Projektdokumentation

HAPTISCHES RENDERING VON OBERFLÄCHENATTRIBUTEN INNERHALB DER VIRTUELLEN REALITÄT

25. November 2019

Stefan Geier
Matrikelnummer: 11112826
Technische Hochschule Köln
Fakultät 10 - Medieninformatik
stefan.geier@smail.th-koeln.de

Inhaltsverzeichnis

1 Einleitung

Das im Folgenden beschriebene Praxisprojekt wurde im Rahmen des gleichnamigen Moduls des Bachelorstudienganges "Medieninformatik" an der TH Köln bearbeitet und dokumentiert. Ziel des Praxisprojekt-Moduls ist es, das von den Studierenden im Studium gelernte umzusetzen und zu vertiefen.

Vor dem Beginn des Praxisprojektes wurde sich bereits für die weite Domäne der Computergrafik entschieden. Um sich innerhalb dieses Teilgebietes der Informatik für ein spezifisches Thema zu entscheiden, wurden wissenschaftliche Mitarbeiter der TH Köln zu Rate gezogen. Nach mehreren persönlichen Gesprächen und eigener Recherche wurde sich schließlich für den Themenbereich der Virtuellen Realität (kurz: VR) entschieden.
Verschiedene, in der Branche relevante, Messen und Konferenzen erwiesen sich dabei als die ergebnisreichsten Quellen – darunter zeigte sich die weltweit größte Computergrafik-Konferenz *SIGGRAPH* [1] als besonders hilfreich. Die Variation der dort vorgestellten Fachgebiete bot gerade in dieser anfänglichen Recherchephase eine wertvolle Möglichkeit, sich aus dem bestehenden Themenpool für einen präziser erfassbaren Zweig der VR zu entscheiden. Aus eigener Interesse und dem sich bereits zu diesem Zeitpunkt abzeichnenden Potential umsetzbarer Möglichkeiten wurde sich daraufhin für die Simulation von Haptik (haptisches Rendering) innerhalb der VR entschieden. Um sich auch innerhalb der Haptik wiederum auf ein Teilgebiet festlegen zu können, wurden einige Projekte betrachtet, in denen Forschungsgruppen die verschiedenen Wahrnehmungsarten des Menschen adressierten. So beispielsweise die Simulation von G-Kräften [2], oder die Illusion der geometrischen Form eines Objektes durch die Verschiebung des Gewichtmittelpunktes [3]. Zum Ende der Recherche wurde sich jedoch für die Simulation einer Textur entschieden – inwiefern dieser Aspekt in dem Bereich des haptischen Renderings, bzw. der Haptik aufzufinden ist, wird im folgenden Punkt 1.1 beschrieben.

1.1 Erläuterung relevanter Fachbegriffe

Um ein klares Verständnis für die Fachbegriffe des haptischen Renderings zu schaffen, werden im Folgenden die relevanten Begriffe aufgeführt und in Bezug zueinander gesetzt. Ziel dieser Erklärung ist es, mögliche Verständnisungenauigkeiten auszubessern, deren Entstehung möglicherweise der Umgangssprache zu Schulden gekommen ist.

Haptik Der Begriff der "Haptik" bezeichnet die Lehre der haptischen Wahrnehmung. Mithilfe dieser ist der Mensch in der Lage, unter Verwendung von unterschiedlichen Erkennungsmustern (siehe Punkt 9.2), Informationen zur Größe, Kontur, Temperatur, Masse und Textur eines Objektes zu erlangen [4]. Für die haptische Wahrnehmung ist eine Bewegung des Menschen notwendig, weswegen die haptische Wahrnehmung als eine aktive Wahrnehmungsart klassifiziert wird. Damit steht sie der taktilen Wahrnehmung gegenüber, welche voraussetzt, dass der Mensch zur Reizaufnahme keine Bewegung durchführt. Bei der taktilen Wahrnehmung werden Informationen demnach ohne aktive Anteilnahme des Menschen aufgenommen, weswegen diese Wahrnehmungsform als passiv kategorisiert wird [5]. Beide Begriffe – die haptische als auch die taktile Wahrnehmung – können unter dem umgangssprachlich bekannten Begriff "Tastsinn" zusammengefasst werden.

Force-Feedback & Oberflächensensibilität Aus einer Publikation von R.E. Sofronia et al. [6] geht hervor, dass zur Erzeugung eines haptischen Feedbacks innerhalb der VR-Domäne sowohl Force-Feedback, als auch die Oberflächensensibilität relevant sind. Mithilfe von Force-Feedback ist der Benutzer des VR-Systems in der Lage, die Härte, das Gewicht und die Trägheit des jeweiligen virtuellen Objektes zu registrieren. Die Oberflächensensibilität ist hingegen keine Feedback-Art sondern bezeichnet die Sensibilität

des Menschen gegenüber den von außen auf die Haut einwirkenden Reizen, welche durch die Hautrezeptoren erfasst werden.

Haptisches Rendering Der Begriff des haptischen Renderings setzt sich aus dem führenden Adjektiv "haptisch" und dem Nomen "Rendering" zusammen. Häufig wird der Begriff des "Renderings" mit dem Rendering von grafischen und dadurch visuell wahrnehmbaren Szenen assoziiert. Die Bedeutung des Ausdruckes ist allerdings technologieunabhängig. So bezeichnet "Rendering" den Vorgang, in dem vorliegende Informationen dem Menschen wahrnehmbar gemacht werden. In Verbindung zur Beschreibung des "Haptik"-Begriffes im vorangegangenen Paragraph "Haptik" folgt daraus, dass "haptisches Rendering" der Vorgang ist, welcher notwendig ist, mit dem ein Benutzer in der Lage ist, vorliegende Informationen zur Größe, Kontur, Temperatur, Masse oder Textur eines Objektes über die Hautrezeptoren wahrzunehmen.

1.2 Kapitelübersicht

In Kapitel 1 wurde die Wahl des Themengebietes und die Bedeutung einiger Domänenbegriffe beschrieben. Dem folgend wird in Kapitel 2 die Relevanz des Themas dargelegt und in Kapitel 3 das aufgestellte Ziel dokumentiert, wonach in Kapitel 4 die Anwendungsmöglichkeit geschildert wird, welche als Fokuspunkt der folgenden Vorbereitung und späteren Entwicklung dienen soll. Daraufhin wird unter Berücksichtigung der Anwendungsmöglichkeit in Kapitel 5 auf den zu Projektbeginn aufgestellten Zeitplan eingegangen. In Kapitel 6 werden die Voraussetzungen genannt, die als notwendig erachtet wurden, um die zuvor in Kapitel 4 beschriebene Anwendungsmöglichkeit umzusetzen. Kapitel 7 dokumentiert und erläutert die Wahl der Hardware, die zur Erfüllung der aufgestellten Voraussetzungen genutzt wurde.

Daraufhin werden in Kapitel 8 die getroffenen Vorbereitungen dargelegt, welche notwendig waren, um das in Kapitel 3 beschriebene Ziel zu erreichen und die Anwendungsmöglichkeit in Kapitel 4 zu ermöglichen. Innerhalb dem Kapitel 8 werden zuerst in Punkt 8.1 mögliche Engines diskutiert und eine ausgewählt. Davon ausgehend wird in Punkt 8.2 & 8.3 die Vor- und Einarbeitung in die gewählte Hardware aus Kapitel 7 beschrieben.
Nachdem sich mit den hardwarebezogenen Voraussetzungen auseinander gesetzt wurde, wird im abschließenden Punkt der Vorbereitungen (Punkt 8.5) auf die Implementation weiterer Voraussetzungen eingegangen.

Im Anschluss zu den Vorbereitungen wird im Kapitel 9 mit der Dokumentation der detaillierten Recherche zur Haptik fortgefahren. Da sich das Projekt mit der Erzeugung von haptisch wahrnehmbarer Textur – und damit einem Teilbereich der Haptik – befasst, werden daraufhin in Kapitel 10 die Ergebnisse der detaillierten Texturrendering-Recherche beschrieben.

Mit dem neu erlangten Wissen wird in Kapitel 11 die genutzte Hardware rückblickend beurteilt und in Kapitel 13 auf Fortsetzungsmöglichkeiten des Projektes eingegangen. Abgeschlossen wird das Projekt mit dem Prozessassessment in Kapitel 14 und dem Fazit in Kapitel 15.

2 Relevanz des Themas

Mithilfe der virtuellen Realität sind Nutzer in der Lage sich visuell, auditiv – und im begrenzten Maße auch räumlich – in Umgebungen jeglicher Art zu begeben. Somit begrenzt sich die physiologische Wahrnehmung der virtuellen Umwelt des Benutzers lediglich auf zwei der fünf klassischen Sinne – die visuelle und die auditive Wahrnehmung [7]. Während diese beiden Sinne, mithilfe von HD- und UHD-Technologie bereits in beachtlicher Qualität angesprochen werden, haben technische Lösungsansätze, welche den Tastsinn adressieren, noch nicht denselben hohen Entwicklungsstand erreicht [8].
Im Hinblick auf die Technologie der VR sind, laut W. R. Sherman und A. Craig, eine

virtuelle Welt, Immersion, sensorisches Feedback und Interaktivität, Schlüsselelemente um die virtuelle Realität zu Erleben [9]. Die durch visuelle und auditive Reize geschaffene Immersion in die virtuelle Umgebung könnte durch die Verwendung von Geräten, die zusätzlich eine Reizung des Tastsinnes ermöglichen, weiter verbessert werden [6]. Mithilfe dieser haptischen Geräte ist der Nutzer eines VR-Systems in der Lage, haptische oder taktile Reize wahrzunehmen und dadurch virtuelle Objekte berühren, fühlen, manipulieren und umgestalten zu können [6] [10]. Mithilfe der dadurch gesteigerten Immersion können Systeme, neben Unterhaltungszwecken, zu Schulungszwecken genutzt werden. So beispielsweise in dem Fall der *Touch*-Reihe der Firma *3D Systems*, mit der die Haptik-Technologie dahingehend genutzt wird, um Chirurgiestudenten eine virtuelle Operation mit "maximaler Präzision"[11] zu ermöglichen, durch die sie "das korrekte Verfahren kennenlernen, bevor sie es im Klinikumfeld einsetzen"[11]. Häufig ist die Simulation von Textur Teil eines solchen haptischen Systems, da die Textur, ebenso wie die Reibungseigenschaften oder die geometrische Form, eine wesentliche Eigenschaft von realen – und damit auch zu simulierenden – Objekten ist [12]. Die Textur-Simulation kann beispielsweise zur 3D-Modellierung von digitalen Inhalten genutzt werden und unterstützt somit eine gesamte Domäne innerhalb der Anwendungsentwicklung bei der Verrichtung der alltäglichen Arbeit. So auch im Falle der von *3D Systems* entwickelten Geräte der *Touch*-Reihe [13] [14]. Zudem zeigt die Anzahl und Größe der Konferenzen [15][16][17][18][19] die Relevanz und die vorhandene Interesse der Wissenschaft an dem Fachbereich des haptischen Renderns, die laut Cagatay Basdogan und Mandayam A. Srinivasan bereits 2001 zu erkennen war [10].

3 Ziel

In Anbetracht der Relevanz der Textursimulation wird sich als absolutes Endziel die Entwicklung einer VR-Anwendung gesetzt, welche es ermöglicht, die in dem *Head mounted display (HMD)* angezeigten Texturen haptisch wahrnehmen zu können. Aufgrund des absehbaren Umfangs, beschränkt sich dieses Praxisprojekt allerdings lediglich auf die Vorarbeit zur Entwicklung der Software. Diese Vorarbeit umfasst die Recherchearbeit, die vor einer praktischen Umsetzung notwendig ist, um sich das dafür notwendige Fachwissen anzueignen. Daher wird der Zielerreichungsgrad aus eigener Sicht dahingehend beurteilt, als wie ergebnisreich sich die Recherchearbeit herausstellt und ob dieses Projekt ein angemessenes Fundament einer praktischen Umsetzung bildet.

4 Anwendungsmöglichkeit der Technologie

4.1 Einleitung

Zum Ende der in Kapitel 1 durchgeführten Domänenrecherche wurde eine mögliche praktische Umsetzung der hier angegangenen Technologie erdacht, welche sich im Laufe des Projektes als hilfreiche Gedankenstütze erwies. Denn durch diese Anwendungsmöglichkeit im Hinterkopf, konnten gefallene Entscheidungen noch einmal aus einem anderen Blickwinkel betrachtet und beurteilt werden um mögliche thematische Ausschweifungen zu vermeiden. Auf diese konzeptuelle Anwendung wird im Laufe dieser Projektdokumentation verwiesen, weswegen die mögliche Umsetzung im Folgenden grob dargestellt werden soll.

4.2 Beschreibung

Die erdachte Anwendungsmöglichkeit bewegt sich in dem Nutzungskontextes des digitalen Zeichnens innerhalb der VR. Dadurch sind Geräte notwendig, mit denen es möglich ist, das gezeichnete erfassen und es entsprechend in Echtzeit in der VR darstellen zu können. Da der Benutzer sich beim Zeichnen in der VR befindet, ergibt sich die Möglichkeit, sich mithilfe von Software Dritter in einen jeden erdenklichen Raum zu begeben und somit die Immersion

mithilfe audiovisueller Reize anzustoßen. Ergänzt und erweitert soll diese schließlich um den haptischen Teil. Dieser haptische Teil umfasst, wie bereits in der Zieldefinition unter Kapitel 3 beschrieben, die Simulation von Textur. In Anbetracht des Nutzungskontextes wird diese Technologie dazu genutzt, um eine haptisch erfassbare Zeichenoberfläche zu simulieren, sodass der Benutzer im erfolgreichen Fall den Eindruck hat, auf einem anderen Material als das des gewählten Gerätes zu zeichnen. Aus dieser Anwendungsmöglichkeit geht außerdem hervor, dass die haptische Wahrnehmung werkzeugvermittelt mithilfe eines stiftähnlichen Objektes zu geschehen hat und die erzeugten Reize nicht direkt auf die Haut des Nutzers zu rendern sind.

5 Zeitplan

Um für das in Kapitel 3 und Kapitel 4.2 beschriebene Vorhaben einen zeitlichen Rahmen zu schaffen, wurden die zu diesem Zeitpunkt bereits absehbaren Arbeitsschritte in kleinere unterteilt. Die vorraussichtliche Bearbeitungszeit dieser Unteraufgaben wurde daraufhin – ausgehend von Erfahrungswerten – geschätzt, mit einem zeitlichen Puffer versehen und dokumentiert. Allein die Bearbeitungszeit des Arbeitsschrittes "Textursimulation implementieren" konnte nicht anhand von gesammelter Erfahrung abgeschätzt werden, daher wurde ein – aus derzeitiger Sicht – großzügig gefasster Zeitraum für die Bearbeitung reserviert. Um nun einen vorraussichtlichen Endtermin des Praxisprojektes ermitteln zu können, wurden schließlich die Bearbeitungszeiten addiert und mit dem Starttermin des 01.03.2019 verrechnet, sodass sich dadurch der zeitliche Rahmen, in dem sich das Praxisprojekt zu bewegen hatte, geschaffen wurde. Im Laufe der Zeit zeigte sich jedoch, dass die Abschätzungen der nötigen Arbeitsschritte zwar größtenteils zutreffend waren, allerdings bei weitem nicht alle Arbeitsschritte einfassten, welche im Laufe des Projektes hinzu gekommen sind. Mehr dazu in Kapitel 14.
Der erstellte Zeitplan ist unter folgendem Link [20] zu finden.

6 Notwendigkeiten zur Umsetzung der Anwendungsmöglichkeit

Nach der Erstellung des Zeitplans wurde damit begonnen, einen groben Überblick über die grundlegend notwendige Hardware und Software zu schaffen. Zu diesem frühen Arbeitsstadium ist noch nicht klar, welche Hard- und Software notwendig ist, um das das Projektvorhaben optimal umzusetzen. Aus diesem Grund wird lediglich der Grundbedarf ausgehend aus der Anwendungsmöglichkeit aufgeführt (siehe Punkt 4.2) – die Voraussetzungen zur Erzeugung haptischer Reize werden während der Recherche gesammelt und werden sowohl in den jeweiligen Kapiteln genannt, als auch in dem Punkt 11 zur Reflexion der Hardwarewahl dokumentiert.

1. Ausgehend aus der Anwendungsmöglichkeit in Kapitel 4 ist ein Computersystem notwendig, das in der Lage ist, VR-Umgebungen berechnen zu könnnen.
2. Um die errechnete VR-Umgebung korrekt darstellen zu können, ist zudem ein VR-System notwendig, welches mindestens aus einem HMD besteht, dessen Position und Ausrichtung erfasst werden kann.
3. Da die Anwendungsmöglichkeit das Zeichnen auf einer Oberfläche vorsieht, sind Geräte notwendig, welche das digitale Zeichnen bzw. Malen auf einer Oberfläche ermöglichen.
4. Um Position, Ausrichtung und möglicherweise andere relevante Informationen über die entsprechenden Geräte erfassen zu können, ist weitere Hard- und/oder Software notwendig.
5. Software ist notwendig, um das auf der realen Oberfläche gemalte in die virtuelle Umgebung projizieren zu können.
6. Zum Zwecke der Zeitersparnis ist die Nutzung einer bestehenden Engine notwendig, welche die Entwicklung einer VR-Anwendung unterstützt.

7 Genutzte Hardware

Da das Projekt ein Studienmodul der TH Köln ausmacht, konnten dem Projekt zur Berechnung der VR-Umgebungen ein Computer der Hochschule gestellt werden – als Betriebssystem wurde *Windows 10 Pro* genutzt. Mit der im Rechner verbauten Hardware konnte dadurch die Voraussetzung 1 aus Punkt 6 erfüllt werden.
Um auch die Voraussetzung 2 zu erfüllen, wurde von der TH Köln zudem das System der *HTC Vive* bereitgestellt, sodass im Laufe des Projektes mit diesem VR-System gearbeitet werden konnte. Um die Voraussetzung 4 zu erfüllen, wurden von der TH Köln zudem *VIVE Tracker* zur Verfügung gestellt. In Verbindung mit der bereitgestellten *HTC Vive* können so die Positionen und Ausrichtungen von Objekten verfolgt werden, die in die virtuelle Umgebung projiziert werden sollen.
Da mithilfe der Zeichengeräte aus Bedarfspunkt 3 haptische Reize erzeugt werden sollen, galt diesen Geräten die größte Aufmerksamkeit. Weil gemäß der Anwendungsmöglichkeit, das Zeichnen auf einer virtuellen Leinwand ermöglicht werden soll, wurde sich auf die Wahl eines digitalen Grafiktabletts mit entsprechendem Stylus entschieden. Dies bot den Vorteil, dass von dem Hardware-Hersteller bereits eine Schnittstelle zu einem Windows-Computer zur Verfügung steht. Von größerer Bedeutung war allerdings, dass Grafiktabletts in der Lage sind, sowohl die Position der Stylus-Spitze, als auch die Druckintensität erfassen zu können [21]. Das Produkt, welches für dieses Projekt ausgewählt wurde, war das *Wacom Intuos CTH-690* [22]. Die Wahl einer Software, um das auf dem Grafiktablett gezeichnete in die VR projizieren zu können (siehe Notwendigkeit 5), wurde bewusst erst nach der Festlegung auf eine Engine getroffen und wird deswegen an dieser Stelle nicht weiter dokumentiert und erst im Kapitel 8.5 erneut aufgegriffen.
Um den noch ausstehenden zweiten Teil der Voraussetzung 3 zu erfüllen, wurde ein *Nintendo Switch JoyCon* [23] in Erwägung gezogen, um die Funktion des Stiftes bzw. Pinsels zu übernehmen.
Zu diesem Zeitpunkt wurde die Theorie aufgestellt, dass das Grafiktablett die Positions- und Druckdaten des *JoyCons* genauso gut aufnehmen kann, wie mit dem mitgelieferten Stylus – unter der Vorraussetzung einer simplen Modifikation: der Montage einer Stylus-Mine an den Controller. Eine weiteres Potential, das der *JoyCon* mit sich brachte, ist der bereits intern verbaute Akzelerometer und Gyroskop. Diese beiden Komponenten bilden – in Verbindung mit einem Magnetometer – üblicherweise eine IMU (Inertiale Messeinheit), mithilfe dessen die Ausrichtung und absolute Position eines Objektes erfasst werden kann. Der Akzelerometer misst die Beschleunigung des Gerätes entlang der x-, y- und z-Achse, während der Gyroskop rotierende Bewegungen registriert. Der Magnetometer erfüllt die Funktion, die absolute räumliche Ausrichtung des Gerätes wahrzunehmen. Dass kein Magnetometer in dem *JoyCon* verbaut ist, wurde registriert und als mögliches Hindernis notiert. Von weitaus größerer Interesse war die Möglichkeit des *JoyCons*, hochfrequente Vibrationen abzuspielen. Mit diesen Vibrationen wurden bereits haptische Eindrücke geschaffen, mit denen das Rollen einer Kugel auf einer Oberfläche nachgestellt werden kann. Eine Anwendung, welche dies Umsetzt ist beispielsweise *Tumbleseed* [24]. Das Minispiel "Kugelschätzen" aus *1-2-Switch* [25] zeigt hingegen, dass selbst das Rollen mehrerer Kugeln innerhalb einer Box simuliert werden kann. Um an die Positions-, Ausrichtungs- und Vibrationsdaten zu gelangen bzw. diese manipulieren zu können, muss eine Kommunikation mit dem *JoyCon* ermöglicht werden. Um dies zu erreichen, wurden zwei Projekte genauer betrachtet – ein Reverse-Engineering-Projekt des *Github*-Nutzers "dekuNukem" [26], aus dem wichtige technische Informationen zum *JoyCon* entnommen werden können, und das "JoyCon Toolkit" des *Github*-Nutzers *CTCaer* [27]. Das Toolkit bietet Funktionen, die für den kommenden Projektverlauf als möglicherweise relevant angesehen und berücksichtigt wurden. Durch die bestehenden Projekte Dritter, welche zur Umsetzung der Anwendungsmöglichkeit hinzu gezogen werden können, und der erfolgsversprechenden Funktionen des *JoyCons*, wurde sich dazu entschieden, dieses Gerät zur Bearbeitung des Projektes zu nutzen, womit auch der zweite Teil der Voraussetzung 3 erfüllt ist.

Um die letzte Bedingung aus Kapitel 6 zu erfüllen, war es notwendig, sich für eine Engine zu entscheiden, mit der das Projekt umgesetzt werden kann. Um diese Entscheidung möglichst fundiert treffen zu können, wurde im nächsten Schritt die für die Auswahl notwendige Recherche durchgeführt (siehe folgendes Kapitel 8.1).

8 Vorbereitungen

Die Auswahl der zu nutzenden Engine markiert den Anfang der Vorbereitungsphase, in der sich sowohl theoretisch als auch praktisch mit noch offenen Themen auseinander gesetzt wurde. Diese offenen Themen wurden in Abhängigkeit zueinander und zu der in Kapitel 7 genannten Hardware bearbeitet. So wird nach der Auswahl einer Engine in Punkt 8.1 mit der kurzen Einarbeitung in der davon in Abhängigkeit gewählten Programmiersprache begonnen (siehe Punkt 8.2). Im Anschluss begann die Einarbeitung in das *HTC-Vive*-System und der gewählten Engine (siehe Punkt 8.3 & 8.4). In dem Punkt 8.5 wird die Implementation der Notwendigkeiten beschrieben und die Vorbereitungsphase damit abgeschlossen.

8.1 Auswahl und Einarbeitung in die gewählte Engine

Da die nachfolgenden Schritte (Punkt 8.2 und 8.4) auf der Entscheidung einer Engine fußen, galt es diese Auswahl als erstes zu treffen. Zur Auswahl standen dabei unter anderen die *Unreal Engine 4*, *Unity* und *Godot*.

8.1.1 Unreal Engine 4

Die *Unreal Engine 4* wird von *Epic Games Inc.* entwickelt, gewartet und herausgegeben und bietet zahlreiche Features. So beispielsweise das Echtzeit-Rendern von photorealistischen Umgebungen mithilfe des Raytracing-Verfahrens [28], ein knotenbasiertes System zum Skripting von Gameplay – genannt *Blueprints* [29], und die Unterstützung der Entwicklung von Augmented Reality-, Virtual Reality (deutsch: virtuelle Realität)- und Mixed Reality-Anwendungen [30].
Zielplattformen der *Unreal Engine 4* sind neben Windows-, Linux- und MacOS-Systemen auch Konsolen wie beispielsweise die *PlayStation 4* oder die *Nintendo Switch* [31] und ermöglicht dadurch eine theoretisch breite Zielgruppe, die in der Lage wäre das entwickelte System zu nutzen. Ein solches System würde mit der *Unreal Engine 4* in der Programmiersprache C++ entwickelt werden, die ihre wohl größte Stärke in der Entwicklung von perfomanten Anwendungen aufweist. Sollte eine Anwendung kommerziell veröffentlicht werden, ist eine Lizenzgebühr von 5% auf den Bruttoumsatz nach den ersten 3000$, pro Spiel und Kalenderquartal zu zahlen. Dies bedeutet im Umkehrschluss, dass falls es nicht zu einem Verkauf einer Anwendung mit anschließendem Bruttoumsatz von mindestens 3000$ kommt, die Engine komplett kostenlos zu nutzen ist [32]. Auch ein *Unreal Marketplace* ist vorhanden, in dem zahlreiche *Assets* heruntergeladen und in dem eigenen Projekt verwendet werden können.

8.1.2 Unity

Unity wird dagegen von *Unity Technologies* entwickelt, gewartet und zur Verfügung gestellt und bietet – wie auch die *Unreal Engine* (siehe 8.1.1) zahlreiche Features. Nebst der *Unreal Engine* wird ein großer Wert auf die Entwicklung von realitätsnahen Umgebungen und Darstellungen gelegt [33] – wenn es auch nicht die einzige Möglichkeit ist, die Engine zu nutzen. Des Weiteren nennt *Unity* einen weiten *Asset Store* sein eigen, in dem unter anderem modellierte 3D-Objekte, Grafiken und Produktivitätstools zu finden sind, die in das eigene Projekt – kostenpflichtig oder kostenfrei – eingebunden werden können [34]. Laut der unternehmenseigenen FAQ-Webseite fallen bei der kommerziellen Veröffentlichung von Anwendungen, die mithilfe dieser Engine entwickelt wurden, keine Lizenzgebühren

an [35] und sollte sich bei der Download-Auswahl der gewünschten *Unity*-Edition für die *Personal*-Lizenz entschieden werden, fallen auch hier keine Kosten an, sodass die Engine kostenfrei genutzt werden kann. Dies beinhaltet allerdings die Einschränkung, dass mit der veröffentlichten Anwendung nicht mehr als 100.000$ Bruttojahresumsatz erziehlt werden darf. Sollte diese Grenze überschritten werden, ist es notwendig die *Plus*-Lizenz für 20€ bzw. 32€ pro Monat [36] zu abonnieren, mit der der maximale Bruttojahresumsatz bis auf 200.000$ gehoben wird. Für Beträge, die auch dieses Limit übersteigen ist das monatliche Abonnement der *Pro*-Lizenz für 125€ pro Monat [37] notwendig [38]. *Unity* unterstützt die Entwicklung von Anwendung in der Programmiersprache C#, die Ähnlichkeiten zu den Programmiersprachen C++ und Java besitzt [39], allerdings anfängerfreundlicher als die Entwicklung mit C++ sein soll. Ein Grund dafür ist, dass C# eine sogenannte *Managed Language* ist und Programmierer sich damit nicht mit dem Speicher-Management auseinandersetzen müssen [39]. Dies kommt besonders Programmieranfängern zugute, da dadurch die Komplexität und Schwierigkeit des Entwickelns einer stabilen Software verringert wird.

8.1.3 Godot

Godot ist eine Open-Source-Engine, mit der es möglich ist, sowohl 2D- als auch 3D-Anwendungen zu erstellen. Anders als in den vorherig genannten Engines, gibt es bei *Godot* keine Firma oder Unternehmen, die es aktiv entwickelt. Stattdessen wird die Engine, aufgrund der Open-Source-Eigenschaft, durch die Gemeinschaft der Programmierer oder Engine-Interessierten stetig verbessert und entwickelt, die ursprüngliche Autoren sind jedoch Juan Linietsky und Ariel Manzur [40]. Programme können mit *Godot* in den Programmiersprachen C#, C++ und der eigens für die Engine entwickelten Skriptsprache *GDScript*, geschrieben werden. Aufgrund des Open-Source-Konzeptes können allerdings noch weitere, durch die Community unterstützte, Sprachen genutzt werden - so beispielsweise Python oder Nim [41].
Neben der weiten Programmiersprachenunterstützung können mithilfe der Engine auch Anwendungen für Systeme nahezu jeglicher Art entwickelt werden, sodass man sich bei der Nutzung der Engine nicht auf ein oder nur wenige Systeme beschränkt [41]. Ein weiterer Vorteil ist, dass mit *Godot* eine Software angeboten wird, die alle teilhabenden Instanzen der Spieleentwicklung bedienen kann. So können mit *Godot* sowohl Skripte geschrieben, Animationen erstellt und Levels designt werden – alles mit einer Software [41]. Dies bietet den Vorteil, dass die besagten Instanzen auf einem höheren Niveau miteinander arbeiten können und die Übergänge von einer Abteilung in die andere reibungsloser verläuft.
Anders als bei den meisten anderen Engines, sind bei kommerziellen Anwendungen, welche mithilfe der *Godot*-Engine entwickelt werden, keine Zahlungen fällig. Dies liegt dem Konzept des Open Source zu Grunde und bietet zudem Vorteile wie die Freiheit, die Engine frei modifizieren und diese wiederum kommerziell anbieten zu können.

8.1.4 Gegenüberstellung der Engines in Bezugnahme zum Projekt

Um aus diesen gesammelten Informationen eine Entscheidung treffen zu können, galt es die Engines einander gegenüber zu stellen, mit dem Vorhaben (siehe Kapitel 3 & 4) in Bezug zu setzen und die entsprechenden Bedingungen und Ansprüche aufzustellen um die Engines gegen diese prüfen zu können. Die Menge der Bedingungen besteht aus einem technischen Teil und einem, der Voraussetzungen anderer Kategorien einfasst. Von den technischen Bedingungen ausgehend, musste die optimale Engine die Entwicklung einer Windows-Anwendung ermöglichen und eine möglichst ausgearbeitete VR-Unterstützung anbieten, mit der es möglich ist, Software für das *HTC-Vive*-System zu entwickeln. Des Weiteren war die Möglichkeit der Programmierung in einer der Programmiersprachen C++ oder C# gewünscht, wobei C++ jedoch C# vorgezogen wurde, da die dadurch angeeigneten C++-Kenntnisse dem späteren Vorhaben in der Computergrafik-Branche zu arbeiten aus eigener Ansicht dienlicher sein könnten. Ein weiteres privat-motiviertes

Bewertungskriterium ist, wie intensiv die Engine in der Industrie genutzt wird. Denn auch wenn dieses Kriterium nicht ausschlaggebend für die Umsetzung des Zieles ist, so ist es doch relevant für die persönliche Weiterbildung und die kommende Bewerbungsphase.
Da dieses Praxisprojekt von einem einzelnen Studenten vorbereitet und durchgeführt wird, waren möglichst geringe Kosten ein weiteres Kriterium der optimalen Engine. Da viele der betrachteten Engines noch nie genutzt wurden, war zudem die Einsteigerfreundlichkeit in die Entwicklung ein weiteres Kriterium. Dieses Kriterium war außerdem von Bedeutung, da die Durchführung des Projektes an einen zeitlichen Rahmen gebunden war und demnach die genutzte Zeit möglichst optimal und effizient zu nutzen war. Je weniger Zeit mit dem Einarbeiten in die Engine verstreicht, umso mehr Zeit konnte für die Erreichung des Projektzieles investiert werden. Die Einsteigerfreundlichkeit wurde dahingehend beurteilt, wie verständlich die Engine-Dokumentation aufgebaut ist, wie viele, und in welcher Qualität, Online-Tools und Hilfsquellen vorhanden sind und wie viel theoretisches Wissen und welche Fähigkeiten in welchem Grad vorhanden sein müssen um die Engine in angemessenem Maße zu nutzen. Um für die Beurteilung dieses Punktes für jedes Produkt allerdings keine mehrtägige Recherche und Tests durchführen zu müssen, wurden möglichst fundierte Einträge aus Online-Foren, Blog-Beiträgen und Online-Artikeln zu Rate gezogen, die jedoch dennoch von einer eigenen groben Recherche angeführt wurde.

Neben den oben genannten Engines (siehe 8.1) wurden noch weitere betrachtet und in Erwägung gezogen. Diese wurden jedoch bereits in einem frühen Zeitpunkt aus der Menge der möglicherweise optimalen Engines ausgesondert, da sie entweder dem Anspruch nicht genügen kostenfrei zu sein (*AppGameKit VR*), die Programmierung nicht in C++ oder C# ermöglichen (*LibGDX*) oder nur eine vergleichsweise geringe Präsenz in der Industrie aufweisen können (*Amazon Lumberyard*, *Urho3D*). Bereits nach der ersten, groben Recherche war zu erkennen, dass die *Unreal Engine 4* und *Unity* diese Bedingungen zu erfüllen scheinen. Auch wenn *Godot* nach der ersten, groben Recherche dem Kriterium der hohen Industriepräsenz nicht zu genügen schien, wurde es dennoch in die tiefergehende Recherche mit einbezogen. Grund dafür ist einerseits das Anraten des betreuenden Professors, allerdings auch zu einem hohen Maße das Konzept des Open Source. Denn aus diesem ging hervor, dass *Godot* in vollem Maße – egal ob kommerziell genutzt oder nicht – kostenlos ist und somit auch dem entsprechendem Kriterium genügt. Auch wenn diese Eigenschaft sehr positiv ist, so besaß sie im Rahmen des Praxisprojektes nicht die nötige Schwere, da das hier entwickelte Projekt ohnehin nicht kommerziell angeboten wird. Aus diesem Grund wurde zumindest der Kostenfaktor in allen 3 Engines gleich bewertet.

Auch die Bedingung der VR-Unterstützung – genauer gesagt die Unterstützung des *HTC-Vive*-VR-Systemes – war von allen Engines gegeben, allerdings bietet *Godot* diese erst seit der 2 Jahre alten Version *Godot 3.0* an. Diese Tatsache selbst reichte nicht für eine fundierte Entscheidung aus und es wäre an dieser Stelle ein Test oder eine noch detailliertere Recherche notwendig, allerdings wurde sich aufgrund der bereits verstrichenen Zeit dazu entschieden, die noch junge VR-Unterstützung vorsichtig zu behandeln und daher etwas geringer zu bewerten als die von *Unreal* und *Unity*.

In Anspruch der Programmiersprache C++ oder C# standen sich die betrachteten Engines allerdings in nichts nach, denn sowohl *Godot* als auch *Unreal* boten die Möglichkeit mit C++ zu programmieren, während *Unity* die Programmierung mit C# ermöglichte – beides gewünschte Programmiersprachen. Aus der Menge der aufgestellten Bedingungen wiesen die Engines lediglich in den Ansprüchen der Einsteigerfreundlichkeit und der Industriepräsenz einen Unterschied zueinander auf. In Puncto Industriepräsenz führte im vorigen Jahr *Unity* den VR-Markt an [42], während *Unreal* bei aufwendigen High-End-PC- und -Konsolenanwendungen eine stärkere Präsenz hatte [43]. Die hohe Präsenz in dem Bereich der High-End-Anwendungen fiel – in Anbetracht des Projektzieles – allerdings nicht in Gewicht, da keine High-End-Anwendung entwickelt werden soll. *Godot* konnte, im Vergleich mit *Unreal* und *Unity*, nur eine vergleichsweise geringe Industriepräsenz aufweisen. Da *Unity* innerhalb der VR-Domäne die höchste Industriepräsenz genießt, wurde *Unity* in

diesem Bewertungskriterium am höchsten bewertet.

Auch bei der Betrachtung der Einsteigerfreundlichkeit wurde *Unity* besser als *Unreal* oder *Godot* bewertet, da *Unity* eine größere Community sein Eigen nennt und im Rahmen eines vorigen Studienmoduls bereits mit der Engine gearbeitet werden konnte. Dies hat die Einarbeitungshürde in *Unity* sehr stark hinuntergesetzt, während die Hürden *Unreals* und *Godots* auf ihrer üblichen Höhe blieb.

Im Hinblick auf die Menge der erfüllten Bedingungen *Unitys* wurde sich dazu entschieden, diese Engine in der kommenden Bearbeitung des Projektes zu nutzen.

8.2 Einarbeitung in C#

Da *Unity* die Programmiersprache C# unterstützt, wurde sich zu Beginn mit einem C#-Buch [44] ausgestattet, um grundlegende Paradigmen und Charakteristika der Sprache C# zu erfahren. Auch ein grundlegendes Verständnis des .NET-Frameworks, auf dem die entwickelten C#-Programme ausgeführt werden, konnte durch diesen Schritt gebildet werden. Durch die frühe Auseinandersetzung konnte sich mit den fundamentalen Grundlagen bereits vor dem komplexeren Arbeitsschritt der kommenden Auseinandersetzung mit *Unity* vertraut gemacht werden.

8.3 Vor- und Einarbeitung - HTC-Vive-System

Nachdem die Einfindung in C# als erste kleine Hürde überwunden wurde, galt es auch die in Punkt 8.1 gewählte Engine genauer zu studieren. Dazu wurde hierbei der etwas andere Weg gewählt die Informationen aus direkter Quelle zu beziehen – der *Unity*-Online-Dokumentation [45]. Dort wurde unter anderem erfahren, wie die Entwicklung von VR-Anwendungen von statten geht und welche Schritte nötig sind, dass mit einer Entwicklung begonnen werden kann.
Die erste zu treffende Vorkehrung ist die Installation des *SteamVR*-Packages, welches die Software-Schnittstelle zwischen dem vorliegenden HTC-Vive-System und *Unity* bildet. Durch das *SteamVR*-Package konnte ohne Programmieraufwand eine lauffähige VR-Anwendung erstellt werden, in denen die grundlegenden Funktionen bereits implementiert gewesen sind. Diese Funktionen umfassten das Tracking des *HTC-Vive*-HMD, der dazugehörigen Controller und dem entsprechendem Verhalten der virtuellen Gegenstücke – also dem Umschauen im Raum, indem das HMD bewegt wird und der virtuellen Repräsentation der Controller. Weitere Funktionen, wie die Teleportation des Spielers im virtuellen Raum konnten daraufhin durch das *SteamVR*-Package mit nur sehr geringem Arbeitsaufwand eingebunden werden. Dadurch konnte in diesem Arbeitsschritt – der Einarbeitung in *Unity* – eine im Zeitplan als groß und herausfordernd bewertete Etappe (Kapitel 5), der Entwicklung einer fundamentalen VR-Anwendung mit grundlegenden Funktionen, mit einem sehr viel geringerem Arbeitsaufwand als geplant abgeschlossen werden.

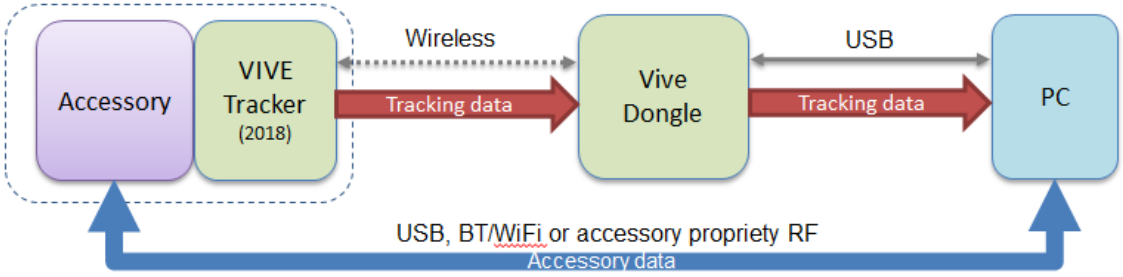
8.4 Einarbeitung in Unity

Nachdem die erste, fundamentale Anwendung erstellt wurde, wurde das Studium der *Unity*-Online-Dokumentation [45] fortgesetzt, wodurch das bereits durch vergangene Erfahrungen gesammelte Wissen erneut abrufbar gemacht und erweitert werden konnte. So wurde sich das *Component*-System *Unitys* genauer angeschaut, mit dem Objekte in der Anwendung skriptbasiert manipuliert werden können, und ein erneuter Überblick über den Editor und dem User Interface halfen, für komplexere Aufgaben innerhalb des *Unity*-Editors bereit zu sein.
Ausgehend aus der erdachten Anwendungsmöglichkeit (Kapitel 4), ging hervor, dass ein Import von 3D-Objekten unabdingbar war, weswegen auch dieses Thema besonders beleuchtet wurde. Erweiternd wurde sich zudem mit dem 3D-Modellierungsprogramm *Blender* [46] vertraut gemacht, um in der Lage zu sein, individuelle, speziell für das

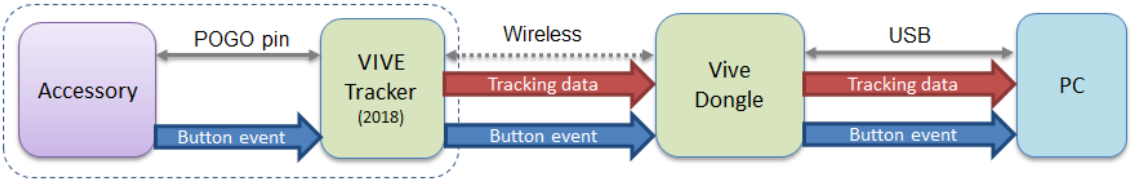
Praxisprojekt angepasste Änderungen an 3D-Modellen vornehmen zu können. Dieses Vorhaben gestaltete sich langwieriger als geplant, erwies sich im späteren Projektverlauf allerdings als äußerst hilfreich.

8.4.1 HTC-Vive-Tracker

Aufgrund der, das *SteamVR*-Package, gewonnenen Zeit (siehe Punkt 8.4) konnte die Recherche der VIVE Tracker umso detaillierter geschehen.
Der *HTC VIVE Tracker* ist in der Lage, seine momentane Position, Ausrichtung und Bewegung dem angeschlossenen System mitzuteilen. Gepaart mit der Möglichkeit, den Tracker an ein nahezu beliebiges Objekt zu befestigen, bietet dies die Option, diese Objekte schnell und genau zu tracken und entsprechend in der VR zu projizieren. Mit den verbauten *POGO-Pins* ist es möglich, anliegende Signale als Buttonbetätigungen der HTC-Vive-Controller zu interpretieren. Die Funktionen, welche durch die Betätigung der Buttons durchgeführt werden sollen, können schließlich über ein Skript implementiert werden, sodass mit dem physischen Objekt Funktionen innerhalb des Programms gesteuert werden können. Da das gesetzte Ziel (Kapitel 3) keine explizite Implementation der *POGO-Pin*-Funktionalität fordert, wurde sich allerdings dagegen entschieden, diese einzubinden um sowohl Zeit als auch Komplexität zu sparen. Als mögliche Erweiterung wäre eine Nutzung der *POGO-Pins* durchaus denkbar, diese entspricht allerdings momentan nicht den Fokus dieses Projektes. Die gewählte Verbindungsart des VIVE Trackers kommt demnach dem in der Abbildung 1a gezeigten Kommunikation gleich, während Abbildung 1b die Kommunikation unter Verwendung der *POGO-Pins* veranschaulicht.



(a) Genutzte Verbindungsart - Wireless ohne Verwendung der *POGO-Pins*



(b) Potentielle Verbindungsart - Wireless mit zusätzlicher Verwendung der *POGO-Pins*

Abbildung 1: Kommunikation des *HTC VIVE Trackers* mit einem PC-System [45]

Übermittelt werden die Daten über die intern im VIVE Tracker verbaute Antenne. Diese befindet sich mittig in der in Abbildung 2 gezeigten, roten Sphäre. Laut Dokumentation ist darauf zu achten, dass sich keinerlei metallischen Objekte in dem rot eingefärbten Bereich befinden sollten, da dies zu Störungen und dadurch zu Tracking-Ungenauigkeiten führen könnte. Für das Projekt bedeutet das allerdings lediglich eine milde Vorsicht, weil das getrackte Objekt, gemäß des Kapitels 7, ein Grafiktablett mit einer geringen Menge an metallischen Objekten ist und überwiegend aus Kunststoffen oder Glas besteht. Je nach Ausführung ist lediglich auf kleinere Schrauben zu achten, auf denen die markierte Stelle des Trackers nicht platziert werden sollte.

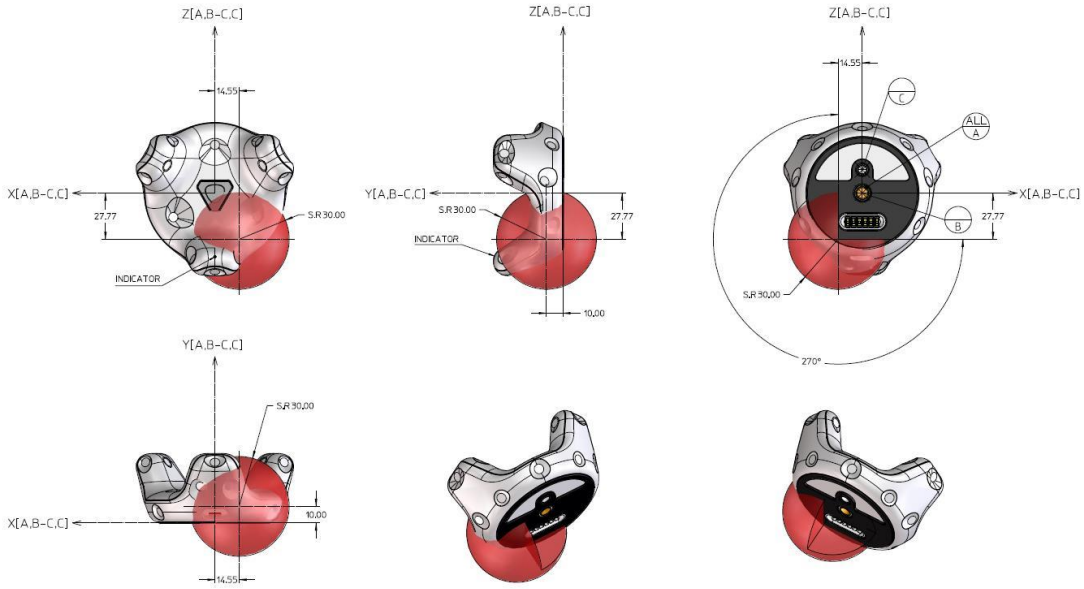
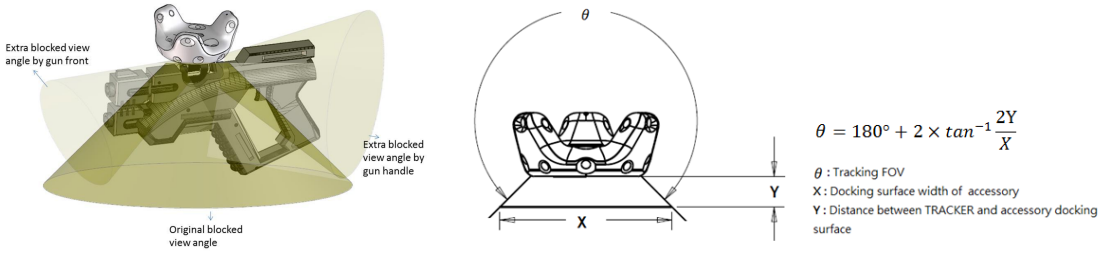


Abbildung 2: Position der Antenne des VIVE Trackers und des metallempfindlichen Bereiches

Ein für dieses Projekt weitaus kritischerer Punkt, der in der Dokumentation genannt wird, ist die sogenannte *no tracking area* (der dunkelgelb gefärbte Bereich in Abbildung 3a), in der das zu trackende Objekt optimalerweise platziert werden sollte.



(a) Darstellung des von dem Objekt blockierten Trackingbereiches

(b) Skizzenhafter Überblick der zur Position des Trackers relevanten Parameter

Abbildung 3: Darstellung des getrackten und des nicht getrackten Bereiches [45]

Der VIVE Tracker besitzt zahlreiche intern verbaute Lichtsensoren (siehe Abbildung 4), welche dazu genutzt werden, um die Position und Ausrichtung des Trackers zu bestimmen. Damit dies möglichst reibungslos funktionieren kann, müssen die von den *lighthouse*-Basisstationen ausgesendeten Laserstrahlen in der Lage sein, diese Lichtsensoren zu erreichen.
Sollte sich jedoch ein physikalisches Objekt vor diesen Lichtsensoren befinden, blockiert dieses Objekt den Laserstrahl, sodass die Zeitdifferenz zwischen Aussenden aus den Basisstationen und Eintreffen auf die Lichtsensoren nicht ermittelt werden kann. Je nach Menge der blockierten Sensoren und der Ausrichtung des Trackers im Raum können dadurch leichte bis schwere Ungenauigkeiten auftreten, die es möglichst zu vermeiden gilt, da wie bereits in Kapitel 2

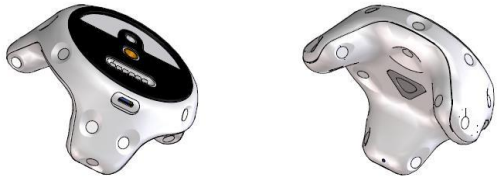
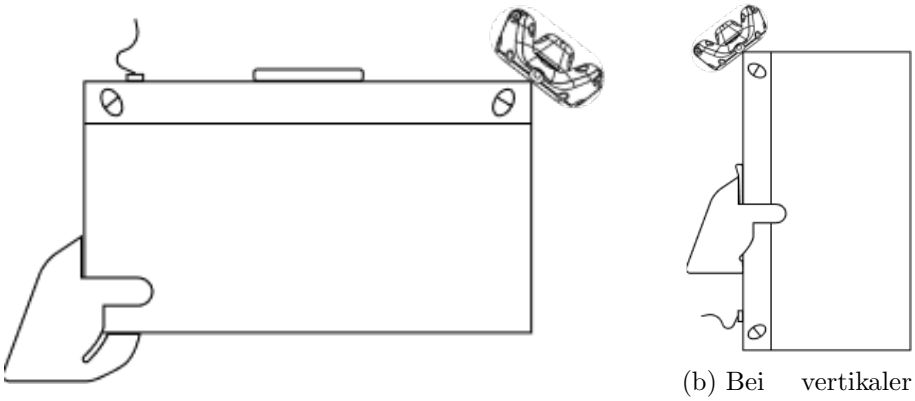


Abbildung 4: Die Positionen der Sensoren des VIVE Trackers[45].

beschrieben, die Immersion des Benutzers
eines der Schlüsselelemente ist, um die VR zu erleben [9]. Auf dieses Projekt und der Anwendungsmöglichkeit (siehe Kapitel 4.2) bezogen bedeutet das, dass besonders bei der Verfolgung des Grafiktabletts Tracking-Probleme auftreten können, sollte der VIVE Tracker nicht korrekt positioniert werden. Nach mehreren Überlegungen und Versuchen ergab sich, dass sich in horizontaler Ausrichtung die obere, rechte Ecke am besten zur Befestigung des Trackers eignet. In der vertikalen bzw. Portraitausrichtung entspricht dies der oberen, linken Ecke (siehe Abbildung 5).



(a) Bei horizontaler Ausrichtung

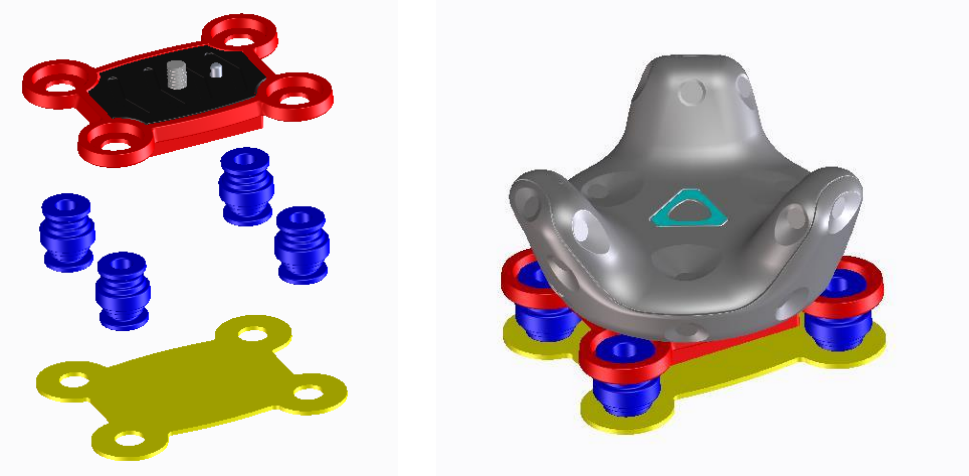
(b) Bei vertikaler Ausrichtung

Abbildung 5: Darstellung der Tracker-Positionierung.
VIVE-Tracker-Symbol bezogen aus der VIVE-Tracker-Dokumentation[45]

Grund für die Wahl dieser Positionierung ist, dass der Großteil der möglichen Nutzermenge rechtshändig ist [47] und daher mit der rechten Hand den Stift hält und die linke Hand dafür nutzt um das Grafiktablett zu halten. In sowohl der horizontalen als auch der vertikalen Ausrichtung würde so der Tracker beim Zeichenprozess und beim Bewegen des Grafiktabletts nicht oder in einem geringen Maße als störend empfunden werden. Sollte allerdings eine linkshändige Person das System nutzen, wäre in Fortführungen der Arbeit eine Möglichkeit zur Befestigung auf der rechten, oberen Ecke (in vertikaler Ausrichtung) bzw. linken, oberen Ecke (in horizontaler Ausrichtung) ratsam.

Neben der *no tracking area* ist auch der Abstand des Trackers zum Objekt relevant. So ist es durch die in Abbildung 3b abgebildete Formel möglich, den Bereich in Grad zu berechnen, der zum Tracking zur Verfügung steht. Je größer dieser ist, desto genauer kann die Verfolgung des Objektes geschehen. Der Maximalwert liegt dabei bei 270°, da unter dem Tracker der Bereits angesprochene Bereich liegt, in dem ohnehin keine Laserstrahlen auf Lichtsensoren treffen können – also der 90° Winkel, unter dem die Objekte platziert werden sollten. So auch das Grafiktablett in diesem Praxisprojekt.

In der Dokumentation wird darauf hingewiesen, dass bei einer kontinuierlichen Vibration des VIVE Trackers die Inertial Measurement Unit (deutsch: inertiale Messeinheit) (IMU) beeinflusst wird und ein IMU-Drift beobachtet wurde. Dies bedeutet, dass der Tracker dem angeschlossenen System zunehmend eine um wenige Grad verschobene Ausrichtung mitteilt. Für den Nutzer beobachtbarer Effekt ist dabei ein langsames Drehen und/oder Verschieben des virtuellen Objektes, während sich das reale Objekt nicht bewegt. Demnach würde dieser Umstand zu einer Unstimmigkeit der optischen und haptischen Wahrnehmung hervorrufen. In Bezug auf dieses Projekt bedeutet dies, dass die zu entwickelnde Software eine Möglichkeit bieten muss, diese Unstimmigkeit zu umgehen oder zu beheben – allerdings nur unter der Bedingung, dass das getrackte Objekt durchgängig vibriert. Die Dokumentation des VIVE Trackers empfiehlt die Installation eines Dämpfungssystems, mit dem die Vibration abgeschwächt werden soll (siehe Abbildung 6).



(a) (b)

Abbildung 6: Darstellung des in der VIVE Tracker-Dokumentation angeratenen Dämpfungssystems zur Milderung der auf den Tracker wirkenden Vibration [45]

Erweiternd zu der in Kapitel 8.3 erstellten Anwendung, wurden die VIVE Tracker in das bestehende *Unity*-Projekt eingebunden, sodass es möglich war, neben dem Tracking des Kopfes und der Controller, auch die VIVE Tracker und dadurch die daran befestigten Objekte zu verfolgen.
Das Ergebnis dieses Kapitels ist demnach das *Unity*-Projekt, in dem eine VR-Anwendung in C# erstellt und um die Komponente der VIVE Tracker erweitert wurde.

8.5 Implementation der weiteren Notwendigkeiten

Aufbauend auf den Vorbereitungsarbeiten aus Punkt 8.4.1 wurde sich erneut auf die aufgestellten Notwendigkeiten (siehe Punkt 6) konzentriert.
So wurde damit begonnen, die Voraussetzung 5 zu erfüllen – die Projektion des auf der realen Oberfläche gezeichneten in die virtuelle Umgebung. Nach der Recherche von möglicher APIs oder Bibliotheken wurde sich dazu entschieden, das Projekt des *Github*-Nutzers "hecomi" [48, 49] zu verwenden. Dieses ist auf die Implementation in Unity ausgelegt und somit, aufgrund der getroffenen Entscheidung in 8.1.4, für dieses Projekt geeignet.
Das "uDesktopDuplication" genannte Projekt ermöglicht die Duplizierung des primären Bildschirmes in eine VR-Anwendung. Der in Echtzeit aufgenommene Bildschirm wird intern als eine Texturdatei behandelt, welche einem gewünschten Objekt zugewiesen werden kann. Dies kann zur Umsetzung der Anwendungsmöglichkeit genutzt werden, indem der Nutzer während der Benutzung des VR-Systems ein externes Grafikprogramm startet und dies in der virtuellen Umgebung auf dem Grafiktablett sehen und nutzen kann. So kann eine bereits bestehende Zeichensoftware genutzt werden, um das tatsächliche digitale Zeichnen zu ermöglichen – dies vermindert die Komplexität dieses Projektes, beseitigt die Notwendigkeit der Entwicklung einer eigenen Zeichensoftware und erspart dem potentiellen Nutzer eine erneute Einarbeitungsphase in ein neues Zeichenprogramm.

Damit der duplizierte Bildschirm auf einem virtuellen Grafiktablett bzw. einer Leinwand angezeigt werden kann, war es zunächst notwendig, in *Blender* ein realitätsgetreues 3D-Modell des *Wacom Intuos* zu erstellen. Nach einigen Iterationen konnte ein geeignetes Modell angefertigt und implementiert werden, welches die Projektion des duplizierten Bildschirmes auf die Zeichenfläche ermöglicht. Dadurch kann das auf der realen Oberfläche gezeichnete in die virtuelle Umgebung projiziert werden – was die Erfüllung der Bedingung 5 aus Punkt 6 bedeutet.
Da somit mithilfe des Grafiktabletts in der VR mit dem Windows-System interagiert

werden kann, wurde daraufhin das Tracking des Tabletts angegangen. Durch die bereits durchgeführte Vorbereitungsarbeit in Punkt 8.4.1 konnte dies ohne Komplikationen mithilfe eines VIVE Trackers umgesetzt werden. Dieser wurde, wie dokumentiert, an der rechten, oberen Ecke des Tabletts angebracht (siehe Abbildung 5).

Um neben dem Grafiktablett den *JoyCon*-Controller tracken zu können, wurde zum Auslesen der Akzelerometer- und Gyroskop-Daten eine von *Looking Glass Factory, Inc.* erstellte Bibliothek [50] verwendet. Auch dieses ist für die Nutzung innerhalb der *Unity*-Engine konzipiert und ermöglicht, neben dem Auslesen der Beschleunigungs- und Rotationsdaten, das gezielte Abspielen von Vibrationen. Dank des innerhalb des Projektes vorhandenen Beispielcodes, konnten diese Funktionen zeitnah implementiert und an die Anwendungsanforderungen angepasst werden. Dadurch ist es möglich die Ausrichtung des *JoyCons* in Echtzeit zu erfassen und in der virtuellen Umgebung entsprechend anzuzeigen, allerdings fehlt nach wie vor die Funktionalität des Magnetometers – die Erkennung der absoluten Position des Gerätes im dreidimensionalen Raum.
Die naheliegende Möglichkeit, den *JoyCon* wie das Grafiktablett mithilfe eine VIVE Trackers zu verfolgen ist theoretisch zwar funktional, aufgrund des dadurch stark verschobenen Gewichtsmittelpunktes allerdings nicht akzeptabel. So würde der *JoyCon* bei der normalen Haltung eines Pinsels (dem sog. Präzisionsgriff) stets aus der Hand kippen – was nicht dem Verhalten eines Pinsels entspräche und damit der Immersion schaden würde.
Weitere Möglichkeiten zur Objektverfolgung wurden kurzzeitig recherchiert, konnten allerdings aus Zeitgründen nicht genauer betrachtet oder dokumentiert werden, weswegen die Positionserfassung des *JoyCons* an dieser Stelle pausiert werden musste.

Das Ergebnis dieses Arbeitsschrittes ist demnach die erfolgreiche Positions- und Ausrichtungserfassung des Grafiktabletts, der Projektion des primären Bildschirmes auf der zugehörigen Zeichenfläche, die Ausrichtungserfassung des *JoyCons* und die Möglichkeit des Abspielens von hochfrequenten Vibrationen mithilfe des *JoyCons*. Das entsprechende *Github*-Repository ist unter diesem Link [51] zu finden.

9 Recherche – Haptik

Die für die Themenwahl und Relevanzdarlegung (Kapitel 1 und 2) notwendige Recherche konnte nun, nach der Entwicklung einer fundamentalen VR-Anwendung, fortgesetzt und weitaus intensiver durchgeführt werden. Aufgrund des festgelegten Zieles (Kapitel 3) und der konzeptuellen Anwendungsmöglichkeit (Kapitel 4.2), wurde der Schwerpunkt der Recherche auf die Disziplin des haptischen Renderings von Texturen gelegt.

Damit das haptische Rendering von Texturen ermöglicht werden kann, wurde es zuerst als notwendig erachtet, eine generelle Recherche zum Bereich der haptischen Wahrnehmung vorzunehmen – jedoch stets unter Berücksichtigung des Zieles und der Anwendungsmöglichkeit. Dies sollte ein besseres Verständnis von fachwissenschaftlichen Texten ermöglichen, sodass dadurch im besten Falle das notwendige Wissen vorhanden ist, um ggf. Änderungen an vorhandenen, öffentlichen Systemen vornehmen zu können – zum Ziel der Anpassung an das gesetzte Projektvorhaben.
Während der gesamten Recherche zu Themen, die das haptische Rendering betreffen (Punkt 9.2 - 9.3 & Kapitel 10), konnten umfassende Informationen gesammelt und festgehalten werden. Diese Informationen sind in dem ersten Punkt 9.1 dokumentiert.
Der Aufbau der darauf folgenden Abschnitte ist ähnlich dem Wahrnehmungsvorgang des Menschen strukturiert. Da das Projekt die haptische und nicht die taktile Wahrnehmung adressiert, wird zu Beginn mit der für die haptische Wahrnehmung notwendigen Bewegung (siehe Punkt 1.1) begonnen. Dort wird beschrieben, welche Vorgänge der Mensch durchführt, um haptische Reize aufnehmen zu können (siehe Punkt 9.2). Parallel zum realen Wahrnehmungsvorgang werden daraufhin in Punkt 9.3 die Rezeptoren beschrieben, die bei Hautkontakt mit dem zu untersuchenden Objekt, die entstandenen haptischen Reize

in elektrische Signale umwandeln. Die dokumentierten Ergebnisse der Mechanorezeptoren-Recherche werden im darauffolgenden Punkt 10.1 aufgegriffen.
Zuletzt werden die technischen Anforderungen und Daten des haptischen Renderings aufgelistet und in Bezug auf das Projektvorhaben gesetzt. Mithilfe diesen neu gewonnenen Informationen wird daraufhin die Hardwarewahl (Kapitel 7) in Kapitel 11 beurteilt.

9.1 Allgemeines zum Haptischen Rendering

Kwon et al. [52] stellten 2010 die Aussage auf, dass sich der Bereich der Haptik-Forschung in ein interdisziplinäres Feld ausgeweitet hat. Fachbereiche wie die menschliche Perzeption, Psychophysik, Virtual Reality, Mechanikdesign und die Entwicklung von Steuerungssystemen bilden dabei die unterschiedlichen Themenfelder. Diese Aussage deckt sich mit den Ergebnissen eigener Recherche, allerdings scheinen seit 2010 weitere Fachbereiche, wie die Schwingungslehre und die Signalverarbeitung, hinzu gekommen zu sein. So werden zum Rendering haptischer Reize aktuell meist Vibrationen erzeugt – andere Technologien, wie die der Elektrostimulation, werden dagegen in diesem Zusammenhang nur vereinzelt genutzt.

Innerhalb des Fachgebietes des haptischen Renderings sind Geräte und/oder Systeme notwendig, mit denen dem Menschen eine Schnittstelle zur Aufnahme haptischer Reize geboten wird. Ein solches Gerät wird in der Fachliteratur meist als haptisches Gerät (engl.: *haptic device*) bezeichnet.
Laut [53] können haptische Geräte nach drei Eigenschaften kategorisiert werden: dem Montierungspunkt, der mechanischen Verhalten und der Anzahl der Freiheitsgrade.

Montierungspunkt Unter dem frei gewählten Begriff des Montierungspunktes ist der Ort zu verstehen, an dem das haptische Gerät je nach Funktion oder Größe montiert oder getragen wird. So können haptische Geräte beispielsweise als Handschuh, Exoskelett oder Handheld genutzt werden, oder auf einer festen Oberfläche stehen bzw. dort befestigt sein.

Mechanisches Verhalten Das "Mechanische Verhalten" (engl. original: *intrinsic mechanical behaviour* [53]) kann laut Salisbury et al. [53] in zwei Ansätze unterteilt werden: den *Admittance haptic devices* und den *Impedance haptic devices*. Beide Ansätze können zur Erzeugung von haptischen Interaktionsmöglichkeiten in der VR genutzt werden [54], allerdings ermöglichen sie dies durch gegensätzliche herangehensweisen. *Admittance haptic devices* erkennen die Kraft, die von dem Nutzer ausgeübt wird und sendet diesem eine entsprechende Bewegung bzw. eine veränderte Position. *Impedance haptic devices* erkennen wiederrum die eigene Position im Raum und rendern Kräfte, die von dem Nutzer registriert werden können [53, 54]. Unter Berücksichtigung des Projektes ermöglicht das Verhalten der *Impedance haptic devices* eine geeignetere Umsetzung des Zieles (siehe Punkt 3 & 4.2). Aus diesem Grund wird sich in der weiteren Recherche näher auf diesen Ansatz fokussiert.

Freiheitsgrad Der Begriff der Freiheitsgrade eines Gerätes gibt an, entlang wie vieler Achsen eine Rotation bzw. Bewegung ermöglicht wird. Unter Punkt 10.2 wird näher auf den Begriff des Freiheitsgrades eingegangen und zudem in Bezug auf das Projekt gesetzt.

Die typische Architektur einer VR-Anwendung, die sowohl visuelle, akustische als auch haptische Reize erzeugen kann, wird von Salisbury et al. in [53] beschrieben (siehe Abbildung 7). Dort wird darauf hingewiesen, dass das Teilsystem zur Erzeugung haptischer Reize aus drei Hauptkomponenten besteht: der Kollisionserkennung (in Abb.: *Collision detection*), der Berechnung der rückwirkenden Kraft (in Abb.: *Force response*) und des Regelungsalgorithmus (in Abb.: *Control algorithms*).

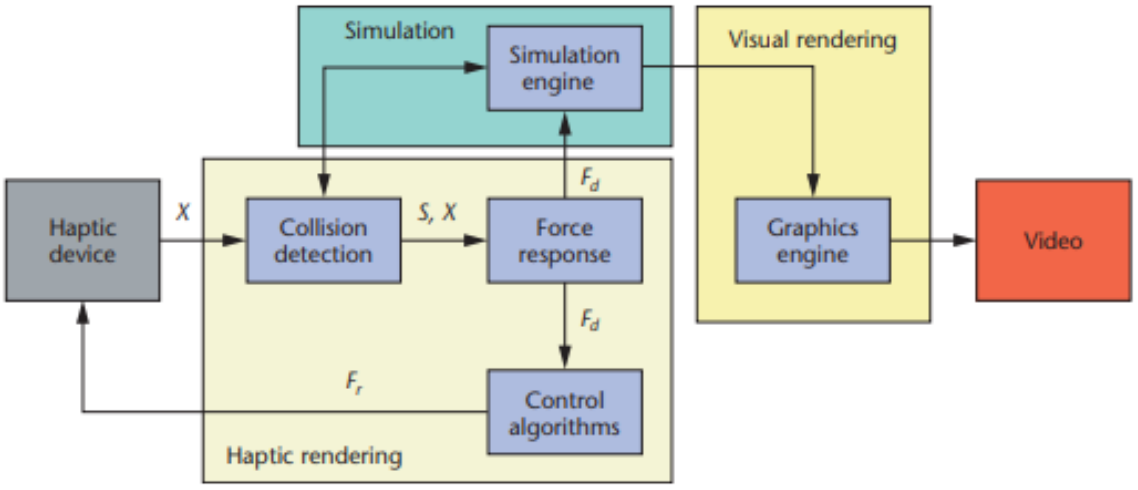


Abbildung 7: Detaillierter Aufbau einer VR-Anwendung zur Erzeugung visueller, akustischer und haptischer Reize nach Salisbury et al. [53].

Salisbury et al. erläutern in ihrer Einführung in die Konzepte des haptischen Renderings [53], welche Aufgaben die einzelnen Softwarekomponenten bewältigen und wie sie zueinander in Bezug stehen. Die folgende Auflistung ist der schematische Hergang zum Rendering haptischer Eindrücke und beschreibt den sich wiederholenden Vorgang zur Kollisionserkennung, Berechnung der Krafteinwirkungen und Rendering der haptischen Reize (entnommen aus [53], übersetzt durch den Autor).

- Hardwarenahe Regelungsalgorithmen erfassen die Positionssensoren an den haptischen Schnittstellenverbindungen.
- Diese Regelungsalgorithmen kombinieren die von jedem Sensor gesammelten Informationen, um die Position der haptischen Schnittstelle im kartesischen Raum zu erhalten - was der Position des Nutzers innerhalb der virtuellen Umgebung entspricht.
- Der Algorithmus zur Kollisionserkennung verwendet Positionsinformationen, um Kollisionen zwischen Objekten und des Nutzers zu erkennen und den daraus resultierenden Grad der Penetration oder Vertiefung zu ermitteln.
- Der Force-Response-Algorithmus berechnet die Kräfte, die während der Interaktion zwischen Nutzer und virtuellen, an einer Kollision beteiligten, Objekten auftreten.
- Der Force-Response-Algorithmus sendet diese Kräfte an die Regelungsalgorithmen, die diese über das haptische Gerät für den Nutzer rendern und dabei gleichzeitig ein stabiles Gesamtverhalten beibehalten.

Dadurch, dass Salisbury et al. damit einen Rahmen zur Entwicklung haptischer Anwendungen ziehen, sollte die abgebildete Softwarearchitektur (Abb.: 7) und die Wechselwirkung der dort aufgeführten Softwarekomponenten übernommen werden. Sollten bereits Erfahrungen in dem Bereich des haptischen Renderns gesammelt werden können, kann der Aufbau und der grundsätzliche Vorgang umstrukturiert werden – da allerdings kein Vorwissen in diesem Fachbereich vorhanden ist, wird sich möglichst an jenen Rahmen gestützt.

9.2 Die sechs Erkennungsmuster

Der Mensch wendet zur Erkennung haptischer Merkmale eines Objektes die folgenden 6 Erkennungsmuster an (siehe Abbildung 8) – bezogen aus [8], erweitert um [55] und ergänzt um die initiale visuelle Wahrnehmung (übersetzt durch den Verfasser):

- Um das Gewicht abschätzen zu können, werden Objekte angehoben.

- Um Informationen über die Wärmeleitfähigkeit zu erhalten, werden Objekte bewegungslos und ggf. lang andauernd berührt.
- Die Härte eines Objektes wird wahrgenommen, indem auf das Objekt gedrückt wird.
- Mithilfe von willkürlichen Gleitbewegungen über die Oberfläche werden Informationen über die Rauheit und Reibungseigenschaften erhalten.
- Um Informationen über das Volumen und die grobe Form der Objekte zu erhalten, werden Objekte mit der Hand umfasst.
- Um die exakte Form eines Objektes wahrzunehmen, fährt der Mensch über die Konturen des Objektes.
- Über den visuellen Wahrnehmungskanal werden räumliche Informationen bezüglich der Form und der Kontureigenschaften bezogen.

Das zuletzt genannte visuelle Erkennungsmuster wird vom Menschen als erstes durchgeführt und ist – obwohl die Wahrnehmung nicht über das haptische Wahrnehmungsorgan des Menschen geschieht – relevant um die haptischen Eigenschaften eines Objektes zu ermitteln. So auch beschrieben in [57], in dem genannt wird, dass es umfangreiche Belege für eine neuronale Verbindung des Sehens und des Berührens gibt. In [8] wird auf diese Tatsache verwiesen und genannt, dass visuelle Informationen relevant zur Übermittlung von Materialeigenschaften sind – neben haptischen Eindrücken.

Da in Anbetracht des Zieles und der Anwendungsmöglichkeit eine haptische Textur simuliert werden soll, kann dadurch die Erkenntnis gewonnen werden, dass neben der bloßen haptischen auch die visuelle Darstellung relevant ist und diese daher in der virtuellen Umgebung auf dem Grafiktablett angezeigt werden sollte, weswegen dies während der Programmierung der Anwendung beachtet und umgesetzt wird.

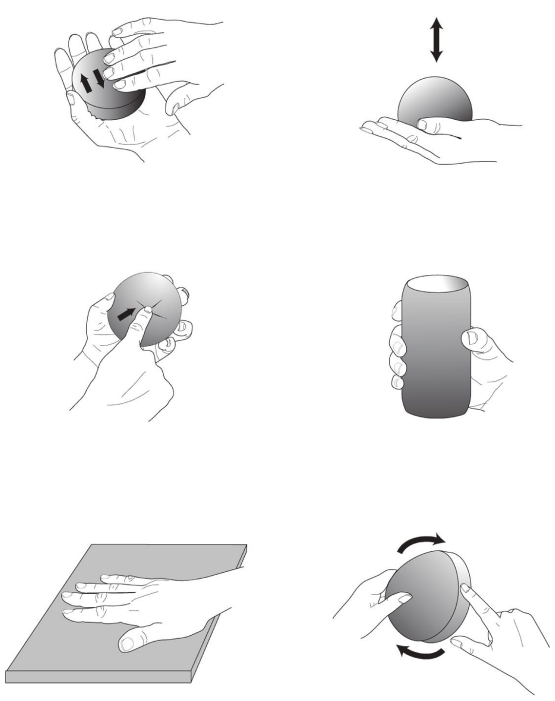


Abbildung 8: Abbildungen der 6 Erkennungsmuster nach Lederman & Klatzky [56]

Willkürliche Gleitbewegungen, Heben, Drücken, Umfassen, statische Berührung, Gleitbewegungen zur Erfassung der Kontur

9.3 Mechanorezeptoren

Wie bereits in Punkt 1.1, unter der Erklärung des Begriffes der Oberflächensensibilität erwähnt, besitzt der Mensch zahlreiche Rezeptoren in der Haut. Die davon für den Tastsinn – und damit das Projekt – relevanten, sind die Mechanorezeptoren. Diese sind durch physiologische Vorgänge in der Lage, von außen auf die Haut einwirkende Reize in elektrische Signale umzuwandeln, welche somit vom zentralen Nervensystem verarbeitet werden können [58]. Gemäß [59] werden die vorhandenen Mechanorezeptoren in niederschwellige Mechanorezeptoren (LTMRs) und hochschwellige Mechanorezeptoren (HTMRs) gruppiert. Die HTMRs reagieren auf schädliche, mechanische Einwirkungen von außen, LTMRs dagegen auf die ungefährlichen, mechanischen Reize [59]. Für dieses Projekt sind die letzteren von Relevanz, sodass im Folgenden näher auf diese eingegangen wird. Innerhalb der LTMR-Gruppierung werden die Mechanorezeptoren des

Weiteren in "*slowly adapting*" und "*rapid adapting*" klassifiziert. Nach dieser Klassifizierung der LTMRs, sind dies die Rezeptoren, welche für dieses Projekt weiterhin betrachtet wurden:

- SA-I-Rezeptoren *(slowly adapting I)*
- SA-II-Rezeptoren *(slowly adapting II)*
- RA-Rezeptoren *(rapid adapting)*
  (können auch RA-I-Rezeptoren genannt werden)
- PC-Rezeptoren *(Pacinian Corpuscle)*
  (können auch RA-II-Rezeptoren genannt werden)

Die für Druckveränderungen verantwortlichen SA-I- und SA-II-Rezeptoren verdanken ihren Namen der Eigenschaft, dass sie sich – verglichen mit den RA-Rezeptoren – den veränderten Reizstärken nur langsam anpassen und somit über einen längeren Zeitraum Signale senden, sollte ein andauernder, statischer Druck auf der Haut anliegen. Dementsprechend senden die RA-Rezeptoren nahezu unmittelbar nach dem Auftreten eines Reizes keine Signale mehr und feuern dadurch nur bei einer Veränderung der Reizstärke. Ähnlich wie die RA-Rezeptoren passen sich auch die PC-Rezeptoren dem neuen Druckverhältnis sehr schnell an, weswegen in Teilen auch die Benennung RA-II-Rezeptor gewählt wird, allerdings findet die Adaption noch schneller statt als bei den RA-Rezeptoren. [59][60][61][62].
Jede der vier unterschiedlichen Mechanorezeptorenarten sind verantwortlich für teils sich überschneidende Aufgabenbereiche. So sind die SA-I-Rezeptoren, aufgrund ihrer Eigenschaft der langsamen Adaption, in erster Linie für die Empfindung der Druckintensität verantwortlich – die SA-II-Rezeptoren, dagegen unter anderen für die Registrierung von Dehnungsreizen der Haut [59]. Aufgrund der Besonderheit, dass RA-Rezeptoren nur bei einer Bewegung der Haut ein Signal aussenden, sind sie für die Registrierung von Berührungen zuständig [58] und ermöglichen die Registrierung von Gleitbewegungen und niedrigfrequenter Vibration (20Hz-40Hz) [59] [62]. Die PC-Rezeptoren sind die emfpindlichsten der Mechanorezeptoren und ermöglichen die Reizaufnahme von hochfrequenten Vibrationen [59]. Das von ihnen ausgesendete Signal passt sich der veränderten Vibrationsfrequenz sehr schnell an und sind aufgrund ihrer Empfindlichkeit in der Lage, auf distanzierte Vibrationsreize zu reagieren [59]. Diese Information entspricht zudem der Aussage in [63], in der darauf hingewiesen wird, dass die PC-Rezeptoren einen wesentlichen Teil zur "Wahrnehmung von übertragenen Vibrationen beim Gebrauch von Werkzeugen" beitragen. Die vier Arten der Mechanorezeptoren, mit den zugehörigen Zellen und Körperchen, sind zur Übersicht der Abbildung 10 zu entnehmen.

Neural Channel[a]	SA I (NP III)	RA (FA I, NP I)	SA II (NP II)	PC (FA II, P)
End Organ	Merkel Disk	Meissner Corpuscle	Ruffini Ending	Pacinian Corpuscle
Sensory Adaptation[b]	Slow	Fast	Slow	Fast
Receptive Field[c]	Small	Small	Large	Large
Frequency Range (Hz)[d]	< 5	3–100	15–400	10–500
Perceiving Property	Pressure, Fine Details	Flutter	Stretch	Vibration

- [a] The name of each psychophysical or neural channel has a few variants as shown inside the parentheses. This paper follows the convention of [8].
- [b] The SA (slowly adapting) channels tend to maintain their response level for a sustained stimulus, and they are responsible for perception of spatial properties. The RA (rapidly adapting) channels cease to respond immediately after a stimulus is applied, relaying temporal properties.
- [c] Each afferent unit responds to a mechanical stimulus only if it is applied within a certain area on the skin. This area is called the receptive field of the afferent unit. A smaller receptive field leads to a higher spatial sensitivity.
- [d] Different sources report minor differences in the frequency range over which each channel responds. This table relies on [5], [7], [9].

Abbildung 10: Charakteristika der vier relevanten Mechanorezeptoren [62]

Es ist allerdings anzumerken, dass die RA-Rezeptoren, nur in unbehaarten Hautbereichen aufzufinden sind – in behaarter Haut übernehmen die Haarfolikelsensoren die Funktion der RA-Rezeptoren, indem sie "die Geschwindigkeit der Auslenkung des Haarschafts" [61] messen. Da das Projekt allerdings vorsieht, lediglich den unbehaarten Hand- und Fingerbereich anzusprechen, wurden die Haarfolikelsensoren nicht weiter recherchiert.

Laut S. Choi und K.J. Kuchenbecker ist unter den Reizen, welche mithilfe der soeben angesprochenen Mechanorezeptoren wahrgenommen werden können, die Erzeugung einer Vibration am weitesten verbreitet. Ansätze, welche Reibung, Dehnung oder Druck auf der Haut simulieren, wurden dagegen seltener umgesetzt [62].

Es ist hervorzuheben, dass diese Aussage für den gesamten Bereich des haptischen Renderings getroffen wurde. Daher wurde es als Indiz dazu aufgefasst, dass zur Umsetzung des Projektvorhabens (Punkt 3 & 4.2) die Erzeugung von Vibrationen – und damit die Ansprache der PC-Rezeptoren – genauer betrachtet werden sollte. Das Ergebnis der Recherche der Mechanorezeptoren ist demnach ein fundamentales Wissen über die haptische Wahrnehmung. Mit diesem Wissen konnte das Verständnis der wissenschaftlichen Berichte zum haptischen Rendering und der Textursimulation stark unterstützt werden, sodass möglichen Missverständnissen vorgebeugt werden konnte.

10 Recherche – Textursimulation

Nachdem in der weiter gefassten Recherche in Kapitel 9 allgemeines Wissen aufgebaut und ein fundamentales Verständnis zu den Terminologien und Paradigmen innerhalb der Haptik aufgebaut werden konnte, wird im folgenden näher auf den projektspezifischen Teilbereich des Texturrenderings eingegangen. Dieses Projekt sieht, gemäß dem aufgestellten Projektvorhaben in Kapitel 4, die Nutzung eines Werkzeuges vor, mit dem ein Nutzer in der Lage ist, über eine virtuelle Textur zu fahren und die haptischen Begebenheiten dieser Textur realitätsgetreu wahrzunehmen.

10.1 Vibrotaktile Stimulation

Um die texturellen Eigenschaften einer Objektoberfläche zu identifizieren, fährt der Mensch – gemäß der Erkennungsmuster in Punkt 9.2 – mit willkürlichen Gleitbewegungen über die gewünschte Oberfläche. Unter Verwendung eines Werkzeuges entstehen dabei hochfrequente Vibrationen [64], die sich in Amplitude, Frequenz und Wellenform ändern können. Beeinflusst werden diese Eigenschaften von der Geschwindigkeit, mit der über die Oberfläche gefahren und von der Kraft, welche auf die Objektoberfläche ausgeübt wird [64]. Diese Erkenntnis verfestigt die in Punkt 9.3 getroffene Annahme, dass Vibrationen nach dem jetzigen Wissensstand der adäquateste Ansatz ist, um haptische Reize zu erzeugen. Es gibt zwar andere Möglichkeiten wie die Nutzung von Magnetismus [65] oder Ultraschall [66], allerdings scheint in Anbetracht des Projektvorhabens die Wahl zur Nutzung von Vibrationen als Reizträger angemessener zu sein. Grund für diese Annahme sind einerseits die Anzahl der Forschungsprojekte, in denen sich die Vibrationserzeugung bereits als erfolgreicher Ansatz bewiesen hat [8, 62, 64, 67, 68, 69, 70, 71, 72]. Andererseits können aufgrund des bereits weiter erforschten Bereiches der Vibration [62] zahlreiche Studien zu Rate gezogen werden, um ein möglichst realitätsnahes Ergebnis erzeugen zu können.

10.1.1 Frequenz, Wellenform, Amplitude

Frequenz So erging aus weiterer Recherche, dass die Mechanorezeptoren in der menschlichen Haut auf Frequenzen im Bereich von 0,4-1000Hz reagieren [68] und unterschiedliche Impulse mit einem Zeitabstand von 5 ms erkennen können [62]. Der Frequenzbereich, in dem der Mensch die höchste Sensibilität aufweist, liegt dabei bei 150-300Hz [68]. In diesem Bereich sind Menschen in der Lage, Oberflächenveränderungen von einer Größe von bereits 0,03 μm zu registrieren [68]. Dies mag den Eindruck verschaffen, dass dies der geeignetste Frequenzbereich zum Rendering von vibrotaktilen Stimuli sei, allerdings ist in wissenschaftlichen Projekten dagegen ein Frequenzbereich um 1kHz üblich und sogar empfohlen [73, 53]. Zur Erzeugung einer haptisch wahrnehmbaren Textur hat sich das

Abspielen einer solch hochfrequenten Vibration bereits in wissenschaftlichen Projekten als erfolgreich erwiesen [73, 69, 64, 67].

Wellenform Wie bereits erwähnt, ist zur Erzeugung einer Vibration neben der Frequenz auch die Wellenform der Schwingung relevant. Gemäß Hoggan et al. [74] kann der Mensch Änderungen der Wellenform besser wahrnehmen als Änderungen an der Amplitude oder der Frequenz. Allerdings werden in Hoggans Forschungsarbeit derart hochfrequente Wellen, wie sie zur Erzeugung von haptischen Eindrücken notwendig sind, nicht beachtet. Stattdessen wird sich in der Forschungsarbeit primär mit dem für den Menschen sensibelsten Frequenzbereich von 150-300Hz befasst [74]. Aus [75] geht laut Lynette A. Jones und Anshul Shinghal [72] hervor, dass der Mensch bei hochfrequenten Schwingungen nicht in der Lage ist, Änderungen der Wellenform wahrzunehmen. Dies lässt die Vermutung zu, dass der Fokus zur Bildung der notwendigen Schwingungen stattdessen auf der korrekten Modifikation und Modulation von Frequenz und Amplitude liegen sollte. Diese Spekulation muss allerdings noch auf Richtigkeit überprüft werden, ehe sie in der Entwicklung eines Systems Anwendung finden kann. Es kann jedoch festgehalten werden, dass durch abklingende Sinuskurven, welche lediglich mithilfe einer Veränderung der Frequenz und Amplitude modifiziert wird, erfolgreiche Ergebnisse erzielt werden konnten [76, 77].

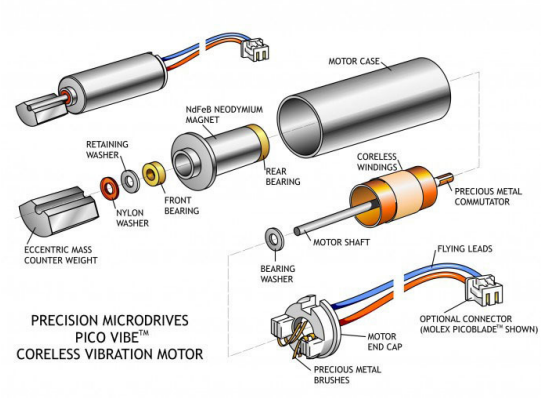
Amplitude Neben der Frequenz und der Wellenform gilt es noch die Amplitude der Schwingung in Betracht zu ziehen – also die Spannung V zu einem Zeitpunkt t. Durch eine Veränderung der Amplitude des vibrotaktilen Signales kann die subjektive Qualität des Reizes erhöht werden. Um dies zu ermöglichen, ist eine Veränderung der Hüllkurve des Signales notwendig, was durch eine sogenannte Amplitudenmodulation erreicht werden kann [62].

Aus diesen Erkenntnissen kann der Schluss gezogen werden, dass alle drei Welleneigenschaften unterschiedliche Effekte bei der Erzeugung haptischer Reize hervorrufen. Während die Relevanz der Wellenformmanipulation noch dargelegt werden muss, sind die Frequenz und die Amplitude bedeutsame Bestandteile bei der Erzeugung vibrotaktiler Stimuli. Um mithilfe dieser Welleneigenschaften möglichst realitätsnahe, haptische Reize zu erzeugen, ist die voneinander unabhängige Manipulation der entsprechenden Werte notwendig [62]. Diese Feststellung wird im folgenden Punkt 10.1.2 zur Beurteilung der recherchierten Hardware genutzt, um entscheiden zu können, welche Komponenten zur Erzeugung vibrotaktiler Stimuli genutzt werden können.

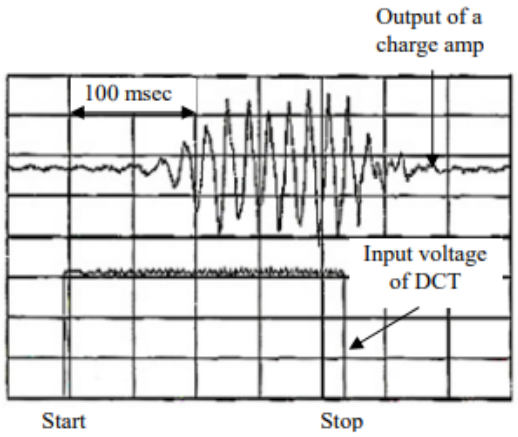
10.1.2 Hardware zur Erzeugung von vibrotaktilen Stimuli

ERM-Motoren Um vibrotaktile Signale korrekt abspielen lassen zu können, gibt es mehrere Technologien, welche in Betracht gezogen werden können. In handelsüblichen Geräten – wie beispielsweise Gamecontrollern – ist häufig ein *Eccentric Rotating Mass (ERM)*-Motor zur Erzeugung von Vibrationen verbaut. Diese Art der Motoren rotieren ein exzentrisches Gewicht, wodurch der gesamte Körper sich zu bewegen und dadurch vibrieren beginnt. Entsprechend der Rotationsgeschwindigkeit verändert sich dabei auch die Vibrationsintensität – je schneller sich also das exzentrische Gewicht dreht, umso höher ist die Frequenz und die Amplitude der enstehenden Vibration. Allerdings können Frequenz und Amplitude dadurch nicht unabhängig voneinander manipuliert werden, was jedoch für ein erfolgreiches Rendering von haptischen Reizen unabdingbar ist [62, 68]. Ein weiterer Nachteil der ERM-Motoren für das haptische Rendering ist die Tatsache, dass das exzentrische Gewicht bei jeder Veränderung der Frequenz und Amplitude Zeit benötigt um entweder zu beschleunigen oder zu bremsen. Mit dem von Niwa et al. [78] getesteten ERM-Motor begann die Rotation nach $50ms$ und die gewünschte Rotationsgeschwindigkeit konnte erst nach mehr als $100ms$ erreicht werden (siehe Abbildung 11b). Diese hohe zeitliche Verzögerung und die gegenseitige Abhängigkeit von Frequenz und Amplitude machen die

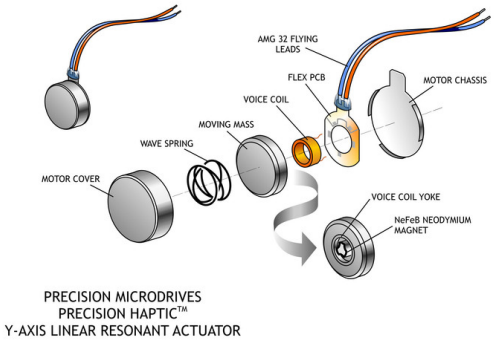
Nutzung von ERM-Motoren für das Projektvorhaben damit gänzlich ungeeignet.



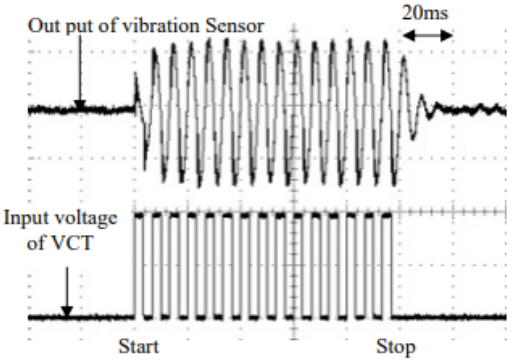
(a) Der Aufbau eines ERM-Motors [79]



(b) Anlauf- und Bremsdauer des von Niwa et al. geprüften ERM-Motors [78]



(c) Der Aufbau eines LRA-Motors [80]



(d) Anlauf- und Bremsdauer des von Niwa et al. geprüften LRA-Motors [78]

Abbildung 11: Der Aufbau und die Anlauf- und Bremsdauer von ERM- bzw. LRA-Motoren

LRA-Motoren Eine mögliche Alternative zu den ERM-Motoren bilden die *Linear Resonant Actuator (LRA)*-Motoren, die sowohl eine getrennte Manipulation von Frequenz und Amplitude ermöglichen, als auch eine hohe Reaktionsgeschwindigkeit aufweisen (siehe Abbildung 11d). Diese Vorteile ergeben sich aus der Bauweise des Motors: Eine elektromagnetische Masse ist in einem Gehäuse an einer Spule befestigt. Sobald Strom durch die Spule fließt, wird die elektromagnetische Masse in Bewegung gebracht – was von dem Menschen als Vibration wahrgenommen wird. Durch das gezielte Anlegen eines elektrischen Signals an die Spule ist es möglich, die Masse schneller oder langsamer schwingen zu lassen, ohne eine Änderung der Schwingungsstärke in Kauf nehmen zu müssen. Im Umkehrschluss kann die Masse entweder eine kürzere oder längere Entfernung pro Schwingung zurücklegen, während die Dauer einer einzelnen Schwingung konstant bleiben kann. Aus diesem Grund können sowohl Amplitude als auch Frequenz getrennt voneinander angesprochen und manipuliert werden. Ein vereinfachter Aufbau eines LRA-Motors ist der Abbildung 11c zu entnehmen, welcher nur entlang einer Achse vibrieren kann. Im haptischen Rendering sind Motoren dagegen in der Lage, eine Vibration in alle drei räumliche Dimensionen zu ermöglichen, sodass die gezeigte Abbildung in 11c nur exemplarische Bedeutung hat.

In dem Bereich des haptischen Renderings gibt es weitere, weniger verbreitete Möglichkeiten um vibrotaktile Stimuli zu erzeugen. So beispielsweise die Nutzung von piezoelektronischen Aktuatoren, in denen durch mechanischen Druck auf piezoelektrische Materialien, elektri-

sche Spannung entsteht [81]. Eine Fortführung der Recherche ist in diesem Punkt möglich, allerdings ergab sich aus der Recherche der LRA-Motoren, dass die Nutzung dieser Motoren zu favorisieren ist. Die Anzahl der betrachteten Forschungsprojekte, welche LRA-Motoren erfolgreich für die Erzeugung haptischer Reize nutzten, stützte diese Annahme. Zudem ging aus der Recherche hervor, dass die Einfachheit des LRA-Motors potentiellen Spielraum bietet, um eigene haptische Geräte zu entwickeln oder bestehende zu modifizieren.
Das Ergebnis dieses Arbeitsschrittes ist demnach die Festlegung zur Nutzung eines LRA-Motors und der Vermerk, bei möglichen hardware-betreffenden Komplikationen weitere Alternativen zu recherchieren.

10.2 Technische Anforderungen

Gemäß [6] besitzen haptische Schnittstellen 15 Hauptcharakteristika. 12 dieser Charakteristika können in direkten Bezug auf dieses spezifische Projekt gesetzt werden um das genutzte haptische Gerät zu bewerten und Verbesserungsmöglichkeiten zu finden. Wiederum 3 dieser 12 Merkmale sind allerdings für die Weiterarbeit im nächsten Schritt von Relevanz – der Freiheitsgrad, der Bereich, in dem der Stift bewegt werden kann, und das Gewicht des Gerätes.

Freiheitsgrad Der Freiheitsgrad (im Folgenden *degrees of freedom (DoF)* genannt) eines Gerätes gibt an, in wieviele Richtungen sich ein Gerät bewegen bzw. ausrichten kann. Häufig aufzufinden sind 3-DoF-Systeme, die im 3D-Raum meist entweder auf einer festen Position in jede beliebige Richtung rotiert, oder beliebig translatiert, jedoch nicht rotiert werden können. Zur Umsetzung der Anwendungsmöglichkeit (Kap. 4) ist allerdings ein 6-DoF-System notwendig, das in der Lage ist, sowohl die Ausrichtung als auch die Position eines Gerätes erfassen bzw. ändern zu können. Dieses Erfordernis erfüllen die in Kapitel 8.4.1 beschriebenen VIVE Tracker, mit dem das Grafiktablett bereits getrackt wird. Laut der Verfasser [6] ist dies das wichtigste Merkmal, weswegen besonders bei der Auswahl der Art des Stift-Trackings darauf geachtet werden muss, dass eine 6-DoF-Verfolgung möglich ist, da ohne diese eine korrekte Überführung des realen Objektes in die VR-Umgebung unmöglich ist.

Nutzbarer Bereich Der Punkt, an dem das Gerät den Kontakt mit einer Oberfläche oder einem Objekt simuliert, wird der haptic interaction point (HIP) genannt. Der räumliche Bereich in dem der HIP bewegt werden kann, ohne auf physische Begrenzungen zu stoßen, ist das zweite wichtige Merkmal. Laut [82] ist für kleinere Geräte ein quadratischer Bereich mit einer Seitenlänge von $15cm$ ausreichend. Auf dieses Praxisprojekt bezogen bedeutet das, dass der zu trackende Stift, mittig positioniert, auf dem Grafiktablett jeweils $7,5cm$ in jede Richtung bewegt können werden soll, ohne auf eine physikalische Barriere zu treffen.

Gewicht Zudem sollte das haptische Gerät laut [6] ein geringes Gewicht aufweisen. Wie gering genau, wurde allerdings nicht genannt, daher wurde für die Erarbeitung dieses Praxisprojektes das Merkmal folgendermaßen angepasst: “Das Gerät sollte ein *möglichst* geringes Gewicht aufweisen”. Dadurch wird stets ein Augenmerk auf das Gewicht gelegt und es wird versucht, das Gewicht des Stift-Tracker-Systems möglichst gering zu halten.

Zudem wird in [6] darauf hingewiesen, dass die Aktualisierungsrate der simulierten Kräfte bei ungefähr 1kHz geschehen sollte um realistische Eindrücke schaffen zu können – andere Aktualisierungsraten können dagegen zu einer regulären Vibration genutzt werden, wie sie bspw. in handelsüblichen Controllern genutzt werden oder um weichere virtuelle Objekte zu simulieren. Aus diesen, für das Projekt relevanten Charakteristika und der Recherche der Vibrotaktilen Stimulation (Punkt 10.1) geht also hervor, dass das System folgende Merkmale aufweisen muss bzw. sollte.

1. Der Stift muss mit 6-DoF getrackt werden.
2. Das Grafiktablett, auf dem gezeichnet wird, sollte eine Zeichenfläche besitzen, die mindestens einem Quadrat mit einer Seitenlänge von $15cm$ entspricht.
3. Die Hardware, bestehend aus Sift, Vibrationsmotor und Tracker, sollte so leicht wie möglich sein.
4. Der Vibrationsmotor muss in der Lage sein, die Frequenz und die Amplitude des Vibrationssignales getrennt voneinander manipulieren zu können.
5. Der Vibrationsmotor sollte eine gezielte Veränderung der Wellenform des Vibrationssignales ermöglichen.
6. Der Vibrationsmotor muss in der Lage sein, eine Vibration entlang der drei räumlichen Achsen ermöglichen.
7. Das für die Simulation der Textur zuständige Teilsystem muss in der Lage sein, die wirkenden Kräfte mit einer Rate von 1kHz zu aktualisieren.
8. Der Vibrationsmotor muss reaktionsschnell und präzise auf eine Veränderung der Frequenz- und Amplitudenwerte reagieren können.

10.3 Grundlegendes Vorgehen

10.3.1 Einleitung

Mithilfe, der im Folgenden beschriebenen, intensiven Recherche wurde die Erkenntnis gewonnen, dass das Vorhaben des haptischen Renderns von Texturen weitaus komplexer und vielschichtiger ist, als zu Anfang angenommen. So bestand der Großteil des recherchierten Materials nicht aus Letfäden oder umfassenden Einführungen, sondern aus Fach- und Forschungsberichten, die sich in den gewählten Forschungsansätzen teils stark unterschieden. Laut [83, 62] ist die Interesse der Wissenschaft an diesem Fachbereich erst in den 2000ern stark angestiegen (siehe auch Otaduy et al. [56]). Diese noch junge Entwicklung kann der Grund dafür sein, weswegen die Hauptinformationsquellen bisher noch Fach- und Forschungsberichte zu sein scheinen. Dadurch, dass dieses Projekt mit keinem Vorwissen zum haptischen Rendering begonnen wurde, bildete die Recherche in dem folgenden Bereich eine große Hürde. Durch das Fehlen eines einsteigerfreundlichen, gesammelten Werkes, gestaltete es sich als äußerst schwierig, die einzelnen Informationsfragmente aus den Forschungsberichten zu einem Gesamtbild zusammen zu setzen. Es konnten allerdings einige Gemeinsamkeiten in den gewählten Vorgehen und eine Tendenz zu bestimmten Konzepten erkannt werden. Diese werden in den folgenden Kapiteln beschrieben.

10.3.2 Texturmodellierung

Dieses Projekt sieht vor, eine Textur haptisch zu rendern. Um dies zu ermöglichen ist zunächst eine digitale Repräsentation der Objektoberfläche notwendig, damit sie von den haptischen Geräten gerendert werden kann. Wie genau diese digitale Repräsentation der Oberfläche strukturiert ist, kann mithilfe verschiedener Ansätze angegangen werden. Der sogenannte *physics-based approach* sieht vor, die physikalischen Ereignisse, welche während der Interaktion mit der Oberfläche auftreten, in Echtzeit zu berechnen [84, 67]. Da eine solche Berechnung allerdings zu komplex und dadurch zu rechenintensiv ist, wurde mit dem *data-driven approach* eine weiterer Ansatz entwickelt [67]. Dieser konnte in jüngster Zeit beachtliche Fortschritte erzielen [85] und sieht vor, dass die gewünschte Textur zuvor gescannt wird [67] um daraus eine digitale Input-Output-Map zu erstellen [86]. Zum haptischen Scannen der Textur verwenden viele Forscher sogenannte Profilometer oder 3D-Akzelerometer, um die feinen Unebenheiten der Objektoberfläche zu erkennen und abzuspeichern [85, 8]. Eine weitere Möglichkeit ist die Verwendung von Infrarotstrahlen [87, 88] oder Fotografie [89], um die Höhenprofile der Textur zu ermitteln.
Während der Texturaufnahme mithilfe eines Akzelerometers, werden Bewegungen mit

unterschiedlicher Beschleunigung, Geschwindigkeit und Druckstärke auf die Oberfläche ausgeführt [64]. Strese und Steinbach zeigten in [8], dass dieser Vorgang mithilfe eines von ihnen entwickelten Systems semi-automatisiert werden kann und nicht zwingend einen Menschen benötigt, wie in [64]. Die bei Strese und Steinbach entstandenen Materialkosten werden in dem entsprechenden Fachbericht aufgeführt und zeigen, dass die Anschaffung eines solchen Systems kostengünstig sein kann – die Kostenübersicht ist der Abbildung 12 zu entnehmen.

Comp.	USB DAQ	Acc.	2 FSR	IR	Mic.	USB Cam.	3D Comp.	Scale	Peltier	Temp. Sensor	Arduino Micro	$\sum$
Price ($)	120	19	14	2	12	11	10	15	20	20	20	263

Abbildung 12: Die Kostenübersicht des von Strese und Steinbach entwickelten Systems zur Aufnahme von Oberflächeneigenschaften [8].

Um Texturen unter Verwendung des *data-driven approach* zu rendern, ist allerdings ein dem Aufnahmegerät mechanisch möglichst ähnliches Gerät notwendig [67]. Der Grund dafür ist, dass die Vibrationserzeugung und -übertragung von dem gesamten Werkzeug-Hand-Oberfläche-System abhängig ist. Durch die Verwendung unterschiedlicher Geräte können zum erzeugten Vibrationssignal weitere Vibrationsartefakte hinzukommen, die die wahrgenommenen Reize verfälschen [67]. Solche Vibrationsartefakte können beispielsweise dadurch entstehen, dass ein anderer Vibrationsmotor verbaut wurde und dieser eine höhere oder niedrigere interne Reibung als das Aufnahmegerät erzeugt.

10.3.3 Texturrendering

Laut Shin et al. [85] und Culbertson et al. [67] sind unter Verwendung eines Werkzeuges, drei Objekteigenschaften von hoher Relevanz: die Härte, die Glätte und die Rauheit der Oberfläche. Wie zuvor in Punkt 9.2 aufgeführt, erfährt der Mensch Informationen über die Härte eines Objektes, indem er darauf Druck ausübt. Die Rauheit und die Reibungseigenschaften – also die Glätte – wiederum, werden mithilfe von willkürlichen Gleitbewegungen identifiziert. Dies resultiert darin, dass auf diese beiden Erkennungsmuster ein besonderes Augenmerk gelegt werden muss, um ein hochwertiges Ergebnis in der Simulation von Texturen erzielen zu können.
Momentan wird die Art und Weise, wie die Texturkomponenten (Härte, Rauheit & Glätte) innerhalb eines gemeinsamen Systems realitätsgetreu gerendert werden könnnen, weiter erforscht. Die Komponente der Rauheit (in manchen englischsprachigen Fachberichten auch *texture*) ist allerdings am weitesten analysiert. So wird in [67] darauf hingewiesen, dass diese Textureigenschaft von den Vibrationen abhängig ist, die während dem herüberfahren des Werkzeuges über die Oberfläche entstehen [67].
Die Texturkomponente der Glätte kann laut Culbertson et al. als Ergebnis einer Verrechnung der auf die Oberfläche wirkenden Normal- und Tangentialkraft simuliert werden [67].
Auch wenn der Mensch die Härte eines Objektes ohne Werkzeug am besten identifiziert, indem er Druck darauf ausübt – ist dies nicht unter Verwendung eines Werkzeuges der Fall. Laut einer Studie LaMotte's ist der Mensch mit einem Werkzeug besser in der Lage, die Härte eines Objektes zu erkennen, indem er mit dem Werkzeug auf die Oberfläche tippt bzw. klopft – und nicht lediglich Druck darauf ausübt [90]. Wenn dieses Tippen bzw. Klopfen mithilfe von Akzelerometern aufgenommen wird, entstehen im Schwingungssignal sogenannte Transienten (beispielhaft zu sehen in Abb. 13).

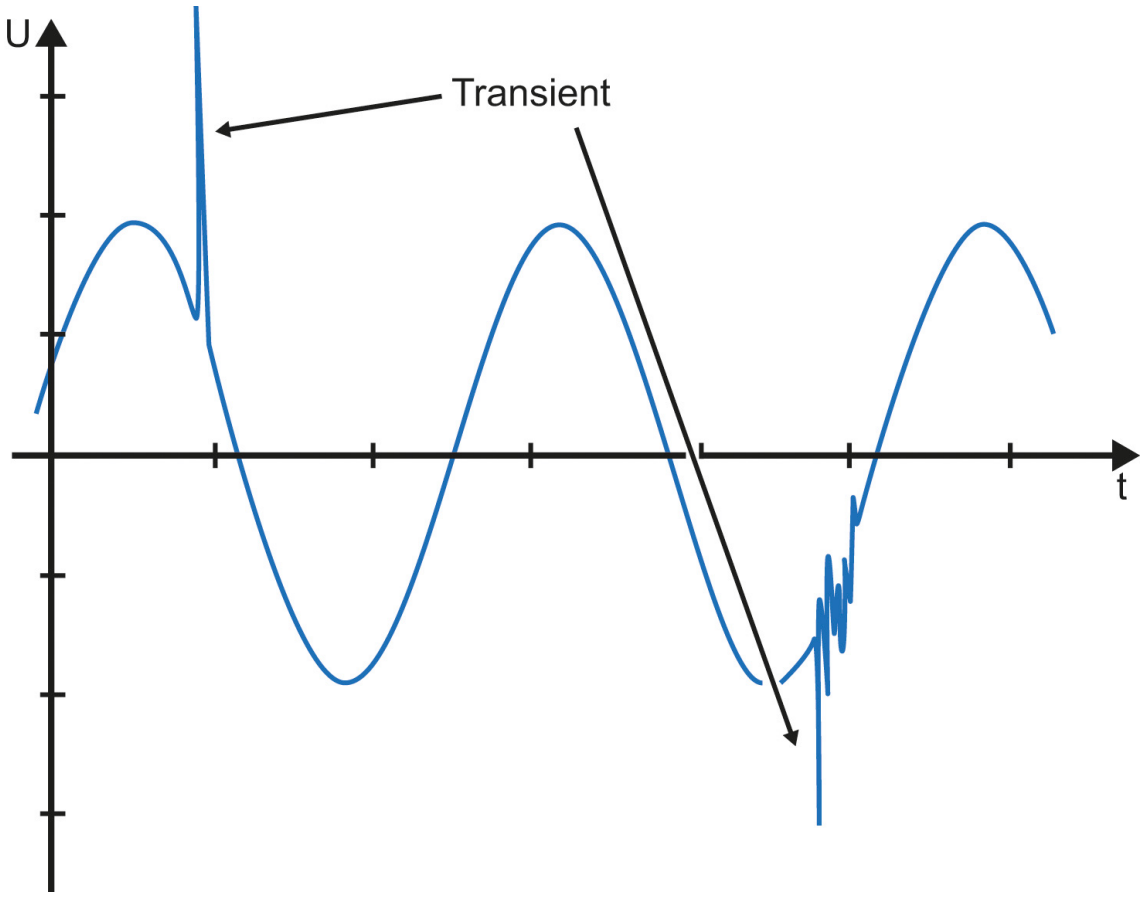


Abbildung 13: Beispielhaft dargestellte Transienten [91].

Um die beim Tippen bzw. Klopfen entstehenden Vibrationen korrekt nachbilden zu können, werden die Transienten dem bestehenden Vibrationssignal beigefügt. Um die Oberfläche beim Tippen weicher oder härter wirken zu lassen, kann die Transientenfrequenz und -amplitude dem gewünschten Härtegrad entsprechend angepasst werden.

In Bezug zum Rendering der Texturkomponenten ist eine noch tiefer gehende Recherche notwendig, um die Erkenntnisse praktisch umsetzen zu können. Da das Projekt zu diesem Zeitpunkt der Recherche allerdings schon lange in Bearbeitung war, musste zum Zweck der Zeitersparnis darauf verzichtet und die Recherche zur Textursimulation an dieser Stelle abgebrochen werden – im Falle der Fortsetzung des Vorhabens sollte allerdings mit der Recherche fortgesetzt werden.

10.4 Zusammenfassung

Es geht aus der Recherche der Textursimulation hervor, dass zur Erreichung des Projektzieles eine eigene Aufnahme von Texturen notwendig ist, um eine realitätsnahe haptische Wahrnehmung zu ermöglichen (siehe Punkt 10.3.2). Durch diese Erkenntnis erhöht sich der zu Beginn abgeschätzte Arbeitsaufwand enorm – denn neben der notwendigen, weiteren Recherchearbeit, ist zudem die Konstruktion einer Vorrichtung notwendig, welche zur Aufnahme von Texturen in der Lage ist. Auch die damit verbundenen finanziellen Kosten zur Erstellung einer dafür notwendigen Vorrichtung würden den Rahmen der tragbaren finanziellen Kosten sprengen. Eine Maßnahme um diesen Hürden entgehen zu können, ist die Verwendung von Texturaufnahmen anderer. Alleinige Möglichkeit dazu, wäre das von Culbertson, Delgado und Kuchenbecker veröffentlichte *Penn Haptic Texture Toolkit (HaTT)* [92, 93, 73]. Mithilfe dieses Toolkits könnte der umfangreiche Arbeitsschritt der Texturmodellierung übersprungen werden, allerdings nur mit gewissen Anpassungen. Culbertson et al. nahmen die Texturen mithilfe eines *SensAble Phantom Omni* auf, weswegen

die Nutzung eines solchen Gerätes nahe zu legen wäre. In dem zum *HaTT* gehörigen Fachbericht [73] werden allerdings Methoden genannt, um das Toolkit dennoch mit anderer Hardware verwenden zu können. Eine solche manuelle Anpassung erfordert jedoch weitere Recherchearbeit und zusätzliche Tests, um die Modifikationen durchzuführen und die getroffenen Änderungen ggf. zu justieren.
Zusätzlich ging aus Punkt 10.3.3 hervor, dass eine weitaus tiefere – und damit zeitintensivere – Recherche von Nöten wäre, um den Softwareteil des haptischen Renderings abzuschließen. Aus diesen Gründen wurde sich an diesem Punkt dazu entschieden, das Projekt an dieser Stelle zum Ende zu führen und es ggf. im Rahmen eines anderen Projektes fortzusetzen. Weiteres dazu in den folgenden Kapiteln 12, 13, 14 & 15.

11 Reflexion der Hardwarewahl

Zum Ende des Projektes wurde die Wahl der zu nutzenden Hardware mithilfe der neu gewonnenen Informationen reflektiert. Anhand dieser Reflektion sollen spätere, auf diesem Praxisprojekt aufbauende Arbeiten unterstützt werden, sowohl im Verständnis der Sachlage als auch in der Wahl der notwendigen Hardware.

11.1 VR-System, VIVE Tracker, Computer

Wie aus Kapitel 7 zu entnehmen ist, wurde als VR-System die *HTC Vive* genutzt. In Verbindung mit den *VIVE Trackern* desselben Herstellers und des verwendeten Computers, boten diese Geräte auch nach der detaillierten Recherche ein solides Fundament. Während der Recherche ergaben sich keine unüberwindbare Hürden, weswegen eine Weiterarbeit mit diesen Geräten nach aktuellem Wissensstand durchaus angestrebt werden kann.

11.2 Grafiktablett

Ausgehend aus der Recherche des allgemeinen Haptik-Feldes (Kapitel 9) und der Textursimulation (Kapitel 10) ergab sich, dass zur Erzeugung haptisch wahrnehmbarer Textur, das Erfassen der auf die Oberfläche wirkenden Normal- und Tangentialkraft von Relevanz ist [64, 67]. Mithilfe des gewählten *Wacom Intuos CTH-690* kann dagegen lediglich die Kraft wahrgenommen werden, mit der der Stylus auf die Zeichenoberfläche gedrückt wird (entspricht der sog. Gewichtskraft). Allerdings kann laut [94] die Normal- und die Tangentialkraft, aus der Gewichtskraft und dem Auflagewinkel des Stylus zur Oberfläche, auf folgende Weise berechnet werden (durch den Autor projektspezifisch angepasst):

$$F_{Normalkraft} = \cos\left(Auflagewinkel\right) * F_{Gewichtskraft} \tag{1}$$

$$F_{Tangentialkraft} = \sin\left(Auflagewinkel\right) * F_{Gewichtskraft} \tag{2}$$

Da mithilfe des genutzten Grafiktabletts die Gewichtskraft erfasst werden kann, ist lediglich das Auslesen des Winkels, mit dem der Stift auf der Zeichenoberfläche aufliegt, notwendig. Dies ist mit dem gewählten Grafiktablett allerdings nicht möglich.
Des Weiteren geht aus Kapitel 10.2 hervor, dass das Grafiktablett mit 6 DoF verfolgt werden muss, um es realitätsgetreu in die virtuelle Umgebung platzieren und bewegen zu können. Mit der Montage der VIVE Tracker war dies problemlos umzusetzen (siehe Punkt 8.5). Neben dem 6-DoF-Tracking ist zudem ein möglichst geringes Gesamtgewicht des haptischen Systems anzustreben, was mit dem genutzen Grafiktablett nach eigenem Ermessen ermöglicht wurde. Allerdings ist anzumerken, dass eine detailliertere Recherche bezüglich dieses Kritikpunktes notwendig ist, um eine objektive Entscheidung treffen zu können, ob das Gewicht im optimalen Bereich liegt. Als dritten und letzten Punkt in Kapitel 10.2, muss der nutzbare Bereich einer Oberfläche mindestens ein Quadrat mit einer Seitenlänge von $15cm$ ausmachen. Mit dem genutzten *Wacom Intuos CTH-690* ist

diese Voraussetzung erfüllt, denn die Zeichenfläche, auf die der Nutzer mit dem *Haptic Interaction Point (HIP)* navigiert, überschreitet nach eigener Messung die notwendigen $225m^2$.
Gemäß der durchgeführten Recherche kann das *Wacom Intuos CTH-690* für das Vorhaben genutzt werden – jedoch mit dem Vorbehalt, dass ggf. Modifikationen an dem Gerät oder an weiterer Hardware vorgenommen werden müssen, um den Auflagewinkel des Stylus erfassen zu können.

11.3 JoyCon

Die Erfassung des Winkels, mit dem der Stylus über die Oberfläche geführt wird, kann unter Benutzung des gewählten *Nintendo Switch JoyCons* durchgeführt werden. Dies bestätigt sowohl die Theorie (siehe Kapitel 7) als auch die Praxis in Punkt 8.5. Dort war es möglich die Ausrichtungsdaten des *JoyCons*, mithilfe des verbauten Gyroskops und einer Softwareschnittstelle [50] zu erfassen und auszulesen.
Aus der Recherche der Textursimulation in Kapitel 10 ging hervor, dass zum Abspielen haptischer Reize, neben den Ausrichtungsdaten des Stylus und der wirkenden Normal- und Tangentialkräfte, auch die Beschleunigungs-, Positions-, und Geschwindigkeitsdaten von Relevanz sind. Die Beschleunigungsdaten können unter der Verwendung der Bibliothek von *Looking Glass, Inc.* [50] mit dem verbauten Akzelerometer ausgelesen und verarbeitet werden. Die Positionsdaten können bei Berührung des *JoyCons* mit der Zeichenoberfläche des Grafiktabletts gewonnen werden, während die Geschwindigkeitsdaten aus dem zurückgelegten Weg und der dabei vergangenen Zeit berechnet werden kann. Allein die absolute Position im Raum, wenn der *JoyCon* die Zeichenfläche nicht berührt, kann mit den intern verbauten Geräten nicht erfasst werden. Um dies zu ermöglichen, sind weitere Maßnahmen notwendig, auf die im Kapitel zur möglichen Weiterarbeit (Kapitel 13) kurz eingegangen wird.
Der *JoyCon* wies nach der Implementation in Punkt 8.5 zudem das Verhalten auf, dass sich die Ausrichtung des virtuellen *JoyCons* stetig verändert hat. Dieses Problem konnte mit einem ständigen Zurücksetzen des entsprechenden Wertes umgangen werden, allerdings wirft dieses Verhalten die Frage auf, ob das Gerät für das gewählte Vorhaben geeignet ist.

Aus der Recherche zur vibrotaktilen Stimulation ging hervor, dass eine getrennte Modifikation der Frequenz und der Amplitude notwendig ist, um realitätsnahe Reize zu erzeugen (siehe Punkt 10.1). Mithilfe des *JoyCons* ist dies möglich – auch der empfohlene Frequenzbereich von $1000Hz$ ist laut der Dokumentation des verbauten LRA-Motors [95] abspielbar. Die maximale Frequenz liegt laut Hersteller bei dem internen Motor bei $1600Hz$ – und damit über dem notwendigen Frequenzbereich. Dadurch, dass in dem *JoyCon* ein LRA-Motor verbaut ist, ist zudem eine hohe Reaktionsgeschwindigkeit gegeben, was ausgehend aus Punkt 10.1.2 notwendig ist, um realitätsnahe Reize zu erzeugen.
Diese Punkte legen nahe, wieso Anwendungen wie *Tumbleseed* [24] oder das Minispiel *Kugelschätzen* der Software *1-2-Switch* [25] mit dem *Nintendo JoyCon* ein realitätsnahes haptische Rendering ermöglichen.
Die Recherche in Kapitel 10 legt allerdings nahe, dass das Rendern einer Textur andere Bedingungen als das Simulieren rollender, mit der Umgebung kollidierender Kugeln, erfordert. So sind bestehende haptische Systeme, welche Texturen haptisch wahrnehmbar machen, stets an einen soliden Untergrund befestigt – wie bspw. die *Touch*-Reihe des Unternehmens *3DSystems* [96].
In Anbetracht der Erfahrung des Unternehmens lässt dies die Vermutung zu, dass das Rendering einer haptisch wahrnehmbaren Textur, nach dem jetzigen Stand der Technik, nicht mit einem Gerät ohne statischen Montierungspunkt umzusetzen ist. Wie auch in Punkt 10.3.3 genannt, ist allerdings eine weitere Recherche in diesem Fachbereich dringend notwendig, um diese Vermutung fundiert belegen oder widerlegen zu können.

Um die Anwendungsmöglichkeit aus Kapitel 4 umsetzen zu können, könnte ein *JoyCon* demnach durchaus verwendet werden, da dieser zahlreiche Funktionen und Voraussetzungen erfüllt, die für das haptische Rendern notwendig sind – allerdings unter Berücksichtigung bzw. Vorbehalt der möglichen Probleme. Um mit der Entwicklung einer Anwendung, unter Verwendung eines *JoyCons* beginnen zu können, ist eine Fortführung der Recherche allerdings zwingend notwendig, um mögliche Problembereiche bereits frühzeitig erkennen zu können.

11.4 Zusammenfassung

In Anbetracht der genutzten Hardware, könnte das Projekt mit dem gewählten Grafiktablett und dem *Nintendo JoyCon* möglicherweise umgesetzt werden. Aufgrund der genannten Probleme, Hürden und Herausforderungen wird allerdings davon abgeraten, ein der Anwendungsmöglichkeit ähnliches Projekt mit den genutzten Geräten umzusetzen. Verbesserungsmöglichkeiten der Gerätewahl, bzw. der hardwarespezifischen Umsetzung werden im Kapitel 13 beschrieben.

12 Rückblick

Das Ziel dieses Projektes und die Bewertungskriterien zur Beurteilung des Zielerreichungsgrades, wurden in Kapitel 3 festgelegt. Aus diesen geht hervor, dass der Erfolg des Projektes dahingehend beurteilt wird, wie ergebnisreich die Recherche war und in welchem Maß dieses Projekt ein angemessenes Fundament einer praktischen Umsetzung bildet.
In Berücksichtigung der Vorbereitungsarbeiten in Kapitel 8, der Haptik- und Textursimulationsrecherche in Kapitel 9 & 10 und der Reflexion der Hardwarewahl in Kapitel 11, wird in diesem Kapitel der Zielerreichungsgrad nach den in Kapitel 3 beschriebenen Kriterien beurteilt.
Die Beurteilung, wie gut das Projekt ein Fundament für eine praktische Umsetzung der Anwendungsmöglichkeit aus Kapitel 4 bildet, wird danach gefällt, ob eine Entwicklung der Anwendungsmöglichkeit unmittelbar nach diesem Projekt begonnen werden kann. Falls nicht, wird beleuchtet, wie viele Aufgaben mit diesem Projekt bereits verrichtet wurden – und im Umkehrschluss, wie viele Arbeitsschritte noch zu bewältigen sind, um mit der Programmierung der Anwendung beginnen zu können.

Allein in Anbetracht, der für die Bearbeitung dieses Projektes, benötigten Zeit, ist erkennbar, dass mit diesem Projekt bereits ein großer Aufwandsanteil zur Entwicklung einer haptischen Anwendung abgeschlossen wurde. Besonders die Recherche des haptischen Renderns und der Textursimulation, ergaben eine wertvolle Basis um die fundamentalen Prinzipien und Paradigmen des haptischen Renderings zu verstehen. Ein solches Wissen ist für die Entwicklung einer entsprechenden Anwendung unerlässlich. Denn da das haptische Rendering nach wie vor ein Feld ist, in dem momentan viele unterschiedliche Herangehensweisen und Technologien erforscht werden, sind bisher lediglich Tendenzen – nicht aber feste Modelle oder Protokolle – zum haptischen Rendering aufzufinden. Eine grundsätzliche Kenntnis der Thematik ermöglicht ein tieferes Verständnis der in den wissenschaftlichen Fachberichten diskutierten Ansätze und Technologien. Um diese Erkenntnisse der Forschenden auf ein eigenes Projekt adaptieren zu können, ist es notwendig, diese gegebenenfalls auf die veränderten Umstände anzupassen – wofür widerrum das soeben angesprochene Verständnis der Thematik notwendig ist.
In diesem Hinblick konnte mit diesem Projekt ein großer Bereich der notwendigen Informationen recherchiert und herausgearbeitet werden können – was wiederrum der Erzeugung eines angemessenen Fundamentes gleichkommt. Es ist allerdings anzumerken, dass die Recherche innerhalb des haptischen Renderings mit den hier aufgeführten Informationen noch nicht als ausreichend betrachtet werden kann, um in der Forschung der Erzeugung haptischer Reize mitwirken zu können. Besonders in dem speziellen Fachbereich der Tex-

tursimulation – wie in Kapitel 10 beschrieben.

In Berücksichtigung dessen, dass weitere theoretische Recherchearbeit zu raten ist, kann dieses Projekt dennoch eine sehr hilfreiche Stütze zur Entwicklung einer haptischen Anwendung bilden. Aus diesem Grund wird dieses Projekt als ein angemessenes Fundament zur Weiterarbeit angesehen und das in Kapitel 3 abgesteckte Ziel als erreicht betrachtet – weswegen das Projekt als erfolgreich bewertet wird.

13 Fortsetzungsmöglichkeiten des Projektes

Wie unter anderen im vorigen Kapitel beschrieben, kann auf diesem Projekt weiter aufgebaut werden. Im folgenden werden die Fortsetzungsmöglichkeiten genannt, mit denen das Projekt fortgeführt und verbessert werden kann.

Um die Anwendung in Kapitel 4 entwickeln zu können, ergab sich aus Kapitel 11, dass die Wahl eines Grafiktabletts zu empfehlen ist. Während und nach der Implementation der Notwendigkeiten (Kapitel 8.5) war jedoch zu erkennen, dass die geringe Auflösung der virtuellen Zeichenfläche zu gering war und dadurch die hochauflösende Repräsentation des primären Bildschirmes nicht zufriedenstellend projiziert werden konnte. Auch wenn die Zeichenfläche die Mindestgröße von $225cm^2$ übersteigt, ist mit diesem Vorgehen dennoch eine größere Fläche notwendig, auf der die Zeichenprogramme projiziert werden können.
Sollte zudem die Funktionalität des Grafiktabletts erweitert werden sollen, können die in Punkt 8.4.1 angesprochenen *POGO-Pins* verwendet werden. Mit diesen ist der Entwickler in der Lage einem realen Objekt, Möglichkeiten zur Softwareinteraktion hinzuzufügen. Dies könnte erwünscht sein, falls man das Grafiktablett um weitere Funktionalität erweitern möchte. So beispielsweise das vergrößerte Einblenden der Bedienelemente.
Die Funktionalität des Stylus, kann allerdings nicht ohne Weiteres auf diese Weise erweitert werden. Da für die Simulation eines Stylus das dafür genutzte Gerät eine stylus-ähnliche Form und Größe haben sollte, wird das Problem bestehen, dass das Gewicht des *VIVE Trackers* das Gerät aus der Hand kippen lässt.
Aus diesem und der in Kapitel 11 beschriebenen Herausforderungen, die die Nutzung eines *JoyCons* als Stylus mit sich bringt, wird empfohlen, stattdessen ein Gerät zu nutzen, das gezielt für das haptische Rendering entwickelt wurde. Ob dieses Gerät bereits kommerzielle Reife erlangt hat, ist dabei von geringerer Relevanz. Denn aus der in Kapitel 9 & 10 dokumentierten Recherche, können die relevanten Informationen entnommen werden, um eigene Geräte zu entwickeln, die speziell für den vorgesehenen Zweck eingesetzt werden können.

Nicht nur die Hardware sollte in einer Fortsetzung angepasst werden – auch die Wahl der zu nutzenden Software. Die *Unity*-Engine kann zwar ohne, zum jetzigen Zeitpunkt, erkennbare Schwierigkeiten genutzt werden, allerdings sind die Projekte Dritter aus Kapitel 7 & 8.5 – ausgenommen der Bildschirm-Duplizierung obsolet, sobald der *JoyCon* mit einem geeigneteren Gerät ausgetauscht wird. Stattdessen werden Frameworks, wie *Chai3D* oder *OpenHaptics* empfohlen, die dazu genutzt werden können, um haptisches Rendering zu ermöglichen. Allerdings ist dabei zu beachten, dass beide der genannten Frameworks lediglich auf bestimmte Hardware ausgelegt sind – so beispielsweise die in diesem Projekt bereits erwähnten haptischen Geräte des Unternehmens *3DSystems*.

Die Recherche der Textursimulation ergab, dass zum korrekten Rendering von Texturoberflächen die Aufnahme einer Textur möglichst mit demselben Gerät durchgeführt werden sollte, mit dem die haptischen Reize später abgespielt werden. Sollte sich für die Wahl eines kommerziellen Gerätes der *Touch*-Reihe von *3DSystems* entschieden werden, kann auf die öffentliche Bibliothek von Culbertson et al. [92] zurückgegriffen werden. In dieser Bibliothek wurden Texturen verschiedener Beschaffenheit nach dem *data-based*

modelling mithilfe eines *Touch*-Gerätes aufgenommen und können bei Verwendung des gleichen abgespielt werden. Dies würde die Komplexität der Weiterarbeit drastisch verringern. Culbertson et al. gehen in dem entsprechenden Fachbericht [93] darauf ein, dass die Daten einem anderen haptischen Gerät angepasst werden können – damit ist die Bibliothek auch für den Fall relevant, sollte eine andere Hardware zum Abspielen der vibrotaktilen Stimuli genutzt werden.
Sollte die Bibliothek nicht genutzt werden können, oder sich aus einem anderen Grund dagegen entschieden werden, müssen die zu rendernden Texturen selbst aufgenommen und modelliert werden. Dies würde nicht nur einen praktischen Aufwand mit sich führen, sondern auch eine fortführende Recherche zur Texturmodellierung erzwingen.

Nach eigenem Ermessen kann das Ziel einer Weiterarbeit demnach entweder als studienorientiert oder entwicklungsorientiert eingeordnet werden. Unter "studienorientert" ist in diesem Zusammenhang gemeint, dass ein fortgeführtes Projekt das Aneignen und Sammeln von Wissen als primäres Ziel hat – bei dem Begriff "entwicklungsorientiert" ist hierbei die Ausrichtung des Projektes gemeint, das haptische Rendering möglichst schnell mithilfe einer Software umzusetzen.
Sollte letzteres das Ziel sein, ist der Erwerb von Hardware zu empfehlen, die bereits kommerzielle Reife beim Rendering haptischer Reize erreicht hat. Während der Recherche zur Haptik ergab sich das Bild, dass im Rahmen wissenschaftlicher Arbeiten die haptischen Geräte der Firma *3DSystem* am häufigsten genutzt werden. Wenn man sich zur Weiterarbeit mit einem solches Gerät ausstattet, können die in den Fachberichten beschriebenen Vorgänge und Ansätze leichter und schneller auf das eigene Projekt adaptiert werden. Besonders durch die bereits bestehende Bibliothek von Culbertson et al. [92], in denen die vorliegenden Texturdaten mithilfe eines "*SensAble Phantom Omni*" (mittlerweile umbenannt zu *3DSystems Touch*) abgespielt werden können.
Sollte allerdings das Studieren der gesamten Domäne des haptischen Renderns im Vordergrund stehen, kann auf die Wahl eines solchen etablierten Gerätes verzichtet werden, da sich stattdessen genauer über die möglichen Vorgänge zur Texturmodellierung und Texturrendering informiert wird. Mit dem dadurch gewonnenen Wissen, kann ein eigenes Gerät angefertigt werden, welche die Anforderungen aus den Punkten 10.1.2 & 10.2 erfüllt. Die studienorientierte Zielsetzung ermöglicht dabei eine speziell für das Vorhaben zugeschnittene Anwendung und umgeht dabei die Restriktionen der bereits bestehenden Geräte – wie beispielsweise der Montierungspunkt des Gerätes auf einer soliden Oberfläche – nimmt allerdings voraussichtlich weit mehr Zeit in Anspruch als eine entwicklungsorientierte Fortsetzung.

14 Prozessassessment

Zu Beginn des Projektes wurde ein Zeitplan aufgestellt, auf den im Kapitel 5 verwiesen wird. Dieser Zeitplan wurde lediglich zu Anfang des Projektes als Richtlinie verwendet, denn je weiter das Projekt vorangeschritten ist, umso mehr stellte sich heraus, dass der notwendige Aufwand sehr viel höher ist als der zu Anfang angenommene. Eine solche Verkalkulierung ist nicht unüblich und der Zeitplan sollte iterativ dem sich klarer werdenden Überblick anpassen. Dies wurde allerdings außer Acht gelassen, was dazu führte, dass es sich für einen beachtlichen Zeitraum als sehr schwierig gestaltete, einen Überblick über das Projekt zu behalten.
Auch das gänzlich neue Thema des haptischen Renderings führte dazu, dass nahezu jeder damit verbundene Arbeitsschritt ein vielfaches mehr an Zeit und Mühe beanspruchen musste. So war es notwendig, sich in Fachbereiche wie die Signalverarbeitung und die Physionomie einzuarbeiten, um den Fachberichten folgen zu können, sodass das dort berichtete auf das Projekt adaptiert werden konnte. Da das haptische Rendering als Technologie momentan noch erforscht wird, waren Fachberichte die Hauptquelle, um Informationen

über das haptische Rendering zu erlangen. Es existieren zwar bereits Bücher, in denen das haptische Rendering beleuchtet wird, nach eigenem Ermessen allerdings nicht mit den notwendigen Informationen um daraus eine Anwendung entwickeln zu können.
Aus diesem Grund war es notwendig, viele einzelne Fachberichte zu lesen, sich das dortige Vokabular und Hintergrundwissen anzueignen und die einzelnen Wissensfragmente zu einem großen Gesamtbild zusammen zu setzen. Dieser Prozess verschlang ein großes Maß an Mühe und Zeit, führte allerdings zu einem erfolgreichen Projekt.

Da sich bezüglich des haptischen Renderings, kein Überblick vorab geschaffen werden konnte, war ständiges Abschweifen in bestimmte Bereiche des haptischen Renderings unumgänglich – da nicht abzusehen war, ob diese Informationen für das Gesamtbild bzw. für dieses Projekt von Relevanz sein könnten.

Eine dadurch leicht erkennbare Verbesserungsmöglichkeit ist, dass mit der tiefgehenderen Recherche früher begonnen werden sollte als es in diesem Projekt der Fall war. Auch wenn die Strukturierung dieses Projektes begründet und mit Bedacht gewählt wurde, so konnte sie aufgrund des mangelnden Überblickes der Domäne des haptischen Renderns nicht einhaltbar festgelegt werden.
Während der Bearbeitung des Projektes führte die unerwartete Komplexität der Thematik zu einem sehr schwerfälligen Vorankommen, sodass sich während dem Projekt nicht über weitere Punkte (wie der Objektverfolgung) informiert werden konnte, die zu Projektbeginn eingeplant waren.

15 Fazit

Auch wenn es während dem Projekt zu vielen Hürden kam – sowohl kleinere als auch größere – konnte sehr viel zu den unterschiedlichsten Themengebieten gelernt werden.
Wenn die Nennung auch ein wenig trivial erscheint, ist der Bereich, in dem am meisten gelernt werden konnte, das haptische Rendering. Da sich während der Vorbereitungsphase jedoch zudem mit der VR-Entwicklung auseinander gesetzt wurde, konnte auch in diesem Bereich viel gelernt werden. So auch in der Entwicklung in C# – das Sichten der *Github*-Projekte und das geringfügige Anpassen der Skripte konnten bereits Vergessenes wieder in das Gedächtnis rufen.
Nach dem haptischen Rendering konnte zu zwei äußerst relevanten Fähigkeiten sehr viel gelernt werden – der Selbstorganisation und Projektplanung.
Wie bereits in dem vorigen Kapitel angesprochen war die Erstellung und Führung des Projektplanes nicht erfolgreich. Aufgrund des Fehlen eines solchen Planes, konnte die Relevanz eines solchen in einem Einzelprojekt direkt wahrgenommen werden – auch wenn der erste Eindruck ein anderer war.

So konnte sich durch dieses Projekt nicht nur auf die Bachelorarbeit vorbereitet werden, indem wissenschaftlich gearbeitet und recherchiert wurde, sondern auch, indem viel dazu gelernt wurde, wie das Arbeiten innerhalb eines gänzlich unbekannten Feldes optimiert und verbessert werden kann.
In Zusammenhang mit dem Rückblick aus Kapitel 12, den aufgeführten Fortsetzungsmöglichkeiten aus Kapitel 13, dem Prozessassessment aus Kapitel 14 und nicht zuletzt dem Gelernten zum haptischen Rendering in Kapitel 9 & 10, war dieses Projekt aus eigener Sicht – trotz diverser zu bewältigenden Hürden – ein Erfolg.

Glossar

Force-Feedback Bezeichnet eine Rückmeldung von Kraft auf den Nutzer.. 1

haptisches Gerät Frei übersetzt vom englischen Begriff *haptic device*. Ein haptisches Gerät bildet eine haptische Schnittstelle zwischen Mensch und Computer.. 15

hochschwelliger Mechanorezeptor Hochschwellige Mechanorezeptoren (englisch: high-threshold mechanoreceptors) sind die Mechanorezeptoren in der Haut, die auf schädliche, mechanische Reize reagieren.. 33

HTC-Vive Ein in Kooperation der Unternehmen Valve und HTC entstandenes VR-System.. 2, 7–10, 32

Immersion "Im Kontext der virtuellen Realität bezeichnet "Immersion" den Zustand, in dem der Nutzer das Bewusstsein, sich in einer künstlichen Welt zu befinden, verliert. Er lässt sich mit allen seinen Sinnen auf das Erlebnis ein und kann, im Gegensatz zur filmischen "Immersion", mit der virtuellen Realität interagieren." Quelle: thinkwithgoogle.com. 3

lighthouse Ein von der HTC-Vive genutztes System zum Verfolgen der einzelnen Hardware-Komponenten des Systems. Die Basisstationen senden Laserstrahlen aus, die auf die Photosensoren der Hardware trifft. Durch die zeitliche Differenz des Aussenden aus der Basisstation und Erkennen auf der Hardware kann schließlich die Position und Ausrichtung des Gerätes ermittelt werden.. 11

Mechanorezeptor Ein Mechanorezeptor ist eine Sinneszelle, die von außen auf die Haut einwirkende Kräfte wahrnehmen und in elektrische Signale umwandeln kann, die von dem zentralen Nervensystem interpretiert werden können.. 32

niederschwelliger Mechanorezeptor Niederschwellige Mechanorezeptoren (englisch: low-threshold mechanoreceptors) sind die Mechanorezeptoren in der Haut, die auf unschädliche, mechanische Reize reagieren.. 33

Oberflächensensibilität "Als Oberflächensensibilität bezeichnet man die durch Rezeptoren vermittelte Sensibilität der Haut gegenüber äußeren Reizen." Quelle: https://flexikon.doccheck.com/de/Oberflächensensibilität DocCheck Flexikon (abgerufen am 01. Oktober 2019). 1

Open-Source Als Open Source wird eine Software bezeichnet, die bei Auslieferung mit ihrem komplett einsehbarem Quellcode zur Verfügung gestellt wird, sodass diese eingesehen und nach belieben verändert werden kann.. 7

Raytracing Raytracing ist lange Zeit ein genutztes Verfahren des Nicht-Echtzeitrenderns gewesen. Das Verfahren funktioniert dabei so, dass die Wege, die die Lichtstrahlen in der 3D-Szene nehmen würden, nachgegangen werden. Kollision des Lichtstrahles mit einem oder mehreren Objekten führen dabei zu verschiedenen Ergebnissen – wie beispielsweise der Bildung einer leichten Reflektion. Nachdem der Lichtstrahl so lange verfolgt wurde, bis er ins Auge des Betrachters fällt, werden alle Ereignisse, die die Farbe des Lichtstrahles manipuliert haben, miteinander verrechnet, sodass am Ende die Farbe des Pixels äußerst realitätsnah dargestellt werden kann.. 6

Akronyme

AR Augmented Reality. 6

DoF degrees of freedom. 22

ERM Eccentric Rotating Mass (deutsch: exzentrische, rotierende Masse). 20, 21

HD High Definition. 2

HIP haptic interaction point. 22, 27

HMD Head mounted display. 3, 4

HTMR Hochschwellige Mechanorezeptoren. 17

IMU Inertial Measurement Unit (deutsch: inertiale Messeinheit). 12

LRA Linear Resonant Actuator (deutsch: Linearaktuator). 21

LTMR Niederschwellige Mechanorezeptoren. 17, 18

MR Mixed Reality. 6

UHD Ultra High Definition. 2

VR Virtual Reality (deutsch: virtuelle Realität). 1, 6

Literatur

[1] ACM SIGGRAPH. *What is ACM SIGGRAPH? What is the SIGGRAPH conference?* The Association for Computing Machinery. 2019. URL: https://www.siggraph.org/about/what-is-acm-siggraph/ (besucht am 05. 06. 2019).

[2] Tomoya Sasaki u. a. *EMERGING TECHNOLOGIES. Leviopole: Mid-air Haptic Interactions Using Multirotor.* The University of Tokyo. 2018. URL: https://s2018.siggraph.org/conference/conference-overview/emerging-technologies/ (besucht am 05. 06. 2019).

[3] Jotaro Shigeyama u. a. *Transcalibur. Weight Moving VR Controller for Dynamic Rendering of 2D Shape using Haptic Shape Illusion.* report. The University of Tokyo, 2018. URL: http://www.cyber.t.u-tokyo.ac.jp/~jotaro/transcalibur_web/downloads/paper.pdf (besucht am 05. 06. 2019).

[4] Dr. Frank Antwerpes. *Haptische Wahrnehmung.* DocCheck Community GmbH. 2013. URL: https://flexikon.doccheck.com/de/Haptische_Wahrnehmung (besucht am 01. 10. 2019).

[5] Martin Grunwald. *Haptikforschung – Schnittstelle zwischen Allgemeiner und Klinischer Psychologie. Gekürzte Fassung des Vortrags in der Klasse für Naturwissenschaften am 13. Dezember 2007.* 2009. URL: https://leibnizsozietaet.de/wp-content/uploads/2012/11/11_grunwald.pdf (besucht am 30. 09. 2019).

[6] Raluca Sofronia, George Savii und Arjana Davidescu. "Haptic Devices in Engineering and Medicine". In: Mai 2010. DOI: 10.1109/ICCCYB.2010.5491247.

[7] Alex Balladares. *Understanding Haptics for VR.* Hrsg. von Virtual Reality Pop. Mai 2017. URL: https://virtualrealitypop.com/understanding-haptics-for-vr-2844ed2a1b2f (besucht am 06. 06. 2019).

[8] M. Strese und E. Steinbach. "Toward high-fidelity haptic interaction with virtual materials: A robotic material scanning, modelling, and display system". In: *2018 IEEE Haptics Symposium (HAPTICS).* März 2018, S. 247–254. DOI: 10.1109/HAPTICS.2018.8357184.

[9] William R. Sherman und Alan B. Craig. *Understanding Virtual Reality - Interface, Application, and Design.* Revised, Update. San Francisco, Calif: Morgan Kaufmann, 2003. ISBN: 978-1-558-60353-0.

[10] Cagatay Basdogan und Mandayam A. Srinivasan. "Haptic Rendering in Virtual Environments". In: 2001.

[11] Inc. 3D Systems. *Touch X.* 2018. URL: https://de.3dsystems.com/haptics-devices/touch-x (besucht am 07. 06. 2019).

[12] G. Campion und V. Hayward. "Fundamental limits in the rendering of virtual haptic textures". In: *First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics Conference.* März 2005, S. 263–270. DOI: 10.1109/WHC.2005.61.

[13] Chris Schneider. *What is a Haptic Device?* Hrsg. von Geomagic. YouTube. 24. Apr. 2018. URL: https://www.youtube.com/watch?v=C_rHAbJJggM#t=0m39s (besucht am 13. 06. 2019).

[14] Ryan Kittleson. *3D Systems – Touch 3D Stylus.* Youtube, TEquipment.NET. 2015. URL: https://www.youtube.com/watch?v=LWzre6D1P4Y#t=26s (besucht am 02. 10. 2019).

[15] *IEEE World Haptics Conference 2019 - Webseite.* IEEE World Haptics Conference 2019. 2019. URL: http://www.worldhaptics2019.org (besucht am 03. 07. 2019).

[16] *EuroHaptics2020 - Webseite.* Eurohaptics 2020 Conference. 2019. URL: http://eurohaptics2020.org (besucht am 03. 07. 2019).

[17] *IEEE Haptics Symposium 2020 - Webseite*. Haptics Symposium 2020 OrgComm. 2019. URL: http://2020.hapticssymposium.org (besucht am 03. 07. 2019).

[18] *Smart Haptics 2019. A Conference For The Haptics Industry*. Smithers Apex. 2019. URL: https://www.smart-haptics.com (besucht am 03. 07. 2019).

[19] *AsiaHaptics 2018 - Webseite*. AsiaHaptics 2018. 2019. URL: http://asiahaptics.org/ (besucht am 03. 07. 2019).

[20] *Projektplan*. URL: https://gitlab.com/sgeier96/PPSS19Geier/wikis/uploads/4087b748734ff396b106552a59316adf/Projektplan_PPSS19.png (besucht am 15. 06. 2019).

[21] Peter Knoll. *Grafiktablett*. Hrsg. von WEKA MEDIA PUBLISHING GmbH. 2008. URL: https://www.pc-magazin.de/ratgeber/grafiktablett-128986-6660.html (besucht am 02. 11. 2019).

[22] Wacom. *Wacom Intuos CTH-690*. Amazon. URL: https://www.amazon.de/Wacom-CTH-690-Intuos-Medium-Tablet/dp/B0154IIJWI (besucht am 13. 11. 2019).

[23] *Technical Specs*. Nintendo Co., Ltd. 2019. URL: https://www.nintendo.com/switch/tech-specs/#joycon-section (besucht am 13. 11. 2019).

[24] Ben Kuchera. *How the Nintendo Switch's HD rumble makes Tumbleseed feel real*. Hrsg. von Polygon. 1. Mai 2017. URL: https://www.polygon.com/2017/5/1/15499328/tumbleseed-hd-rumble-nintendo-switch (besucht am 14. 11. 2019).

[25] Ltd. Nintendo Co. *1-2-Switch*. 2019. URL: https://www.nintendo.de/Spiele/Nintendo-Switch/1-2-Switch-1173186.html (besucht am 14. 11. 2019).

[26] dekuNukem. *Nintendo_Switch_Reverse_Engineering*. https://github.com/dekuNukem/Nintendo_Switch_Reverse_Engineering. 10. März 2019.

[27] CTCaer. *Joy-Con Toolkit*. https://github.com/ctcaer/jc_toolkit/. 30. Apr. 2018.

[28] Ken Pimentel. *Unreal Studio 4.22 advances real-time photreal rendering*. Epic Games, Inc. 16. Apr. 2019. URL: https://www.unrealengine.com/en-US/blog/unreal-studio-4-22-advances-real-time-photoreal-rendering (besucht am 17. 06. 2019).

[29] *Unreal Engine - Blueprints Visual Scripting*. Epic Games, Inc. URL: https://docs.unrealengine.com/en-US/Engine/Blueprints/index.html (besucht am 17. 06. 2019).

[30] *Unreal Engine for AR, VR & MR*. Epic Games, Inc. URL: https://www.unrealengine.com/en-US/vr (besucht am 17. 06. 2019).

[31] *Unreal Engine - Frequently Asked Questions (FAQ). What platforms are supported?* Epic Games, Inc. URL: https://www.unrealengine.com/en-US/faq (besucht am 17. 06. 2019).

[32] *Unreal Engine - Frequently Asked Questions (FAQ). How much do I have to pay for Unreal Engine 4?* Epic Games, Inc. URL: https://www.unrealengine.com/en-US/faq (besucht am 17. 06. 2019).

[33] *Unity - Products. Grafik-Rendering*. Unity Technologies. URL: https://unity3d.com/de/unity (besucht am 17. 06. 2019).

[34] *Unity - Products. Unity Asset Store*. Unity Technologies. URL: https://unity3d.com/de/unity (besucht am 17. 06. 2019).

[35] *Unity - FAQ: Neues Abbonement. Erhebt Unity Lizenzgebühren?* Unity Technologies. URL: https://unity3d.com/de/unity/faq (besucht am 19. 06. 2019).

[36] *Unity Plus*. Unity Technologies. URL: https://store.unity.com/de/products/unity-plus (besucht am 26. 06. 2019).

[37] *Unity Pro*. Unity Technologies. URL: https://store.unity.com/products/unity-pro (besucht am 26. 06. 2019).

[38] *Unity Personal. Allgemeine FAQ*. Unity Technologies. URL: https://store.unity.com/de/products/unity-personal (besucht am 19. 06. 2019).

[39] *Programmieren in Unity. Welche Programmiersprachen können in Unity verwendet werden?* Unity Technologies. URL: https://unity3d.com/de/programming-in-unity (besucht am 19. 06. 2019).

[40] *Godot - Contact*. URL: https://godotengine.org/contact (besucht am 26. 06. 2019).

[41] *Godot - Features*. URL: https://godotengine.org/features (besucht am 26. 06. 2019).

[42] *Kennen Sie die besten Game-Engines im Jahr 2018? Unity*. Viscircle GmbH. 2018. URL: https://viscircle.de/kennen-sie-besten-game-engines-im-jahr-2018/ (besucht am 28. 06. 2019).

[43] *Kennen Sie die besten Game-Engines im Jahr 2018? Unreal Engine*. Viscircle GmbH. 2018. URL: https://viscircle.de/kennen-sie-besten-game-engines-im-jahr-2018/ (besucht am 28. 06. 2019).

[44] Walter Doberenz und Thomas Gewinnus. *Visual C# 2015 – Grundlagen, Profiwissen und Rezepte* -. Carl Hanser Verlag GmbH Co KG, 2015. ISBN: 978-3-446-44606-9.

[45] *Unity Documentation*. Version 2019.1-002W. Unity Technologies. 2019. (Besucht am 29. 06. 2019).

[46] *Blender - Homepage*. Blender Foundation. URL: https://www.blender.org/ (besucht am 29. 06. 2019).

[47] *Fakten über Linkshändigkeit*. 15. Juli 2015. URL: http://www.linkshaender-fakten.de/die-in-etwa-fast-genaue-anzahl-an-linkshaendern/ (besucht am 01. 07. 2019).

[48] hecomi. *Unity de Windows no desuku toppu gamen o tekusucha toshite hyōji suru puragu in o tsukuttemita*. 4. Dez. 2016. URL: http://tips.hecomi.com/entry/2016/12/04/125641 (besucht am 14. 11. 2019).

[49] hecomi. *uDesktopDuplication*. https://github.com/hecomi/uDesktopDuplication. 20. Sep. 2019.

[50] Looking Glass Factory Inc. *Joycon Lib*. https://github.com/Looking-Glass/JoyconLib. 22. März 2019.

[51] Stefan Geier. *Github Repository – PPSS19Geier*. URL: https://github.com/sgeier96/PPSS19Geier (besucht am 25. 11. 2019).

[52] D. Kwon, T. Yang und Y. Cho. "Mechatronics Technology in Mobile Devices". In: *IEEE Industrial Electronics Magazine* 4.2 (Juni 2010), S. 36–41. DOI: 10.1109/MIE.2010.936763.

[53] K. Salisbury, F. Conti und F. Barbagli. "Haptic rendering: introductory concepts". In: *IEEE Computer Graphics and Applications* 24.2 (März 2004), S. 24–32. DOI: 10.1109/MCG.2004.1274058.

[54] K. Wen, D. Necsulescu und J. Sasiadek. "Haptic force control based on impedance/admittance control aided by visual feedback". In: *Multimedia Tools and Applications* 37.1 (März 2008), S. 39–52. ISSN: 1573-7721. DOI: 10.1007/s11042-007-0172-1. URL: https://doi.org/10.1007/s11042-007-0172-1.

[55] S. J. Lederman und R. L. Klatzky. "Haptic perception: A tutorial". In: *Attention, Perception, & Psychophysics* 71.7 (Okt. 2009), S. 1439–1459. ISSN: 1943-393X. DOI: 10.3758/APP.71.7.1439. URL: https://doi.org/10.3758/APP.71.7.1439.

[56] M.C. Lin und M.A. Otaduy. *Haptic rendering: Foundations, algorithms, and applications*. Jan. 2008, S. 1–603.

[57] Simon Lacey, Christine Campbell und Krish Sathian. “Vision and touch: multiple or multisensory representations of objects?” In: *Perception* 36 10 (2007), S. 1513–21.

[58] Dr. Frank Antwerpes und Thomas Rohner. *Mechanorezeption*. DocCheck Community GmbH. März 2017. URL: https://flexikon.doccheck.com/de/Mechanorezeption (besucht am 04. 10. 2019).

[59] Yann Roudaut u. a. “Touch sense. Functional organization and molecular determinants of mechanosensitive receptors”. In: *Channels (Austin)* 6.4 (Juli 2012), S. 234–245. DOI: 10.4161/chan.22213.

[60] Dr. Frank Antwerpes. *SA-Rezeptor*. DocCheck Community GmbH. März 2017. URL: https://flexikon.doccheck.com/de/SA-Rezeptor (besucht am 05. 10. 2019).

[61] Dr. Frank Antwerpes. *RA-Rezeptor*. DocCheck Community GmbH. März 2017. URL: https://flexikon.doccheck.com/de/RA-Rezeptor (besucht am 05. 10. 2019).

[62] S. Choi und K. J. Kuchenbecker. “Vibrotactile Display: Perception, Technology, and Applications”. In: *Proceedings of the IEEE* 101.9 (Sep. 2013), S. 2093–2104. DOI: 10.1109/JPROC.2012.2221071.

[63] Dr. Frank Antwerpes. *PC-Rezeptor*. DocCheck Community GmbH. März 2017. URL: https://flexikon.doccheck.com/de/PC-Rezeptor (besucht am 05. 10. 2019).

[64] H. Culbertson u. a. “Refined methods for creating realistic haptic virtual textures from tool-mediated contact acceleration data”. In: *2012 IEEE Haptics Symposium (HAPTICS)*. März 2012, S. 385–391. DOI: 10.1109/HAPTIC.2012.6183819.

[65] Masa Ogata. “Magneto-Haptics: Embedding Magnetic Force Feedback for Physical Interactions”. In: Okt. 2018, S. 737–743. DOI: 10.1145/3242587.3242615.

[66] Daniel Spelmezan, Rafael Morales Gonzalez und Sriram Subramanian. “SkinHaptics: Ultrasound focused in the hand creates tactile sensations”. In: Apr. 2016, S. 98–105. DOI: 10.1109/HAPTICS.2016.7463162.

[67] H. Culbertson und K. J. Kuchenbecker. “Importance of Matching Physical Friction, Hardness, and Texture in Creating Realistic Haptic Virtual Surfaces”. In: *IEEE Transactions on Haptics* 10.1 (Jan. 2017), S. 63–74. ISSN: 1939-1412. DOI: 10.1109/TOH.2016.2598751.

[68] M. Azadi und L. Jones. “Identification of vibrotactile patterns: building blocks for tactons”. In: *2013 World Haptics Conference (WHC)*. Apr. 2013, S. 347–352. DOI: 10.1109/WHC.2013.6548433.

[69] R. Zhang, A. J. Boyles und J. J. Abbott. “Six principal modes of vibrotactile display via stylus”. In: *2018 IEEE Haptics Symposium (HAPTICS)*. März 2018, S. 313–318. DOI: 10.1109/HAPTICS.2018.8357194.

[70] J. M. Romano, T. Yoshioka und K. J. Kuchenbecker. “Automatic filter design for synthesis of haptic textures from recorded acceleration data”. In: *2010 IEEE International Conference on Robotics and Automation*. Mai 2010, S. 1815–1821. DOI: 10.1109/ROBOT.2010.5509853.

[71] J. M. Romano und K. J. Kuchenbecker. “Creating Realistic Virtual Textures from Contact Acceleration Data”. In: *IEEE Transactions on Haptics* 5.2 (Apr. 2012), S. 109–119. DOI: 10.1109/TOH.2011.38.

[72] L. A. Jones und A. Singhal. “Perceptual dimensions of vibrotactile actuators”. In: *2018 IEEE Haptics Symposium (HAPTICS)*. März 2018, S. 307–312. DOI: 10.1109/HAPTICS.2018.8357193.

[73] H. Culbertson, J. J. López Delgado und K. J. Kuchenbecker. “One hundred data-driven haptic texture models and open-source methods for rendering on 3D objects”. In: *2014 IEEE Haptics Symposium (HAPTICS)*. Feb. 2014, S. 319–325. DOI: 10.1109/HAPTICS.2014.6775475.

[74] Eve Hoggan, Stephen Brewster und Stephen Brewster. “New Parameters for Tacton Design”. In: *CHI ’07 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’07. San Jose, CA, USA: ACM, 2007, S. 2417–2422. ISBN: 978-1-59593-642-4. DOI: 10.1145/1240866.1241017. URL: http://doi.acm.org/10.1145/1240866.1241017.

[75] Ian R. Summers u. a. “Information from time-varying vibrotactile stimuli”. In: *The Journal of the Acoustical Society of America* 102.6 (1997), S. 3686–3696. DOI: 10.1121/1.420154. eprint: https://doi.org/10.1121/1.420154. URL: https://doi.org/10.1121/1.420154.

[76] A. M. Okamura, J. T. Dennerlein und R. D. Howe. “Vibration feedback models for virtual environments”. In: *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*. Bd. 1. Mai 1998, 674–679 vol.1. DOI: 10.1109/ROBOT.1998.677050.

[77] V. L. Guruswamy, J. Lang und W. Lee. “Modelling of haptic vibration textures with infinite-impulse-response filters”. In: *2009 IEEE International Workshop on Haptic Audio visual Environments and Games*. Nov. 2009, S. 105–110. DOI: 10.1109/HAVE.2009.5356133.

[78] Masataka Niwa u. a. “Vibrotactile apparent movement by DC motors and voice-coil tactors”. In: (Jan. 2004).

[79] Precision Microdrives Limited. *Eccentric Rotating Mass Vibration Motors - ERMs*. URL: https://www.precisionmicrodrives.com/vibration-motors/eccentric-rotating-mass-vibration-motors-erms/ (besucht am 21. 10. 2019).

[80] Precision Microdrives Limited. *Linear Resonant Actuators - LRAs*. URL: https://www.precisionmicrodrives.com/vibration-motors/linear-resonant-actuators-lras/ (besucht am 21. 10. 2019).

[81] *Piezoelektrizität*. URL: https://www.chemie.de/lexikon/Piezoelektrizit%C3%A4t.html (besucht am 21. 10. 2019).

[82] Randy Ellis, Ossama Ismail und M Lipsett. “Design and Evaluation of a High-Performance Haptic Interface”. In: *Robotica* 14 (Mai 1996), S. 321–327. DOI: 10.1017/S0263574700019639.

[83] Alberto Gallace, Hong Z. Tan und Charles Spence. “The Body Surface as a Communication System: The State of the Art after 50 Years”. In: *PRESENCE: Teleoperators and Virtual Environments* 16 (2007), S. 655–676.

[84] Thomas Knott und Torsten Kuhlen. “Geometrically Limited Constraints for Physics-Based Haptic Rendering”. In: *Haptics: Neuroscience, Devices, Modeling, and Applications*. Hrsg. von Malika Auvray und Christian Duriez. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, S. 343–351. ISBN: 978-3-662-44196-1.

[85] S. Shin und S. Choi. “Geometry-based haptic texture modeling and rendering using photometric stereo”. In: *2018 IEEE Haptics Symposium (HAPTICS)*. März 2018, S. 262–269. DOI: 10.1109/HAPTICS.2018.8357186.

[86] S. Yim, S. Jeon und S. Choi. “Data-Driven Haptic Modeling and Rendering of Viscoelastic and Frictional Responses of Deformable Objects”. In: *IEEE Transactions on Haptics* 9.4 (Okt. 2016), S. 548–559. DOI: 10.1109/TOH.2016.2571690.

[87] Miroslav Ďulík und Libor Ladányi. “Surface detection and recognition using infrared light”. In: Mai 2014, S. 159–164. ISBN: 978-1-4799-3721-9. DOI: 10.1109/ELEKTRO.2014.6847893.

[88] T. Aytaç und B. Barshan. “Surface recognition by parametric modeling of infrared intensity signals”. In: *2004 12th European Signal Processing Conference*. Sep. 2004, S. 1107–1110.

[89] Y. Ikei, K. Wakamatsu und S. Fukuda. “Vibratory tactile display of image-based textures”. In: *IEEE Computer Graphics and Applications* 17.6 (Nov. 1997), S. 53–61. DOI: 10.1109/38.626970.

[90] Robert H. LaMotte. “Softness Discrimination With a Tool”. In: *Journal of Neurophysiology* 83.4 (2000). PMID: 10758090, S. 1777–1786. DOI: 10.1152/jn.2000.83.4.1777. eprint: https://doi.org/10.1152/jn.2000.83.4.1777. URL: https://doi.org/10.1152/jn.2000.83.4.1777.

[91] Janitza electronics GmbH. *Transients*. URL: https://www.janitza.com/transients.html (besucht am 30. 10. 2019).

[92] H. Culbertson, J. J. Lopez Delgado und K. J. Kuchenbecker. *The Penn Haptic Texture Toolkit for Modeling, Rendering, and Evaluating Haptic Virtual Textures*. Departmental Papers (MEAM). 2014. URL: https://repository.upenn.edu/meam_papers/299 (besucht am 30. 10. 2019).

[93] H. Culbertson, J. J. Lopez Delgado und K. J. Kuchenbecker. “[D11] The Penn Haptic Texture Toolkit”. In: *2014 IEEE Haptics Symposium (HAPTICS)*. Feb. 2014, S. 1–1. DOI: 10.1109/HAPTICS.2014.6775540.

[94] H.G. Steger. *Maschinenbau für Elektrotechniker*. Teil 2. Vieweg+Teubner Verlag, 2013. ISBN: 9783322927767. URL: https://books.google.de/books?id=0eygBgAAQBAJ.

[95] STMicroelectronics. *LSM6DS3*. URL: https://www.st.com/en/mems-and-sensors/lsm6ds3.html#overview (besucht am 16. 11. 2019).

[96] 3DSystems Inc. *Touch*. URL: https://de.3dsystems.com/haptics-devices/touch (besucht am 16. 11. 2019).

Abbildungsverzeichnis