Webseiten dar, sondern eine Webseite ist ein dreidimensionaler Raum, welcher durch VR erlebbar ist. Dieser Raum wird 'Webspace' genannt. Zunächst war JanusVR nur ein eigenständiger Browser, allerdings existiert seit Januar 2016

JanusVRWeb, ein WebVR Client, welcher in jedem WebVR fähigen Browser läuft und so die Funktionalitäten von JanusVR anbietet.[Bozorgzadeh (2018)]
Ein großer Vorteil von JanusVR ist, dass er Multi-User fähig ist, solange man es in den Einstellungen aktiviert hat. So kann man auf verschiedenen Webspaces andere Nutzer treffen und mit ihnen per Text- oder Sprachchat kommunizieren und zusammen den Webspace erleben.

### Supermedium

Supermedium ist ein 'Full VR Browser' von den Entwicklern von WebVR und A-Frame. Supermedium stellt, bisher, nur WebVR Webseiten dar. Die Idee dahinter ist, das Browsen zwischen Seiten zu erleichtern, die eigenständige WebVR Erlebnisse sind. Derzeit kann man nur Seiten besuchen, welche von den Entwicklern ausgestellt und angeboten werden, aber traditionelle URL-Eingabe soll noch hinzugefügt werden.[Ngo (2018b)]


Abbildung 2.3: Screenshot vom Supermedium Browser (Quelle: Supermedium Webseite, zuletzt abgerufen 17.03.18)

# 3 Konzept

Da nun die Grundlagen über die Möglichkeiten der WebVR Technologie bekannt sind, wird im folgenden Kapitel das Konzept für das digitale Museum vorgestellt.
Ein mögliches Hindernis für das Erleben von VR-Anwendungen ist die 'VR-Krankheit', welche bei einigen Menschen auftritt, während oder nachdem sie ein VR-Erlebnis hatten. Die Symptome sind ähnlich der Reisekrankheit und reichen von einem Unwohlsein bis zu Übelkeit [Kolasinski (1995)]. Obwohl es keine Bestätigungen dafür gibt, soll eine niedrige Latenz, eine hohe Auflösung des Displays sowie eine hohe Bildwiederholrate der Krankheit gegen wirken. Während hardwarespezifische Eigenschaften nur schwer beeinflusst werden können, muss sichergestellt sein, dass die Anwendung von der Softwareseite diese Eigenschaften erreicht. Sprich, sie soll in höchster Auflösung und mit hoher Framerate laufen.

## 3.1 Mögliche Features eines virtuellen Museums

Die Idee für ein VR-Museum ist, dass zum einen viele Ausstellungsstücke in echten Museen hinter Absperrungen oder Glaskästen verwahrt werden. Deshalb kann man oft bestimmte Ausstellungsstücke nicht genauer ansehen, geschweige denn sie berühren. Die Fähigkeit, Objekte im VR zu 'berühren' und sich dadurch besser anzuschauen ist ein großer Vorteil gegenüber einem normalen Museum. Zum anderen ist so ein Virtuelles Museum für ein Museum selbst eine interessante Idee. Es könnte Teile ihres eigenen Inventars digitalisieren und im Netz ausstellen, um Besucher in das echte Museum zu ziehen. Beispielsweise wäre auch denkbar, dass verschiedene Museen ihre Ausstellungsräume untereinander verbinden, welches einen Besucher dazu verleiten kann, auch das andere Museum zu besuchen.
Ein weiterer Vorteil von einem digitalen Museum ist es, dass leicht mehr Informationen zu den Objekten geliefert werden können, als es es in der Realität möglich ist. So könnte beispielsweise zu einem Skelett eines Dinosauriers die Funktion gegeben sein, welche aus dem Skelett eine Replikation des Sauriers macht. Oder zu einem Ausstellungsstück könnten Videos dargestellt werden, oder auf Literatur verwiesen werden, welche genutzt werden kann, um sich zu bilden.

Natürlich muss das Museum nicht nur seine eigenen Räume nachstellen. Durch die Quasi-Unendlichkeit von virtuellen Welten, kann in der Theorie jeder Ort der Erde, oder auch anderen Planeten, nachgestellt werden. So könnte man durch ein virtuelles Museum die Ruinen der Akropolis in Athen oder die Chinesische Mauer besuchen, selbst ein Spaziergang auf dem Mars oder dem Mond ist denkbar.

## 3.2 Anforderungen an eine VR Anwendung

Im Folgenden werden die umzusetzenden Punkte für den Prototypen, in Form von Anforderungspunkten, beschrieben.

### 3.2.1 Funktionale Anforderungen

#### Ausstellen von ausgewählten Modellen

Ausgewählte 3D Modelle sollen in einem Raum virtuell zum Betrachten ausgestellt werden. Die Modelle sollen größtmögliche Qualität haben. Qualität lässt sich zwar visuell schwer messen, da Schönheit subjektiv ist, allerdings lässt sich aus den Anzahl der Verticies (oder Knoten) ein gewisser Grad an Realitätstreue ableiten.
Kleine Modelle, bis 0,5x0,5x0,5 Meter Größe in der echten Welt, sollen mindestens 500 Verticies besitzen. Mittelgroße Modelle bis 2x2x2 Meter mindestens 10.000 Verticies besitzen und große Modelle ab 2x2x2 Meter mindestens 100.000. Der Raum soll mindestens 5 Modelle unterstützen, wovon mindestens eines mittelgroß und mindestens eins groß sein soll.

#### Steuerung über Maus & Tastatur und Controller

Die Navigation im Museum soll ausschließlich über Tastatur und Maus bzw. vorhandener Controller des VR-Headsets erfolgen.

#### Interaktion mit Ausstellungsstücken

Solange die Größe des Ausstellungsstücks es erlaubt, soll es möglich sein, das Modell mit Hilfe der Eingabemethoden in die Hand zu nehmen, um es besser betrachten zu können.

#### Informationen zu Ausstellungsstücken

Zu den Ausstellungsstücken sollen relevante Informationen wie Alter, Herkunft, Künstler oder Ähnliches mit Hilfe von Texteinblendungen, Ton-Overlay oder weiteren Grafiken angezeigt werden.

#### Hyperlinks zu (VR-) Seiten

Hyperlinks im Museum sollen auf andere Webseiten weiterleiten können, welche selbst auch VR-Webseiten sein können. Im Fall, dass die verlinkte Seite keine VR-Anwendung ist, wird normal der Browser wie auf dem Desktop-Bildschrim dargestellt. Testwebseite für die Webseite soll https://aframe.io/examples/showcase/museum/ sein.

#### Firefox und Chrome

Als Vorreiter der WebVR Technologie beschränkt sich die Nutzbarkeit der Anwendung in vollem Umfang auf Google Chrome und Mozilla Firefox.

#### Anwendung auf PC und Smartphone

Da der Prototyp in einem Browser läuft und daher plattformunabhängig ist, soll er auch auf Smartphones einsetzbar sein, die reibungslose Benutzung soll allerdings nur auf Android-Geräten festgelegt sein.

#### Framerate

Um ein angenehmes Erlebnis zu gewährleisten, muss die Framerate sehr hoch sein. Hierbei sollen konstant 60 Frames pro Sekunde (FPS) erreicht werden, gerade um der VR-Krankheit vorzubeugen.

### 3.2.2 Nicht Funktionale Anforderungen

#### Ladezeit in 10 Sekunden

Der Benutzer soll weniger als 10 Sekunden brauchen, bis das vollständige Museum geladen ist. Dies bezieht sich ausschließlich auf das Laden von dem Museum und nicht das Wechseln zu anderen Seiten.

#### Intuitive Navigation

Die Navigation und Steuerung soll simpel gehalten sein. Dies lässt sich am besten durch eine Nutzerstudie messen, in der mehrere Personen die Anwendung testen und die Intuitivität mit Hilfe eines Fragebogens einschätzen.

#### Latenz zwischen Aktion und Effekt

Die Interaktionen, wie das in die Hand nehmen von Modellen, oder die Wiedergabe von Informationen zu Ausstellungsstücken, sollen möglichst schnell erfolgen, damit der Komfort der Bedienung gewährleistet wird. Der Ablauf soll weniger als 5 Frames betragen.

#### Änderbarkeit der Modelle/Räume

Das Umgestalten des virtuellen Museums soll nicht zu umständlich sein, damit es in Zukunft verschiedene Erfahrungen bieten kann. Hierfür soll ein 3D-Editor verfügbar sein, in denen Modelle bewegt und modifiziert werden können. Die Grundfunktionen sollen sein, dass man Modelle durch den Raum bewegen kann, sowie neue Modelle hinzugefügt werden können.

### 3.2.3 Dateitypen

Um Objekte in 3D darzustellen, gibt es viele Möglichkeiten. WebVR beziehungsweise WebGL unterstützt hier mehrere Dateitypen mit verschiedenen Vor- und Nachteilen.
Zum einen gibt es OBJ-Dateien, welche weit verbreitet sind und für kleine Objekten problemlos nutzbar sind. Zusätzlich zu einer OBJ-Datei wird eine MTL-Datei benötigt, in der die Materialeigenschaften des Objektes beschrieben sind, also Informationen wie die Textur oder Reflexionen am Modell.

Collada DAE Dateien beinhalten sowohl Modellinformationen sowie Materialeigenschaften und können optional auch physikalische Eigenschaften enthalten, wie Animation, Friktion eines Modells oder Lichtquellen.

Die oben genannten Dateitypen haben das Problem, dass sie an sich sehr groß sind und lange Ladezeiten haben. Der Grund hierfür ist, dass diese Dateien als Textdateien formatiert sind. OBJ-Dateien sind einfache ASCII Textdateien [Murray und VanRyper (1996)] und Collada-Dateien sind XML-Basiert [Barnes und Finch (2008)]. Der Nachteil besteht hier, dass diese erst noch geparsed, also 'übersetzt' werden müssen, bevor der Computer damit arbeiten kann.
Die Khronos Group hat hierfür das Format glTF (GL Transmission Format) entwickelt, welches

im Grunde die selben Eigenschaften wie DAE-Dateien besitzt, aber schneller geladen werden kann aufgrund der kompakten Dateigröße und Dateistruktur. Ein Modell wird durch eine .gtlf-Datei, eine .bin-Datei und Texturdateien beschrieben. Eine Alternative ist eine .glb-Datei, welche die .bin und die .gltf in einer einzelnen Binärdatei enthält. Die .gltf-Datei ist wie eine JSON Datei aufgebaut und beinhaltet die Knotenhierarchie des Modells, Materialinformationen und Kameras. Die .bin enthält die geometrischen Informationen und Animationsinformationen. Zwar muss die .gltf-Datei erst in den Arbeitsspeicher geladen werden, jedoch kann die .bin direkt in den GPU Speicher geladen werden, ohne sie parsen zu müssen. Dies verkürzt die Ladezeit gegenüber OBJ- und Collada-Dateien.[Bhatia u. a. (2017)]

Aufgrund der Tatsache, dass glTF direkt für OpenGL und der schnellen Übertragung entwickelt wurde, liegt es auf der Hand, dass dieses Dateiformat sich am Besten für eine WebVR Anwendung eignet.

# 4 Implementation

In diesem Kapitel wird die Umsetzung des Prototyps beschrieben, welcher anhand des vorangegangenen Konzepts entwickelt wurde. Zuerst wird auf die Wahl der genutzten Software eingegangen, bevor Schritt für Schritt die Implementierung beschrieben wird.

## 4.1 Genutzte Software

### 4.1.1 A-Frame

A-Frame von Mozilla ist ein Open-Source Entity-Component Framework, welches auf HTML aufsetzt. Es ist schnell in eine Webseite eingebunden und die Sprache, welche HTML selbst sehr ähnelt, ist schnell erlernt. Des weiteren hat A-Frame eine große Community, welche für das Framework bereits viele Erweiterungen hervorgebracht hat. Ebenfalls besitzt es eine relativ gut ausgebaute Dokumentation.
Die Basis für den Prototypen ist die 'A-Frame Boilerplate', welche eine einfache Szene und eine package.JSON enthält. Die Boilerplate wird von den Entwicklern von A-Frame selbst über GitHub bereitgestellt. Eine der Dependencies in dieser package-Datei ist ein Node-Server, welcher die Webseite lokal laufen lässt, hierdurch ist schnelles Prototyping möglich.

### 4.1.2 Git

Um Sourcecode zu sichern und zwischen verschiedenen Geräten zu übertragen, wurde Git benutzt. Das Repository, welches den Code sowie die Assets, wie die Modelle enthält, liegt auf GitHub. Um die Anwendung testen zu können, als läge sie auf einem Server, wurde GitHub Pages verwendet. GitHub Pages ist ein einfacher Hosting-Service von GitHub, welcher zwar keinen Serverseitigen Code, wie php ausführt, aber HTML darstellt, was für diese Zwecke vollkommen ausreichend ist.

### 4.1.3 NPM

Ehemals 'Node Paket Manager', jetzt nur noch npm genannt, ist ein JavaScript Paketmanager, welcher über die Kommandozeile bedient wird. A-Frame sowie viele Erweiterungen sind über npm bereitgestellt.

### 4.1.4 Sketchfab

Sketchfab ist eine Webseite, welche 3D Modelle hostet. Seit 2016 werden alle angebotenen Modelle, welche heruntergeladen werden können, sowohl in ihrem Originalformat als auch als GLTF-Datei zur Verfügung gestellt. Die Modelle werden hierbei automatisch durch den Anbieter konvertiert.

### 4.1.5 Blender

Viele Modelle, die zu finden sind, haben eine sehr hohe Anzahl an Verticies und Dreiecken, um möglichst detailliert zu sein. Jedoch wirkt dies sich ggf. nachteilig auf die Performance der Anwendung aus. Die 3D-Grafiksoftware Blender wurde hier genutzt, um die Modelle zu 'vereinfachen', sodass sie leichter zu Rendern sind. Ebenfalls wurde eine Erweiterung hinzugefügt, welche es ermöglicht, ein Modell als GLTF zu exportieren. Diese Erweiterung wurde von der Khronos-Group entwickelt.

## 4.2 Ausstellungsstücke

### 4.2.1 Wahl der Modelle

Nach den Anforderungen aus Abschnitt 3.2.1, soll die Anwendung mindestens 5 Ausstellungsstücke besitzen. Mindestens eines davon soll nach der dortigen Definition Mittelgroß und eines Groß sein. Um die Anwendung realistisch zu halten, werden Modelle gewählt, die auch in einem echten Museum ausgestellt werden könnten. Die genutzten Modelle, inklusive Quelle, werden im Folgenden genannt.

'Fiat C.R 42 'Falco' - Italian Biplane Fighter' von Dan Hardo [https://sketchfab.com/hzgamestudio], Lizenziert unter CC BY 4.0, URL: https://sketchfab.com/models/1817d370a2c24f2d8e39b97b9a368a5c

'David by Michelangelo' von jerryfisher [https://sketchfab.com/jerryfisher], Lizenziert unter CC BY 4.0, URL: https://sketchfab.com/models/8f4827cf36964a17b90bad11f48298ac

'Marcus Aurelius' von Geoffrey Marchal [https://sketchfab.com/geoffreymarchal], Lizenziert unter CC BY 4.0, URL: https://sketchfab.com/models/03d7639ecbe943bba20b22ba1f9746d3

'Bust of Emperor Commodus, Getty Villa' von arck-project [https://sketchfab.com/arck-project], Lizenziert unter CC BY 4.0, URL: https://sketchfab.com/models/cc91ffb8e4264047a21a36c1e1af01d7

'Roman helmet' von Choges Craft Channel [https://sketchfab.com/sergish1], Lizenziert unter CC BY 4.0, URL: https://sketchfab.com/models/97c323fd65ad44b59a4dcc88cba6dbb1

'Spotlight' von Ozhogi [https://sketchfab.com/ozhogi], Lizenziert unter CC BY 4.0, URL: https://sketchfab.com/models/84f9a5ee411342b0b0e9fc94e84a9f29

Es wird empfohlen, Modelle zu wählen, die Schatten und Lichter 'eingebacken' haben. Dies bedeutet, dass sie ein Teil der Textur sind und nicht auf die in A-Frame eingebaute Beleuchtung zurückgegriffen werden muss. Denn dies bedeutet zusätzlichen Rechenaufwand und kann zu Rucklern führen.
Da hier jedoch auf vorhandene Modelle zurückgegriffen wird, ist dies bei unseren Modellen leider nicht der Fall. Jedoch sollte ein Team für ein Projekt, in dem die Modelle selber erstellt werden können, dies hier beachten. Beispiel für ein Modell mit 'baked lighting' ist

Abbildung 4.2. Modellquelle: 'Roma Pincio Statua - Pincian Hill Statue' von spogna [https://sketchfab.com/spogna], Lizenziert nach CC BY-NC 4.0, URL: https://sketchfab.com/models/a868b24b116840c6

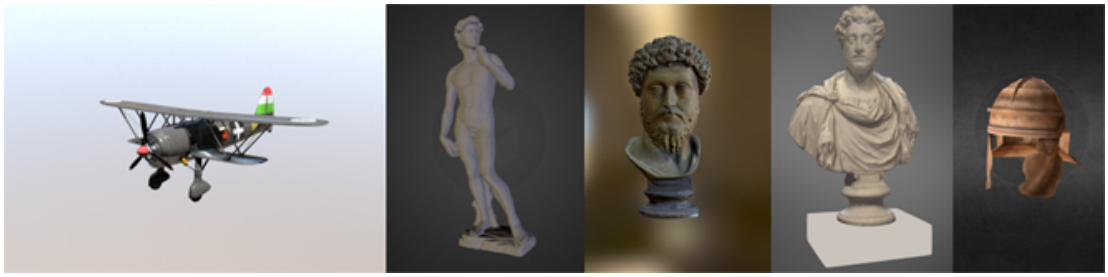

Abbildung 4.1: Die ausgewählten Modelle (Quelle: Screenshots von Sketchfab.com)

Abbildung 4.2: Modell mit 'baked lighting' (Bildquelle: Screenshot aus frühen Phasen des Prototypen)

### 4.2.2 Simplifizierung der Modelle

Um die Dateigröße der Modelle etwas zu verkleinern und die Framerate hoch zu halten, muss die Anzahl der Knoten, Kanten und Dreiecke in einem Modell verkleinert werden. So sind die Modelle für den WebGL Renderer leichter darzustellen. Um dies zu erreichen, kann beispielsweise der Garland-Heckbert-Algorithmus angewendet werden. Hier wird zunächst

für jede Kante in einem Netz eine Fehlerquadrik berechnet. Diese beschreibt, wie sehr eine Fläche durch das zusammenfassen beider Knoten dieser Kante verändert wird. Entsprechend werden nach und nach die Kanten mit den kleinsten Fehlerquadriken entfernt, bis die vorher angegebene Knotenanzahl erreicht ist. [Garland und Heckbert (1997)]
Für den Prototypen wurden die Modelle in ihrer Originalform in Blender importiert und mit dem Modifikator 'Decimate' reduziert.
Mit diesem Verfahren wurden alle Modelle mit über 200.000 Verticies weitestgehend reduziert, ohne, dass sie optisch zu sehr an Qualität verlieren.

Abbildung 4.3: Beispiel vom Simplifizieren. Links vor, Rechts nach dem Anwenden eines 0,2 Decimate Modifiers in Blender. Eine Reduzierung von 200.000 Verticies auf 40.000

## 4.3 Bau der Szene

### 4.3.1 Grundaufbau des Ausstellungsraumes

Da nun die Modelle bereit sind, ist es nun daran, die Szene aufzubauen. Dies ist durch A-Frame sehr einfach gehalten, da das Framework primitive Objekte mitbringt. So ist Boden und Decke des Raumes eine sogenannte 'a-plane' und die Wände des Raumes bestehen aus 4 Boxen. Beiden Objekten wurden dann noch Texturen hinzugefügt. Dem Raum wurde dann noch ein Ambient-Licht hinzugefügt.

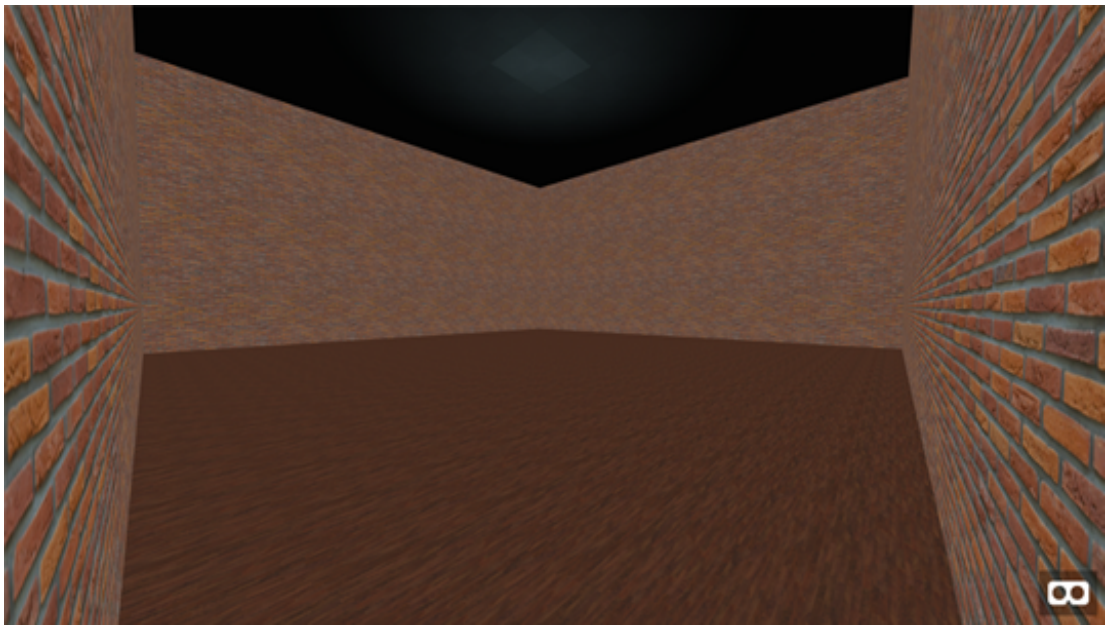

Abbildung 4.4: Grundszene

### 4.3.2 Hinzufügen der Modelle

A-Frame besitzt einen Asset-Manager, welches ein System ist, um alle Assets, wie Texturen, Modelle, Sounds oder Videos vorzuladen, bevor die Szene aufgebaut wird. Dies stellt sicher, dass keine Assets geladen werden, während gerendert wird und sie somit visuell fehlen. Durch diesen Asset Manager werden alle 5 Modelle als GLTF-Format geladen und in der Szene entsprechend positioniert. Kleinere Modelle, wie der Helm oder die Büste von Marc Aurel haben noch eine texturierte Box als Podest bekommen. Alle genutzten Texturen sind bezogen von https://www.textures.com. Die Nutzung ist nach https://www.textures.com/terms-of-use.html zu nicht kommerziellen Zwecken gestattet.

### 4.3.3 Beleuchtung und Schatten

Um die Darstellung visuell etwas ansprechender zu gestalten, wurden an die Decke Modelle von Scheinwerfern 'montiert'. Diese haben jeweils zusätzlich noch ein Spotlight-Licht hinzugefügt bekommen, welches auf das jeweilige Ausstellungsstück gerichtet wird. Da A-Frame ein Entity-Component System ist, hat jedes Objekt die Möglichkeit Eigenschaften im Sinne von Komponenten zu erhalten. Auf diesem Weg kann einem Modell die Komponente 'shadow' hinzugefügt werden. Diese sorgt dafür, dass das Modell Schatten wirft, solange es von einer Lichtquelle beschienen wird, die Schatten werfen kann.

Abbildung 4.5: Der Fertig ausgestattete Raum mit Lichtern und Schatten

### 4.3.4 Informationen zu Ausstellungsstücken

Um ein paar Informationen zu den Modellen anzuzeigen, wurden einfache Texteinblendungen neben die Modelle gehängt. Diese Texteinblendungen sind ebenfalls ein A-Frame Primitivtyp.

## 4.4 Steuerung in der Virtuellen Welt

In diesem Abschnitt wird auf die Navigation und Bewegung in der virtuellen Welt eingegangen.

### 4.4.1 Steuerung ohne VR-Headset

Wenn das Museum am PC ohne VR Headset geöffnet wird, ist die Steuerung mit den Pfeiltasten sowie W/A/S/D automatisch aktiviert. Mit der Maus wird die Kamera geschwenkt. An dem Kamera-Objekt wird ein Cursor angebracht, welcher fest in der Mitte des Bildschirms sitzt. Mit Hilfe dieses Cursors werden Portale aktiviert und Objekte aufgenommen. Das Aufheben von Objekten ist umgesetzt mit Hilfe einer Erweiterung, einer sogenannten Komponente, welche mit JavaScript die entsprechende Funktion hinzufügt.

### 4.4.2 Steuerung mit VR-Headset

A-Frame hat die Kommunikation zwischen Browser und VR-Headset schon übernommen, weshalb die Navigation durch Bewegen durch den Raum schon enthalten ist. Alternativ wurde eine Komponente hinzugefügt, welches die Controller nutzt, um sich teleportieren zu können. Für das Aufheben und Bewegen von Objekten wird die selbe Komponente genutzt, die auch für die Steuerung ohne VR-Headset genutzt wurde.

### 4.4.3 Darstellen von Informationen

Das A-Frame Framework hat nativ die Funktion enthalten, Text darzustellen. Dieser erscheint dann als Objekt in der Szene und kann wie jedes andere Objekt behandelt werden. Diese Funktion wird hier genutzt um zu jedem Austellungsstück ein Informationstext anzuzeigen. Alternativ wäre es möglich Videos und Audiodateien als Assets zu laden und in der Szene abzuspielen. Da jedoch keine solche Dateien für die hier ausgestellten Modelle zur Verfügung standen, wurde darauf verzichtet.

## 4.5 Hyperlinks

Mit A-Frame 0.6.0 wurden Links eingeführt, welche wie a-href tags bei HTML eingesetzt werden können. Das Framework generiert dann ein 'Portal', welches klickbar ist und eine Seitenweiterleitung einleitet. Man kann dem Link ein 360° Bild hinzufügen, welches hinter dem Portal angezeigt wird.


Abbildung 4.6: Portal zu dem in Abschnitt 3.2.1 angegebenen Link, mit einem Screenshot von https://cecropia.github.io/thehallaframe/ als Vorschau.

# 5 Evaluation

Nachdem die Anwendung nach besten Möglichkeiten umgesetzt wurde, wird in diesem Kapitel nun darauf eingegangen, inwieweit die gestellten Anforderungen in Kapitel 3 erfüllt wurden. Daraufhin wird auf die Probleme und Schwierigkeiten eingegangen, welche bei der Entwicklung dieser Arbeit aufgetreten sind.

## 5.1 Überprüfen der Anforderungen

### 5.1.1 Funktionale Anforderungen

#### Ausstellen von ausgewählten Modellen

Es war gefordert, dass die im Prototypen dargestellten Modelle je nach Größe in der Realität eine gewisse Menge an Verticies besitzen sollen. So sollten kleine Modelle, bis ca 0,5x0,5x0,5 Meter Größe in der Realität sollen mindestens 500 Verticies besitzen. Mittelgroße Modelle bis 2x2x2 Meter mindestens 10.000 Verticies besitzen und große Modelle ab 2x2x2 Meter mindestens 100.000.
Hier waren kleine und mittelgroße Modelle kein Problem in der Darstellung, allerdings machten große und komplexe Modelle, mit vielen Verticies Probleme. Sobald der Anwendung ein Cursor hinzugefügt wurde, gab es Einbrüche in der Framerate, sowie ein starkes 'Umherrutschen' der Betrachterkamera, was unangenehm ist. Dies passierte allerdings nur, wenn der Cursor ein großes Objekt, oder eines mit vielen Verticies traf oder auch nur in die unmittelbare Nähe des Modells kam. Dies ist möglicherweise dadurch, dass der Cursor so implementiert ist, dass er einen Strahl in die Szene wirft, welcher mehrmals pro Sekunde prüft, ob er ein Objekt schneidet [Ngo (2018a)]. Durch die hohe Anzahl an Dreiecken, die die Modelle besitzen, ergeben sich hierdurch viele Berechnungen, was dieses Problem verursacht.
Um dieses Problem zu beheben, wurde die Anzahl der Verticies der Modelle verringert, welches das Problem löste, allerdings lag der 'sweet spot' dafür für jedes Modell zwischen etwa 40.000 bis 50.000 Verticies, was der Anforderung, dass große Modelle mindestens 100.000 Verticies besitzen sollen, widerspricht. Leider hat aus diesem Grund auch das Modell der Davidstatue darunter gelitten, welches weniger ansehnlich war, nachdem es simplifiziert wurde.

Alternativ müsste eine andere Art von Cursor implementiert werden, um dieses Problem zu lösen. Möglicherweise können für diesen Fall angefertigte, optimierte Modelle auch helfen. Jedoch war es für den Prototypen kein Problem 5 Modelle darzustellen, solange die Menge an Verticies klein gehalten werden kann, ist es sicher möglich, noch mehr Modelle einzufügen.

#### Steuerung über Maus & Tastatur und Controller

Die Steuerung im Sinne der Navigation hat sich alleinstehend als kein Problem herausgestellt, da diese schon im Framework A-Frame enthalten war. Sowohl mit Maus & Tastatur als auch die 3D-Navigation mit einem VR-Headset und Positionstracking. Alternativ wurde eine Teleportations-Komponente hinzugefügt, welche die Controller nutzt, um sich durch den Raum zu bewegen.

#### Interaktion mit Ausstellungsstücken

Interaktion mit Objekten ist nicht in Version 0.7 von A-Frame enthalten gewesen. Hierfür wurde ebenfalls eine Komponente hinzugefügt, welche entweder mit Hilfe des Cursors oder mit den Controllern es möglich macht, Objekte aufzuheben, solange sie dafür gekennzeichnet sind. Leider war es nicht möglich, diese Funktion korrekt einzubinden, solange die Teleportations-Komponente ebenfalls implementiert war. Grund ist, dass das Script, welches das Teleportieren ermöglicht, als Komponente der Kamera bzw. den Controllern hinzugefügt werden muss, während die Komponente, welche die Interaktion ermöglicht von sich aus eine Kamera generiert, was verhindert, dass die Teleport-Funktion hinzugefügt werden kann.

#### Informationen zu Ausstellungsstücken

A-Frame beinhaltet schon die Möglichkeit, Videos und Audiodateien abzuspielen, ebenfalls konnte sehr leicht Text eingeblendet werden. Dies kann dazu genutzt werden, Informationen wiederzugeben. Die Modelle für diesen Prototypen wurden eher zufällig gewählt, entsprechend konnten leider keine weiterführenden Informationen zu eingeblendet werden, mit der Ausnahme von ein paar Texten.

#### Hyperlinks zu (VR-) Seiten

Links in A-Frame sind seit Version 0.6 unterstützt. Durch einfaches Einfügen eines 'a-link' Tags, wie ein a-href in 'normalem' HTML wird dann ein klickbares Portal eingefügt. Dieses kann wie jedes andere Objekt in der Szene bewegt werden. Hierbei ist es egal, zu welcher Seite der Link führt.

#### Firefox und Chrome

Das Testen des Prototyps mit einem VR-Headset wurde im Creative Space for Technical Innovations (https://csti.haw-hamburg.de/) an der HAW Hamburg vollzogen. Genutzt wurde hier das VR-Headset HTC Vive und der Browser Mozilla Firefox. Die Nutzung von WebVR lief hier Problemlos, jedoch musste zum Testen in der Anwendung Steam erst SteamVR gestartet werden, bevor das Headset im Browser nutzbar war. Die Nutzung der VR-Funktion war im Chrome-Browser nicht möglich.
Die Anwendung wurde von mehreren Personen auf dem Smartphone getestet. Hier wurden unterschiedliche Browser getestet. Allerdings wurde nur die Magic Window Funktion genutzt, da leider keine der Personen eine VR-Brille besaß. Die Rückmeldungen waren zumeist positiv, jedoch gab es Berichte, dass es Ruckler gab, sobald große Modelle angeguckt wurden. In einem Fall gab es einen Absturz des Browsers, als ruckartig direkt nach oben geschaut wurde.

#### Framerate

Die angepeilten durchgängigen 60 Frames pro Sekunde wurden zum Teil erreicht, sowohl beim Testen mit dem VR-Headset als auch im Magic-Window Modus in Google Chrome. Dies war nur dann der Fall, als nichts außer der Anwendung lief, sprich, nur die Seite des Prototyps war geöffnet und geladen. Während die Seite lädt, gibt es immer wieder Einbrüche in der Framerate, sobald jedoch alles geladen ist, wird die 60 FPS-Marke erreicht. Ohne ersichtlichen Grund bricht jedoch die Framerate zwischenzeitlich auf etwa 40 FPS herunter.

### 5.1.2 Nicht Funktionale Anforderungen

#### Ladezeit in 10 Sekunden

Die Ladezeit der Seite ist von mehreren Faktoren abhängig. Zum einen, von wo sie aufgerufen wird, da je nach Standort die Internetanbindung besser oder schlechter sein kann. Zum anderen speichern heutige Browser bestimmte Information zwischen, wodurch die Ladezeit nach dem ersten Aufruf verkürzt wird.
Zehn Sekunden wurden allerdings zumeist verfehlt, vor allem, wenn die Seite das erste mal geöffnet wird, kann die Ladezeit sehr lang sein. Gemessen wurde dies mit Hilfe der Laufzeitanalyse-Funktion in Firefox. Es wurde so vorgegangen, dass die Seite geöffnet wurde, während die Messung lief, bis die Szene vollständig geladen wurde und alle Modelle der Szene auf dem Bildschirm dargestellt sind. Daraufhin wurde eine neue Messung gestartet und die Seite neu geladen. Da der Browser die Möglichkeit hatte, Seiteninformationen zwischenzuspeichern, ist die Annahme, dass der Ladevorgang schneller sein sollte. Danach wurde der Cache des

Browsers geleert, der Browser neu gestartet und eine neue Messung begonnen.
Aus diesen Messungen ergab sich, dass, wenn der Prototyp lokal auf einem Server läuft, das erstmalige Laden im Schnitt Acht Sekunden dauert. Beim Neuladen etwa Sechs Sekunden. Wenn der Prototyp extern läuft, kann die Ladezeit beachtlich ansteigen. Im Schnitt lag es beim Erstaufruf der Seite zwischen 20 und 30 Sekunden, ein Neuladen der Seite hat dies auf etwa Zehn Sekunden reduziert. Tabelle 5.1 zeigt Ladezeiten des Museums basierend auf den Standort des Servers. Beispielhafte Aufnahmen sind in Abbildung 5.1 zu sehen.
Tabelle 5.2 sind Messungen einer anderen Szene, nicht des Museums, mit unterschiedlicher Menge an Modellen. Hier wurden die Modelle des Museums genommen und dupliziert. Der Server für diese Szene lief Lokal, da das Testen auf einem gehosteten Server nur zusätzlich die Zeit messen würde, welche durch das Downloaden entstehen würde. Zusätzlich ist anzumerken, dass keine 30 verschiedenen Modelle genutzt wurden. Das bedeutet, dass keine 30 verschiedene Objekte im Speicher gelagert waren, sondern nur die 5 Originale, welche insgesamt 30 mal dargestellt wurden. Jedoch ändert dies nichts daran, dass knapp 1,2 Millionen Verticies gerendert werden mussten.
Die Messungen zeigen, dass eine hohe Anzahl an Modellen die Ladezeit beeinträchtigt, auch wenn der Browser die Möglichkeit hat, ein Objekt mehrmals darzustellen. Selbst bei den 30 Modellen war keine visuelle Beeinträchtigung zu bemerken. Ausnahme ist jedoch wieder der Cursor, welcher einen Framerate-Einbruch verursacht hat, sobald er auf Modelle gerichtet wurde. Dies war schon bei einer Anzahl von 10 Modellen zu merken.

| Server | Ladetyp | Zeit durchschnitt |
|---|---|---|
| Lokal | Erstaufruf | 9 Sekunden |
| Lokal | Neuladen | 6 Sekunden |
| Gehostet | Erstaufruf | 25 Sekunden |
| Gehostet | Neuladen | 10 Sekunden |

Tabelle 5.1: Ladezeiten des Museums entsprechend des Ladetyps und Standort des Servers

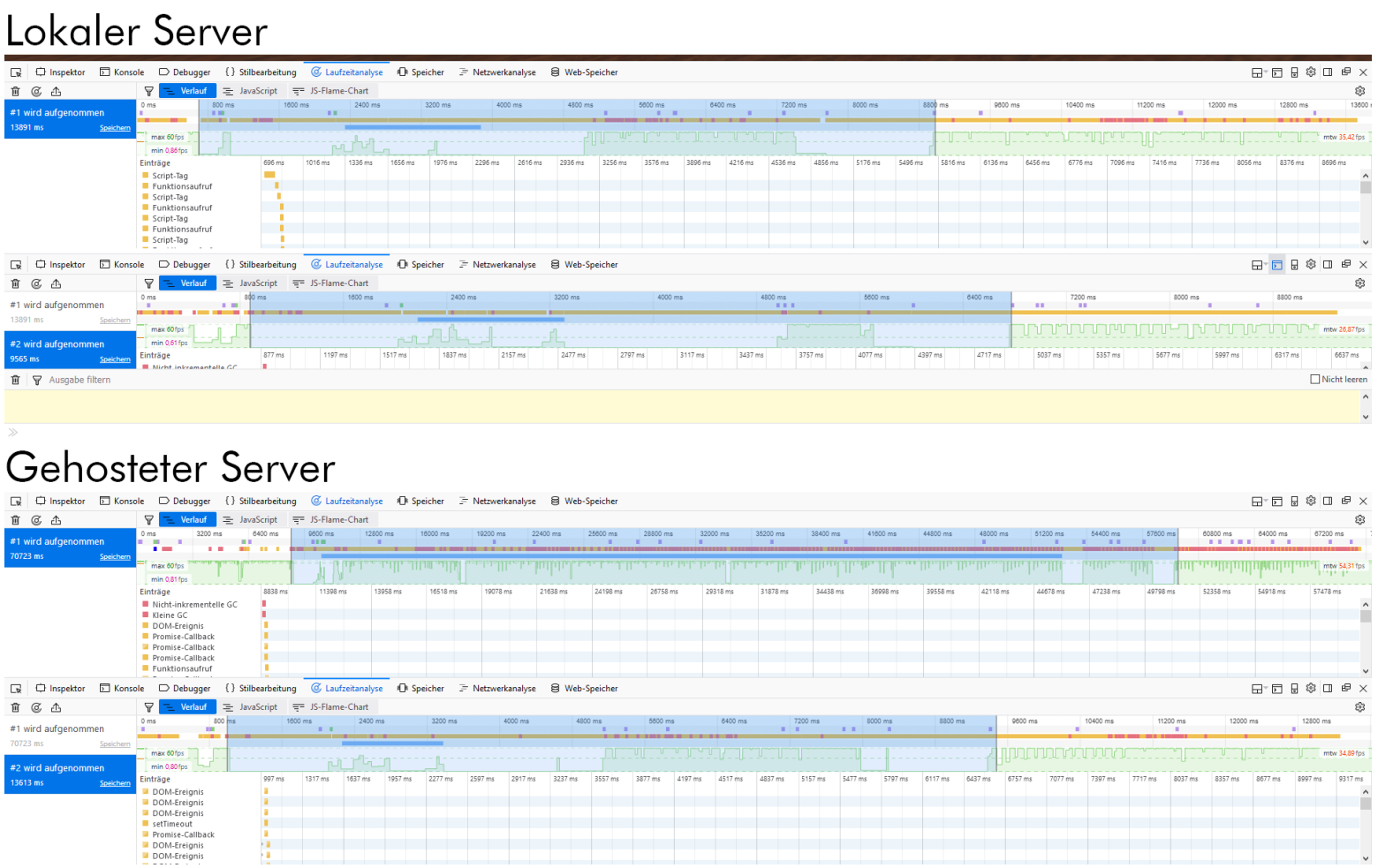


Abbildung 5.1: Aufnahmen vom Laden des Prototyps. Oben, wenn die Webesite lokal läuft und unten, wenn über das Internet darauf zugegriffen wird. Die Ladezeit selbst ist in Blau markiert

| Anzahl der Modelle | Gesamtverticies | Ladezeit Erstaufruf | Ladezeit Neuladen |
|---|---|---|---|
| 1 | 40.500 | 1 Sekunde | 1 Sekunde |
| 2 | 82.000 | 2,3 Sekunden | 2 Sekunden |
| 3 | 128.300 | 3,2 Sekunden | 3,2 Sekunden |
| 4 | 155.700 | 3,9 Sekunden | 3,3 Sekunden |
| 5 | 197.300 | 5,1 Sekunden | 4,2 Sekunden |
| 10 | 405.300 | 6,1 Sekunden | 4,7 Sekunden |
| 30 | 1.193.900 | 9,6 Sekunden | 6,5 Sekunden |

Tabelle 5.2: Ladezeiten einer anderen Szene mit bestimmter Anzahl von Modellen

### Intuitive Navigation

Leider konnte keine Nutzerstudie durchgeführt werden, welche testen kann, wie intuitiv die Bedienung ist. Grund hierfür ist die relative Unausgereiftheit des Prototyps und Zeitmangels. Jedoch lässt sich sagen, dass für Besitzer eines VR-Headsets die 3D Navigation einfach zu verstehen ist. Für die Steuerung der Teleportation sowie die Möglichkeit des Greifens von Objekten wäre es jedoch hilfreich, wenn entsprechende Tooltips eingeblendet werden würden. Für die Steuerung ohne VR-Headset lässt sich sagen, dass die Nutzung der Pfeiltasten oder von W/A/S/D durch andere 3D-Anwendungen wie Computerspiele relativ standardisiert ist. Deshalb kann diese Option der Steuerung als intuitiv gewertet werden.

### Latenz zwischen Aktion und Effekt

Zwischen dem Anklicken und dem Aufheben eines Objekts sollten weniger als 5 Frames vergehen, um eine flüssige Interaktion zu ermöglichen, die konkrete Dauer war leider nicht zu messen, jedoch konnte durch manuelles Testen festgestellt werden, dass das Aufheben augenblicklich erfolgte, sowohl beim ersten Mal, sowie beim wiederholten aufheben.

### Änderbarkeit der Modelle/Räume

Jede A-Frame Szene lässt sich durch einen sogenannten Inspektor mit der Tastenkombination Strg-Alt-I öffnen, welcher sich ebenfalls wie ein 3D-Szeneneditor nutzen lässt. Hier kann jedes Objekt bewegt, rotiert und skaliert werden. Ebenfalls können hier neue Entitäten hinzugefügt werden, was es auch möglich macht, neue Modelle der Szene hinzuzufügen. Die Szene aktualisiert sich dann nicht sofort, stattdessen muss sie als HTML-Datei neu heruntergeladen, und dem Server hinzugefügt werden, damit die Änderungen angewendet werden. Dies ist etwas kompliziert, jedoch ist es leichter, als in den HTML-Dokumenten selbst zu arbeiten.

## 5.2 Probleme und Schwierigkeiten

Die größten Probleme in der Bearbeitung dieses Projekts waren zum einen die Arbeit mit den zusätzlichen Komponenten bzw. Erweiterungen, und zum anderen das Finden von passenden Modellen und deren Einbindung in den Prototypen.

### 5.2.1 Probleme mit Komponenten

Da WebVR und A-Frame noch relativ jung sind, war zu erwarten, dass gewisse Kinderkrankheiten bestehen. Dies war gerade bei den Komponenten zum Aufheben von Objekten und dem Teleportieren Auffällig. Diese sind zwar schon etwas dokumentiert und es gibt auch Beispiele, welche getestet werden können. Jedoch laufen selbst diese Beispiele zum Teil nicht reibungslos. So kann es sein, dass man beim Teleportieren beispielsweise nicht auf der Position landet, welche eigentlich angezeigt wurde.
Außerdem werden diese Beispiele auch 'isoliert' demonstriert, das heißt, sie stellen die Funktionalität einzeln dar. Da es so gut wie keine Überschneidungen in den Anwendungen der Komponenten gibt, kommt es auch selten zu Kompatibilitätsproblemen, die sonst vielleicht zu erwarten gewesen wären. Dies ist eingetreten bei der Implementierung der Teleportation und dem Aufheben, wie bereits unter Abschnitt 5.1.1 beschrieben.

### 5.2.2 Probleme mit Modellen

Die Modelle haben sich als zweites Problemgebiet enttarnt. Zum einen mussten passende Modelle gefunden werden, welche die Anforderungen erfüllen. Hier gibt es ein paar Quellen im Internet, bei denen man Modelle findet, welche unter den Creative Commons veröffentlicht werden. Jedoch müssen diese auch im passenden Format vorhanden sein. Das automatische Konvertieren von Modellen von der Webseite Sketchfab war hier zwar hilfreich, jedoch ist sie auch nicht perfekt. So kam es vor, dass die Modelle nicht korrekt konvertiert wurden, und deshalb fehlerhaft dargestellt wurden, siehe Abbildung 5.2.

Außerdem bestand das Problem mit den Framerate-Einbrüchen beim Betrachten von komplexen Modellen, wie bereits unter Abschnitt 5.1.1 beschrieben, welches jedoch mehr oder weniger durch das Simplifizieren der Modelle gelöst werden konnte. Allerdings bedeutet dies, dass sehr komplexe Modelle nicht gut dargestellt werden können, wenn gleichzeitig der vorhandene Cursor genutzt wird.

Abbildung 5.2: Beispiel für ein Fehlerhaft konvertiertes Modell. Links die Darstellung auf Sketchfab, rechts die von Sketchfab konvertierte Version im Prototypen

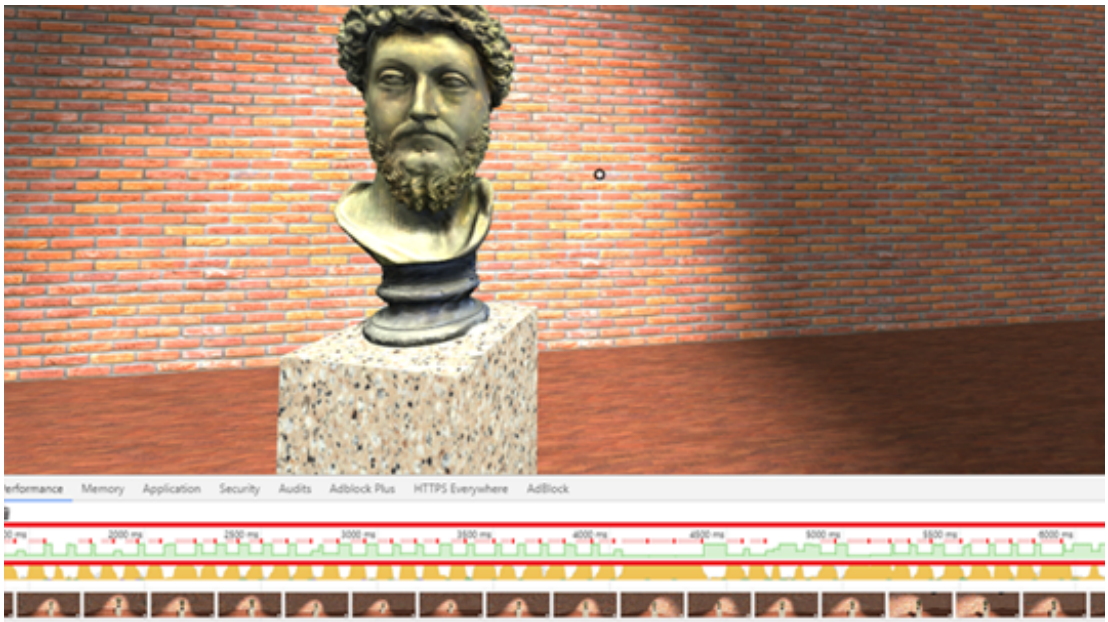


Abbildung 5.3: Ruckeln der Anwendung, sobald ein komplexes Modell angesehen wurde. Framerate grafisch im roten Kasten hervorgehoben

Ein weiteres Problem, welches allerdings nichts mit WebVR zu tun hat, ist, dass für manche Modelle die Texturdateien so groß waren, dass sie nicht auf GitHub hochgeladen werden konnten. Deshalb ist die Büste von Marcus Aurelius auf dem Server nicht texturiert. Dies bedeutet allerdings auch, dass weniger Daten übertragen werden müssen, was die Ladezeit verkürzen kann.

# 6 Ausblick

## 6.1 Weiterentwicklung des virtuellen Museums

Um den Prototypen weiterzuentwickeln, lassen sich noch einige Aspekte verbessern und ausbauen. Zum einen muss das Problem zwischen dem Teleportieren und dem Aufheben von Objekten gelöst werden. Denkbar wäre hier, das Teleportieren ganz aus dem Projekt herauszunehmen, da die Komponente für das Aufheben ebenfalls eine Möglichkeit der Bewegung durch eine Szene anbietet, bei der man die Controller nutzt und sich 'zieht'.
Ebenfalls muss eine Unterscheidung implementiert werden, welche erkennt, welche Darstellungsmethode verwendet wird. Denn für eine VR-Brille muss die Kamera auf Position 0.0 der Y-Achse starten, damit die Darstellung korrekt ist. Das dies eher problematisch für die Darstellung auf einem normalen Monitor oder Smartphone ist, ist naheliegend, da keine Möglichkeit besteht, sich nach oben oder unten zu bewegen, wodurch die Kamera 'im Boden' hängt.
Ebenfalls wäre es sicherlich von Nutzen, wenn bessere Modelle zur Verfügung stünden, welche extra für diese Anwendung angefertigt werden. Der Vorteil ist, dass diese dann optimiert werden können, dass eine bessere Balance zwischen Qualität und Dateigröße erreicht werden kann. Jedoch reichte hierfür mein Wissen in der 3D Modellierung nicht aus, um solche Optimierungen anzuwenden.
Außerdem wären tiefgreifende Informationen, sowie weitere Medien für das Museum von Vorteil, wenn es einen bildenden Zweck erfüllen soll. A-Frame besitzt bereits die Möglichkeit, Videos und Audiodateien wiederzugeben, jedoch müssen diese zuerst aufgetrieben werden, was mir leider nicht möglich war. Eine Zusammenarbeit mit einem Museum wäre hier perfekt, um diese Informationen bereitstellen zu können.
Zusätzlich wäre es gut, wenn Tooltips eingeblendet würden, welche dem Besucher der Seite die Steuerung je nach Gerät erklären, um die volle Nutzbarkeit zu gewährleisten.
Zuletzt ist ein großer Vorteil an WebVR, dass es ebenfalls leicht auf dem Smartphone genutzt werden kann. Da die Geräte jedoch nicht so leistungsfähig sind wie ein PC, muss hier eventuell darauf zurückgegriffen werden, dass unterschiedliche Versionen der Anwendung in Form von mehreren HTML-Dokumenten existieren. Sobald die Seite dann besucht wird, wird das

Gerät erkannt und man wird entsprechend weitergeleitet. Zudem muss für Smartphones auch noch eine Form der Steuerung eingebunden werden, da diese generell keine 6-DOF besitzen. Aufgrund dessen ist eine Steuerung durch Positionstracking nicht machbar. Ein Einblenden von einem Steuerkreuz ist ebenfalls nicht möglich, da die Anwendung dafür ausgelegt ist, dass das Smartphone in einer Brille eingesetzt wird. Dies zieht nach sich, dass das Display nicht mehr zu berühren ist. Eine mögliche Form der Bewegung wäre es, mit Hilfe von Markern auf dem Boden sich Teleportieren zu können, wenn diese eine gewisse Zeit angesehen werden.

## 6.2 Zukunft von WebVR

Der Zweck dieser Bachelorarbeit war es, den aktuellen Stand von WebVR zu erfahren, sowie die Möglichkeiten zu erkunden, welche man mit dieser Technologie hat. Während der Prototyp nur einen kleinen Ausschnitt der Möglichkeiten aufzeigt, die man hat, war das gesamte Projekt jedoch ein guter Einblick in den aktuellen Zustand. Das Entwickeln, gerade mit A-Frame ist vom Prinzip her einfach, das Aufsetzen ging leicht von der Hand und man bekommt einfache Szenen sehr schnell eingerichtet. Ebenfalls von Vorteil ist es, dass A-Frame auf HTML sitzt, und es ein leichtes ist, mittels JavaScript Erweiterungen für WebVR zu schreiben. Es existieren deshalb auch schon viele Komponenten, welche einfach mittels 'script-tag' eingebunden und genutzt werden können. Allerdings sind sowohl WebVR als auch A-Frame noch sehr jung, weshalb damit gerechnet werden muss, dass nicht alles problemlos funktioniert.
Vom Standpunkt der Performance aus schlägt sich WebVR sehr gut, auch wenn die Ladezeiten beachtlich sein können. Jedoch liegt dies aber hauptsächlich an der Internetanbindung und der Größe von genutzten Assets, wie Modellen oder Audiodateien. Sobald aber eine Szene geladen ist, funktioniert die Darstellung problemlos. Eine Erweiterung für A-Frame ist Beispielsweise eine Physik-Komponente, welche Kollisionen ermitteln und Gravitation simulieren kann. Auch diese war zu Testzwecken eine Zeit lang in dem Prototypen implementiert, aber wurde später wieder entfernt, weil sie nicht notwendig war. Die Performance war jedoch nicht beeinflusst.
Viele Anwendungen, die derzeit mit WebVR entwickelt werden, dienen hauptsächlich der Unterhaltung, wie kleine Spiele oder 'Galerien', in denen Objekte präsentiert werden. Oft sind es auch einfach nur Anwendungen, welche einfach als Experimente mit der Technologie dienen. Google selbst hat hierfür eigens eine Webseite eingerichtet. 'Richtige' Anwendungen, welche die vollen Möglichkeiten von WebVR ausschöpfen existieren leider gar nicht.
Jedoch steckt in WebVR nach meiner Meinung viel Potential. Die Tatsache, dass der Zugang zu einem VR-Erlebnis darin besteht, dass man nur einen Link öffnet, spricht sehr dafür, dass diese Möglichkeit auch genutzt wird. Vor allem die Tatsache, dass Smartphones WebVR nutzen

können bringt eine große Nutzerreichweite für solche VR-Anwendungen.
Allerdings müssen mehr Browser diese Technologie unterstützen, mit Hilfe eines Polyfills haben zwar andere Browser theoretisch schon die Möglichkeit, die VR-Inhalte darzustellen, aber dies funktioniert nicht überall und native Unterstützung ist in jedem Fall besser. Außerdem fehlt es an konkreten Anwendungen, welche über die Funktion einer Präsentation von WebVR hinausgehen, wobei dies jedoch nur eine Frage der Zeit ist.

# Literaturverzeichnis


[Adams 2004] Adams, Ernest: *Postmodernism and the Three Types of Immersion*. 2004. – URL https://www.gamasutra.com/view/feature/130531/the_designers_notebook_.php. – Zugriffsdatum: 07.03.2018

[Apple Inc 2018] Apple Inc: *ARKit - Apple Developer*. 2018. – URL https://developer.apple.com/arkit/. – Zugriffsdatum: 18.03.2018

[Barnes und Finch 2008] Barnes, Mark ; Finch, Ellen L.: COLLADA – Digital Asset Schema Release 1.5.0 Specification. (2008). – URL https://www.khronos.org/files/collada_spec_1_5.pdf. – Zugriffsdatum: 17.03.2018

[Berners-Lee u. a. 2001] Berners-Lee, Tim ; Hendler, James ; Lassila, Ora: *The Semantic Web*. 2001. – URL https://www.scientificamerican.com/article.cfm

[Bhatia u. a. 2017] Bhatia, Saurabh ; Cozzi, Patrick ; Knyazev, Alexey ; Parisi, Tony: *GLTF 2.0 Specification*. 2017. – URL https://github.com/KhronosGroup/glTF/blob/master/specification/2.0/README.md. – Zugriffsdatum: 17.03.2018

[Bounds 2014] Bounds, Tucker: *Facebook to Acquire Oculus*. 2014. – URL https://newsroom.fb.com/news/2014/03/facebook-to-acquire-oculus/. – Zugriffsdatum: 17.03.2018

[Bozorgzadeh 2018] Bozorgzadeh, Amir: *The state of WebVR heading into 2018*. 2018. – URL https://venturebeat.com/2018/01/17/the-state-of-webvr-heading-into-2018/. – Zugriffsdatum: 05.03.2018

[Brandon Jones u. a. ] Brandon Jones ; NellWaliczek ; Tom Ritter ; Yan, Shaobo ; Ningxin Hu ; Newton Calegari ; C. Van Wiemeersch: *w3c/webvr*. – URL https://github.com/w3c/webvr/blob/master/explainer.md. – Zugriffsdatum: 24.10.2017

[Crecropia Solutions 2017] Crecropia Solutions: *Crecropia Hall AFrame*. 2017. – URL https://github.com/Cecropia/thehallaframe. – Zugriffsdatum: 11.03.2018

[Dan Callahan ] Dan Callahan: *Firefox 55: first desktop browser to support WebVR*. – URL https://hacks.mozilla.org/2017/08/firefox-55-supports-webvr/. – Zugriffsdatum: 23.10.2017

[Elliott 2017] Elliott, Matt: *These iPhones and iPads work with ARKit: iOS 11 is set to usher in a flood of AR apps*. 2017. – URL https://www.cnet.com/how-to/these-iphones-and-ipads-work-with-arkit/. – Zugriffsdatum: 18.03.2018

[Franklin 2017] Franklin, Rachel: *Facebook Spaces: A New Way To Connect With Friends In VR | Facebook Newsroom*. 2017. – URL https://newsroom.fb.com/news/2017/04/facebook-spaces/. – Zugriffsdatum: 19.03.2018

[Garland und Heckbert 1997] Garland, Michael ; Heckbert, Paul S.: Surface simplification using quadric error metrics. In: Owen, G. S. (Hrsg.): *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. New York, NY : ACM Press/Addison-Wesley Publishing Co, 1997, S. 209–216. – URL http://mgarland.org/files/papers/quadrics.pdf. – Zugriffsdatum: 17.03.2018. – ISBN 0897918967

[Google 23.02.2018] Google: *ARCore Overview | ARCore | Google Developers*. 23.02.2018. – URL https://developers.google.com/ar/discover/. – Zugriffsdatum: 17.03.2018

[Kolasinski 1995] Kolasinski, Eugenia M.: *Simulator Sickness in Virtual Enviroments*. 1995. – URL http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA295861. – Zugriffsdatum: 19.03.2018

[Microsoft ] Microsoft: *Immersive headset setup*. – URL https://developer.microsoft.com/en-us/windows/mixed-reality/immersive_headset_setup. – Zugriffsdatum: 24.10.2017

[Mills u. a. 02.08.2017] Mills, Chris ; Doyle, Erika ; Lowell, Dayton: *Gamepad API Documentary*. 02.08.2017. – URL https://developer.mozilla.org/en-US/docs/Web/API/Gamepad_API#Experimental_Gamepad_extensions. – Zugriffsdatum: 02.11.2017

[mozilla ] mozilla: *WebVR Polyfill*. – URL https://github.com/immersive-web/webvr-polyfill. – Zugriffsdatum: 05.03.2018

[mozilla 17.08.2017] MOZILLA: *WebVR Rocks*. 17.08.2017. – URL https://webvr.rocks/. – Zugriffsdatum: 19.10.2017

[mozilla 28.01.2018] MOZILLA: *This Week of A-Frame*. 28.01.2018. – URL https://aframe.io/blog/. – Zugriffsdatum: 30.01.2018

[Murray und VanRyper 1996] MURRAY, James D. ; VANRYPER, William: *Encyclopedia of graphics file formats: [the complete reference on CD-ROM with links to Internet resources; for PC, Macintosh and UNIX platforms]*. 2. ed. Bonn : O'Reilly, 1996. – ISBN 1565921615

[Ngo 2018a] NGO, Kevin: *A-Frame Cursor Documentation*. 2018. – URL https://github.com/aframevr/aframe/blob/master/docs/components/cursor.md. – Zugriffsdatum: 17.03.2018

[Ngo 2018b] NGO, Kevin: *Born to Be Web*. 2018. – URL https://www.supermedium.com/blog/launch. – Zugriffsdatum: 05.03.2018

[Robering 2008] ROBERING, Klaus (Hrsg.): *Semiotik der Kultur*. Bd. Bd. 6: *Information technology for the virtual museum: Museology and the semantic web*. Wien and Zürich and Berlin and Münster : Lit, 2008. – ISBN 9783037359358

[samsung 17.08.2017] SAMSUNG: *WebVR Rocks*. 17.08.2017. – URL https://webvr.rocks/samsung_internet. – Zugriffsdatum: 24.10.2017

[Schwan 06.07.2017] SCHWAN, Ben: *Apple macht bei WebVR mit*. 06.07.2017. – URL https://www.heise.de/mac-and-i/meldung/Apple-macht-bei-WebVR-mit-3765131.html. – Zugriffsdatum: 19.10.2017

[Steuer 1992] STEUER, Jonathan: Defining Virtual Reality: Dimensions Determining Telepresence. In: *Journal of Communication* 42 (1992), Nr. 4, S. 73–93. – URL https://faculty.washington.edu/farkas/TC510-Fall2011/SteuerMediaRichnessTheory.pdf. – ISSN 00219916

[Sutherland 1968] SUTHERLAND, Ivan: A head-mounted three dimensional display. In: *Proceedings of AFIPS* (1968), Nr. 68, S. 757–764. – URL https://www.cise.ufl.edu/research/lok/teaching/ve-s07/papers/sutherland-headmount.pdf. – Zugriffsdatum: 07.03.2018

[VRChat Inc ] VRCHAT INC: *VRChat - VRChat is now available on Steam*. – URL https://vrchat.com/blog-steam. – Zugriffsdatum: 19.03.2018

[Vukicevic u. a. 2017] Vukicevic, Vladimir ; Jones, Brandon ; Gilbert, Kearwood ; van Wiemeersch, Chris ; Waliczek, Nell ; Cintron, Rafael: *WebVR Editors Draft latest.* 2017. – URL https://w3c.github.io/webvr/spec/1.1/. – Zugriffsdatum: 24.10.2017

[w3c ] w3c: *Participants in the WebVR Community Group | WebVR Community Group.* – URL https://www.w3.org/community/webvr/participants. – Zugriffsdatum: 19.10.2017

[Wang und Jia 2009] Wang, Wei ; Jia, Jinyuan: An incremental SMLAOI algorithm for progressive downloading large scale WebVR scenes. In: Fellner, Dieter (Hrsg.): *Proceedings of the 14th International Conference on 3D Web Technology.* New York, NY : ACM, 2009, S. 55. – ISBN 9781605584324

*Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.*

Hamburg, 20. März 2018 Eric Salomon