

Leap Motion in Unity

Spielerisches Lernen von Mathematik umgesetzt in einem uFrame-Projekt

Tobias Schweisfurth
2169629



Hochschule für Angewandte Wissenschaften Hamburg
Fakultät Design, Medien und Information
Department Medientechnik

Prüfer: Matthias Kuhr

Hamburg, 29. 9. 2016

Inhaltsverzeichnis

1 Einleitung	5
2 Produktvision	6
2.1 Setting	6
3 Umsetzung	8
3.1 Unity	8
3.1.1 Menü und Schwierigkeitsstufen	8
3.1.2 Fingererkennung	8
3.2 Entity Component System uFrame	10
4 Probleme	12
4.1 uFrame	12
4.2 Leap Motion	12
5 Erweiterungen	13
5.1 Worldbuilding	13
5.2 Audiovisuelles Feedback	13
5.3 Virtual Reality	13
5.4 Models	14
5.5 Videos und Animationen	14
6 Fazit	15
6.1 Leap Motion	15
6.2 Gedanken zu VR	15
Abbildungsverzeichnis	16

1 Einleitung

Leap Motion bietet viele noch unentdeckte Möglichkeiten im Bereich Virtual und Augmented Reality. Ich persönlich sehe für Leap Motion jedoch keine Zukunft in Virtual Reality außerhalb von Kleinstanwendungen und Experimenten. Viel mehr sollte Leap Motion seine Stärken in Augmented Reality fördern. Leap Motion könnte dafür in erzieherischen und bildenden Anwendungen genutzt werden.

Deswegen handelt dieses Projekt von dem Versuch Bildung in Mathematik in spielerischer Form darzustellen. Speziell geht es dabei um die Zielgruppe der Kleinkinder, welche sich im Vorschulalter zwischen vier und sechs Jahren befinden.

Für das Projekt wird die Leap Motion verwendet. Diese soll Gesten des Spielers entgegennehmen und somit Interaktion ermöglichen.

2 Produktvision

Der Spieler befindet sich auf einem Trainingscamp, auf welchem er gerade als Neuling angekommen ist. Ein Betreuer begleitet ihn durch seine erste Trainingseinheit. Für den Verlauf seiner Mission ist es erforderlich, dass er Mathematik-Aufgaben lösen kann. Dafür werden ihm in unterschiedlichen Schwierigkeitsstufen verschiedene Aufgaben gestellt, die er zu lösen hat. Wenn der Spieler die Lösung gefunden hat, zeigt er sie mit seinen Fingern an.

2.1 Setting

Da es sich um ein Trainingsspiel handelt, ist es für die Atmosphäre erforderlich, dass man sich als Spieler in einer erforschenden Rolle wiederfindet. Der Spieler befindet sich an einem unbekannten Ort, den es zu erforschen gilt.

Deswegen ist er Astronautenlehrling auf einem gerade neu entdeckten Planeten oder im Trainingsquartier auf dem Raumschiff. Hierbei ist der Design Spielraum groß. Es könnte ein eher trockener oder eher feuchter Planet sein und abhängig davon mit mehr oder weniger Flora überseht sein.



Abbildung 2.1: Monolith aus dem Spiel NO MAN'S SKY auf zu erforschendem Planeten

2 Produktvision



Abbildung 2.2: Mögliche Trainingsstation auf dem Raumschiff

Bevor sich der Spieler den forschenden Aufgaben widmen kann, muss sichergestellt sein, dass er richtig ausgebildet ist, wodurch man in die Spielsituation gelangt.

Der Betreuer ist dabei ein weiterer erfahrener Astronaut oder aber ein Roboter mit künstlicher Intelligenz. Dieser würde auch eine game-typische Companion-Rolle im Laufe des Spiels übernehmen. Das Abschließen der Trainingseinheiten ermöglicht das Fortschreiten im Spielgeschehen. Dieses wird jedoch im Prototypen nicht behandelt.

3 Umsetzung

Das Spiel wird in Unity mithilfe des uFrame-Frameworks programmiert. Für die Gesetenerkennung des Spielers wird Leap Motion verwendet. Dabei werden die LeapMotion-CoreAssets und LeapMotion-HandsModule Packages benutzt.

3.1 Unity

Im Spiel sind die Hände des Spielers sichtbar, wenn er sie in den Aufnahmebereich der Leap Motion bewegt. Vor ihm ist eine halb-transparente Tafel aufgebaut, die ihm verschiedene Aufgaben anzeigen wird. Dieses Canvas Objekt ist im Prototypen mit einem festen Abstand an dem HeadMount befestigt. In einer vollständigen Version würde diese Tafel von dem Betreuer oder einem Beamer ausgehen.

In Unity ist ein leeres GameObject angebracht, welches sich um die Game- und Canvas-Logic kümmert. Dort werden Schwierigkeitsstufen gespeichert und das Anzeigen der Aufgaben abhängig von der Schwierigkeitsstufe vorbereitet.

Um das vermeintliche richtige Erkennung einer Lösung zu verhindern, wird durch einen Timer rechts oben auf der Tafel sichergestellt, dass die Geste, bevor sie als gelöst erkannt wird, eine feste Zeit gehalten werden musste.

3.1.1 Menü und Schwierigkeitsstufen

Für das Einstellen der Schwierigkeitsstufe ist ein Menü auf der linken Hand implementiert. Dieses öffnet sich, wenn man den größeren Button auf der Handfläche betätigt, wobei die Handfläche zu dem Spieler zeigen muss.

Bei Betätigung öffnet sich das Menü an den Fingerspitzen der restlichen Finger. Diese können über den Zeigefinger der rechten Hand bedient werden. Hierbei steht jeweils ein Button für eine jeweilige Schwierigkeitsstufe.

3.1.2 Fingererkennung

Wenn dem Spieler eine Aufgabe gestellt wird, kann er diese über das Anzeigen der richtigen Anzahl an Fingern lösen. Hierbei ist noch nicht spezifiziert, dass eine spezielle Geste gezeigt muss (bspw. dass zwei nur durch das Ausstrecken von Zeigefinger und Daumen korrekt erkannt wird); es zählt lediglich das korrekte Ausstrecken der richtigen Anzahl an Finger.

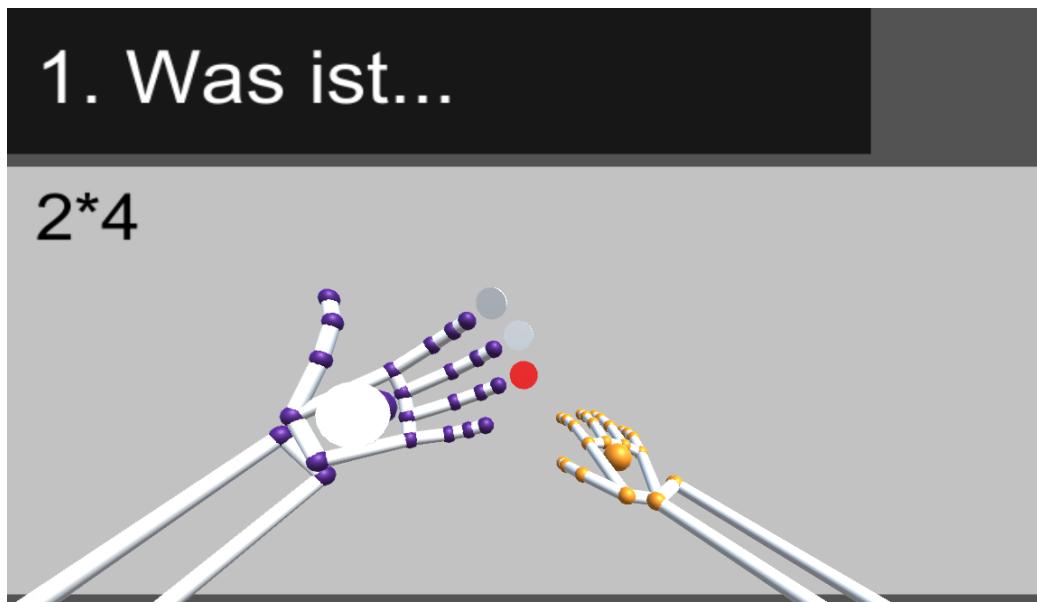


Abbildung 3.1: Das geöffnete Menü auf der linken Hand. Schwierigkeitsstufe Schwer ist ausgewählt

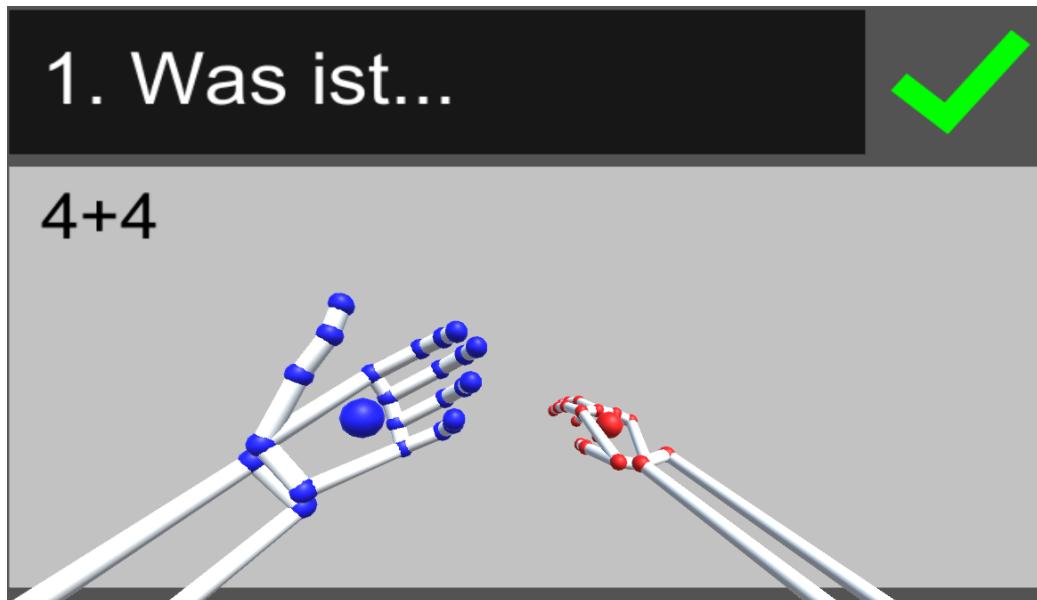


Abbildung 3.2: Richtige Antwort mit den Fingern anzeigen

3 Umsetzung

Für das Erkennen der Fingerstellung wurde das Plugin von Patrick Hilgenstock verwendet. Hierbei wird eine Geste daran erkannt, wie viele Finger gleichzeitig ausgestreckt oder eingezogen sind.

3.2 Entity Component System uFrame

Die Gamelogic ist in uFrame umgesetzt. Für das Projekt wurden effektiv vier Systeme erstellt, die verschiedene Aufgaben übernehmen. PalmSystem erkennt das Zeigen der Handfläche, MenuSystem das Aktivieren und Deaktivieren des Menüs, DifficultySystem das Einstellen der Schwierigkeitsstufe und das NumberDetectionSystem sowohl das Anzeigen der Aufgaben als auch das Erkennen der Lösung und die damit verbundenen Aktionen im Spiel.

Für die GameObjects in Unity wurden verschiedene Components erstellt, die in den meisten Fällen gut zu Gruppen zusammengefasst werden konnten. Im Beispiel der Buttons besteht eine Gruppe immer aus einer Button-Component und einer Schwierigkeits-Component (bspw. Easy-Component). Dadurch kann ein Event wie OnTriggerEnter leichter auf eine spezielle Situation gefiltert werden.

Für den Zeigefinger gibt es eine IndexFingerComponent. Nur der Zeigefinger soll in der Lage sein, das Menü zu bedienen. Dafür wird diese Component auf den Zeigefinger des HeadMountRigs gelegt.

Einzelne Gamelogic, die einmalig oder nicht allzu häufig auftreten sollte, wird über Events geregelt. Dazu gehört unter anderem das Abgeben einer richtigen Antwort, aber auch das Auswählen einer Schwierigkeitsstufe.

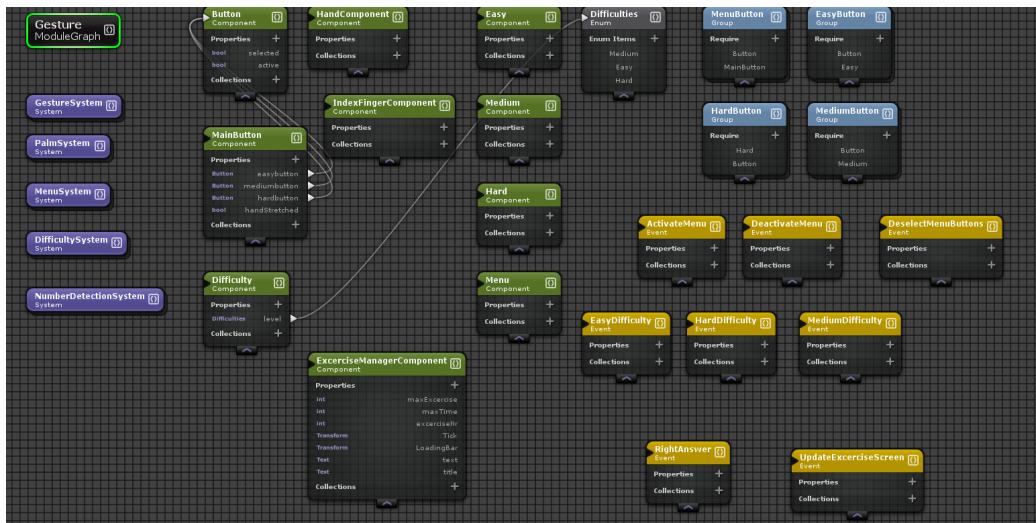


Abbildung 3.3: Die verschiedenen Components, Groups, Systems und Events, die in diesem Modul genutzt wurden

3 Umsetzung

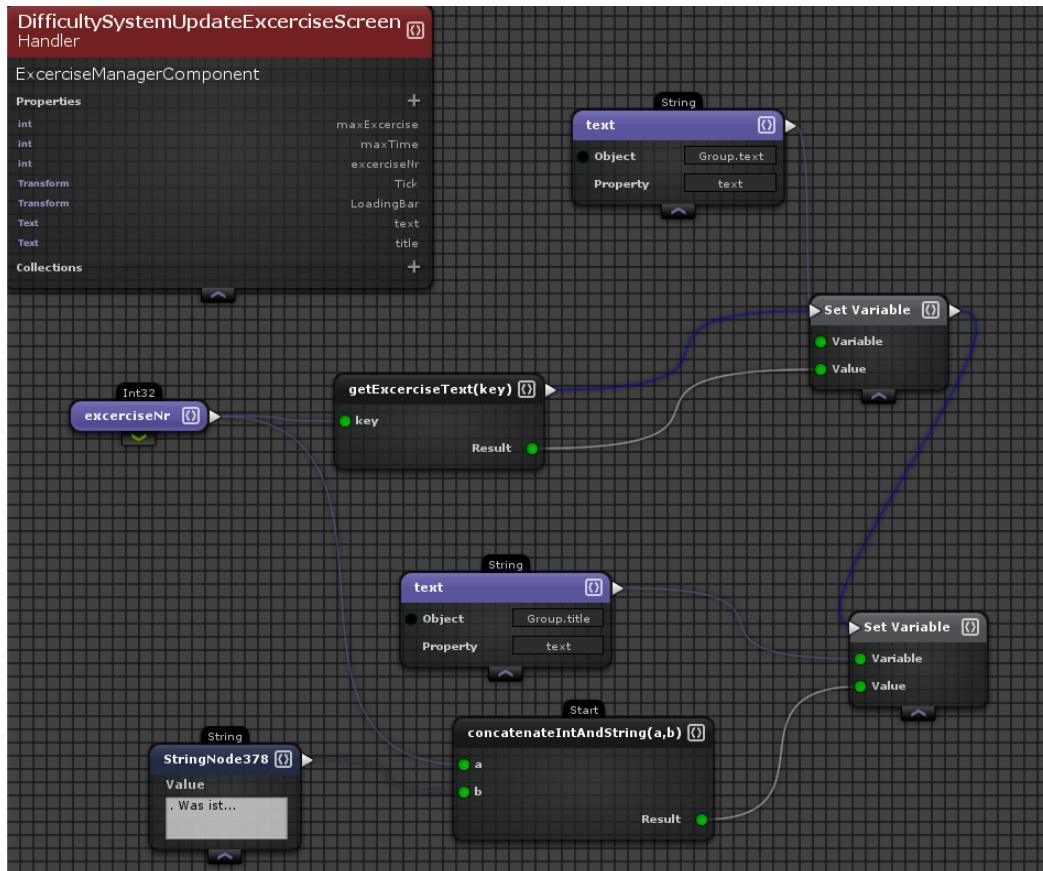


Abbildung 3.4: Beispiel eines Steckmusters in uFrame (Anzeigen der neuen Aufgabe auf der Tafel)

4 Probleme

4.1 uFrame

uFrame hat in seiner doch eher anfänglichen Version nicht ausreichend grundlegende Funktionen implementiert. Dazu zählen nicht nur Unity Funktionen, sondern auch fundamentale logische Operationen. Dies ist vor allem der Weiterleitung der Aktionen durch die Funktionen (Actions) yes() und no() geschuldet. Es wird zwar zum Beispiel geprüft, ob eine boolsche Variable True oder False ist, jedoch erhält man nicht das Ergebnis, sondern lediglich die Möglichkeit basierend auf dem Ergebnis eine Aktion auszuführen. In vielen Fällen brauchte man aber das Ergebnis dieser Abfrage dringender, als einfach nur eine darauf anschließende Funktion.

Da uFrame sein Design auch grundsätzlich darauf ausgelegt hat, dass bei Bedingungsabfragen eine Aktion folgt, mussten für das Projekt einige Funktionen mit minimaler Änderung neu geschrieben werden.

Auch einzelne Bugs wie das zufällige Verstellen des Start-Nodes und das Löschen von einzelnen Grafen erschwerten die Arbeit an dem Projekt.

Mir war es auch im Laufe des Projekts nicht möglich gut und effizient Timer basierte Aufgaben zu erledigen. Der Luxus von Coroutinen blieb mir vorenthalten, weswegen einige Aspekte nicht umgesetzt werden konnten.

4.2 Leap Motion

Eines der größten Defizite der Leap Motion ist nach wie vor die Ungenauigkeit. Durchaus komplexe Gesten können immer mit einer verhältnismäßig großen Chance nicht erkannt werden, welches für den Nutzer schnell frustrierend wirkt, da er diese Bewegungen solange wiederholen muss, bis es funktioniert. Diese entsteht größtenteils durch das gegenseitige Verdecken der Hände, wodurch die Kamera Fokus auf die verdeckte Hand verliert.

Dennoch ist Leap Motion genau genug um einfache Gesten zu machen. Beim Setup muss viel auf Blickwinkel und Entfernung geachtet werden, da dieses sehr starken Einfluss auf die Genauigkeit hat.

5 Erweiterungen

Da es sich bei dem Projekt nur um einen Prototypen handelt, sind einige Änderungen durchaus denkbar.

5.1 Worldbuilding

Im Prototypen steht man nur vor einer Tafel. Für ein vollwertiges Spiel ist es notwendig, die Welt rund um das Geschehen genauer zu beleuchten. Dabei gerät vor allem das Design des Planeten, auf welchem man gelandet ist, im Fokus. Kommt hinzu, dass man sich in der Welt bewegen kann, anstatt starr an einer Position zu verharren, wird Interaktion mit der Umwelt wichtiger.

Man könnte sich durchaus unterschiedliche Planeten vorstellen, die unterschiedliche Schwierigkeitsstufen der Aufgaben durch Design und Farbschemata repräsentieren. Beim Auswählen einer anderen Schwierigkeitsstufe wird dann zu dem jeweiligen Planeten gereist.

5.2 Audiovisuelles Feedback

Mit Leap Motion kann zwar Berührung und Bewegungen der Hand simuliert werden, jedoch fehlt im Gegensatz zur Realität das fühlbare Feedback. Der Tastsinn muss über genügend audiovisuelles Feedback kompensiert werden.

Dazu gehören Tastentöne beim Drücken der Buttons, aber auch bestätigende oder verneinende Töne bei jeweils richtiger oder falscher Antwort.

Bisher färben sich die Buttons farbig, wenn sie ausgewählt sind. Bei besserem Design der Buttons, sollte das auch beibehalten werden. Ein typisches "button-pressed"-Feeling reicht da aus.

5.3 Virtual Reality

Mit dem HeadMountRig ist die Steuerung zwar schon auf Virtual Reality ausgelegt, der Prototyp an sich jedoch noch nicht. Dafür muss noch die Bewegung des Kopfes in der Welt implementiert werden. Damit ist es dann auch besser möglich, dass man sich auf dem Planeten begrenzt bewegen kann. Andernfalls säße oder stände man ähnlich wie in einem Klassenzimmer an einer Position.

5.4 Models

Die angesprochenen Punkte Worldbuilding und Virtual Reality müssen vor allem durch bessere Models ersetzt werden. Buttons sollte futuristischer Aussehen, ein Betreuer der die Tafel projiziert könnte als fahrender Roboter dargestellt werden und die Oberfläche des Planeten von bestimmten Gegenständen und/oder Pflanzen überwuchert sein.



Abbildung 5.1: Mit besseren Models, einem Betreuer und einer holographischen Tafel könnte das Projekt diese Form annehmen

Weiterhin würde die Immersion enorm verstärkt werden, wenn das HeadMountRig nicht nur das Skellet darstellt, sondern von einer Haut oder im Falle eines Astronauten mit einem Handschuh/Raumanzug überzogen werden würde.

5.5 Videos und Animationen

Das Spiel - speziell der Betreuer im Spiel - sollte am Anfang des Spiels kurz erklären, wie das Zählen mit den Fingern funktioniert. Dafür könnte eine kleine Animation gezeigt werden, die wieder abrufbar unter einem Menüpunkt "Tips" wird.

Andere Aspekte im Spiel, die durchaus animiert werden könnten, sind Geräte in der Welt. Text sollte auf der Tafel eingefahren werden, sodass man den Eindruck erweckt, dass die Aufgabe gerade geschrieben wird (im Schreibmaschinen-Stil). Außerdem könnte sich die holographische Tafel ausklappen, indem erst der Tafel-"Kopf" erscheint, woraus der Tafel-"Körper" mit den Aufgaben ausfährt.

6 Fazit

6.1 Leap Motion

Die Einfachheit in Programmierung und Bedienung ist Leap Motion hoch anzurechnen. Neben seinen offensichtlichen Problemen und meiner Meinung nach Schwächen in VR, ist es trotzdem für jeden Nutzer intuitiv. Hände weiß jeder sofort zu bedienen und wenn Knöpfe erscheinen, sind die meisten Menschen schon so stark für Buttons sensibilisiert, dass sie wissen werden, was sie damit zu tun haben.

Bei Kinderspielen ist dies vielleicht nicht sofort der Fall, aber ich bin überzeugt, dass die spielerische, experimentierfreudige Natur das ausgleicht.

In Retrospektive fällt auf, dass Menüs an der Hand grundsätzlich eher für Aspekte genutzt werden sollten, die getriggert werden. Feste Einstellungen wie Options, Schwierigkeitsstufen sollten in einem festen Menü ausgelagert werden, welches nicht versehentlich bedient werden kann. Für das Hand-Menü musste extra ein Main-Button eingebaut werden, sodass die einzelnen Auswahloptionen nicht versehentlich getriggert wurden.

6.2 Gedanken zu VR

Einer der großen Pluspunkte von Virtual Reality wird neben der Spieleindustrie der Virtual Classroom oder Virtual Meeting Room sein. Menschen, die zu weit voneinander entfernt sind, können in den Unterricht gehen oder ein Meeting antreten ohne große Reisen bestreiten zu müssen.

Virtual Classroom hat demnach den Vorteil, dass man einen Betreuer programmieren kann, der spielerisch Aufgaben durchgeht, andererseits aber auch einfach den richtigen Lehrer hinzuschalten, der dann persönlich mit dem Schüler ins Gespräch kommt.

Lernspiele sind über Virtual Reality durchaus möglich und Gamification trägt meiner Meinung nach einen positiven Beitrag zum Lernen hinzu; es muss in Zukunft nur noch die Verbindung untereinander aufgestellt werden können. Wie in einer Art MMORPG loggen sich mehrere Spieler oder Nutzer ein und treffen sich online.

Abbildungsverzeichnis

2.1	Monolith aus dem Spiel NO MAN'S SKY auf zu erforschendem Planeten	6
2.2	Mögliche Trainingsstation auf dem Raumschiff	7
3.1	Das geöffnete Menü auf der linken Hand. Schwierigkeitsstufe Schwer ist ausgewählt	9
3.2	Richtige Antwort mit den Fingern anzeigen	9
3.3	Die verschiedenen Components, Groups, Systems und Events, die in diesem Modul genutzt wurden	10
3.4	Beispiel eines Steckmusters in uFrame (Anzeigen der neuen Aufgabe auf der Tafel)	11
5.1	Mit besseren Models, einem Betreuer und einer holographischen Tafel könnte das Projekt diese Form annehmen	14