

*Entwicklung und Analyse gestenbasierter
Steuersysteme von digitalen Musikinstrumenten
in einer Augmented Reality Anwendung*

Abschlussarbeit
zur Erlangung des akademischen Grades
Bachelor of Science (B.Sc.)

Eingereicht an der Hochschule für Technik und Wirtschaft Berlin
Fachbereich 4: Informatik, Kommunikation und Wirtschaft
Studiengang: Angewandte Informatik

1. Prüfer: Herr Prof. Dr.-Ing. Thomas Jung
2. Prüfer: Herr Prof. Dr. Burkhard Messer

Eingereicht von Nicolai Schiele (Matrikelnr. 556480)

Berlin, den 11.03.2019

Inhaltsverzeichnis

1 Einleitung.....	1
1.1 Zielsetzung.....	1
1.2 Gliederung.....	2
2 Grundlagen.....	3
2.1 Digitale Musikinstrumente.....	3
2.2 Augmented Reality.....	4
2.3 Usability	5
2.3.1 UI Prototyping.....	5
2.3.2 Usability Test.....	6
2.3.3 Iterative Tests.....	7
2.3.4 Datenerhebung in der Usability.....	8
2.3.4.1 Strukturiertes Interview.....	10
2.3.4.2 Fragebogen.....	10
2.3.4.3 aktives teilnehmendes Beobachten.....	10
3 Analyse.....	11
3.1 Entwicklung der Musikinstrumente.....	11
3.1.1 Instrumenten-inspirierte Benutzerschnittstelle.....	11
3.1.2 Erweitertes Musikinstrument.....	12
3.1.3 Alternative Benutzerschnittstelle.....	12
3.1.4 Fazit.....	12
3.2 Theoretische Analyse.....	13
3.2.1 Theoretische Szenarien.....	13
3.2.1.1 Use Case "Keypad".....	13
3.2.1.2 Use Case "Sequencer".....	13
3.2.1.3 Use Case "Drone Generator".....	14
3.2.1.4 Fazit.....	14
3.2.2 Prototyping.....	14
3.2.2.1 3D Prototypen der Instrumente.....	14
3.2.2.2 Erhebung der Testdaten.....	20
3.2.2.3 Auswertung der Daten.....	21
3.2.3 Anforderungsanalyse	25
3.2.3.1 Funktionale Anforderungen.....	25
3.2.3.2 Nicht-Funktionale Anforderungen.....	28
3.3 Technische Analyse	29
3.3.1 Verwendete Hard-, Software	29
3.3.1.1 Hololens.....	29
3.3.1.2 Unity.....	30
3.3.1.3 MixedReality Toolkit.....	30
3.3.1.4 C#.....	31
3.3.1.5 Max/MSP.....	31
3.3.1.6 Open Sound Control.....	32
3.3.2 Machbarkeitsstudie.....	32
3.3.2.1 Unity.....	32

3.3.2.2 MixedRealityToolkit.....	33
3.3.2.3 Unity build für Hololens-Anwendungen.....	33
3.3.2.4 Hololens Emulator.....	33
3.3.2.5 Max/MSP.....	34
3.3.2.6 Verbindung zwischen Unity und Max/MSP.....	34
3.4 Fazit.....	34
4 Design.....	35
4.1 Komponenten.....	35
4.1.1 User Interface.....	36
4.1.2 Klangerzeugung.....	37
4.2 Instrumente.....	41
4.2.1 Keypad.....	41
4.2.2 Sequencer.....	42
4.2.3 Drone Generator.....	43
4.3 Kommunikation.....	44
4.3.3 Befehlsstruktur.....	44
5 Implementierung.....	45
5.1 Technische Umsetzung	45
5.1.1 User-Interface.....	45
5.1.1.1 Steuerung.....	45
5.1.1.2 Hololens.....	48
5.1.1.3 Netzwerkverbindung.....	48
5.1.2 Klangerzeugung.....	49
5.1.2.3 Datenfluss.....	49
5.1.2.1 Datenanpassung.....	49
5.1.2.2 Synthesizer.....	49
5.1.3 Fremdcode.....	50
5.2 Evaluation.....	51
5.2.1 Anforderungen.....	51
5.2 Abschluss der Evaluation.....	54
6 Erste Testphase.....	55
6.1 Aufbau der Tests.....	55
6.2 Ablauf der Tests.....	56
6.3 Auswertung der Testdaten.....	58
7 Anpassung anhand der Testergebnisse.....	61
7.1 Konzeption der Anpassungen.....	61
7.1.1 Gestensteuerung	61
7.1.2 Keypad.....	62
7.1.3 Sequencer.....	64
7.1.4 Drone Generator.....	65
7.2 Implementierung der Anpassungen.....	66
7.2.1 Gestensteuerung.....	66
7.2.2 Keypad.....	66
7.2.3 Sequencer.....	69
7.2.4 Drone Generator.....	69
8 Zweite Testphase.....	71

8.1 Aufbau der Tests.....	71
8.2 Ablauf der Tests.....	71
8.3 Auswertung der Testdaten.....	72
8.4 Fazit.....	75
9 Fazit.....	76
9.1 Zusammenfassung	76
9.2 Ausblick.....	77
Glossar.....	78
Abbildungsverzeichnis.....	79
Tabellenverzeichnis.....	80
Quellenverzeichnis.....	81

1 Einleitung

Die Digitalisierung hat innerhalb der letzten Jahrzehnte in so ziemlich jedem Aspekt des menschlichen Daseins Einzug gefunden. Auch in der Musikbranche sind digitale Musikanstrumente heutzutage allgegenwärtig und nicht mehr weg zu denken. Sie ermöglichen die Erzeugung komplett neuer Klänge und Steuerkonzepte, die von analogen Instrumenten nicht umsetzbar sind und haben die Art wie Musik produziert wird revolutioniert. Seit dem Aufkommen der ersten digitalen Synthesizer, wie dem Yamaha DX7 in 1983¹ ist die Musikbranche stetig bemüht sich neuesten Technologien zu bedienen, um die Qualität der Klangerzeugung aber auch die Bedienung neuer Instrumente immer weiter zu entwickeln. Gerade im Amateurbereich führen diese Entwicklungen dazu, dass digitale Musikanstrumente mittlerweile nicht mehr nur auf klassischen Computersystemen existieren, sondern bereits als Apps in Tablets oder auch auf Spielekonsolen zu finden sind. Durch das Aufkommen neuer Mixed Reality Systemen, die in der neuesten Zeit auch in der breiten Öffentlichkeit vermehrt an Aufmerksamkeit gewonnen haben, ist es also nur noch eine Frage der Zeit bis kommerzielle digitale Musikanstrumente auch auf diesen umgesetzt werden. Da die Steuerungen von digitalen Musikanstrumenten heute nicht mehr nur über klassische MIDI Controller realisiert werden sondern sich stetig mit neu aufkommenden Technologien verändern, ist es also erforderlich neue Konzepte der Bedienung von Mixed Reality Audiosystemen zu erforschen, entwickeln und analysieren.

1.1 Zielsetzung

Ziel dieser Arbeit ist es, digitale Musikanstrumente für die neu aufkommende Technologie der Augmented Reality umzusetzen und diese auf ihre Bedienkonzepte und Bedienbarkeit zu überprüfen. Dazu werden zunächst die unterschiedlichen Arten von digitalen Musikanstrumenten analysiert und deren Bedienkonzepte betrachtet. Die für die Entwicklung der Instrumente nötigen Technologien müssen untersucht werden um festzustellen, ob ein solches System umsetzbar ist. Hierfür werden verschiedene Hardware- und Softwarekomponenten, die für ein solches System in Frage kommen, betrachtet.

Anschließend werden Konzepte der Usability verwendet, um Bedienkonzepte schon in der

¹ VintageSynth (2019)

Planungsphase anhand von potentiellen NutzerInnen zu evaluieren. Hierbei werden auch mögliche Interaktionen mit dem System identifiziert. Um digitale Musikinstrumente allgemein auf ihre Bedienbarkeit überprüfen zu können, werden mehrere unterschiedliche Instrumente mit unterschiedlichen Bedienkonzepten entworfen. Die entwickelten Konzepte werden daraufhin mithilfe von iterativen Entwicklungsverfahren realisiert und mit Methoden der Usability immer wieder an ProbandInnen getestet um so eine Verbesserung in der Bedienbarkeit der Instrumente zu erzeugen. Hierbei wird eine prototypische Anwendung realisiert, um die Umsetzbarkeit des Systems zu belegen.

1.2 Gliederung

Die Arbeit gliedert sich in neun Abschnitte, wobei Das erste Kapitel die Einleitung darstellt. Hierbei soll ein kurzer Überblick über den gesamten Inhalt und Ablauf der Arbeit gegeben werden.

Das zweite Kapitel stellt die Grundlagen dar. Hier werden, die für das Verständnis nötigen Begriffe und Konzepte erklärt.

Im dritten Kapitel findet eine Analyse der Zielsetzung statt. Die theoretische Analyse dient dazu die funktionale und nicht-funktionale Anforderungen an das System zu extrahieren. In der technischen Analyse werden die technischen Anforderungen an das System untersucht.

anhand der identifizierten Anforderungen in Kapitel drei werden in Kapitel vier verschiedene Designalternativen erwägt und ein realisierbares Design entworfen.

Kapitel fünf beschreibt die Realisierung des Systems. Hier wird die erste Iteration des entwickelten Prototyps vorgestellt.

Im sechsten Kapitel findet die erste Testphase statt. Hier werden ProbandInnen herangezogen um einige, mit Methoden der Usability entwickelten Tests zu absolvieren. Diese Tests sollen im folgenden Kapitel dazu verwendet werden die Benutzbarkeit des Systems zu verbessern.

Kapitel 7 beschreibt die Anpassung des Systems anhand der Ergebnisse aus Kapitel sechs. Hier wird der Prototyp abgeschlossen.

Kapitel 8 stellt die zweite Testphase dar. Durch diese Tests sollen mögliche Verbesserungen des Systems aufgezeigt werden.

In Kapitel 9 wird ein Fazit gezogen. Hier werden die Ergebnisse der Arbeit Reflektiert und ein möglicher Ausblick auf Erweiterungen gegeben.

2 Grundlagen

Dieses Kapitel soll ein Grundwissen über digitale Musikanstrumente, Augmented Reality, Usability, sowie die genutzte Hard- und Software vermitteln. Es wird umfangreich vor allem auf die in der Arbeit genutzten Komponenten der einzelnen Bereiche eingegangen.

2.1 Digitale Musikanstrumente

Digitale Musikanstrumente werden von Wanderley und Depalle (2004) als computerbasierte Systeme beschrieben, die von Menschen in Echtzeit zur Generierung von Tönen verwendet werden können. Hierzu ist immer eine Schnittstelle zwischen Mensch und Maschine nötig, welche im Folgenden als Interface bezeichnet wird. Interfaces sind heute meist Hardware-Komponenten die mit Kommunikationsprotokollen, wie dem MIDI-Standard, eine Verständigung der beiden Akteure ermöglichen.

Miranda und Wanderley (2006) unterscheiden digitale Musikanstrumente vor allem nach der Art ihrer Bedienung und legen hierfür vier grundlegende Bereiche fest.

Instrumenten-simulierende Controller wie beispielsweise ein E-Piano, welches durch gewichtete Hammermechaniken das Gefühl eines akustischen Klaviers simulieren soll. *Instrumenten-inspirierte Benutzerschnittstellen*, welche grundlegende Steuerungen und Konzepte von akustischen Instrumenten übernehmen, diese aber meist um Steuerelemente erweitern, die ihre akustischen Vorbilder nicht aufweisen. Beispielsweise Max Mathews sequential Drum, das wie eine akustische Trommel Klangerzeugung durch Anschläge mit Stöcken ermöglicht, darüber hinaus aber die auftreffende Position eines Stockes ermitteln kann und die X und Y Achsen Werte zurückliefert, womit wiederum verschiedene Parameter gesteuert werden können.² *Erweiterte Musikanstrumente*, welche nach wie vor akustische Instrumente als Grundlage nehmen, deren Funktionen aber weit von ihren Vorbildern abweichen können. Als Beispiel kann hier das ROLI Seaboard genannt werden, welches zwar ein klassisches Klaviertastenlayout aufweist, in der Spielweise aber kaum noch an akustischen Klavier erinnert, da sich hier über die Längsachse der Tasten frei zuweisbare Parameter verändern lassen, was ein völlig neues Spielgefühl erzeugt.

Als vierten Bereich definieren Miranda und Wanderley *alternative Benutzerschnittstellen*, welche den gesamten übrigen Bereich von digitalen Musikanstrumenten beschreiben. Diese Instrumente können sehr abstrakte Formen annehmen und Nutzungskonzepte

² Nunzio (2013)

realisieren, die in der Welt der akustischen Instrumente undenkbar wären. Als Beispiel lässt sich hier der Reactable nennen. Ein Instrument, das durch Platzieren von Blöcken auf einem interaktiven Display einen virtuellen Modularen Synthesizer anspricht, welcher daraufhin Klänge und Rhythmen erzeugt. Mit neu aufkommenden Technologien wächst vor allem der Bereich der alternativen Benutzerschnittstellen immer weiter und verschiedene Experimental-musiker erarbeiten immer abstraktere Konzepte. Hierfür werden oft neue Technologien herangezogen, sodass viele der Instrumente als rein konzeptionelle Prototypen entwickelt werden.

2.2 Augmented Reality

Wolfgang Broll (2013) bezeichnet Augmented Reality als die Verbindung zwischen der realen Umgebung mit virtuellen Objekten. Wichtig sei hierbei vor allem eine kontinuierliche Anpassung der virtuellen Objekte an die Umgebung, da beispielsweise durch eine Veränderung des Standortes der Nutzenden Position und Perspektive der virtuellen Objekte an den Blickwinkel anpassen sollen. Broll unterteilt den Prozess der Augmented Reality in fünf Schritte. *Videoaufnahme*, bei der die Umgebung mit einer Kamera aufgezeichnet wird. Das *Tracking* der Blickwinkelposition und -Lage durch Erfassung verschiedener Sensordaten (meist Inertialsensoren, Gyrosensoren und Magnetometern) und Übertragung der dadurch erhaltenen Transformation in ein virtuelles Koordinatensystem. Als nächstes erfolgt eine geometrische *Registrierung*, bei der die virtuellen Objekte auf Basis der Positions- und Lageabschätzung so in Relation mit der realen Umgebung gesetzt werden, dass sie fest verankert erscheinen. Bei der *Darstellung* können virtuelle Inhalte anhand der durch die geometrische Registrierung erzeugten Transformation und der Blickwinkelposition nun perspektivisch korrekt angezeigt werden³. Hierbei können zwei grundsätzliche Formen der Darstellung unterschieden werden. Systeme, bei denen die Realität abgefilmt und auf einem Bildschirm zusammen mit den augmentierten Objekten dargestellt wird und Geräte, bei denen die direkte Realität mithilfe von verschiedenen Techniken um virtuelle Objekte erweitert wird. Hierbei kommen beispielsweise halbtransparente Spiegel, optische Prismen oder direkte retinale Projektion zum Einsatz.

3 Broll et al. (2013): 242

2.3 Usability

Usability (dt. Benutzbarkeit) ist ein sehr vieldeutiger Begriff, doch beschreibt im wesentlichen meist die Qualität von Benutzeroberflächen⁴. Gerade bei konventioneller Software, wie Webseiten oder Anwendungsprogrammen die vorwiegend auf Monitoren angezeigt und mit Maus und Tastatur bedient werden, sind die Anordnung von Bedienelementen und die Anzahl nötiger Klicks oft Qualitätskriterien, die eine Messbarkeit der Usability ermöglichen.

Der Begriff Usability und die damit einhergehenden Methoden zur Qualitätsmessung einer Bedienbarkeit sind jedoch weitgreifender. Die ISO-Norm 9241-11 beschreibt Usability wie folgt:

„das Ausmaß, in dem ein Produkt durch bestimmte Benutzer in einem bestimmten Nutzungskontext genutzt werden kann, um bestimmte Ziele effektiv, effizient und zufriedenstellend zu erreichen.“

DIN EN ISO 9241-11, 1998

Richter und Flückinger (2010): 4

Folglich sind Nutzungskontext und Art der BenutzerInnen wichtige Faktoren, ohne die eine Software nicht auf Ihre Usability geprüft werden kann. Bei der Usability wird oft von den 7±2 wichtigsten Usability Methoden⁵ gesprochen. Im Folgenden werden aber ausschließlich die für die Analyse dieses Systems wichtigen Methoden der Usability beschrieben.

2.3.1 UI Prototyping

Noch vor der eigentlichen Entwicklung eines Systems ist es oft hilfreich, die Benutzerschnittstelle einer Software mithilfe von Prototypen zu erarbeiten. Dies dient nicht nur dazu die Bedürfnisse der Nutzenden bei der Bedienung zu verstehen um sie direkt in die Konzeptionierung mit aufnehmen zu können, es hilft auch den EntwicklerInnen, Probleme bei der Entwicklung frühzeitig zu erkennen.

Prototypen⁶ werden häufig als so genannte Mock-ups⁷ realisiert. Hierbei kann die Nutzeroberfläche auf verschiedene Arten umgesetzt werden (Paper Prototype,

4 Richter & Flückinger (2010): 3

5 Malinowski, S. (2019)

6 Anm.: Vereinfachte Abbildung des Zielsystems

7 Anm.: Kopie eines Systems mit eingeschränktem Funktionsumfang

Grafikprogramme, etc.). Meistens enthält ein UI Mock-up lediglich die rudimentären Steuerfunktionen der grafischen Oberfläche, ohne die eigentliche Funktionen der Software auszuführen, da diese zum Zeitpunkt des Prototypings meist noch gar nicht entwickelt wurden. Eine Nutzung solcher Prototypen durch Testpersonen hilft vor allem das allgemeine Layout der Software, die Struktur von Informationen, das Funktionsverhalten von Bedienelementen, das Design der Software und die Verständlichkeit des Dargestellten zu testen, aber kann je nach Aufbau des Mock-ups jeden erdenklichen Aspekt einer Benutzerschnittstelle prüfen.

2.3.2 Usability Test

Grundsätzlich lassen sich Usability-Tests in zwei Kategorien einteilen. Formative und summative Evaluation. Während bei der formativen Evaluation die Verbesserung eines zu prüfenden Systems im Vordergrund steht, wird bei der summatischen Evaluation das gesamte Produkt zusammenfassend geprüft, um eine Einschätzung der allgemeinen Qualität zu erhalten⁸.

Usability-Tests bestehen immer aus einer Reihe von Aufgaben, die von Testpersonen mithilfe des zu prüfenden Systems abgearbeitet werden sollen. Das zu prüfende System sollte hierbei den Eindruck schaffen, vollwertig lauffähig zu sein selbst wenn nur einzelne spezifische Teile geprüft werden. Es ist außerdem darauf zu achten, dass die Tests realistische Anwendungsfälle darstellen und für die Testpersonen lösbar, aber dennoch nicht zu einfach sind. Bei der Aufgabenstellung ist eine zu technisch detaillierte Formulierung zu vermeiden. Beispielsweise ist "Suchen Sie nach dem günstigsten Produkt..." besser als "Stellen Sie die Sortierkriterien auf 'Preis aufsteigend'..."⁹.

Wichtig sei außerdem, dass die Testpersonen möglichst aus der Benutzergruppe des zu prüfenden Systems sind, sie also zu späteren potentiellen NutzerInnen gehören und ein dementsprechendes Fachwissen mitbringen und nicht erst eingelernt werden müssen. Dennoch kann in sehr spezifischen Anwendungsfällen eine kurze Einführung in das System sinnvoll sein. Die Anzahl der Testpersonen hängt zwar auch von den Zielen der Tests ab, in der Regel sind zwischen fünf und sieben Tester aber ausreichend.¹⁰ Unter Optimalbedingungen werden Usability-Tests in extra dafür eingerichteten Testlaboren durchgeführt. Wichtig sei hierbei vor allem, dass die Testpersonen ihre Aufgaben ohne Einflüsse der EntwicklerInnen in einem separaten Raum absolvieren können, diese dennoch jederzeit Sicht auf die Testpersonen haben und den Ablauf des Tests genau

8 Richter & Flückinger (2010): 61

9 Richter & Flückinger (2010): 62

10 Richter & Flückinger (2010): 62

verfolgen und protokollieren können. Nach Ablauf eines Tests gibt es meist eine Nachbesprechung, in der Testende und Entwickler gemeinsam mögliche Verbesserungen in einem Interview festhalten. Dieses ergibt zusammen mit den Ergebnissen des Tests den Testbericht.

Als Alternative zum Usability-Test wird oft der nicht so formale Usability-Walkthrough verwendet. Testpersonen werden hierbei von TestleiterInnen begleitet, welche die Absolvierung der Aufgaben beobachten gleichzeitig aber schon Fragen stellen und den Testpersonen bei Bedarf helfen können. Gerade im frühen Stadium eines Prototypen eignet sich diese Variante besonders, da Testende selbst mit unfertigen Prototypen arbeiten können, da sie bei Unklarheiten jederzeit auf die Hilfe der TestleiterInnen zurückgreifen können.

Usability-Tests sind grundsätzlich sehr zeitaufwändig. Die Erarbeitung der Testaufgaben, die Vorbereitung des Prototypen, das Anwerben der Testpersonen, die eigentliche Durchführung und die Nachbereitung und Erstellung eines Testberichts sind nicht zu unterschätzende Aufgaben für die in der Entwicklung einer Software oftmals die Zeit fehlt.¹¹

2.3.3 Iterative Tests

Iteratives Testen beinhaltet das Wiederholen bestimmter Schritte, die im Folgenden erläutert werden. Für ein verbessertes Verständnis werden diese Schritte hier stark vereinfacht beschrieben und enthalten normalerweise mehrere Teilschritte. Nach der (1) *Entwicklung eines lauffähigen Systems* steht ein (2) *Usability Test* an. Dieser kann im Laufe des Iterativen Testens variieren und von Usability-Walkthroughs der allgemeinen Steuerung, bis hin zu formalen Usability Tests einzelner Teilkomponenten des Systems reichen. Anschließend steht immer eine (3) *Auswertung der Tests* an, bei der entschieden wird, wie anschließend vorgegangen wird. Ist eine weitere (4) *Anpassung des Systems* nötig, folgt daraufhin ein erneuter Usability-Test und die Iteration beginnt erneut von vorne. Abgeschlossen wird das Iterative Testen durch eine Letzte Auswertung, nach der beschlossen wird, dass keine weiteren Anpassungen erfolgen, womit in den meisten Fällen ein (5) *Ende der Entwicklung* erreicht ist.¹² (siehe Abb. 1)

11 Richter & Flückinger (2010): 64

12 Coughlan (2016)

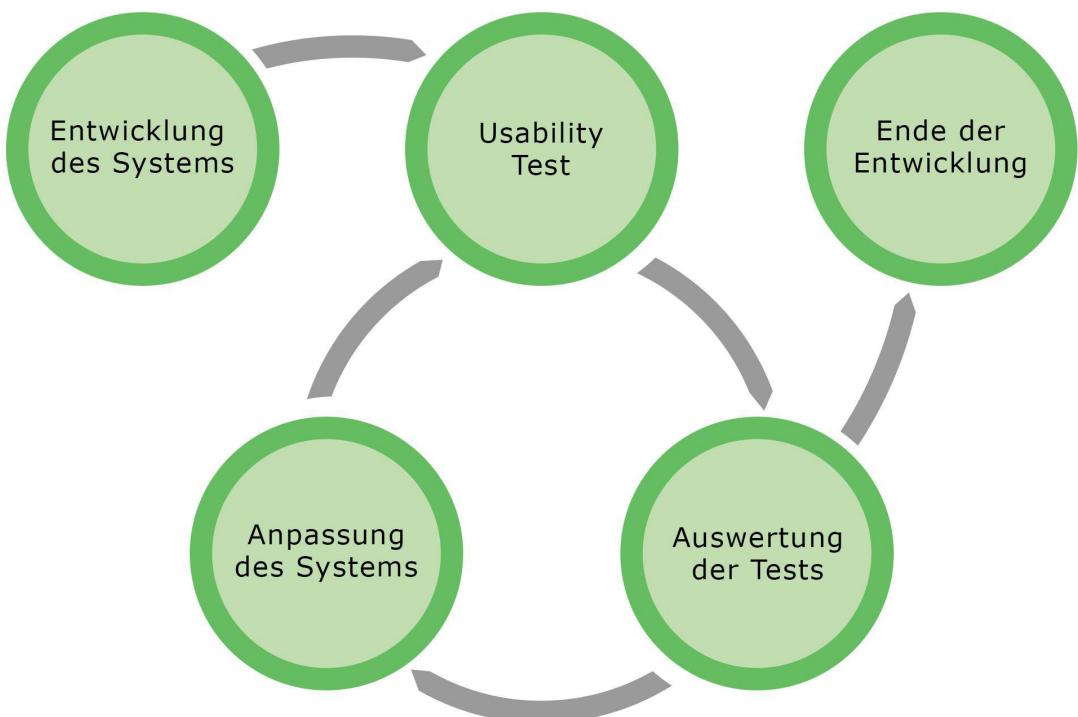


Abbildung 1: Schaubild des iterativen Testings (eigene Darstellung)

2.3.4 Datenerhebung in der Usability

Brigitte Eller (2009) unterteilt die verschiedenen Arten der Datenerhebung der Usability in drei übergeordnete Bereiche. Dialogische, hermeneutische und praktische Erhebungsmethoden. Dialogische Erhebungsmethoden beinhalten verschiedene Formen von Interviews, wie beispielsweise Gruppenbefragungen oder strukturierte Interviews. Zu den hermeneutischen Erhebungsmethoden gehören Fragebögen und Dokumentenanalysen. Als Praktische Erhebungsmethode nennt Eller die Beobachtung, wobei diese Methode selbst wiederum in einzelne Unterkategorien aufgeteilt werden kann, wie zum Beispiel passives, verdecktes Beobachten oder auch offenes, teilnehmendes Beobachten.

Die verschiedenen Erhebungsmethoden können in Kombination verwendet werden um so alle Vorteile dialogischer, hermeneutischer und praktischer Datenerhebung zu verbinden. (siehe Abb. 2) Im Folgenden werden ausschließlich die Erhebungsmethoden genauer beschrieben, die für die Datenerhebung dieser Arbeit in Frage kommen.

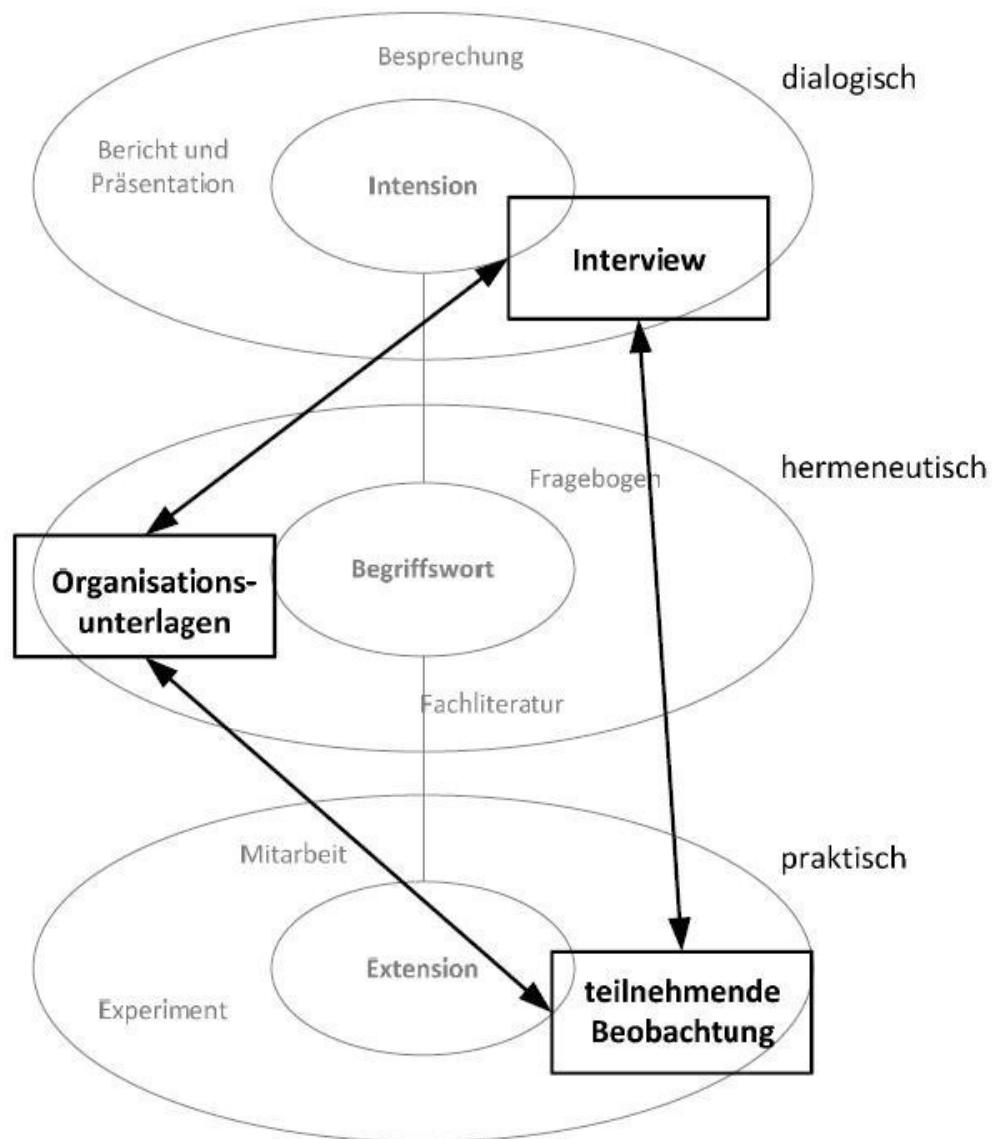


Abbildung 2: Erhebungsmethoden der Usability (Eller 2009)

2.3.4.1 Strukturiertes Interview

Grundsätzlich dienen Interviews dazu, Informationen von Testpersonen anhand von Befragungen zu erfahren. Bei einem Strukturierten Interview wird eine vorab festgelegte Liste mit Fragen gestellt, die sich auf die Eigenschaften des Systems beziehen.¹³ Diese Art des Interviews liefert sehr detaillierte Informationen, die auf Grund der immer gleichen Befragung aller Teilnehmenden anschließend leicht miteinander verglichen und bewertet werden kann.

2.3.4.2 Fragebogen

Die Datenerhebung durch Fragebögen eignet sich unter anderem dafür, zu Beginn einer Entwicklung Informationen über die Anforderungen der potentiellen NutzerInnen an das System zu erhalten. Fragebögen können außerdem als schriftliche Ergänzung zu einem Interview oder einer Befragung verwendet werden.¹⁴

2.3.4.3 aktives teilnehmendes Beobachten

Bei der aktiven teilnehmenden Beobachtung kann sich der Beobachter durch Anleitung an ein komplexeres Problem oder Beantwortung von Fragen an der Aufgabenerfüllung beteiligen. Das ist vor allem bei Systemen wichtig, die für die Testpersonen eine größere Herausforderung darstellen, bzw. bei denen nicht abgeschätzt werden kann, wie gut Testende ohne Anleitung mit Ihnen umgehen können. Wichtig ist, dass bei reinen Beobachtungen nur wahrnehmbares Sachverhalten erfasst werden kann, weshalb diese oft um Befragungen erweitert werden.¹⁵

13 Eller (2009): 201

14 Eller (2009): 202

15 Eller (2009): 203

3 Analyse

Ziel der Analyse ist es, durch theoretische Nutzungsszenarien Prototypen der digitalen Musikinstrumente erzeugen zu können, welche dann wiederum mithilfe von TestnutzerInnen dazu genutzt werden können um funktionale und nicht-funktionale Anforderungen für das System zu extrahieren. In der anschließenden technischen Analyse werden die Anforderungen an das System geprüft. Dazu zählt die Einbindung der Hololens, des MixedRealityToolKits, die Kommunikationsschnittstellen mit UDP zwischen Unity und Max/MSP und die Voraussetzungen für den Build und Deploy einer Hololens-Anwendung unter Unity.

3.1 Entwicklung der Musikinstrumente

Da digitale Musikinstrumente verschiedenste Formen der Steuerung zulassen und gerade diese später analysiert werden soll, werden bei der Entwicklung drei Instrumente verwirklicht, die sich grob in drei der vier von Miranda und Wanderley (2006) definierten digitalen Instrumententypen unterteilen lassen (siehe 2.1), da sich hierbei grundsätzliche Unterschiede in der Steuerung zeigen.

3.1.1 Instrumenten-inspirierte Benutzerschnittstelle

Bei der Entwicklung eines Instrumentes, das sowohl Steuerelemente von akustischen Instrumenten als auch erweiterte Steuermöglichkeiten aufweist, eignet sich ein Synthesizer, der zur Klangerzeugung eine klassische Klaviatur besitzt. Die erweiterte Steuerung wird durch Elemente zur Veränderung des Klangs realisiert. Hierbei sind Schieberegler, beispielsweise zur Einstellung der Lautstärke oder der Tonhöhe, aber auch Schalter und Taster zur Nutzung verschiedener Klangerzeuger (Sinus-, Rechteck-Oszillatoren) denkbar. Die gesamte Steuerung des Instruments wird gestenbasiert mit der Air-Tap Geste der Hololens umgesetzt, welche momentäre und verriegelnde Schalteraktionen sowie haltende Aktionen (bei Schieberegbern) ausführen kann.

3.1.2 Erweitertes Musikinstrument

Als erweitertes Musikinstrument wird ein Sequencer entwickelt. Diese heut meist digitalen Instrumente basieren ursprünglich auf mechanischen Musikautomaten. Die Bedienung dieser Instrumente ist eher kompositiv, da bei ihrer Nutzung Klänge nicht einzeln angespielt werden, sondern auf einer Partitur Moment und Dauer der Töne definiert und anschließend automatisch abgespielt werden.

Moderne Sequencer erweitern die Musikautomaten doch meist um einige Funktionen. So wird bei der Entwicklung ein Kontrollfeld implementiert, mit dem verschiedene Einstellungen am Sequencer vorgenommen werden können. Die Partitur selbst besteht aus einer Matrix einzelner Punkte, die durch Gestensteuerung aktiviert werden können. Die einzelnen Zeilen der Partitur geben beim Durchlaufen der Sequenz unterschiedliche Töne aus. Durch einen über die Partitur laufenden Balken wird optisch dargestellt, welche Spalte der Partitur zu jedem Zeitpunkt abgespielt wird.

3.1.3 Alternative Benutzerschnittstelle

Alternative Benutzerschnittstellen stellen digitale Musikinstrumente dar, die keinerlei Vorbild aus der realen Welt verwenden und meist komplett neue Nutzungskonzepte verwirklichen. Da es durch Augmented Reality möglich ist, einen digitalen Raum auf die reale Welt zu projizieren und jedoch keinerlei Einschränkungen der realen Welt mit sich bringt, wird bei diesem Konzept ein Instrument realisiert, bei dem einzelne Objekte schwerelos im virtuellen Raum schweben. Die Klangsynthese wird hierbei über die Position der einzelnen Objekte im Raum erzeugt und kann somit pro Objekt beispielsweise die X, Y, und Z Koordinaten der Objekte auf Klangerzeuger übertragen, welche diese wiederum als Parameter verwenden.

3.1.4 Fazit

Durch die unterschiedlichen Steuerkonzepte der einzelnen Instrumente soll eine möglichst umfassende Art der Interaktion mit dem System gewährleistet werden, welche im weiteren Verlauf dieser Arbeit anhand von Usability Tests Aufschluss über die Qualität der Benutzerschnittstellen und Steuerung geben soll.

3.2 Theoretische Analyse

3.2.1 Theoretische Szenarien

Die Erhebung theoretischer Szenarien soll die unterschiedlichen Interaktionen, des Systems aufzeigen. Hierbei sind sowohl Mensch-Maschine-, als auch Maschine-Maschine-Interaktionen denkbar. Im Folgenden werden Nutzungsszenarien für die drei entwickelten Instrumente vorgestellt. Bei allen Instrumenten interagieren die AnwenderInnen per Hololens mit der Benutzerschnittstelle, welche wiederum per OSC Steuerbefehle an einen verbundenen Computer sendet, die von einem Max/MSP Patch entgegengenommen und verarbeitet werden.

3.2.1.1 Use Case "Keypad"

Der/die AnwenderIn nutzt in diesem Szenario das Keypad. Da es sich hierbei um eine Instrumenten-inspirierte Benutzerschnittstelle handelt, können NutzerInnen einige Bedienelemente des Instruments bekannt sein, andere vielleicht nicht. Das Anwenden der Air-Tap Geste auf einer Klaviaturtaste löst einen Ton aus. Durch Interaktion mit den zusätzlichen Schiebereglern und Tasten kann der Ton verändert werden. Diese Bedienfelder sind beschriftet, um Anwendenden jederzeit klar zu vermitteln womit Sie interagieren. Die Klangerzeugung in Max/MSP erhält bei der Bedienung eines Elements eine Nachricht, in der das anzusprechende Element und der einzustellende Wert enthalten ist und leitet diese an das korrespondierende Klangelement weiter, welches je nach Element einen Klang erzeugt oder auch die Werte des Klangerzeugers verändert.

3.2.1.2 Use Case "Sequencer"

Bei der Nutzung des Sequencers kann der/die AnwenderIn über die Gestensteuerung die Partitur des Instruments ansteuern. Ein Air-Tap auf einen Punkt in der Partitur aktiviert diesen, wodurch er beim nächsten Durchlauf dieser Spalte abgespielt wird. Aktivierte Punkte sind optisch hervorgehoben um Nutzenden eine bessere Übersicht über die Sequenz zu geben. Durch eine über die Spalten laufende Leiste ist die aktuelle Position, in der sich die Sequenz befindet, erkenntlich. Über das Kontrollfeld kann das Abspielen der Sequenz gestartet und gestoppt und die Abspielgeschwindigkeit verändert werden. Bei jedem Sequenzschritt wird automatisch eine Nachricht mit allen angewählten Partitelpunkten an Max/MSP gesendet, welches diese daraufhin an die Klangerzeuger weiterleitet, welche wiederum den gewünschten Klang erzeugen.

3.2.1.3 Use Case "Drone Generator"

Bei der Verwendung des Drone Generators kann der/die AnwenderIn mit Objekten Interagieren welche durch die reine Interaktion Töne erzeugen. Durch den Raum Bewegt werden die Objekte beispielsweise durch gestenbasiertes Halten und Ziehen oder auch durch Ein- und Ausschalten der Gravitation, der Engine und Abprallen an den Wänden des digitalen Raums. Bei einer Veränderung der Positionen der Objekte leitet Unity diese an Max/MSP weiter, welches daraufhin die Klangerzeugung anpasst.

3.2.1.4 Fazit

Die Beschreibung der Use Cases ist zu diesem Zeitpunkt noch relativ ungenau, da aus Ihnen pro Use-Case mehrere Prototypen entwickelt werden sollen, welche daraufhin mithilfe von Testpersonen zur Entwicklung spezifischer Anforderungen helfen sollen.

3.2.2 Prototyping

Die Entwicklung von Prototypen soll zur frühzeitigen Erkennung von Problemen und Lösungsansätzen beitragen. Die Einbindung potentieller NutzerInnen bei Prototyp-Tests soll vor allem Feedback vor Beginn der eigentlichen Entwicklung des Systems bringen. Im Folgenden werden die unterschiedlichen Prototypen vorgestellt und die Erhebung der Testdaten analysiert und anschließend ausgewertet.

3.2.2.1 3D Prototypen der Instrumente

Das Verwenden von virtuellen Musikinstrumenten in einer dreidimensionalen Umgebung mit einem Augmented-Reality-Headset bietet komplexe Interaktionsmöglichkeiten. Folglich müssen auch Prototypen eines solchen Systems eine gewisse Komplexität aufweisen, sodass diese Interaktionen nachgestellt werden können. Bei der Entwicklung der Prototypen wurde daher ebenfalls Unity verwendet. Um hierbei schnell zu Ergebnissen zu kommen, werden die Prototypen anders als das spätere System aber mit Maus und Tastatur gesteuert, auch weil es bei diesen Prototyp-Tests vor allem um die Funktionen der Instrumente und weniger um die Art der Interaktion geht. Die Prototypen sind lediglich für die User-Interface-Interaktion entwickelt und erzeugen keine Töne. Dadurch können Testpersonen gefragt werden wie gewisse UI-Elemente den Ton verändern sollen, was wiederum zur späteren Entwicklung der Klangerzeuger beiträgt.

Für jedes der drei Instrumente wurden zwei unterschiedliche Prototypen entwickelt. Es sollte dabei jeweils eine konventionelle und eine experimentelle Variante des Instruments geben, um den Testpersonen ein möglichst breites Spektrum an Interaktionsmöglichkeiten zu bieten. Dies soll dazu dienen eine Qualitative Aussage der Testenden über möglichst viele unterschiedliche UI-Elemente zu erhalten. Die Elemente sind bewusst unbeschriftet, sodass die Testpersonen nicht vorab beeinflusst werden, sondern ihre Wünsche über die Funktionsweisen der Elemente frei äußern können.

Keypad

Die konventionelle Variante des Keypads besteht aus einer klassischen Klaviatur, fünf unbeschrifteten Tasten und zwei Schiebereglern. Eine Interaktion mit den Tasten gibt ein optisches Feedback durch kurze Verfärbung dieser und die Schieberegler lassen sich per halten der Maustaste nach Anklicken nach Links und Rechts bewegen.

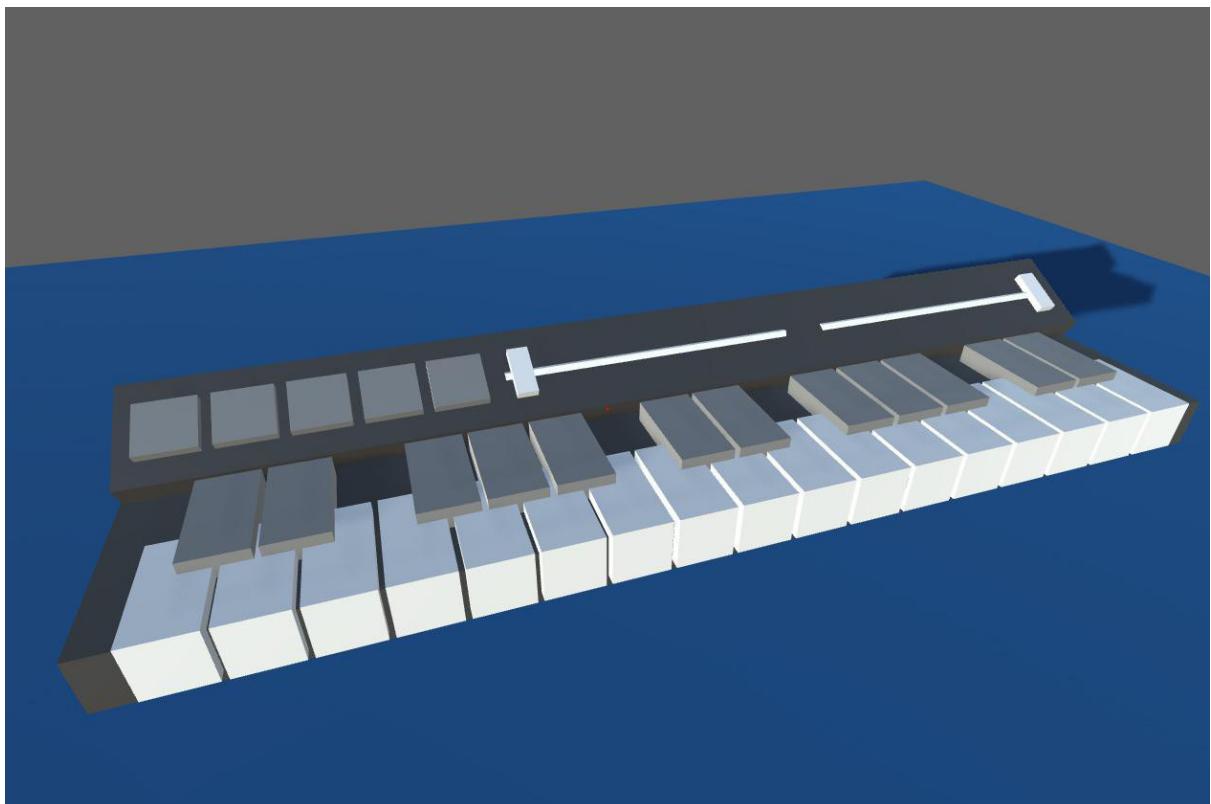


Abbildung 3: Prototyp 1 (eigene Darstellung)

Bei dem experimentellen Keypad wurde die klassische Klaviatur durch eine vom Akkordeon inspirierte Klaviatur, bestehend aus Knopfreihen, ersetzt. Das Instrument bietet wesentlich mehr Steuerelemente und eine größere Bedienoberfläche, mit angewinkelten Panelen. Es gibt nun sechs Schieberegler, die mehr Freiheit zur

Interpretation lassen, eine Tastermatrix neben den Reglern und als wiederkehrendes Element fünf Taster am oberen Panel des Instruments. Die Fläche neben den Tastern ist bewusst nicht mit Elementen belegt um die Testenden nach alternativen Regelmöglichkeiten für diese Stelle fragen zu können. Die Elemente geben, genau wie bei der klassischen Variante, optische Feedbacks.

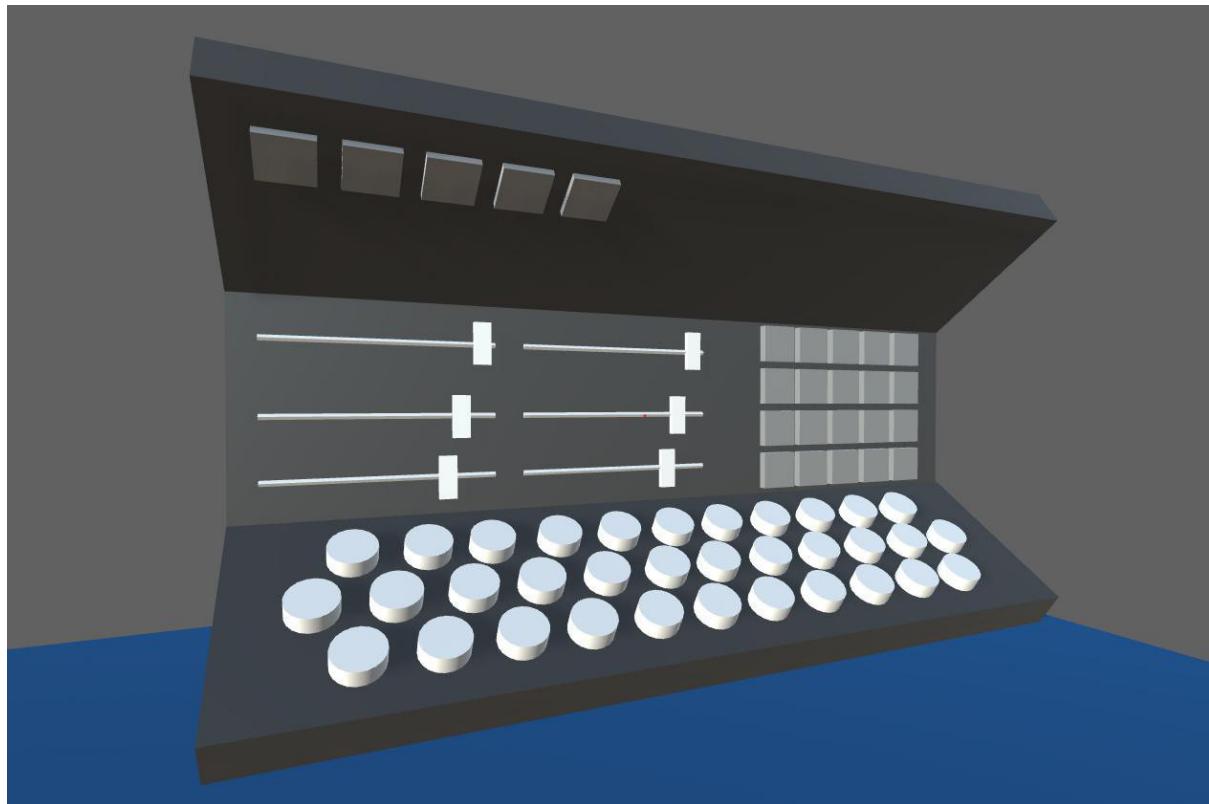


Abbildung 4: Prototyp 2 (eigene Darstellung)

Sequencer

Der klassische Sequencer weist eine zweidimensionale 16x8 Matrix auf, die von einem dahinterliegenden Balken abgelaufen wird. Die einzelnen Punkte der Partitur können angewählt werden. Im unteren Teil des Instruments befinden sich erneut fünf Taster, die bei Bedienung ein optisches Feedback geben.

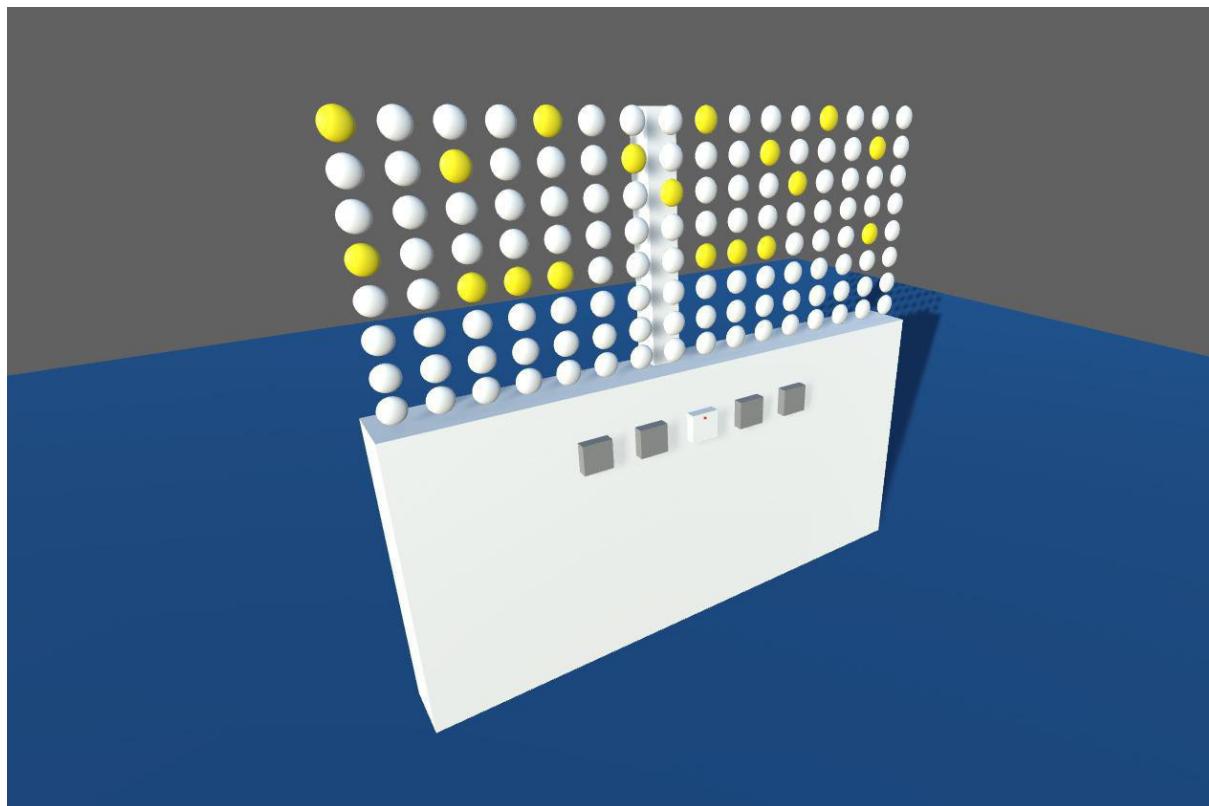


Abbildung 5: Prototyp 3 (eigene Darstellung)

Bei der experimentellen Version sind die Bedienelemente und Funktionsweisen grundsätzlich gleich. Die Partitur ist hierbei aber auf einer 16x8 Matrix abgebildet, die zylindrisch angeordnet ist und von einem sich im Kreis bewegenden Block abgelaufen wird. Dadurch gibt es keinen definierten Anfang der Partitur.

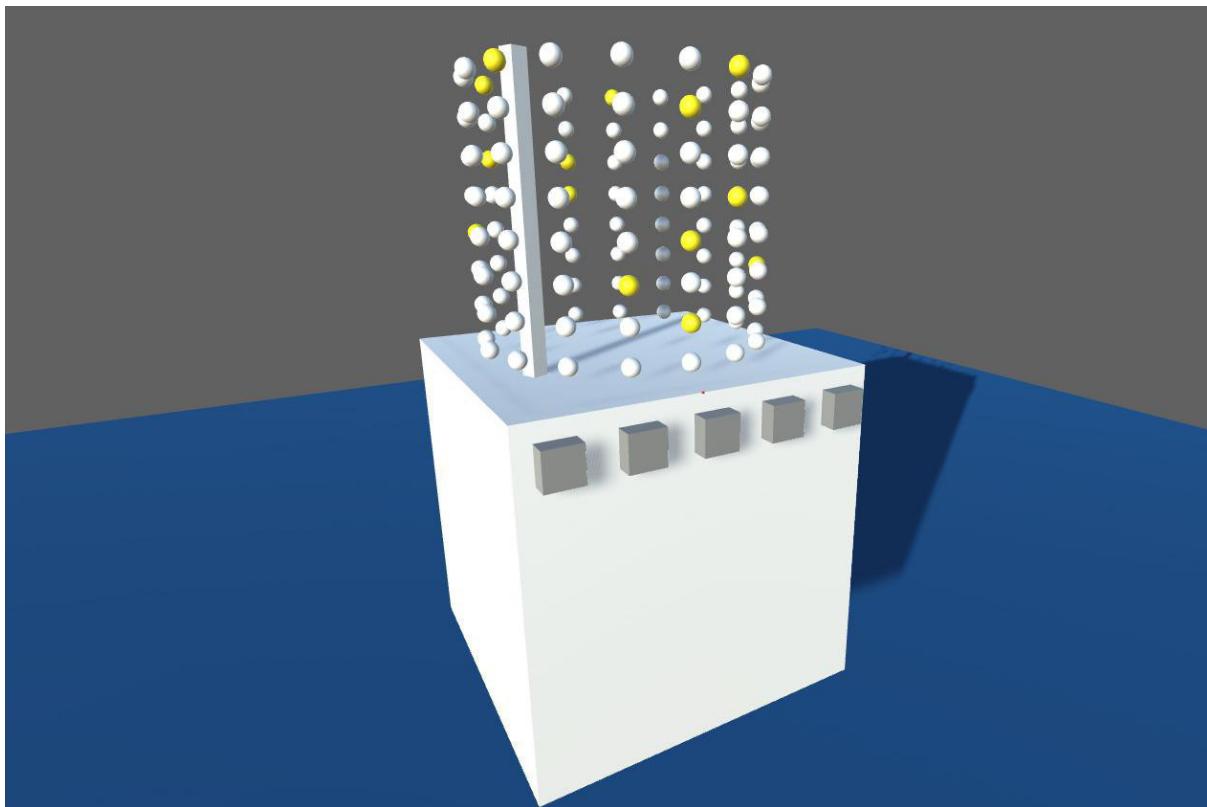


Abbildung 6: Prototyp 4 (eigene Darstellung)

Drone Generator

Beim Drone Generator kann man weniger von einer klassischen Variante sprechen, da es für beide Prototypvarianten keine Vorbilder gab. Die erste Variante ist in ihrer Bedienung aber intuitiver. Es gibt fünf Kugeln die durch Anklicken und Halten frei im Raum positioniert werden können. Diese sollen als Tonerzeuger dienen und durch das Verändern ihrer Positionen im Raum, den Klang des Instruments beeinflussen. Ebenfalls gibt es wieder fünf unbeschriftete Taster, die Bedienmöglichkeiten bieten.

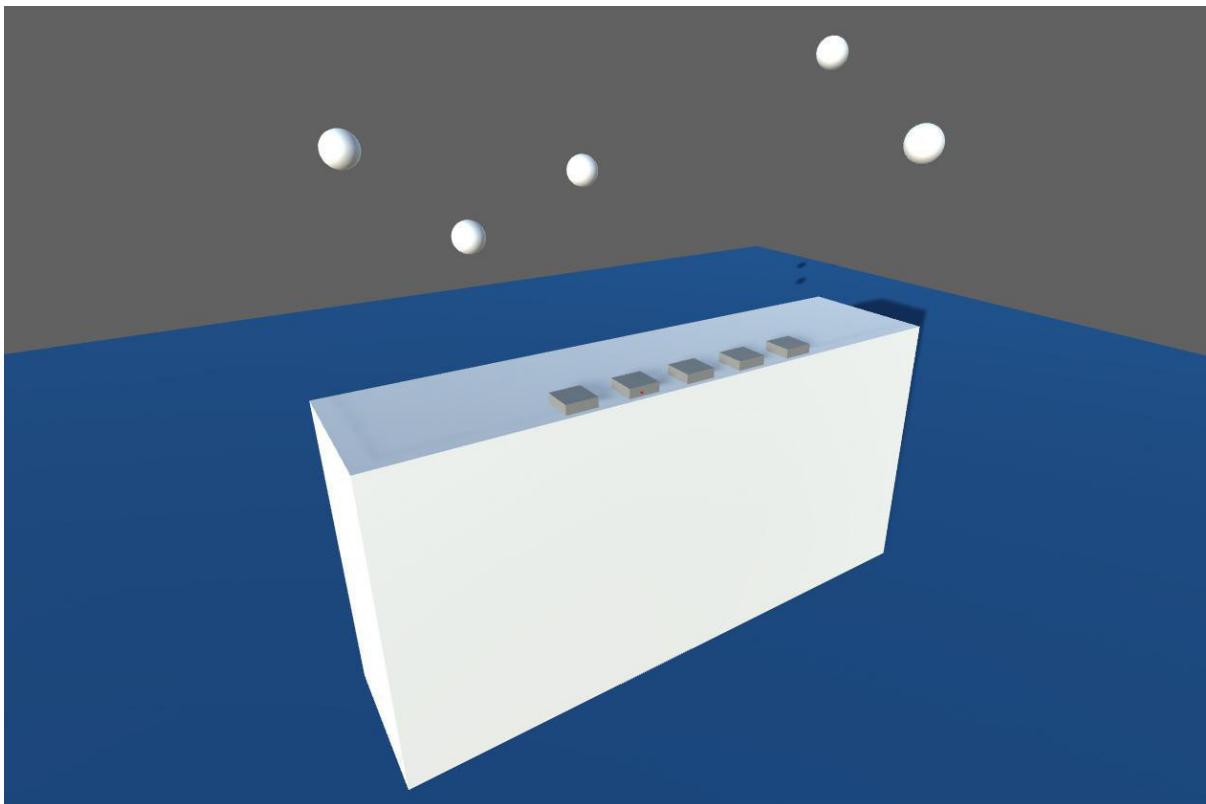


Abbildung 7: Prototyp 5 (eigene Darstellung)

Die zweite Variante besitzt ebenfalls fünf frei schwebende Kugeln. Diese befinden sich in einem durchsichtigen Würfel, der sich durch anklicken und bewegen um die eigenen Achsen drehen lässt. Durch Betätigen der rechten Maustaste wird für kurze Zeit die Gravitation eingeschaltet, was die Kugeln in Bewegung versetzt. Nach der ersten Berührung der Kugeln mit dem Würfel, beginnen diese frei in diesem herumzuschweben. Die Klangerzeugung soll hier ebenfalls durch die Positionen der Kugeln, aber auch durch die Kollisionen mit dem Würfel passieren.

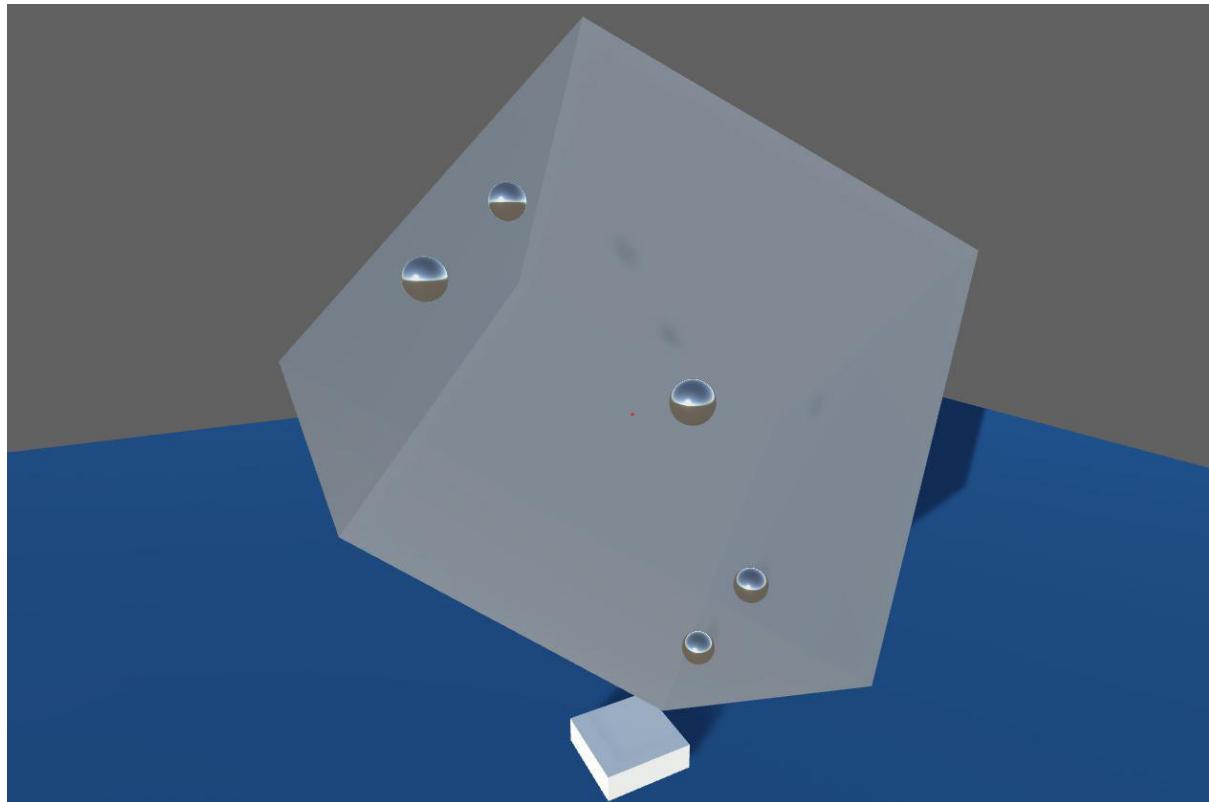


Abbildung 8: Prototyp 6 (eigene Darstellung)

3.2.2.2 Erhebung der Testdaten

Für die Erhebung der Testdaten werden fünf Testpersonen herangezogen, welche die einzelnen Prototypen nacheinander benutzen und währenddessen Fragen beantworten. Vor Beginn der Tests gibt es eine kurze Einweisung in die Steuerung. Bei jedem Prototypen wird vorab kurz beschrieben welche grundsätzliche Funktionsweise das Instrument haben soll. Die Testperson bekommt eine Minute Zeit, den Prototyp frei zu nutzen bevor Fragen gestellt werden. Für jede Prototypgruppe (Keypad, Sequencer, Drone Generator) gibt es spezielle Fragen, aber für Fragen zur allgemeinen Benutzertauglichkeit oder Erscheinung werden wiederkehrende skalierte Fragen verwendet. Die verwendete Skala ist siebenstufig und besteht aus den folgenden

Abstufungen: 1 stimmt absolut nicht, 2 stimmt nicht, 3 stimmt eher nicht, 4 teils/teils, 5 stimmt eher, 6 stimmt, 7 stimmt absolut. Frei formulierte Antworten werden in Absprache mit den Testpersonen in Stichpunkten notiert.

3.2.2.3 Auswertung der Daten

Im Folgenden werden die Daten, die das Prototyp-Testing ergeben hat, ausgewertet. Die skalierten Fragen sind hierbei vorwiegend dafür, Präferenzen zwischen den zwei Prototypen einer Instrumentengruppe zu erkennen. Die Werte der Antworten (bspw. 5 für "stimmt eher zu") können ähnlich wie bei der Likert Skala aufaddiert werden und ergeben somit ein Ergebnis, mit dem sich die Qualität der Prototypen vergleichen lässt¹⁶. Hierbei können Vergleiche ganzer Fragebögen zwar zeigen welcher Prototyp allgemein besser ist, warum das so ist, zeigt ein solcher Vergleich aber nicht. Dies funktioniert vor allem beim Vergleich einzelner Fragen miteinander. Die frei formulierten Antworten hingegen geben eine deutlich spezifischere Einsicht in die Wünsche der Nutzenden. vor allem bei den experimentellen Instrumenten haben die Testpersonen durch das freie Formulieren die Möglichkeit, eigene Ideen einzubringen und komplett eigene Nutzerkonzepte zu erklären.

Keypad

Beim Vergleich der Keypad Prototypen zeigt sich in den skalierten Fragen, dass ein einfaches Design zu einer verbesserten Verständlichkeit in der Funktionsweise führt (Vergleich Frage 5), allerdings bei der Frage nach der Anzahl der Bedienelemente, dass ein zu ein simples Design hierbei schlechter abschneidet (Vergleich Frage 3). Die Fragen 1, 2 und 4 dienen eher zur Einschätzung der Qualität der einzelnen Elemente der Prototypen (siehe Tabelle 1 und 2). Da sich diese oft selbst innerhalb der Instrumentengruppen unterscheiden, ist ein Vergleich der Antworten nicht möglich.

Keypad 1	1	2	3	4	5	6	7	Ergebnis
Die allgemeine Erscheinung des Instruments spricht mich an				1	3	1		22
Die Bedienung der einzelnen Elemente ist intuitiv						3	2	32
Die Anzahl der Bedienelemente ist ausreichend			2	2		1		20
Das optische Feedback bei Tastern macht Eingaben deutlich						2	3	33
Die Funktionsweise des Instruments ist verständlich			1	1	1	2		24

Tabelle 1: Auswertung Prototyp 1

16 Brosius et al. (2009): 62

Keypad 2	1	2	3	4	5	6	7	Ergebnis
Die allgemeine Erscheinung des Instruments spricht mich an			1	2	1	1		22
Die Bedienung der einzelnen Elemente ist intuitiv			2		1	2		23
Die Anzahl der Bedienelemente ist ausreichend					1	3	1	30
Das optische Feedback bei Tastern macht Eingaben deutlich						1	4	34
Die Funktionsweise des Instruments ist verständlich		1	2	1		1		18

Tabelle 2: Auswertung Prototyp 2

Bei den frei formulierten Antworten gaben drei von fünf Nutzenden an, dass Ihnen der experimentelle Prototyp mit der klassischen Klaviatur, die der erste Prototyp enthält, besser gefallen würde. Des Weiteren konnten alle Testpersonen, bei den Fragen nach der Funktionsweise der Schiebereglern und der Bedienmatrix, Vorstellungen über diese äußern. Dies, in Verbindung mit dem Ergebnis aus Frage 3, lässt darauf schließen, dass die erweiterten Bedienmöglichkeiten des experimentellen Prototypen durchaus erwünscht sind. Einige der frei formulierten Antworten wurden mehrfach genannt und werden daher direkt in das Design des Keypads übernommen. Die Frage nach der Funktionsweise der Taster, ergab bei dem ersten Prototyp bei drei von fünf Testenden den Wunsch, ein Umschalten der Oktave in der sich die Klaviatur befindet, zu ermöglichen. Bei dem experimentellen Prototypen ergab dieselbe Frage bei drei Testpersonen den Wunsch, die Einstellungen der Schiebereglern speichern und laden zu können. Die Antworten auf die Frage nach den Schiebereglern ergab bei allen Testenden den Wunsch, den Klang des Instruments damit verändern zu können. Hierbei gab es wiederum einige gleiche Antworten, die in das Design einfließen. Bei der Frage nach zusätzlichen Bedienelementen gaben nur zwei der Testenden überhaupt eine Antwort, was darauf schließen lässt, dass der zusätzliche Platz nicht unbedingt nötig ist. Dies in Verbindung mit der erhöhten Akzeptanz des simpleren Prototypen, führt beim Design des Keypads zu einer Bedienoberfläche, die mehr Steuerelemente als der erste Prototyp, aber auch weniger als der Zweite enthält. Außerdem wird die klassische Klaviatur verwendet.

Sequencer

Die skalierten Fragen ergaben bei der Sequencer Instrumentengruppe nur wenig Aufschluss. Die Antwortpunktzahlen liegen zu nah aneinander, was daran liegen kann, dass sich die Prototypen lediglich in der Form der dargestellten Partitur unterscheiden

und sonst identisch funktionieren. Bei den frei formulierten Antworten gab es eine Reihe an verwertbaren Antworten. Die Frage nach der Funktionsweise der Taster ergab in beiden Tests bei allen Testpersonen den Wunsch nach Start und Stop Tasten und Bedienmöglichkeiten, um die Geschwindigkeit des Sequencers einzustellen, was sich mit dem Use-Case Szenario deckt. Bei drei von fünf Testenden wurde der Wunsch geäußert, erstellte Sequenzen speichern und laden zu können. Diese Anforderungen werden in die Anforderungsanalyse mit aufgenommen. Drei von fünf Testpersonen gaben an, dass sie weitere Bedienflächen nützlich fänden um den Sequenzer noch umfangreicher steuern zu können, was ebenfalls in das Design mit aufgenommen wird.

Sequencer 1	1	2	3	4	5	6	7	Ergebnis
Die allgemeine Erscheinung des Instruments spricht mich an				1	2	2		26
Die Bedienung der einzelnen Elemente ist intuitiv				2		2	1	27
Die Anzahl der Bedienelemente ist ausreichend		1	2		1	1		16
Das optische Feedback bei Tastern macht Eingaben deutlich						1	4	34
Die Funktionsweise des Instruments ist verständlich				2	1	2		25

Tabelle 3: Auswertung Prototyp 3

Sequencer 2	1	2	3	4	5	6	7	Ergebnis
Die allgemeine Erscheinung des Instruments spricht mich an					2	3		28
Die Bedienung der einzelnen Elemente ist intuitiv						3	2	32
Die Anzahl der Bedienelemente ist ausreichend		3		1	1			15
Das optische Feedback bei Tastern macht Eingaben deutlich						1	4	34
Die Funktionsweise des Instruments ist verständlich					1	3	1	30

Tabelle 4: Auswertung Prototyp 4

Drone Generator

Bei der Auswertung der skalierten Fragen für die dritte Instrumentengruppe zeigt sich, dass der erste Prototyp in allen Fragen besser abschneidet als der Zweite. Da diese beiden Prototypen sowohl in ihrer Funktionsweise, als auch in ihrer Erscheinung und ihren Bedienmöglichkeiten stark voneinander abweichen, ist es nicht möglich weitere Vergleiche der Instrumente anhand der Fragen anzustellen.

Es sei noch zu erwähnen, dass Frage 4 nicht auf den zweiten Prototyp angewendet werden kann, da dieser keine Taster besitzt und somit die Frage nach dem optischen Feedback dieser nicht anwendbar ist. Da die allgemeine Akzeptanz für den ersten

Prototyp höher ausgefallen ist als für den zweiten Prototypen (73 zu 64 Punkte, wenn man Frage 4 nicht wertet) wird vor allem der erste Prototyp für die Auswertung zur Anforderungsanalyse betrachtet.

Drone Generator 1	1	2	3	4	5	6	7	Ergebnis
Die allgemeine Erscheinung des Instruments spricht mich an				2	2	1		24
Die Bedienung der einzelnen Elemente ist intuitiv		2	3					15
Die Anzahl der Bedienelemente ist ausreichend		2	1			2		21
Das optische Feedback bei Tastern macht Eingaben deutlich						1	4	34
Die Funktionsweise des Instruments ist verständlich		3	1	1				13

Tabelle 5: Auswertung Prototyp 5

Drone Generator2	1	2	3	4	5	6	7	Ergebnis
Die allgemeine Erscheinung des Instruments spricht mich an			2	1	2			20
Die Bedienung der einzelnen Elemente ist intuitiv	1	3	1					10
Die Anzahl der Bedienelemente ist ausreichend		2	2		1			17
Das optische Feedback bei Tastern macht Eingaben deutlich	-	-	-	-	-	-	-	-
Die Funktionsweise des Instruments ist verständlich	1	1	2	1				17

Tabelle 6: Auswertung Prototyp 6

Die Auswertung der frei formulierten Antworten zeigt, dass alle Testpersonen eine Veränderung des Klanges durch Positionieren der Kugeln im Raum erreichen wollen (siehe Tabelle 5 und 6). Die Frage nach der Funktionsweise der Taster ergab, dass drei Testpersonen ein Speichern und Laden der Positionen wünschen. Beide Punkte werden in das Design des Drone Generators aufgenommen.

Es sei noch zu erwähnen, dass zum Beispiel beim Test mit der zweiten Instrumentengruppe eine Inkonsistenz in den Antworten aufgetreten ist. Die Frage nach der intuitiven Bedienbarkeit einzelner Steuerelemente (Frage 2) ergab bei dem zweiten Prototyp eine höhere Punktzahl als beim ersten, obwohl beide Prototypen, dieselben Bedienelemente besitzen. Dies ist wohl darauf zurückzuführen, dass die Testpersonen bei der Nutzung des ersten Prototypen bereits Erfahrungen mit den Bedienelementen sammeln konnten und somit beim Test mit Prototyp Zwei schon vertrauter mit ihnen sind. Auch die erhöhte Punktzahl in Frage 5 bei dem zweiten Prototyp kann auf dieses Problem zurückzuführen sein. Bei dieser Art des Testens ist ein solcher Fehler nicht vermeidbar und muss beachtet werden.

3.2.3 Anforderungsanalyse

Anhand der in 3.2.1 entwickelten Szenarien und der anschließenden Auswertung der Daten die das Prototypings in 3.2.2 ergeben haben, lassen sich sowohl funktionale als auch nicht-funktionale Anforderungen für die Steuerung des Systems ableiten. Diese werden im Folgenden dargelegt und beschrieben.

3.2.3.1 Funktionale Anforderungen

Must-Have Anforderungen

Bedienbare Taster

Hierzu zählen alle Objekte, die beim Anwenden einer einfachen Air-Tap Geste eine Aktion auslösen. Das Bedienen von Tastern ist ein wesentlicher Bestandteil jedes zu entwickelnden Instruments. Ausgelöst wird ein Taster durch die Air-Tap Geste nachdem er über den Gaze angewählt wurde. Nach dem Auslösen soll automatisch die dem Taster zugewiesene Funktion ausgelöst werden.

Schieberegler

Durch auswählen mit dem Gaze und Auslösen mit der Air-Tap Geste folgt ein Schieberegler so lange der relativen Position des Gaze Punktes, wie die Air-Tap Geste gehalten wird. Währenddessen sendet er kontinuierlich von seiner Position abhängige Werte, die auf float Werte zwischen 1 und 256 skaliert werden an Max/MSP.

Greifbare Objekte

Wie Schieberegler werden greifbare Objekte durch Gaze und gehaltene Air-Tap Geste ausgewählt. Sie verändern ihre Position relativ zum Nutzenden durch Kopfbewegungen aber auch relativ zu ihrer Umgebung durch die Bewegungen des Nutzenden im Raum. Nach Beendigung der Air-Tap Geste verbleiben die Objekte an ihrer aktuellen Position. Die Objekte senden bei jeder Positionsänderung ihre X,Y und Z Koordinatenwerte auf float Werte zwischen 1 und 256 skaliert an Max/MSP.

Klangerzeugung des Keypads

Durch Betätigen der Klaviatur werden Töne erzeugt. Nach Senden eines Tonwertes als MIDI Note an Max/MSP wird diese hier als Erstes zu dem Keypad Modul gesendet und anschließend zu einer Frequenz gewandelt. Diese Frequenz wird als Hauptanteil an einen Frequenzmodulierten Synthesizer übergeben. Gleichzeitig löst das Eintreffen eines Tons die Hüllkurve aus, deren Öffnungs- und Schließzeiten über zwei Schieberegler eingestellt werden können. Die restlichen fünf Schieberegler bedienen die Beimischung der Modulationsfrequenz, der Verzerrung des Signals, einen Low-Pass-Filter, einen Vibrato Effekt und die Lautstärke des Signals. Über die Matrix kann pro Spalte ein Synthesizer hinzugefügt werden. Drei der vier Zeilen geben die verschiedenen Wellenformen des Synthesizers an (Sinus, Rechteck, Sägezahn), die vierte schaltet diesen Synthesizer ab.

Klangerzeugung des Sequencers

Nach Auslösen eines Partiturpunkts durch den Sequencer, wird der dazugehörige Synthesizer in Max/MSP mit einem "Bang" angesprochen. Jede Zeile des Sequencers spricht einen einzelnen Synthesizer an, wobei jeder Synthesizer verschiedene Klänge eines Schlagzeuges imitiert. Die Klangsynthese funktioniert hierbei subtraktiv mit Filterung verschiedener Oszillatoren und Noise Generatoren.

Klangerzeugung des Drone Generators

Die Klangerzeugung des Drone Generators gibt kontinuierlich Klänge aus. Durch Verändern der Position der einzelnen Kugeln können fünf Instanzen eines Synthesizers angesprochen werden. Die X,Y und Z Koordinaten stellen bei jedem Synthesizer die Lautstärke und die Frequenz zweier frequenzmodulierenden Oszillatoren ein.

Should-Have Anforderungen

Speichern und Laden von Zuständen

Durch Betätigung spezieller dafür vorgesehene Taster, können für Nutzende wichtige Werte eines Instruments (Bei Keypad: alle Sliderposition, bei Sequencer: aktuelle Partitur, bei Drone Generator: die Position der Kugeln) gespeichert und wieder abgerufen werden. Jedes speicherbare Element kann den zu speichernden Wert selbst setzen und abrufen.

Optisches Feedback der Taster

Bei der Auslösung eines Tasters verändert dieser für einen kurzen Zeitraum seine Farbe um Nutzenden eine Bedienung klar zu suggerieren.

Beschriftung der Taster

Alle Taster werden mit einer Kurzbeschreibung ihrer Funktion beschriftet. Hierbei werden auch im Musikbereich gängige Abkürzungen verwendet.

Nice-to-Have Anforderungen

Sequencer Funktion (Keypad)

Durch Betätigen eines Tasters wird ein einfacher Step Sequencer gestartet, welcher mit jeder folgenden Betätigung einer Keyboard Taste, diesen Ton in die Sequenz mit aufnimmt. Erneutes Betätigen der Taste beendet den Sequenzer wieder.

Freie Bewegung der Kugeln (Drone Generator)

Die Kugeln des Drone Generators stehen nicht starr im Raum, sondern können durch Anstoßen von anderen Kugeln oder durch Loslassen der Greif-Funktion während einer Bewegung frei durch den Raum schweben.

Bedienung ganzer Zeilen der Partitur (Sequencer)

Durch Betätigen spezieller Tasten können einzelne Zeilen der Sequencer Partitur geleert, oder auch mit einfach Rhythmen gefüllt werden.

3.2.3.2 Nicht-Funktionale Anforderungen

Performanz

Um eine möglichst hohe Immersion bei der Nutzung einer Augmented Reality Anwendung zu erzeugen, ist ein performantes System unverzichtbar.¹⁷ Sowohl die Interaktion mit Steuerelementen, als auch die Übertragung der Steuerdaten und die Klangerzeugung müssen so latenzfrei wie möglich sein, um den Nutzenden ein möglichst "echtes" Spielgefühl, bei der Nutzung der Instrumente geben zu können.

Benutzbarkeit

Eine hohe Benutzbarkeit einer Software ist für das Erlernen und die Wiederverwendung des Systems unerlässlich.¹⁸ Nutzende verlieren schnell das Interesse an schlecht bedienbarer Software und da bei der Vielzahl an digitalen Musikinstrumenten eine hohe Konkurrenz existiert, ist es umso wichtiger, dass diese eine hohe Benutzbarkeit aufweisen, da Nutzende sonst schnell auf andere Systeme wechseln würden, selbst wenn die Funktionalität der Anwendung umfangreicher ist. Durch Methoden der Usability soll ein möglichst gut benutzbare System erzeugt werden.

Erweiterbarkeit

Die Musikbranche ist eine sich stetig verändernde Welt, in der die Erweiterbarkeit und Verbesserung digitaler Musikinstrumente jederzeit möglich sein muss. Durch eine Kapselung der einzelnen Systemkomponenten in User-Interface und Audiogenerierung ist dies von vornherein vereinfacht. Denkbar wären beispielsweise die Implementierung weiterer Instrumente oder neue Steuerkonzepte an den bestehenden Instrumenten zu verwirklichen. Auch die Benutzbarkeit und Performanz muss nachträglich verbessert werden können.

17 Eller (2009): 196

18 Richter und Flückinger(2010): 7

3.3 Technische Analyse

In der technischen Analyse wird ermittelt, ob ein System, dass die in 3.2 erhobenen funktionalen und nicht-funktionalen Anforderungen erfüllt, mithilfe der im Folgenden vorgestellten Hard- und Software Komponenten, realisierbar ist. Hierbei werden vor allem die für dieses Projekt wichtigsten Komponenten und Aspekte der einzelnen Systeme erläutert, da eine allumfängliche Beschreibung nicht relevant ist.

3.3.1 Verwendete Hard-, Software

3.3.1.1 Hololens

Die im März 2016 erschienene Microsoft Hololens ist eine Durchsichtdatenbrille¹⁹, welche als Stand-Alone²⁰ Gerät mit Windows 10 als Betriebssystem genutzt wird. Die Darstellung der virtuellen Objekte wird hierbei mithilfe von zwei Waveguide-Displays²¹ umgesetzt, die angezeigte Grafiken eines Displays auf die Augen umlenken, ohne die direkte Sicht auf die Realität zu verhindern. Mit einer Vielzahl von Sensoren und Kameras ist es möglich, die Position des Nutzenden sowie die der virtuellen Objekte, als auch die Form der Umgebung per Spatial Mapping²² zu bestimmen. Nutzende können über Handgesten mit der virtuellen Welt interagieren. Die Hololens bietet zwei implementierte Gesten. Zum Einen die Air-Tap Geste, die durch Zusammenführen des Zeigefingers und Daumens im Sichtfeld der Kameras ausgelöst wird. Die Geste wird automatisch auf dem Objekt angewendet, die durch das Gaze²³ Symbol ausgewählt ist. Die zweite von Hololens bereitgestellte Geste ist die Bloom Geste. Sie wird durch Öffnen der nach oben zeigenden Hand ausgelöst und ist für das Zurückkehren ins Hauptmenü reserviert.

19 Broll et al. (2013): 273

20 Anm.: Hard-/Software, die eigenständig, ohne weitere Hard-/Software funktioniert

21 Anm.: Durch Brechung von Unidirektionalem Licht mit einer Vielzahl kleiner Prismen auf einer Linse wird diese zu einem durchsichtigen Display

22 Anm.: Erstellung einer virtuellen Repräsentation der reellen Umgebung

23 Anm.: Mittelpunkt des Sichtfeldes der Hololens, das meist durch einen weißen Punkt dargestellt wird

3.3.1.2 Unity

Die am 08.06.2005 erschienene Laufzeit- und Entwicklungsumgebung Unity ist eine cross-platform²⁴ Spiele-Engine der Firma Unity Technologies. Sie ermöglicht Multiplattform-builds²⁵ auf über 25 Plattformen und ist derzeit die von Microsoft empfohlene Entwicklungsumgebung für die Entwicklung für Microsoft Hololens-Anwendungen²⁶. Der Unity Editor ist in seiner Funktionsweise an Animationsprogramme wie Blender angelehnt und ermöglicht die Entwicklung von 2D und 3D Anwendungen. Unity bietet als Skript-API C# für sowohl den Editor, als auch die Spiele selbst. Skripte werden hierbei einfach an sogenannte Game Objects angehängt. Über die C# using-direktive "using UnityEngine;" kann der Unity-spezifische namespace angesprochen werden, der alle für das Skripting nötigen Klassen enthält. Jedes Unity Skript erbt von der Klasse "MonoBehaviour" und kann dadurch auf die Methoden der Basisklasse (bspw. Start(), Update(), etc.) zugreifen. Unity unterstützt sowohl Direct3D als auch OpenGL Grafik APIs, was wiederum Multiplattform-builds ermöglicht. Gerade deswegen und da die Entwicklung von Software mit Unity für den privaten Gebrauch kostenlos ist, die Engine aber dennoch ausgereift genug ist, dass mit ihr auch kommerzielle Titel entwickelt werden können, gibt es eine sehr große Community, die Assets²⁷ bereitstellen, in Foren diskutieren und Feedback an Unity geben, sodass sich die Entwicklungsumgebung in den letzten Jahren zu einer der größten ihrer Art entwickeln konnte.

3.3.1.3 MixedReality Toolkit

Das Mixed Reality Toolkit ist eine Reihe von Skripten und Komponenten, die dazu dienen sollen, die Entwicklung von Anwendungen für die Microsoft Hololens und Mixed Reality immersive (VR) headsets zu erleichtern. Durch die Bereitstellung einer Dokumentation, Beispielszenen und Prefabs²⁸, ermöglicht das Toolkit schnell Erfolge bei der Entwicklung zu erzielen.²⁹

24 Anm.: Entwicklung auf unterschiedlichen Plattformen (hier Betriebssystemen)

25 Anm.: Kompilieren eines Projekts auf unterschiedliche Plattformen (bspw. Windows, iOS, Android, etc.) möglich

26 Microsoft (2019a)

27 Anm.: In Unity entwickelte Elemente die von Entwicklern oder Unity selbst bereitgestellt werden um in anderen Unity Projekten eingesetzt zu werden

28 Anm.: Zusammenstellung verschiedener Unity Software Elementen

29 Microsoft (2019b)

3.3.1.4 C#

C# ist eine objektorientierte Programmiersprache der Firma Microsoft. Sie ist eine der von Microsoft unterstützten Sprachen für die Bereitstellung von Microsoft Hololens Anwendungen³⁰. Bei einem Build einer Universal-Windows-Platform Anwendung in Unity wird eine C# Visual Studio Solution erstellt, welche in Visual Studio für die Hololens kompiliert werden kann.

3.3.1.5 Max/MSP

Max/Msp ist eine grafische Programmiersprache für die Audio und Multimedia Entwicklung. Anders als bei herkömmlichen Programmiersprachen werden bei Max keine Zeilen mit Anweisungen geschrieben, die Sprache entspricht optisch eher einem Datenflussdiagramm.

Einzelne Blöcke mit unterschiedlichen Funktionen werden mit Linien verbunden um Referenzen zu erzeugen oder Informationen zu übertragen. Max Objekte sind hierbei vergleichbar mit Klassen einer konventionellen Sprache und können wiederum selbst geschrieben und in anderen Max Patchern als Blöcke integriert werden. Durch komplexes Routing von Daten lassen sich Nachrichten aufteilen, auf Grund ihres Inhalts an verschiedene Ausgänge eines Objekts und somit in verschiedene Bereiche eines Patches senden, oder auch durch Schaltungsobjekte von verschiedenen Bereichen eines Patches beziehen. Dies geschieht hauptsächlich mit Objekten zur Steuerung des Datenflusses, wie `gate`, `switch` und `route`. Hierbei werden empfangene Daten oft auch zur Steuerung anderer Daten verwendet. Beispielsweise besitzt ein "route integer string float" Objekt durch die Angabe der drei Argumente "integer string float" drei Ausgänge. Beim Erhalt einer Nachricht "float 0.2" sendet dieses Objekt dann den Wert "0.2" aus dem dritten Ausgang weiter. Neben Objekten zur Steuerung bietet Max/MSP alle gängigen arithmetischen Operationen als eigenständige Objekte. Ein "+" Objekt beispielsweise hat immer zwei Eingänge, die jeweils einen Zahlenwert erwarten. Am Ausgang des Objektes erhält man dann die Summe beider Objekte zur Weiterverarbeitung. In Max werden Ereignisse und Funktionen von Objekten durch so genannte "Bangs" ausgelöst. Eine Bang Nachricht kann in unterschiedlichen Objekten unterschiedliche Funktionen auslösen. Wird bspw. ein Bang an ein "Toggle" Objekt gesendet, sendet dieses beim Einschalten eine 1 beim Ausschalten aber wiederum eine 0 an das nächste Objekt weiter. In vielen Objekten werden Bangs auch durch erhalt einer Nachricht ausgelöst. Im obigen Beispiel des `route` Objekts löst der erhalt der Nachricht "float 0.2" automatisch die Funktion des Objektes aus und sendet "0.2" weiter.

³⁰ Microsoft (2018)

Dadurch kann eine Datenverarbeitung nur durch eingeben einer Nachricht in einen Patch erfolgen, welche durch Steuer Objekte an verschiedene andere Objekte verteilt, dort unterschiedliche Funktionen auslösen kann, die wiederum zur Anpassung, Verarbeitung oder Ausgabe weiterer Daten dienen können.

Max/MSP liefert von sich aus eine Vielzahl von Objekten, die speziell für die Erzeugung, Verarbeitung und Ausgabe von Audiosignalen konzipiert wurden. Dies ermöglicht umfangreiche Klangsynthese und Arrangements, die eine hohe Komplexität aufweisen können, durch die mögliche Strukturieren mit eigens erzeugten Patches aber weitestgehend übersichtlich bleiben. Max/MSP enthält darüber hinaus Objekte für die Verarbeitung von MIDI Signalen und dem OSC Protokoll, was als Schnittstelle eine Anbindung an viele Hardware und Software Komponenten ermöglicht.

3.3.1.6 Open Sound Control

ist ein Netzwerkprotokoll, das für die Übertragung von Steuersignalen in Echtzeit verwendet wird. Da es möglich ist vom MIDI Standard lesbare Signale per OSC zu übertragen findet es oft bei Audioanwendungen Verwendung.

3.3.2 Machbarkeitsstudie

Im folgenden werden die Möglichkeiten der einzelnen Komponenten für die Umsetzung des Systems vorgestellt.

3.3.2.1 Unity

Unity (LTS) ist die von Microsoft empfohlene Entwicklungsumgebung zur Erstellung von Microsoft Hololens Software.³¹ Hierfür stellt Microsoft ein für die Entwicklung von Mixed Reality Anwendungen optimiertes Toolkit names Mixed-Reality-Toolkit bereit. Für die Kommunikation mit Max/MSP gibt es mehrere kostenpflichtige aber auch ein kostenloses Unity Package, dass OSC Kommunikation über Netzwerke ermöglicht.³²

31 Microsoft (2019b)

32 UnityOSC (2019)

3.3.2.2 MixedRealityToolkit

Das MRTK stellt unter anderem mehrere Interfaces zur Verfügung, mit denen verschiedene Interaktionen mit Unity Game Objects realisiert werden können. Durch Einbinden in, an Game Object angehängte Skripte, werden Event-basierte Methoden bspw. beim Anwählen durch einen Gaze ausgelöst.

IInputClickHandler

Stellt die Methode `OnInputClicked(InputClickedEventData eventData)` bereit, die bei der Anwendung der Air-Tap Geste auf ein Game Object ausgelöst wird.

IInputHandler

Stellt `OnInputDown(InputEventData eventData)`; zur Verfügung, das bei Auslösen und Halten einer Air-Tap Geste ausgelöst wird. Nach Beendigung der Geste wird `OnInputUp(InputEventData eventData)`; ausgelöst.

IFocusable

Die Methoden `OnFocusEnter()`; und `void OnFocusExit()`; werden ausgelöst wenn ein Gaze auf einem Game Objekt mit diesen implementierten Methoden ausgeführt oder beendet wird.

3.3.2.3 Unity build für Hololens-Anwendungen

Der Build einer Unity Anwendung für die Hololens kann auf zwei Arten geschehen. Zum einen ist es möglich ein Unity Projekt als UWP zu kompilieren und über C# einen Build für die Hololens auszuführen. Zum anderen kann mit dem MRTK direkt in Unity für die Hololens kompiliert werden. Das MRTK enthält ein eigenes Build Window, das alle Konfigurationen für einen Build einstellbar macht, je nachdem ob für die Hololens, oder ein immersive Headset, kompiliert werden soll.

3.3.2.4 Hololens Emulator

Der Hololens Emulator ist ein von Microsoft bereitgestelltes Programm zur Simulation von Hololens-Anwendungen auf Windows PCs. Mit dem Emulator ist es möglich, alle Steuerbefehle der Hololens zu simulieren und somit die Funktionalität eines Hololens Projekts schnell testen zu können, ohne dass ein Deploy auf der Hololens selbst nötig ist.

3.3.2.5 Max/MSP

Durch die Spezialisierung auf Audiosignalverarbeitung und eine erleichterte Konnektivität mit anderen Systemen, eignet sich Max/MSP besonders gut dafür, aus Unity gesendete Steuersignale entgegen zu nehmen und diese zur Erzeugung von Audiosignalen in Echtzeit zu verwenden. Max/MSP stellt eine Reihe von Spezialisierten Objekten für die Audioverarbeitung bereit. Verschiedene Oszillatoren zur Klangerzeugung, Filter und andere Effekte zur Verarbeitung der Signale, Routing Objekte zur Ausgabe und Anpassung sowie Analyse Objekte, die das Sounddesign vereinfachen, stehen nativ zur Verfügung und können in Verbindung mit Objekten aus der klassischen Programmierung (Variablen, Verzweigungen, Schleifen, Arrays, etc.) zur Erzeugung komplexer Klangsynthesen verbunden werden.

3.3.2.6 Verbindung zwischen Unity und Max/MSP

Es ist nicht möglich Max/MSP Builds direkt auf der Hololens zu integrieren, da hierfür immer ein Max/MSP Runtime Environement nötig ist, welches nicht auf der Hololens gestartet werden kann. Max/MSP muss daher auf einem separaten PC gestartet und per UDP mit Unity verbunden werden. Das in 3.3.2.1 erwähnte OSC Package enthält ein Script, das Daten aus Unity heraus per UDP an eine gewünschte IP Adresse senden kann. In Max können diese Daten mit einem "UDPReceive" Objekt auf einem beliebigen Port abgegriffen und verwendet werden. Des weiteren kann Max, da es auf einem separaten PC betrieben werden muss, während der Usability Tests als Kontrollmonitor fungieren. Da man als Beobachter eines Tests mit der Hololens nur schwer Einblicke in die Nutzung der Software erhalten kann, ist es mit Max möglich, Eingaben sichtbar zu machen und somit zu sehen, wie eine Testperson mit dem System zurecht kommt.

3.4 Fazit

Unter Verwendung der vorgestellten Komponenten ist eine Umsetzung des Projekts inklusive aller, durch die Use-Cases und das Prototyping entwickelten Must-Have Anforderungen, theoretisch möglich. Durch Kapselung der einzelnen Komponenten ist ein performantes System erreichbar und zukünftige Erweiterungen durchaus denkbar. Für eine möglichst hohe Bedienbarkeit wurde bereits mit dem Prototyping ein erster Schritt getan. Durch folgendes Usability Testing wird diese noch weiter optimiert werden können.

4 Design

Im Folgenden wird ein Design für das zu entwickelnde System entworfen. Auf Basis der durch die Use Case Analyse und dem Prototyp-Testing entwickelten Anforderungen muss dieses gewisse technische Voraussetzungen erfüllen, um die Anforderungen umsetzen zu können. Hierbei stellt die Entwicklung eines lauffähigen Systems den ersten Teil einer Umsetzung im User-Centered-Design dar. Alle folgenden Schritte wie Usability Tests und die spätere Anpassung des Systems werden in diesem Teil nicht beschrieben.

Auf Grund verschiedener technischer Einschränkungen (beschrieben in 3.3.2.6), die durch die Verwendung der Microsoft Hololens auftreten, ist eine besondere Architektur für dieses System notwendig. Die Klangerzeugung kann nicht auf der Hololens erfolgen und wird auf ein separates System ausgelagert. Auf Grund dessen ergeben sich zwei Hauptkomponenten, das User-Interface und die Klangerzeugung, welche Hardware- und Softwareseitig voneinander zu trennen sind.

4.1 Komponenten

Das zu entwickelnde System besteht im Wesentlichen aus zwei Komponenten, die miteinander über eine Netzwerkverbindung kommunizieren. Zum einen das Augmented Reality User Interface, dass Nutzenden eine Interaktion mit dem System ermöglicht und zum anderen die Klangerzeugung, welche auf einem separaten System im Stand Alone Betrieb läuft und dort Audiosignale erzeugt, welche dann wiederum ausgegeben oder auch weiterverarbeitet werden können. Bei der Übertragung der Daten sind mehrere Möglichkeiten zu erwägen. Sowohl die Übertragung per USB, als auch per Bluetooth oder mittels einer klassischen Netzwerkverbindung sind denkbar. Diese bietet neben einer Einfachen Anbindung von in der Software auf die Möglichkeit die Hololens ohne Kabelverbindung zu verwenden. Mithilfe eines LAN Netzwerks kommunizieren diese beiden Komponenten in Echtzeit, sodass den Anwendenden ein immersives Spielgefühl vermittelt werden kann(siehe Abb. 9).

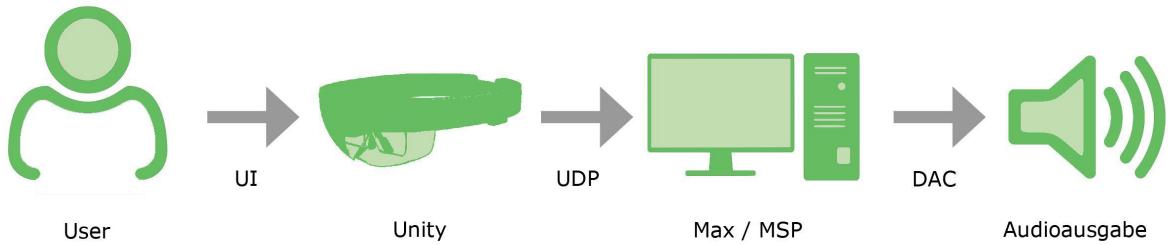


Abbildung 9: Darstellung der einzelnen Komponenten (eigene Darstellung)

4.1.1 User Interface

Das User Interface soll AnwenderInnen ermöglichen, nach Start der Anwendung eines der drei Instrumente zu laden, um mit diesem anschließend möglichst intuitiv und immersiv interagieren zu können. Es wird als Windows UWP Anwendung auf der Microsoft Hololens umgesetzt. Zur Entwicklung dieser Anwendung wird Unity verwendet. Alternativen sind hierbei nicht denkbar, da eine Entwicklung eines so komplexen Programms auf der Hololens im Rahmen dieser Arbeit ausschließlich mit einer 3D Engine wie Unity möglich ist und Unity derzeit die einzige unterstützte 3D Engine für die Hololens darstellt³³. Ein Fadenkreuz zeigt Anwendenden jederzeit an, welches Element sie gerade bedienen. Mit Hilfe von Interfaces und angepassten Skripten des Hololens Toolkit von Microsoft werden die verschiedenen, durch Gesten ansprechbare Elemente der einzelnen Instrumente realisiert, um die entwickelten Anforderungen umzusetzen. Schalter und Taster werden per Air-Tap Geste angesprochen, Elemente wie Schieberegler oder bewegbare Objekte können mit derselben Geste "gehalten" und bedient werden.

Unity Architektur

In der ersten Iteration der Entwicklung, die ausschließlich den darauf folgenden Usability Tests dient, ist jedes einzelne Instrument in einer eigenen Unity Szene angelegt. Eine Umsetzung mit nur einer Szene könnte das Programm unübersichtlich machen. Jede dieser Szenen enthält, neben dem eigentlichen Instrument, alle für die Lauffähigkeit mit der Hololens nötigen Komponenten und alle nötigen Netzwerkkomponenten. Die Instrumente bestehen aus einer Reihe von Game Objects, die einer Unity-typischen hierarchischen Strukturierung unterliegen um möglichst einfach miteinander kommunizieren zu können und mit Zugriffen, wie bspw. "Parent.Transform()", in Relation gebracht werden können. Skripte, die Eingaben der Nutzer verarbeiten, hängen immer an den dazugehörigen Game Objects. Hierbei wäre auch eine Manager basierte

33 Microsoft (2019b)

Architektur denkbar, bei der Skripte in speziellen Gameobjects gesammelt werden. Dies macht Zugriffe auf diese Skripte aber komplexer und erschwert nötige Veränderungen einzelner Elemente. Eine Event-basierte Eingabenverarbeitung ist in beiden Fällen möglich und wird beachtet. Für die Kommunikation werden zwei Game Objects angelegt die jeweils ein Skript enthalten. OSC enthält das OSC.cs Skript mit dem OSC Nachrichten generiert und versendet werden können. Diesem Skript wird das UDPCommunication.cs Skript, welches an dem UDP Game Object angehängt ist, übergeben. Damit werden die OSC Nachrichten per UDP versendet. Alternativen sind hierbei nur schwer denkbar, da die UDP Kommunikation auf der Hololens unter Unity sehr spezifisch erfolgen muss. Alle für die Interaktion mit der Hololens nötigen Game Objects werden als Prefabs aus dem Hololens Toolkit entnommen. Das DefaultCursor Game Objekt und alle ihm unterstellten Objekte sorgen für die Darstellung des Fadenkreuzes. Das Input Manager Objekt enthält eine Reihe an Game Objekten, unter anderem das GesturesInput Objekt, welches die Gestenerkennung der Hololens steuert und das ControllerPointerStabilizer Objekt zum Stabilisieren des 'Gaze'. Das Objekt MixedRealityCameraParent beinhaltet die Kamera, die für die Hololens benötigt wird, und verarbeitet die Steuereingaben, die durch die Kopfbewegung erfolgen. Alternativ wäre denkbar die Funktionen die das Hololens Toolkit bietet selbst zu implementieren. Dieses wäre jedoch sehr zeitaufwändig und würde den Rahmen dieser Arbeit überschreiten.

4.1.2 Klangerzeugung

Die Klangerzeugung findet auf einem separaten PC statt und wird mit Max/MSP im Stand-Alone Betrieb umgesetzt. Denkbar wäre hier auch eine Umsetzung in PureData, das Max/MSP im Grunde stark ähnelt, mittlerweile aber leicht veraltet ist und weniger Möglichkeiten in der Erzeugung und Verarbeitung von Audiodaten bietet. Max/MSP bietet außerdem eine große Community in der viele aktive NutzerInnen Hilfestellungen bei Problemen anbieten. Alle drei Instrumente werden hierbei in einem Patch realisiert. Dieser kann Anweisungen per UDP im OSC Format über Local Area Network vom User-Interface entgegen nehmen. Intern werden diese Anweisungen dann an die einzelnen Synthesizer weitergeleitet, welche daraufhin Töne erzeugen oder Parameter verändern. Anschließend werden die erzeugten Klänge an einen DAC³⁴ weitergeleitet und zur Ausgabe gebracht.

34 Anm.: Digital-to-Analog Converter

Max/MSP Architektur

Die Architektur der Klangerzeugung besteht im Wesentlichen aus drei Sektionen und ist, wie für Max/MSP typisch, datenstromorientiert aufgebaut (siehe Abb. 10). Am Anfang steht das UDP Receive Objekt über das alle Steuerdaten des User Interface in den Patcher gelangen. Anschließend werden diese je nach Adresse auf eine der drei Instrumentsektionen verteilt. Denkbar wäre hier auch die übergeordnete Aufteilung in Steuerung und Klangerzeugung. Die beiden Sektionen könnten dann jeweils in drei einzelne Teile pro Instrument aufgeteilt werden. Dieses Design würde sich im Grunde aber kaum von dem oben gewählten unterscheiden, da jede Instrumentensektion nun jeweils eine eigene Sektion für Steuerung und Klangerzeugung besitzt.

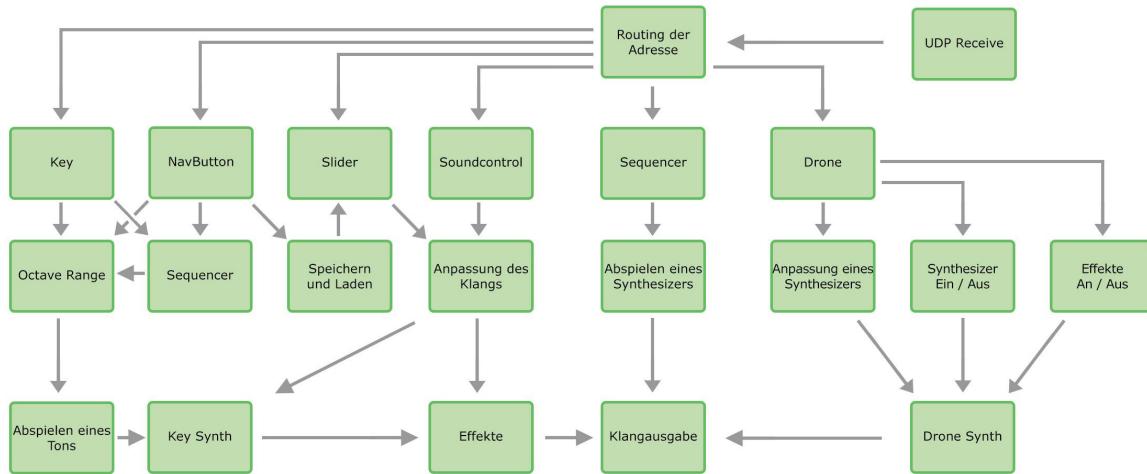


Abbildung 10: Schematische Darstellung der Architektur von Max/MSP (eigene Darstellung)

Keypad

Die vier vom User-Interface kommenden Adressen Key, NavButton, Slider und SoundControl (Siehe Abb.) sind für die Steuerung des KeyPads zuständig. Der Befehl "Key" besitzt immer eine nachfolgende Ganzzahl zwischen 0 und 26, welche mit dem derzeitig gewählten Oktavenbereich verrechnet, einen passenden MIDI Wert ergeben. Dieser wird den fünf zur Verfügung stehenden Synthesizer Objekten übergeben, welche die zum MIDI Wert passende Frequenz annehmen. Der Key Befehl löst außerdem die Hüllkurve aus, die das Signal der Synthesizer mit Werten zwischen 0 und 1 multipliziert um sie hörbar zu machen und somit bestimmt, welche Länge und Form ein Ton hat. Befehle mit der Adresse "NavButton" können abhängig vom anschließenden Ganzzahl

Wert drei verschiedene Funktionen erfüllen. "0-1" lädt und speichert die Schieberegler Werte. "2" startet und stoppt den Sequencer des Instruments. "3-4" Stellt den Oktavenbereich ein. Dieser besteht aus einem Ganzzahl-Wert, der bei Erhalt von "Navbutton 3 / 4" um zwölf erhöht oder verringert wird um eintreffende "Key" Werte um eine Vielzahl von zwölf Halbtönen (eine Oktave hat 12 Halbtöne) zu erhöhen. Nach Erhalt von "Navbutton 2" wird der Sequencer gestartet. Alle eintreffenden Key Werte werden in eine Liste gespeichert, die mithilfe eines metro Objekts in festgelegten zeitlichen Abständen wiederholt ausgelesen wird. Die ausgelesenen Werte ergeben die Sequenz, die wiederum über die Octave Range an die Synthesizer weitergeleitet werden, sodass laufende Sequenzen auch im Nachhinein in ihrer Oktave verändert werden können. Die Nachrichten "Navbutton 0 / 1" speichern und laden die Schiebereglerwerte in einem Speicherplatz. Bei mehrmaligem Erhalt eines Speicherbefehls werden die alten Werte überschrieben, sodass beim Laden immer die zuletzt gespeicherten Werte ausgelesen werden. Die Schiebereglerwerte werden mit der Nachricht "Slider", einer Ganzzahl zwischen 0 und 6 und einer anschließenden Gleitkommazahl versendet. Die Ganzzahl unterscheidet die einzelnen Schieberegler und die Gleitkommazahl stellt den neuen Wert dar. Die Werte werden nach dem Versenden als Ganzzahlen zwischen 1 und 256 in 7 unterschiedliche Zahlen-Felder gespeichert. Die Slider Adressen "0" und "1" geben die Länge des Attacks und Releases der Hüllkurve in Millisekunden an. Adresse "2" wird als Frequenz eines Sinus Oszillators eingestellt, der durch Frequenzmodulation den Carrier³⁵ moduliert und somit die Klangfarbe anpasst. Adresse "3" wird in einen veränderten Wertebereich als Filter Frequenz und Resonanz Intensität eines Low Pass Filters umgerechnet. Adresse "4" steuert ebenfalls mit umgerechneten Werten den Grad der Verzerrung eines Overdrive Blocks. Adresse "5" wird zu Werten zwischen 1 und 20 umgerechnet, welche dann die Frequenz eines Low-Frequency-Oszillators darstellt, um einen Vibrato Effekt zu erzeugen. Adresse "6" wird umgerechnet zu Werten zwischen 0 und 1 und mit dem Signal multipliziert um die Lautstärke zu variieren. Die Nachricht "SoundControl" ist immer gefolgt von zwei Ganzzahlen. Erstere ist eine Adressierung mit Werten zwischen 0 und 4, welche dazu dient den zweiten Wert an den passenden Klangerzeuger zu senden. Der zweite Wert wählt innerhalb der einzelnen Klangerzeuger Oszillatoren mit unterschiedlichen Wellenformen aus, welche anschließend mit dem frequenzmodulierbaren Teil der Klangerzeugung verrechnet werden. Die zweiten Werte sind zwischen 0 und 3. Wert "0" stellt den Klangerzeuger aus. Wert "1" wählt den Sinus, Wert "2" den Sägezahn und Wert "3" den Rechteck Oszillator eines einzelnen Klangerzeugers aus. Neben einer dieser drei Oszillatoren beinhaltet jeder Klangerzeuger noch einen weiteren frequenzmodulierten Sinus Oszillator, und einen Oszillator, den

35 Anm.: Oszilator der die Trägerfrequenz einer Frequenzmodulation erzeugt

reinen Sinus Oszillator. Alle drei schwingen in ihrer Trägerfrequenz immer mit der Frequenz passend zur gespielten Taste.

Sequencer

Da viele Funktionen des Sequencers im User Interface stattfinden, muss die Klangerzeugung lediglich die einzelnen Instrumente bereitstellen, welche alle durch eine Nachricht angesprochen werden können. Die Adresse "sampler" beinhaltet immer acht binäre Werte (1 oder 0). Jeder Einzelne steht für ein Instrument. Beim Erhalt einer "1" wird dieses getriggert und spielt einen Ton ab. Bei der Klangerzeugung handelt es sich um acht verschiedene Synthesizer, die durch verschiedene Methoden der additiven und subtraktiven Klangsynthese gängige Klänge eines Drum-Computers³⁶ erzeugen. Die acht Klänge simulieren eine Bass Drum, eine Snare, geschlossene Hihats, offene Hihats, eine tiefe und eine hohe Tom, ein Becken und eine Tieffrequenz.

Drone Generator

Mit der Adresse "sphere" gelangen eingehende Daten in die Sektion des Drone Generators. Diese besteht aus fünf gleichen Synthesizern, deren Ausgangssignale gemischt werden. Der Adresse folgen immer Werte zwischen 0 und 6, welche die einzelnen Funktionen der Klangerzeuger adressieren. Die Adressen 0-4 tragen nachfolgend immer drei Werte zwischen 1 und 256 welche je nach Adresse auf die fünf Klangerzeuger verteilt werden. Die nachfolgenden Werte stellen unterschiedliche Parameter in der Klangerzeugung ein. Der erste Wert wird als Frequenz eines simplen sinus Oszillators eingestellt. Der Zweite stellt die Lautstärke des Klangerzeugers ein. Der dritte Wert wird ebenfalls als Frequenz eines Sinus Oszillators genutzt, welcher sich aber in der Effektsektion des Klangerzeugers befindet. Diese wird durch eine Nachricht mit der Adresse "6" eingeschaltet. Die Effektsektion sendet Schritt-Zufallswerte³⁷ im niederfrequenten Bereich (zwischen 0 und 20) an zwei Oszillatoren, die dadurch in unterschiedlichen Geschwindigkeiten als Schwingung wahrnehmbare Frequenzen erzeugen, welche mit den Trägersignalen multipliziert werden. Des Weiteren wird das Ausgangssignal der Synthesizer beim Erhalt der Adresse "6" durch ein Delay und ein Reverb Objekt gesendet. Eine Adresse "5" schaltet die Klangerzeuger durch Multiplizieren einer 1 oder einer 0 mit dem Ausgangssignal ein oder aus.

36 Anm.: System zur auditiven Simulation von Schlagzeugen

37 Anm.: Ein Wert wird zufällig entweder um eine Zahl erhöht oder verringert

4.2 Instrumente

4.2.1 Keypad

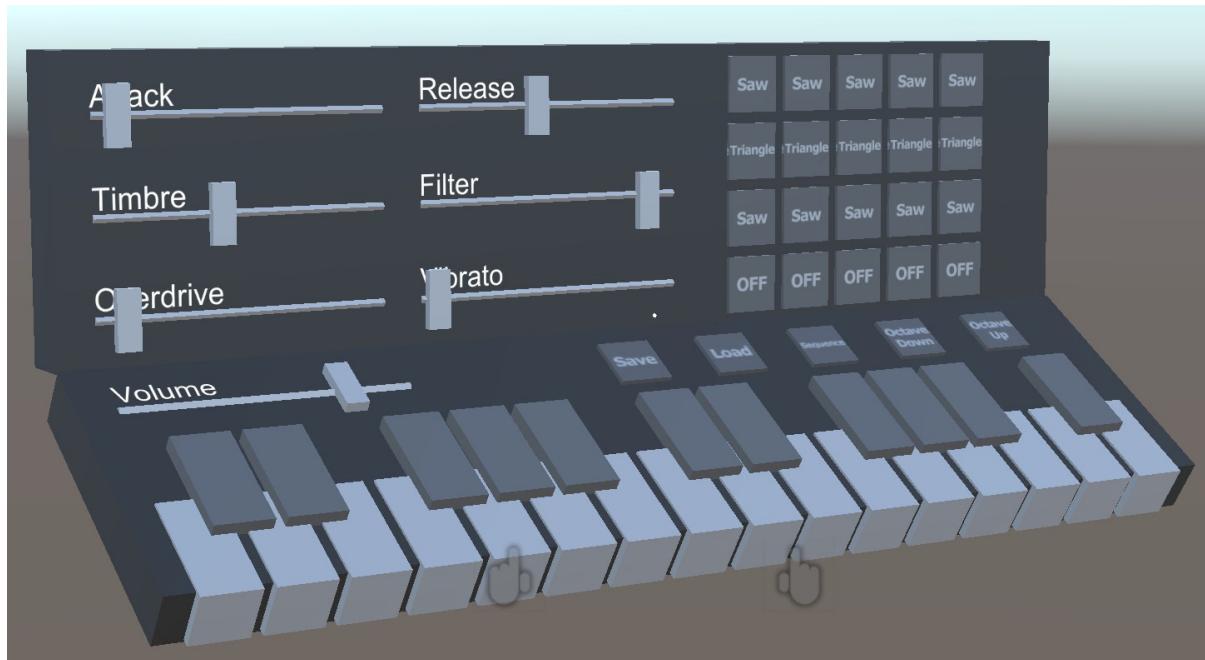


Abbildung 11: Keypad der ersten Iteration (eigene Darstellung)

Das Keypad wird auf Basis der entwickelten Prototypen und der anschließenden Analyse dieser entwickelt. Es zeigte sich, dass eine vergrößerte Bedienoberfläche deutlich ansprechender für die BenutzerInnen ist und dass gewisse Nutzungselemente gewünscht sind. Es kommt eine klassische Klaviatur zur Klangzeugung zum Einsatz, die durch einfaches Bedienen mit der Air-Tap Geste ausgelöst wird. Direkt über der Klaviatur befindet sich ein Schieberegler zur Anpassung der Lautstärke und fünf Schalter zur Steuerung verschiedener Funktionen. Diese Funktionen sind von Links nach Rechts: Speichern der Schieberegler Einstellungen; Laden der Schieberegler Einstellungen; Starten einer Sequenz; Klaviatur um eine Oktave nach unten transponieren; Klaviatur um eine Oktave nach oben transponieren. Direkt über den Funktionstasten befindet sich die Klangmatrix. Sie enthält fünf Spalten mit denen jeweils zwischen verschiedenen Oszillator-Arten umgeschalten werden kann. Jede Spalte spricht in der Klangzeugung einen separaten Klangzeuger an, sodass die unterschiedliche Klänge geschichtet werden können. Links neben der Matrix befinden sich die Schieberegler zur Veränderung des Klangs. Die sieben Schieberegler steuern Attack und Release der Hüllkurve, Timbre zur Veränderung der Klangfarbe mithilfe von Frequenz Modulation, Filter steuert die

Filterfrequenz und Resonanz-Intensität eines Low Pass Filters, Overdrive steuert den Grad der Verzerrung, Vibrato die Geschwindigkeit eines Low-Frequency-Oszillators um einen Vibrato Effekt zu erzeugen und Volume steuert die Lautstärke des Klangs. Jede Interaktion mit dem Instrument wird über die Air-Tap Geste realisiert. Bei den Schiebereglern muss diese während des Einstellens des Wertes gehalten werden. Da die Klangerzeugung bei diesem Instrument hauptsächlich durch Interaktionen der NutzerInnen (Bedienen der Tasten und Schiebereglern mit der Air-Tap Geste) ausgelöst wird, kommt eine Ereignis-basierte Architektur zum Einsatz. Beispielsweise beim Betätigen einer Taste, wird Code-seitig ein Event ausgelöst, das eine Nachricht an die Klangerzeugung sendet um den dazugehörigen Klang abzuspielen. Die Klangerzeugung besteht aus fünf frequenzmodulierbaren Synthesizern, welche beim Betätigen einer Klaviertaste mit einer Hüllkurve multipliziert werden. Anschließend geht der erzeugte Ton durch die verschiedenen Effektgruppen (bspw. Verzerrung, Filter, etc.)

4.2.2 Sequencer

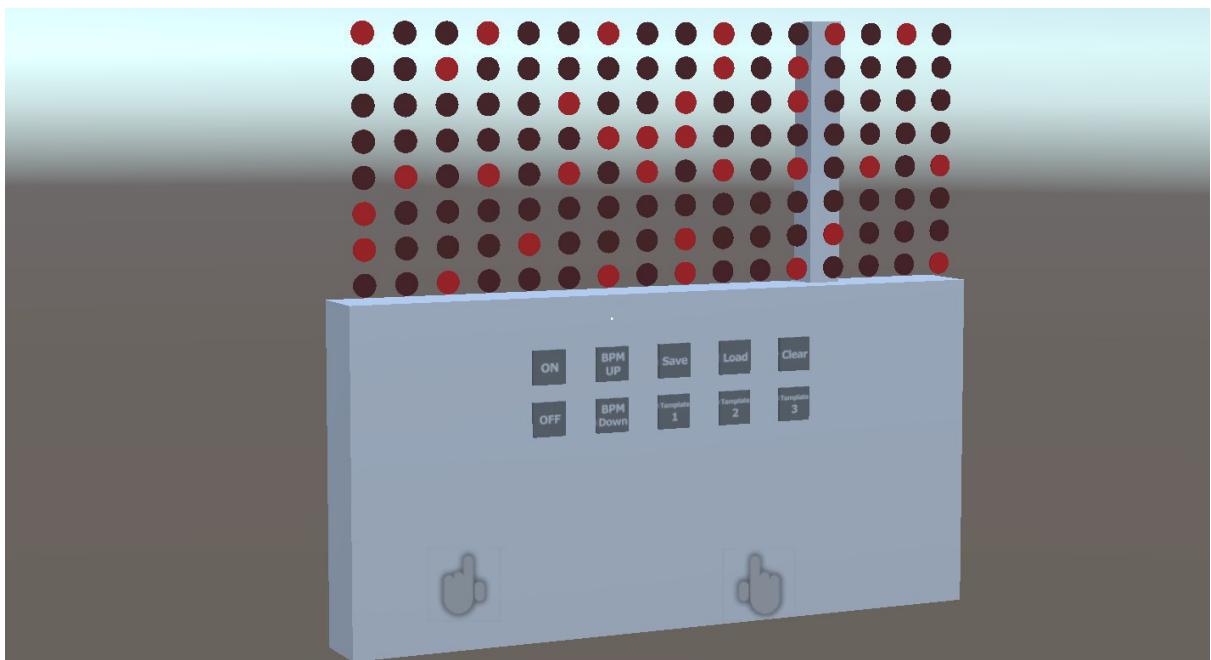


Abbildung 12: Sequencer der ersten Iteration (eigene Darstellung)

Beim Sequencer werden aufgrund der erhöhten Akzeptanz des ersten Prototypen große Teile dessen Designs übernommen. Das Bedienfeld mit den Auswahltastern wird um fünf weitere Taster erweitert. Insgesamt erfüllen diese folgende Funktionen: Starten und Stoppen der Sequenz, Geschwindigkeit der abgespielten Sequenz erhöhen und

verringern. Ein Schalter zum Speichern, einer zum Laden von Sequenzen. Diese beiden Schalter funktionieren in Verbindung mit den darunter liegenden Schaltern. Nach Betätigen einer der beiden Schalter muss anschließend einer der drei Speicherplätze gewählt werden, um die Sequenz darin zu speichern oder eben daraus zu laden. Ein weiterer Taster leert die Partitur. Die einzelnen Elemente der Partitur lassen sich per Air-Tap Geste anwählen. Ein angewähltes Partiturelement leuchtet wenn es angewählt wurde in einem helleren Rotton und erzeugt beim Durchlaufen der Partitur, erkenntlich gemacht durch die über die Partitur laufende Leiste, einen Ton, der sich pro Zeile unterscheidet. Der Sequenzer sendet pro Partitur Schritt automatisch eine Ausgabeanweisung an die Klangerzeugung.

4.2.3 Drone Generator

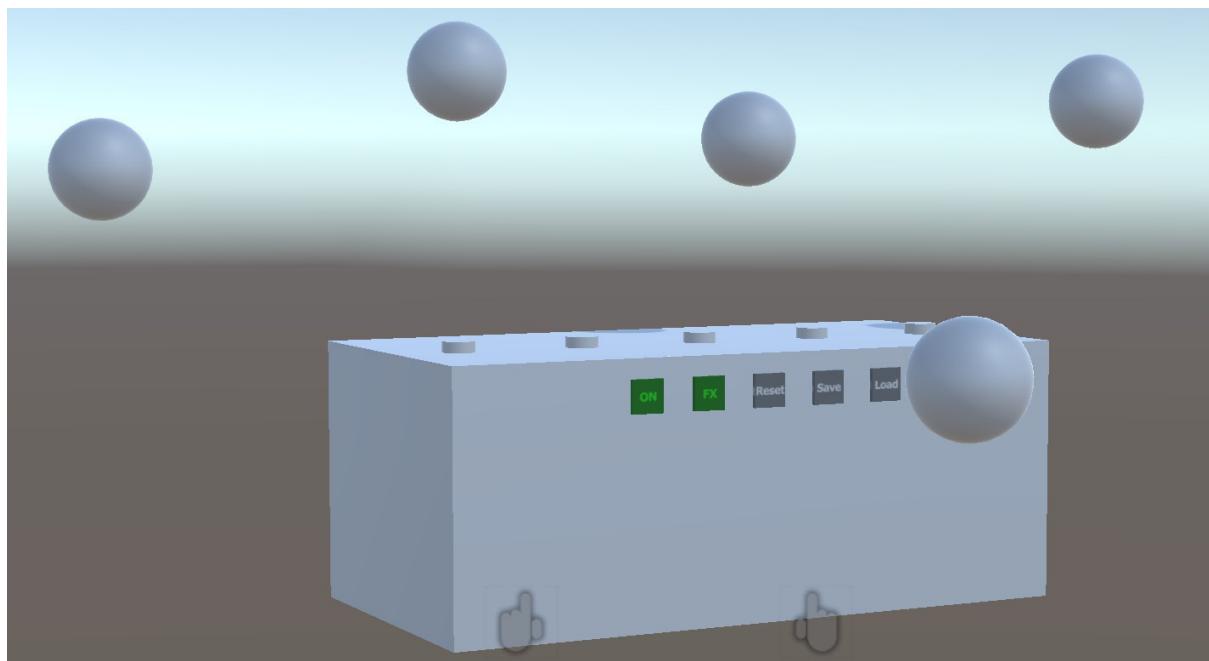


Abbildung 13: Drone Generator der ersten Iteration (eigene Darstellung)

Beim Drone Generator wird aufgrund der erhöhten Akzeptanz des ersten Prototypen eine angepasste Version dessen Design umgesetzt. Das Instrument hat fünf frei im Raum positionierbare Kugeln, welche, durch ein Anwenden und Halten der Air-Tap Geste, anschließend dem Fadenkreuz folgen und somit den erzeugten Klang durch ihre Position verändern. Jede Kugel spricht dabei einen eigenen Klangerzeuger an. Diese fünf erzeugten Töne können zusammen eine Vielzahl unterschiedlicher Bordun³⁸ Klänge

38 Anm.: einen meist tiefen Halteton.

erzeugen. Am Vorderen Teil des Instruments befinden sich fünf Funktionstasten die wie folgt belegt sind: Ton Ein-/Ausschalten, Effekte An-/Ausschalten, Kugeln in Ausgangsposition bringen, Aktuelle Position speichern, zuletzt gespeicherte Position laden. Der Drone Generator sendet auf der User Interface Seite durchgehend alle Positionsdaten der fünf Kugeln an die Klangerzeugung sobald der Ton aktiviert wurde. In der Klangerzeugung stehen fünf Instanzen eines Synthesizers bereit, welche die Werte entgegennehmen und mit ihnen durch Frequenzmodulation zweier Oszillatoren und der Lautstärke der einzelnen Instanzen den Klang verändern. Das Benutzen des "FX" Tasters leitet den erzeugten Klang durch einen Effektweg, welcher ein Delay und einen Hall beinhaltet. Des Weiteren werden durch die "FX" Funktion die Frequenzmodulierten Oszillatoren durch Zufallsdaten leicht verändert, was den Klang lebendiger gestalten soll.

4.3 Kommunikation

Wie bereits in 4.2 erwähnt, wird die Verbindung des User-Interface und der Klangerzeugung mittels eines Local Area Network realisiert. Als Netzwerkprotokoll wird UDP verwendet. Da die stabile Lauffähigkeit der beiden Komponenten nicht von der Verbindung abhängig ist und ausschließlich Steuerdaten versendet werden, eignet sich UDP sehr gut, da es schnell ist, eventuelle Übertragungsfehler verkraftbar sind, es sowohl von Unity als auch von Max/MSP versend- und empfangbar ist und mit dem Open Sound Control Protocol kompatibel. Dieses wird verwendet, da das in Max/MSP zum Einsatz kommende Modul "udpreceive" ausschließlich OSC Daten entgegen nimmt.

4.3.3 Befehlsstruktur

Die Befehlsstruktur der Steuerdaten folgt einem simplen Muster, dass immer aus Nachrichtenadresse und einer variablen Anzahl an Nachrichtenwerten besteht. Als Adresse wird in Unity immer der Name des Objektes das sendet (bspw. Slider 3) verwendet. So können die einzelnen Nachrichten leicht unterschieden werden. Außerdem kann die gesendete Nachricht durch diese Form der Adressierung in Max/MSP leicht an die richtige Stelle geroutet und verarbeitet werden. Des Weiteren benötigen lediglich die Nachrichten der Schiebereglern des Keypads und die der Kugeln des Drone-Generators Werte, da alle Schalter durch alleiniges senden der Adresse binär geschaltet werden.

5 Implementierung

5.1 Technische Umsetzung

Im Folgenden werden die wichtigsten Bestandteile der Programmierung der Steuerung und Netzwerkverbindung des User-Interfaces, sowie der Datenverarbeitung und Klangsynthese der Klangerzeugung beschrieben. Auf eine allumfängliche Beschreibung des Codes wird aufgrund der für das Verständnis dieser Arbeit vorausgesetzten Programmierkenntnisse sowohl in Unity als auch in Max/MSP verzichtet.

5.1.1 User-Interface

Die einzelnen Instrumente werden in Unity in eigenen Szenen entwickelt. Jedes Instrument besteht hierbei aus einem parent Gameobject und einer Vielzahl an weiteren child Gameobjects, welche teils mit Skripten versehen sind, um gewisse Funktionalitäten realisieren zu können. Alle Funktionstasten wurden mit eigens angefertigten Materials versehen um eine Beschriftung dieser zu ermöglichen. Neben den Instrumenten enthält jede Szene eine Vielzahl an weiteren Gameobjects, die teilweise für die Netzwerkverbindung mit der Klangerzeugung und teilweise für die korrekte Darstellung und Steuerung auf der Hololens zuständig sind.

5.1.1.1 Steuerung

Alle Interaktionen der NutzerInnen mit dem System erfolgen über das Anwählen von Gameobjects über den Gaze und das Auslösen derer Funktionen mit der Air-Tap Geste. Das Hololens Toolkit stellt hierfür einige fertige Prefabs zur Verfügung. Auf Basis dieser, wurde die gesamte Funktionalität des User Interfaces realisiert. Alle Funktionen die aus Prefabs des Hololens Toolkit hervorgehen, werden in 5.1.3 beschrieben. Um eine Funktionalität auszulösen, sobald Nutzende die Air-Tap Geste verwenden, kommt das Interface `IInputClickHandler` zum Einsatz. Dieses stellt die Funktion `OnInputClicked()` zur Verfügung, welche bei jeder Air-Tap Geste auf ein Gameobject mit einem Skript, dass dieses enthält, ausgelöst wird. Jedes einfach Klickbare Objekt enthält diese Funktion. Da das Keypad alle Funktionalitäten der Buttons an die Klangerzeugung weiterleitet, werden diese über ein einzelnen Skript `ButtonScript.cs` realisiert. Dieses Skript wird durch Hinzufügen von zwei public boolean Werten `Toggle`

und `Key` variabel gestaltet. So führt ein `True` Wert des `Toggle` Feldes bei einem Game Object mit dem `ButtonScript.cs` dazu, dass ein einmaliger Klick den Button grün leuchten lässt bis ein zweiter Klick durchgeführt wurde. Ohne `Toggle` leuchtet der Knopf nur kurz auf um eine Taster Funktion zu vermitteln. Bei einem `True` Wert des `Key` Feldes wird die farbliche Reaktion komplett abgestellt, da die Tasten der Klaviatur bereits eine auditive Reaktion zeigen. Lediglich die Speicher- und Ladeknöpfe enthalten neben dem `ButtonScript.cs` jeweils ein weiteres Skript, für die Speicherung und Wiederherstellung der Positionen der Slider Game Objects. Bei den beiden anderen Instrumenten werden durch die Buttons oft spezifische Funktionen innerhalb Unity's umgesetzt, weshalb hier pro Button ein eigenes Skript zum Einsatz kommen muss. Neben diesen enthalten alle Buttons des Sequencers und des Drone Generators das Skript `BtnBlink.cs` welches sich ähnlich wie im `ButtonScript.cs` um die optische Reaktion der Gameobjects kümmert. Auch eine `Toggle` Funktion wurde hier realisiert um für unterschiedliche Reaktionsarten dasselbe Skript verwenden zu können.

Die Funktionalität der Schieberegler wird über das an den Reglern befindliche `SliderScript.cs` realisiert. Dieses ist ein abgewandeltes Skript aus dem Hololens Toolkit, das ein Greifen und Verändern der Position eines Gameobjects realisiert. Das Verändern der Position ist bei den Schieberegler auf die X Achse beschränkt und darf nur in einem eingeschränkten Bereich möglich sein, da die Regler sonst über ihren Bereich hinaus bewegbar wären. Bei jedem Verändern der Position sendet das `SliderScript.cs` eine Nachricht mit der Adresse des Gameobject Namens (bspw. Slider 3) und einen Gleitkommawert, welcher auf Werte zwischen 1 und 256 umgerechnet wurde. Beim Laden und Speichern der Schieberegler Werte wird auf der User-Interface Seite die aktuelle Position des Gameobjects im `SliderScript.cs` gespeichert, sodass sie beim Laden wieder ausgelesen und auf den Transform des Gameobjects angewendet werden kann.

Alle an den Funktionstasten befindlichen Skripte des Sequencers greifen auf eine von zwei Funktionalitäten des Instruments zu. Die Tasten "On", "Off", "BpmUp" und "BpmDown" steuern hierbei die Funktionen des "Mover" Game Objects. Dieses enthält das Skript `MoveScript.cs`, welches wiederum eine Coroutine `MoveBlock()` enthält. Die Coroutine sendet, wenn das boolsche Feld `activated` durch Betätigung des "On" Tasters `True` gestetzt wurde, in einstellbaren Zeitabständen eine Nachricht mit der Adresse "sampler" und den dazugehörigen Werten an die Klangerzeugung. Des Weiteren bewegt die Coroutine das Gameobject durch einen Transform Befehl so über den Sequencer, dass dieses einen optischen Indikator der gerade abgespielten Spalte darstellt. Die

Zeitabstände lassen sich durch die beiden Funktionstasten "BpmUp" und "BpmDown" einstellen. Beim Betätigen greifen diese auf das "Mover" Gameobject zu und verändern den Integer Wert `bpm` entsprechend. Der Off Button stellt das Feld `activated` wiederum auf `False`, was die Funktion der Coroutine ausstellt. Die Tasten "Save", "Load", "Reset", "Template 1", "Template 2" und "Template 3" agieren allesamt mit der Partitur. Save und Load enthalten simple Skripte, welche jeweils einen booleschen Wert setzen. Ist der Save Wert `True` gesetzt worden, speichert ein Auslösen eines der Template Tasters alle Trigger Elemente der Partitur in einer Matrix. Selber Vorgang mit dem Load Taster, nimmt die in der Matrix enthaltenen Werte und setzt die Elemente der Partitur dementsprechend. Jedes Element der Partitur enthält ein Skript, welches das Element beim Auslösen über einen booleschen Wert "aktiviert". Aktivierte Elemente erhalten einen helleren Rotton und speichern eine 1 in einer 16x8 Matrix an der Position, die sich aus den zwei im Namen enthaltenen Nummern zusammensetzt. Dadurch repräsentiert die Matrix alle Elemente der Partitur in ihrer Position und ihrem Aktivierungszustand. Die vom Mover Block gesendeten Werte werden spaltenweise aus dieser Matrix gelesen, sodass ein Werteset immer 8 Werte (0 oder 1) enthält.

Die an den Funktionstasten befindlichen Skripte des Drone Generators senden teilweise nur Nachrichten an die Klangerzeugung, ähnlich wie dem `ButtonScript.cs`, teilweise enthalten sie aber auch Unity interne Steuerbefehle. Der "On" Taster greift auf alle Skripte die an den "sphere" Objekten hängen zu und aktiviert sie über einen booleschen Wert. Dadurch senden die Skripte anfänglich eine Nachricht mit der Adresse "sphere" und dem nachfolgenden Wert 5 an die Klangerzeugung, was diese aktiviert. Bei jeder Veränderung der Position ihrer Gameobjects, senden die einzelnen sphere Objekte dann eine Nachricht mit ihrem Namen (bspw. `sphere 2`) und ihrer X, Y und Z Positionsvaluen. Diese werden vor dem Versenden in einen Bereich zwischen 1 und 256 umgerechnet, sodass die Klangerzeugung sie verarbeiten kann. Die "sphere" Objekte werden wie die Schieberegler ebenfalls durch ein abgewandeltes Skript aus dem Hololens Toolkit bewegt. Durch das Halten der Air-Tap Geste auf ihnen, sind sie frei im Raum positionierbar. Wird ein Objekt von einem anderen "sphere" Objekt berührt bewegt sich dieses frei im Raum, sodass sich der Klang ohne Zutun verändert. Das angehängte `sphereScript.cs` enthält neben den Funktionen für das Greifen und Halten der Objekte weitere Funktionen, die für die Funktionalität des Instruments nötig sind. Die "FX" Funktionstaste sendet eine Nachricht mit der Adresse "sphere" und dem nachfolgenden Wert 6 an die Klangerzeugung. Die "Reset" Taste bringt alle Sphere Objekte zurück in ihre Ausgangspositionen. Dies geschieht ebenfalls im `sphereScript.cs` durch die

Funktion `LoadPositions()`. Mit ihr wird auch das Laden, von durch die NutzerInnen gespeicherten Positionen der sphere Objekte, realisiert, welches durch den "Load" Taster ausgelöst wird. Der "Save" Taster ruft die im `sphereScript.cs` befindliche Methode `SavePositions()` auf.

5.1.1.2 Hololens

Für die korrekte Darstellung der Szenen und die Steuerung durch die Hololens wurden Prefabs des Hololens Toolkits verwendet. Dabei enthält jede Szene folgende Prefabs: "DefaultCursor" für die Anzeige des Fadenkreuzes. "InputManager" für alle Steuereingaben die von der Hololens ausgehen. Sowohl Kopfbewegungen, als auch die Gestensteuerung, wird hier umgesetzt. "MixedRealityCameraParent" ist die Standard Kamera für die Entwicklung von Hololens-Anwendungen. Die Prefabs "InputManager" und "MixedRealityCameraParent" sind voneinander abhängig und greifen an vielen Stellen aufeinander zu. Die Instrumente greifen über die Interfaces `IFocusable`, `IInputHandler`, `ISourceStateHandler` und `IInputClickHandler` auf die Funktionalitäten der Prefabs zu.

5.1.1.3 Netzwerkverbindung

Für die Netzwerkverbindung stehen in jeder Szene zwei Gameobjects zur Verfügung. "OSC" kümmert sich hierbei um die korrekte Formatierung der versendeten Nachrichten. Das enthaltene `OSC.cs` beinhaltet neben der Klasse OSC eine weitere Klasse `OscMessage`. Diese enthält die Felder `adress` und `values` mit denen eine Nachricht versehen werden kann. Die Klasse OSC enthält die Funktion `Send(OscMessage)` welche die übergebene Nachricht in ein byte Array mit dem korrekten Format einer OSC Nachricht(OSC-String: ASCII-Zeichenfolge mit einer durch 4 teilbaren Gesamtlänge) wandelt. Über das "UDP" Gameobject werden die Nachrichten dann mittels eines asynchronen Outputstreams über einen Datawriter an die gewünschte IP Adresse mit dem gewünschten Port versendet. Hierbei verlangt die Hololens einen sehr spezifischen Code, welcher online in mehreren Foren als die einzige funktionale Lösung für UDP Kommunikation mit der Hololens gilt³⁹. Große Teile des Codes sind Unity nicht bekannt und werden deshalb durch Präprozessoranweisungen(`#if !UNITY_EDITOR`) erst auf der Hololens aktiviert.

39 Microsoft Mixed Reality Developer Forum (2017)

5.1.2 Klangerzeugung

Die gesamte Klangerzeugung wird auf einem separaten System in einer Stand-Alone Max/MSP Runtime umgesetzt. Es gibt einen Haupt Patcher in dem sich mehrere unterpatcher befinden, welche allesamt Synthesizer für die einzelnen Instrumente darstellen.

5.1.2.3 Datenfluss

Nach Eintreffen einer Nachricht im `udpreceive` Objekt wird diese mithilfe eines `route` Objekts an das richtige Instrument gesendet. `route` Objekte steuern den Datenfluss auch innerhalb der Objekte bspw. zur Unterscheidung einzelner Synthesizer. Neben diesen kommen auch `switch` Objekte zur Auswahl verschiedener Objekte und zum Ein und Ausschalten bestimmter Pfade. In Kombination mit `gate` Objekten können so anpassbare Signalpfade erzeugt werden.

5.1.2.1 Datenanpassung

Manche eingetroffenen Nachrichten müssen in ihrem Format angepasst werden, da sie vom User Interface teilweise nicht optimiert versendet werden können. Dazu kommen reguläre Ausdrücke, mit denen bspw. Teile von Strings entfernt oder angepasst werden können, zum Einsatz. Mit Hilfe von `expr` Objekten können mathematische Ausdrücke realisiert werden. Dies dient meist dazu, die eingetroffenen Werte, welche immer zwischen 1 und 256 sind, in einen angepassten Bereich (bspw. 0 bis 1) zu bringen.

5.1.2.2 Synthesizer

Alle Synthesizer befinden sich in Subpatchern. Alle Subpatcher, von denen mehr als eine Instanz existiert, werden mit dem `patcher` Objekt erzeugt, was bei den Synthesizern des Keypads und des Drone Generators geschieht. Die Subpatcher, von denen nur eine Instanz benötigt wird, werden vorab in eigenen Patchern geschrieben und dann im Hauptpatcher instanziert. Die fünf Klangerzeuger des Keypads erzeugen Töne durch additive Klangsynthese. Hierbei werden mehrere Oszillatoren teils parallel, teils seriell, miteinander verkettet. Die serielle Verkettung sorgt für eine Frequenzmodulation. Neben der Klangsynthese enthalten die Synthesizer des Keypads jeweils eine eigene Hüllkurve, die mithilfe eines `line~` Objekts verwirklicht werden. Durch einen zeitversetzten Anstieg auf eine 1 und eine anschließende Absenkung auf 0 über einen gewissen Zeitraum,

können Töne geformt werden. Dabei wird der Ausgabewert der Hüllkurve mit dem Signal der Synthesizer multipliziert. Die acht Instrumente des Sequencers erzeugen ihre Klänge durch subtraktive Synthese. Hierbei werden Signale von Oszillatoren und Noise Generatoren durch Filterung bestimmter Frequenzen geformt. Hierbei kommen oft mehrere Filter zum Einsatz, um einen Klang in verschiedene Frequenzbereiche aufzuteilen. Da die Instrumente des Sequencers perkussive Eigenschaften aufweisen sollen, werden teils mehrere Synthesizer mit mehreren Hüllkurven verwendet um eine komplexere Klangdynamik erzeugen zu können. Der Drone Generator besitzt fünf Instanzen eines Synthesizers mit additiver Klangsynthese. Hierbei modulieren zwei Oszillatoren mit niederfrequenten Schwingungen den Hauptklangzeuger. Diese können zusätzlich kontinuierlich mithilfe von `drunk` Objekten in ihrer Frequenz angepasst werden, was das Klangbild immer wieder verändert, um somit interessant für die NutzerInnen zu bleiben.

5.1.3 Fremdcode

Im Folgenden wird der gesamte Fremdcode der in diesem System verwendet wurde aufgeführt. Die in Unity und Max MSP in standard Installation zur Verfügung stehenden Objekte und Skripte werden hier nicht erwähnt.

Hololens Toolkit

Das Hololens Toolkit ist komplett im Unity Projekt importiert. Genutzt werden davon die Interfaces `IFocusable`, `IInputHandler`, `ISourceStateHandler` und `IInputClickHandler`, die Prefabs `DefaultCursor`, `InputModule`, `MixedRealityCameraParent` und eine abgewandelte Version des Scripts `HandDraggable.cs`. Des Weiteren wird der vom Hololens Toolkit bereitgestellte Reiter `MixedRealityToolkit` zum Konfigurieren und Build der Anwendung genutzt.

UDP

Für die Kommunikation per UDP wird eine abgewandelte Version des Skripts `UDPCommunication` verwendet⁴⁰.

40 Microsoft Mixed Reality Developer Forum (2017)

OSC

Für die Erzeugung von Nachrichten im OSC Format wird eine abgewandelte Version des OSC Prefabs verwendet⁴¹.

5.2 Evaluation

Im Folgenden wird die gesamte Realisierung auf ihre Funktionalität und Benutzbarkeit hin überprüft. Hierzu wird das umgesetzte System und dessen Verwendung mit den in 3.2.3 gestellten Anforderungen verglichen.

5.2.1 Anforderungen

Funktionale Anforderungen

Bedienbare Taster:

Diese Anforderung wurde erfüllt. Nutzende können Objekte per Air-Tap Geste bedienen und dadurch Funktionen auslösen.

Schieberegler:

Diese Anforderung wurde erfüllt. NutzerInnen können Schieberegler durch Halten der Air-Tap Geste bewegen und damit Werte anpassen.

Greifbare Objekte:

Diese Anforderung wurde erfüllt. Nutzende können die Sphere Objekte des Drone Generators mit Halten der Air-Tap Geste frei im Raum positionieren und dadurch den Klang des Instruments verändern.

Klangerzeugung des Keypads:

Diese Anforderung wurde erfüllt. Die Klangerzeugung des Keypads ist in ihrem vollen Umfang implementiert. Nutzende können Klänge auslösen und sie über die Schieberegler und die Klangmatrix verändern.

Klangerzeugung des Sequencers:

Diese Anforderung wurde erfüllt. NutzerInnen können durch Anwählen der

41 UnityOSC(2019)

Partiturelemente Klänge für das Abspielen des Sequencers auswählen, welcher diese dann abspielt.

Klangerzeugung des Drone Generators:

Diese Anforderung wurde erfüllt. Durch Positionieren der "sphere" Objekte können NutzerInnen die Klänge des Drone Generators verändern.

Speichern und Laden von Zuständen:

Diese Anforderung wurde erfüllt. Alle Instrumente haben eine speicherbare Sektion, die von Nutzenden durch Betätigen von "Save" und "Load" Taster gespeichert und in ihren gespeicherten Zustand zurückversetzt werden kann.

Optisches Feedback der Taster:

Diese Anforderung wurde erfüllt. Alle Taster besitzen ein optisches Feedback in Form von einer kurzen Veränderung der Tasterfarbe nach dem Auslösen.

Beschriftung der Taster:

Diese Anforderung wurde erfüllt. Alle Taster sind beschriftet.

Sequencer Funktion (Keypad):

Diese Anforderung wurde erfüllt. Die Sequencer Funktion ist vollumfänglich nutzbar.

Freie Bewegung der Kugeln (Drone Generator):

Diese Anforderung wurde teilweise erfüllt. Durch das Anstoßen anderer Kugeln bewegen sich diese frei durch den Raum, ein Loslassen während einer Bewegung setzt die Kugel allerdings starr im Raum ab.

Bedienung ganzer Zeilen der Partitur (Sequencer):

Diese Anforderung wurde nicht erfüllt. Die Bedienung ganzer Zeilen durch spezielle Tasten ist bisher nicht möglich.

Nicht-Funktionale Anforderungen

Performanz:

Die Performanz des Systems wird durch eine Reihe von Designentscheidungen gewährleistet. Durch die Verwendung von UDP, das eine schnelle Übertragung bietet, ist eine Latenz zwischen User-Interface und Klangerzeugung nicht erkennbar. Selbst bei der Verwendung des Drone Generators, welcher kontinuierlich die Werte der fünf "Sphere" Objekte sendet, kommt es zu keiner hörbaren Verzögerung. Im User-Interface wurde darüber hinaus darauf geachtet, möglichst wenige Gameobjects mit wiederum möglichst wenigen Polygonen pro Gameobject zu verwenden, um die Grafik- und Rechenleistung der Hololens nicht zu sehr zu beanspruchen. Ziel ist es, eine flüssige Darstellung (> 25 fps) zu erreichen. Hierbei gab es zu Beginn noch Probleme bei der Anzeige des Sequencers aufgrund der 128 Gameobjects, aus der die Partitur besteht. Durch Verwendung möglichst simpler Game Objects konnte die Frame Rate hier aber sichtbar verbessert werden, sodass bei der Verwendung der Instrumente nun kein Ruckeln wahrzunehmen ist.

Benutzbarkeit:

Die Benutzbarkeit des Systems wird erst durch die Usability Tests und anschließende Anpassung des Systems gewährleistet. Zwar wurde durch das Prototyp-Testen an Nutzenden schon ein Grundstein für eine verbesserte Benutzbarkeit gesetzt, dies wird sich aber vor allem in der zweiten Iteration des Systems und den anschließenden Usability Tests deutlich zeigen.

Erweiterbarkeit:

Die Erweiterbarkeit des Systems soll vor allem durch die Kapselung der Klangerzeugung und des User-Interface gewährleistet werden. Durch das Trennen von Steuerung, Darstellung und Verarbeitung in einzelne Game Objects innerhalb des User-Interfaces, soll eine Erweiterung all dieser Komponenten vereinfacht werden. Die Entscheidung einzelne Instrumente in einzeln gekapselten Szenen zu entwickeln, vereinfacht die Erweiterung um weitere Instrumente ebenfalls.

5.2 Abschluss der Evaluation

Die Evaluation zeigt, dass eine Umsetzung des Systems wie es in 3 geplant wurde, realisierbar ist. Bis auf eine Nice-to-Have Anforderung konnten alle funktionalen Anforderungen umgesetzt werden und auch die Nicht-Funktionalen Anforderungen konnten, soweit es zum jetzigen Entwicklungsstand möglich ist, umgesetzt werden. Die in 4 beschriebene Architektur des Systems konnte integriert werden und bietet dem System durch Kapselung in verschiedenen Bereichen eine verbesserte Erweiterbarkeit. Bei der Realisierung aufgetretene Probleme, wie die Netzwerkverbindung der Hololens und Performanzschwächen, konnten gelöst werden.

6 Erste Testphase

Nach der erfolgreichen Implementierung eines lauffähigen Systems wird dieses im Folgenden auf seine Benutzbarkeit und Qualität überprüft. Die Testphase wird zweimal durchgeführt. Zwischen den Tests steht eine Auswertung dieser und eine Anpassung anhand der Ergebnisse an. Im folgenden wird der Begriff Usability Test verwendet, auch wenn die Methoden der Tests nicht ganz den klassischen Usability Test Methoden entsprechen. Beim Testen von Augmented Reality Anwendungen sind diese nur schwer umsetzbar und ein allumfänglicher Usability Test würde den Rahmen dieser Arbeit überschreiten.

6.1 Aufbau der Tests

Der Aufbau der Tests ist klassischen Usability Methoden nachempfunden, doch muss dieser aufgrund der Verwendung eines Augmented Reality Headsets stark angepasst werden. Hierbei werden die meisten Methoden aus Andrew J. Hunsuckers Artikel über Usability Testing bei Augmented Reality Anwendungen entnommen⁴². Durchführungsort der Tests sollte laut Hunsuckers eine Umgebung sein, die der späteren Umgebung in dem das finale System genutzt wird so ähnlich wie möglich ist. Da dieses System aber an keine spezielle Umgebung gebunden ist, kommt ein einfacher Wohnraum zum Einsatz. Aufgrund der bestehenden Netzwerkverbindung des Systems und der eingeschränkten Rechenleistung der Hololens wird auf ein Live-Monitoring der Hololens verzichtet, da das Aufnehmen des Systems zu sichtbaren Einbrüchen in der Frame-Rate führt. Ein Monitoring wird also hauptsächlich über die eingehenden Daten in Max/MSP realisiert, was den Umgang mit einigen Steuerelementen wiedergibt, aber bei User-Interface internen Prozessen keinen Einblick bieten kann. Des Weiteren werden alle ProbandInnen zum „Lauten Denken“ aufgefordert. Dies ist eine häufig angewandte Methode zur nutzerbasierten Untersuchung. Die dabei entstehenden Daten können einen ersten allgemeinen Überblick über mögliche Problembereiche bieten. Dabei sollen unverständliche und problematische Elemente der Anwendung aufgezeigt werden, jedoch auch positive Auffälligkeiten beleuchtet werden⁴³. Da die Methode des „Lauten Denkens“ in dieser Arbeit nicht im Vordergrund steht und lediglich eine Ergänzung zur besseren Einschätzung der Benutzbarkeit der Steuerelemente sein soll, wurden die Aussagen der ProbandInnen nicht direkt dokumentiert, sondern in der späteren Befragung verwendet

42 Hunsucker (2017)

43 Yom et al. (2007): 637

um die ProbandInnen auf ihre eigenen Aussagen anzusprechen. Dies in Kombination mit einfacher Beobachtung der ProbandInnen während der Nutzung, bietet bei diesem System die meisten Einblicke in die Interaktion der ProbandInnen mit dem System. Alle fünf ProbandInnen sind MusikerInnen und entweder vertraut mit digitalen Musikinstrumenten oder zumindest vertraut mit elektronischen Musikinstrumenten. Dies soll für ein Grundverständnis in der Funktionsweise der präsentierten Instrumente sorgen, sodass sich die ProbandInnen schneller zurecht finden und auch objektiver bewerten können.

6.2 Ablauf der Tests



Abbildung 14: Probandin während Test (eigene Fotografie)

Grundsätzlich behandelt ein Test immer alle implementierten Instrumente nacheinander. Pro Instrument wird den ProbandInnen eine Reihe von Aufgaben gestellt, welche sie so eigenständig wie möglich lösen sollen. Da alle Teilnehmenden zum ersten Mal ein Augmented Reality Headset mit gestenbasierter Steuerung verwenden wird die Lösung der Aufgaben aber begleitet und bei Problemen auch Hilfestellung angeboten. Außerdem gibt es vor Beginn der Tests eine kurze Eingewöhnungsphase in den Umgang mit der Hololens. Dieser Testrahmen soll dafür sorgen, dass die ProbandInnen möglichst intuitiv mit der Hololens und der Gestensteuerung umgehen können, sodass sie sich auf die Bedienung der Instrumente konzentrieren und deren Qualität unabhängig von ihren Fähigkeiten im Umgang mit der Hololens bewerten können. Die Aufgaben an den

einzelnen Instrumenten sehen wie folgt aus:

Keypad:

- Spiele 4 vorab ausgewählte Töne deiner Wahl in regelmäßigen Zeitabständen
- Nutze den Sequenz Button um anschließend einige Töne in eine Sequenz zu speichern
- Stelle dir nun den Klang des Instrumentes so ein, dass er dir gefällt
- Speichere deine Einstellungen, verändere Sie und lade Sie anschließend wieder

Sequencer

- Stelle eine Sequenz deiner Wahl ein
- Verstelle die Geschwindigkeit nach deinen Vorstellungen
- Speichere zwei unterschiedliche Partituren in zwei Speicherplätzen
- Lade die zuerst gespeicherte Partitur und lösche sie

Drone Generator:

- Stelle dir einen Ton nach deinen Vorstellungen ein
- Speichere diesen Ton und setze die Kugeln in die Ausgangsposition
- Stelle die Effekte ein
- Lade den Ton erneut

Nach Abschluss der Aufgaben an einem Instrument können die ProbandInnen noch eine kurze Zeit frei mit den Instrumenten umgehen. Das soll den ProbandInnen die Möglichkeit geben, Schwachpunkte des Systems zu entdecken, an die bei der Entwicklung der Aufgaben nicht gedacht wurde. Anschließend gibt es einen Fragebogen mit jeweils sechs Fragen, die sowohl die Qualität der Bedienung, als auch die Qualität des Instruments selbst überprüfen. Der Fragebogen soll bei den zweiten Tests auch ein Qualitätsindikator sein, an dem Verbesserungen erkennbar sind. Neben den Fragebögen gibt es ein offenes Gespräch bei dem die ProbandInnen ihre Erfahrungen, aber auch Verbesserungsvorschläge äußern können. Nach Beendigung der Tests an allen drei Instrumenten gibt es einen abschließenden Fragebogen, welcher ebenfalls der Qualitätskontrolle dient. Es handelt sich hierbei um die standardisierten Fragen der System Usability Scale von John Brooke. Dieser Fragebogen eignet sich zur Bewertung dieses Systems, da er als technologieunabhängig gilt und die grundsätzliche Gebrauchstauglichkeit eines Systems testet. Des Weiteren lässt sich aus den Ergebnissen ein Score berechnen, dessen Wert mit einer festgelegten Skala verglichen wird um die

Qualität des Systems einschätzen zu können (siehe Abb. 15). Der von John Brooke vorgeschlagene standardisierte Fragebogen enthält abwechselnd positiv und negativ formulierte Fragen, die in ab- und aufsteigenden Skalen kodiert werden⁴⁴.

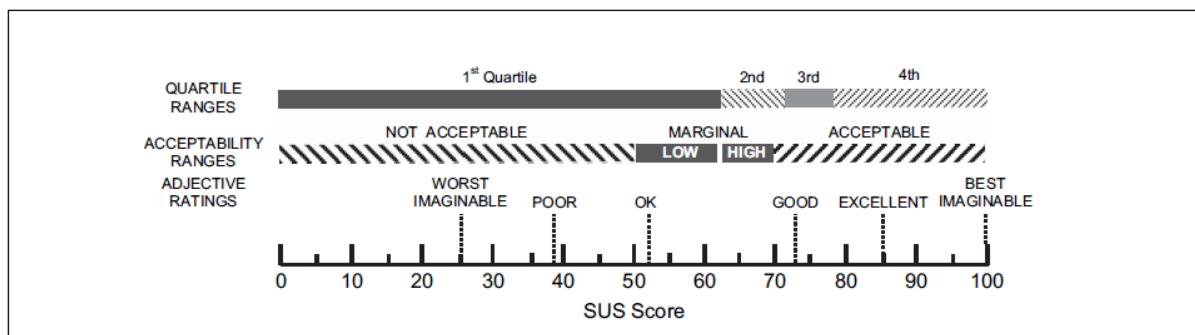


Abbildung 15: Darstellung der SUS Skale (Brooke 2011)

6.3 Auswertung der Testdaten

Das Beobachten der ProbandInnen und das Monitoring über Max/MSP dienten vor allem dazu, zu erkennen, ob die Nutzenden Probleme mit der Steuerung haben. Ob Air-Tap Gesten mehrfach durchgeführt werden mussten bis das gewünschte Ergebnis eintrat und ob das Bewegen von Objekten durch Halten und Bewegen des Kopfes intuitiv funktioniert.

Es zeigte sich schnell, dass fast alle ProbandInnen nach kurzen Einlernphasen kaum noch Probleme mit der Air-Tap Geste hatten. Es kam nur selten zu Klicks die mehrfach ausgeführt werden mussten und es wurden keine Probleme während dem lauten Denken diesbezüglich geäußert. Bei dem Bewegen von Objekten kam es häufiger zu Problemen. Dies konnte im Monitoring beobachtet werden und wurde auch von ProbandInnen verbalisiert. Gerade bei den frei beweglichen "sphere" Objekten des Drone Generators war dies der Fall. Auch dadurch, dass diese nicht den Raumgrenzen des realen Raums folgen. Die insgesamt eher positive Bewertung der ProbandInnen bei der Gestensteuerung könnte durchaus an der fehlenden Erfahrung in der Nutzung mit gestenbasierten Steuerungen liegen. Dies hätte sich durch die Einbindung von erfahreneren NutzerInnen prüfen lassen, wurde hier jedoch vernachlässigt. Das laute Denken ergab außerdem, dass ProbandInnen die verschiedenen Speicherfunktionen oft unklar waren. Gerade die Speicherfunktion des Sequencers verursachte hier vermehrt Probleme.

44 Brooke (2011)

Die Auswertung der Fragebögen (siehe Tabelle 7) der einzelnen Instrumente erfolgt ähnlich wie in 3.2.2.3. Die Ergebnisse der einzelnen Fragen können aber erst ab der zweiten Iteration mit anderen verglichen werden. Zum jetzigen Zeitpunkt können diese nur in Verbindung mit den von den ProbandInnen getroffenen Aussagen über die Qualität und die Funktionen des Systems genutzt werden. Gerade bei Fragen, die in der Bewertung schlechter abschnitten, sollten Verbesserungen erwägt werden. Aber auch Bereiche mit höheren Ergebnissen können angepasst werden, gerade da die Äußerungen der ProbandInnen nach den Tests nicht immer deckungsgleich mit den Ergebnissen der Fragebögen waren.

Usability Test 1

Keypad	1	2	3	4	5	6	7	Ergebnis	Bemerkungen
Die allgemeine Erscheinung des Instruments spricht mich an				2	2	1		24	3x Matrix überflüssig
Die Bedienung der einzelnen Elemente ist intuitiv					2	3		28	Sequencer sollte mehr Funktionen haben
Die Anzahl der Bedienelemente ist ausreichend				1		1	3	31	Mehr Speicherplätze
Die Steuerung des Instruments funktioniert gut						1	4	34	
Mehrere Gesten würden die Bedienung erleichtern (n)			2	3				22	
Die Möglichkeiten der Klangerzeugung sprechen mich an					1	3	1	30	

Sequencer	1	2	3	4	5	6	7	Ergebnis	Bemerkungen
Die allgemeine Erscheinung des Instruments spricht mich an				1	3		1	26	optische Anzeige das Template gefüllt ist
Die Bedienung der einzelnen Elemente ist intuitiv				2	1	2		25	Speicherfunktion umständlich
Die Anzahl der Bedienelemente ist ausreichend					1	4		29	Mehr Funktionen für die Partitur
Die Steuerung des Instruments funktioniert gut						1	4	33	
Mehrere Gesten würden die Bedienung erleichtern (n)	1	1	2	1				27	
Die Möglichkeiten die Partitur zu Bedienen sind ausreichend				3	2			22	

Drone	1	2	3	4	5	6	7	Ergebnis	Bemerkungen
Die allgemeine Erscheinung des Instruments spricht mich an				2	1	1	1	26	2x mehr klangliche Vielfalt, verschiedene sounds pro Kugel
Die Bedienung der einzelnen Elemente ist intuitiv			1	1	1	2		24	
Die Anzahl der Bedienelemente ist ausreichend				2	1	2		25	
Die Steuerung des Instruments funktioniert gut				1		2	2	30	
Mehrere Gesten würden die Bedienung erleichtern (n)		1	2	1		1		22	bedeutung der Beschriftung oft unklar
Die Möglichkeiten der Klangerzeugung sprechen mich an	1		1		2		1	21	optisches Feedback verbessern

(n) Bewertungsskala umgekehrt

Tabelle 7: Auswertung der Ergebnisse der ersten Iteration

Die Gespräche nach der Beendigung der Tests wurden stichpunktartig dokumentiert (Abb. siehe Spalte “Bemerkungen”) die Anmerkungen “Bedeutung der Beschriftung oft unklar” und “optisches Feedback verbessern” in den untersten Zeilen gelten dabei dem gesamten System. Es zeigt sich, dass die Matrix zum Verändern des Klangs von der Mehrheit der ProbandInnen als überflüssig eingestuft wurde. Während der Tests wurde dies auch von den ProbandInnen verbalisiert. Die Funktion der Matrix war allen Nutzenden nicht ganz klar. Dies veränderte sich auch nicht während der Nutzung. Die Speicherfunktion des Sequencers wurde nach den Tests auch vermehrt kritisiert, was sich mit den Aussagen der ProbandInnen während der Nutzung des Sequencers deckt. Bei den Gesprächen nach dem dritten Test mit dem Drone Generator gaben drei von fünf ProbandInnen an, sich eine veränderte Klangerzeugung zu wünschen.

Die Auswertung des SUS Fragebogens ergab einen Score von 80. Dieser ergibt sich durch die Addition aller Ergebnisse, der anschließenden Skalierung durch Multiplikation mit 2.5 und der Teilung durch die Anzahl an ProbandInnen. Auf der SUS Skala ist ein Wert von 80 zwischen "Good" und "Excellent" einzustufen(Siehe Abb.). Erst bei der zweiten Iteration der Tests wird sich zeigen ob sich dieser Score durch die Anpassungen noch verbessern wird.

System Usability Scale					
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.		1	2	2	
Ich empfinde das System als unnötig komplex.	4	1			
Ich empfinde das System als einfach zu nutzen.				3	2
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	4		1		
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.			2	1	2
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	5				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.			2	2	1
Ich empfinde die Bedienung als sehr umständlich.	3	2			
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.			2	1	2
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	1	2	1	1	
Insgesamt 160 Punkte					
Score: $160 \times 2.5 / 5 = 80$					

Tabelle 8: Auswertung SUS Test

Die Auswertung aller Testdaten ergab, dass das System von den ProbandInnen bereits gut angenommen wird(siehe Tabelle 8). Die Beobachtungen zeigten, dass Nutzende nur wenige Probleme mit der Steuerung des Systems haben. Bei den Instrumentenfragebögen gab es nur selten unterdurchschnittliche Ergebnisse und auch der SUS Fragebogen ergab einen Score im akzeptablen Bereich. Dennoch wurden beim lauten Denken und auch in den Befragungen nach den Tests einige Wünsche und Verbesserungsvorschläge geäußert. Aus diesen wird im Folgenden die Anpassung für die zweite Entwicklungsphase erzeugt.

7 Anpassung anhand der Testergebnisse

Die in 6. gewonnenen Erkenntnisse bezüglich der Steuerung und der Steuerelemente der Instrumente sollen im Folgenden genutzt werden, um sowohl die Gestensteuerung, als auch die Bedienelemente in ihrer Funktionalität zu verbessern. Die Grundfunktionalität und Erscheinung der Instrumente soll weitestgehend erhalten bleiben, damit die beiden Iterationen vergleichbar sind.

7.1 Konzeption der Anpassungen

7.1.1 Gestensteuerung

Da die Gestensteuerung von der Testgruppe vorwiegend positiv bewertet wurde, wird in der zweiten Iteration auf eine Erweiterung dieser verzichtet. Da es aber bei den bewegbaren Objekten vermehrt zu Problemen kam, wird sowohl bei den Schiebereglern des Keypads, als auch bei den Kugeln des Drone Generators eine Anpassung vorgenommen. Die Bewegung von Objekten wird durch Interpolation der Position des Objekts zwischen der Position des Objekts selbst und der des Fadenkreuzes erreicht. Diese Interpolation geschieht mit einer gewissen Geschwindigkeit. Je höher diese Geschwindigkeit ist, desto schneller folgt das Objekt den Bewegungen des Fadenkreuzes. Alle Objekte haben in der ersten Iteration einen Standardwert von 0.2 erhalten. In der zweiten Iteration soll nun ermittelt werden, ob eine Anpassung dieses Wertes eine Verbesserung in der Benutzbarkeit der Objekte darstellt. Dazu bekommen die verschiedenen bewegbaren Objekte eines Instruments unterschiedliche Interpolationsgeschwindigkeiten ($0 > x > 1$) zugewiesen. Durch eine Befragung soll dann ermittelt werden, welches dieser Objekte sich am besten Steuern lässt. Dies wird sowohl bei sechs Schiebereglern des Keypads als auch bei den fünf Kugeln des Drone Generators getestet. Bei eigenen Tests zeigt sich, dass vor allem im unteren Wertebereich stärkere Veränderungen wahrnehmbar sind. Aufgrund dessen werden die Interpolationsgeschwindigkeitswerte ungleichmäßig aufgeteilt und erhalten bei den Schiebereglern die Werte 0.01, 0.05, 0.1, 0.2, 0.6, 1 und bei den Kugeln 0.01, 0.05, 0.1, 0.4, 0.8.

7.1.2 Keypad

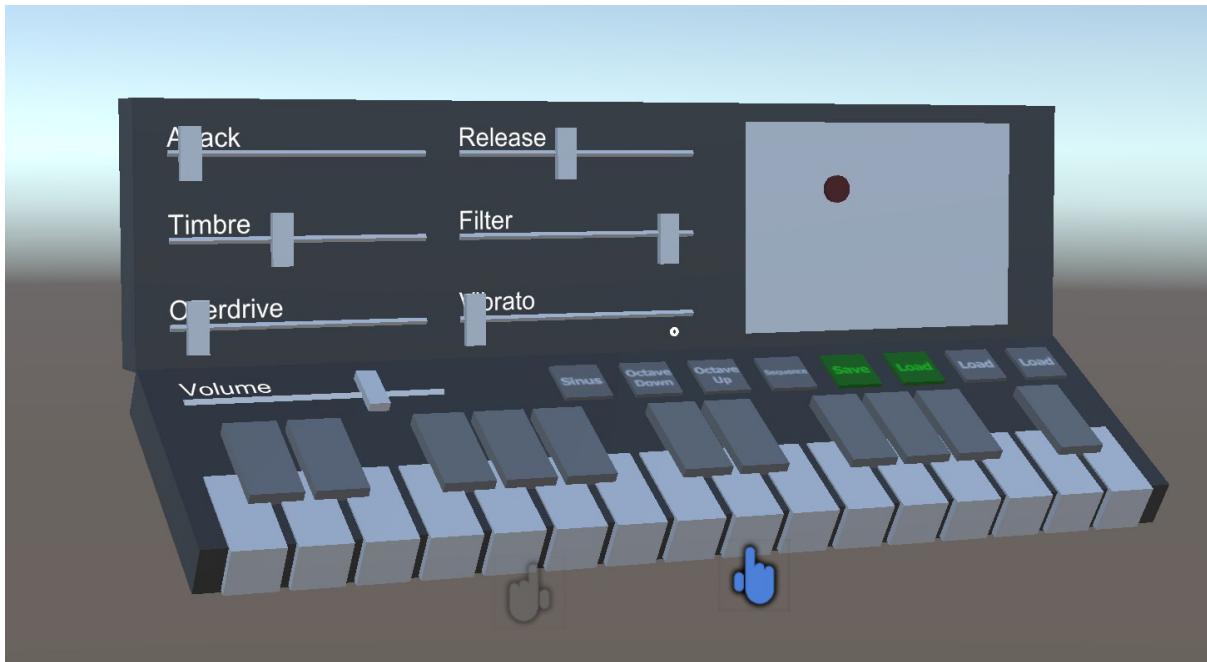


Abbildung 16: Keypad der zweiten Iteration (eigene Darstellung)

Die Auswertung der Tests ergab, dass die Mehrheit der ProbandInnen die Matrix des Keypads als überflüssig empfanden. Aufgrund dessen wird die Umschaltung der Wellenform der Klangerzeuger in der zweiten Iteration auf einen Taster beschränkt. Die Funktionalität der Klangerzeugung ändert sich dadurch nicht. Die Funktion der Matrix wird dadurch aber nicht mehr gebraucht. Durch Wegfallen der Matrix wird die Klangerzeugung aber auf einen Synthesizer beschränkt. Dieser kann nach wie vor mit unterschiedlichen Wellenformen betrieben werden. Es gibt, genau wie zuvor, drei Wellenformen (Sinus, Sägezahn, Rechteck) und die Möglichkeit, den Synthesizer auszuschalten. Das Umschalten der Wellenformen geschieht nun über einen einzelnen Taster, welcher beim Synthesizer durch Anwenden der Air-Tap Geste die Wellenformen durchschaltet. Anstelle der Matrix wird ein weiteres Element zur Veränderung des Klangs implementiert. Hierbei handelt es sich um das während der Prototyping Tests gewünschte X/Y Pad zur Bedienung zweier Effektparameter. Dieses besteht aus einer freien Fläche auf der sich ein bewegbares Objekt positionieren lässt. Mit dem bewegbaren Bedienelement lassen sich so die Parameter eines Effektes anpassen. Bei dem Effekt handelt es sich um einen Delay-Effekt, dessen anpassbare Werte, Verzögerungszeit und Rückspeisungsintensität, jeweils von der X und Y Koordinate des Pads abgebildet werden. Je weiter rechts sich der Punkt auf dem Pad befindet, desto höher die Verzögerungszeit, je höher der Punkt, desto höher die Lautstärke der Rückspeisung. Da die Bedienung der Speicherfunktionen der Instrumente während der

Tests von den ProbandInnen oft missverstanden und kritisiert wurde, wird diese in der zweiten Iteration komplett neu gestaltet. Gleichzeitig soll die Funktion aber auch bei allen Instrumenten gleich umgesetzt werden, sodass hierbei nicht mehrere Bedienungskonzepte erlernt und verstanden werden müssen. Als Grundlage der neuen Bedienung dient hierbei das Konzept des Sequencers. Das Keypad erhält in der zweiten Iteration nun auch drei eigenständige Speicherplätze mit denen die Schiebereglervorstellungen gespeichert und anschließend wieder geladen werden können. Da die Bedienung der Speicherfunktion des Sequencers von den ProbandInnen aber als umständlich empfunden wurde, kommt ein neues Bedienungskonzept zum Einsatz. Die drei Speicherplätze sind nun grundsätzlich nicht auf dem Instrument zu sehen, bis der "Save" Taster bedient wird (siehe Abb. 17-1). Durch die Bedienung leuchtet der "Save" Taster grün auf und neben ihm erscheinen die Speicherplatz Taster, welche nun zur Speicherung angewählt werden können (siehe Abb. 17-2). Wurde eine Speicherung auf einem der Elemente durchgeführt, leuchtet diese fortan grün auf um eine Veränderung des Zustands zu vermitteln. Durch das Auswählen eines Speicherfeldes wird eine Speicherung abgeschlossen und die Speicherfunktion springt automatisch zurück in den Ausgangszustand. Alle noch nicht belegten Speicherplätze sind nun wieder unsichtbar. Nur die belegten Speicherplätze bleiben sichtbar, da eine Bedienung in diesem Zustand einen Ladevorgang auslöst (siehe Abb. 17-3). Erst durch eine erneute Bedienung der "Save" Taste werden die freien Speicherplätze wieder bedienbar. Auch bereits belegte Speicherplätze können in diesem Zustand wieder zur Speicherung neuer Werte genutzt werden (siehe Abb. 17-4)

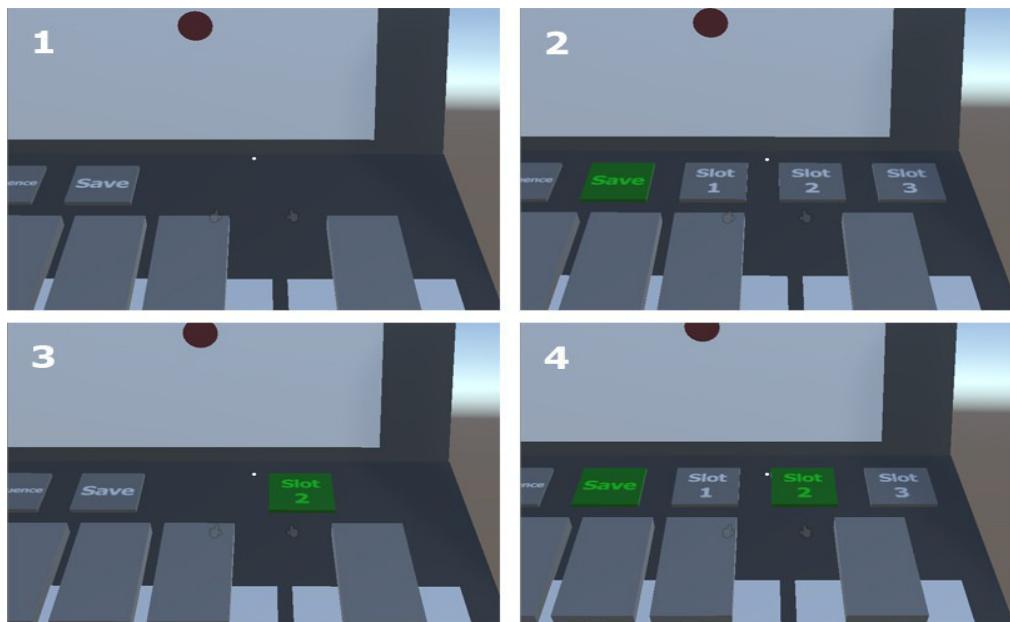


Abbildung 17: Darstellung der Speicherfunktion (eigene Darstellung)

7.1.3 Sequencer

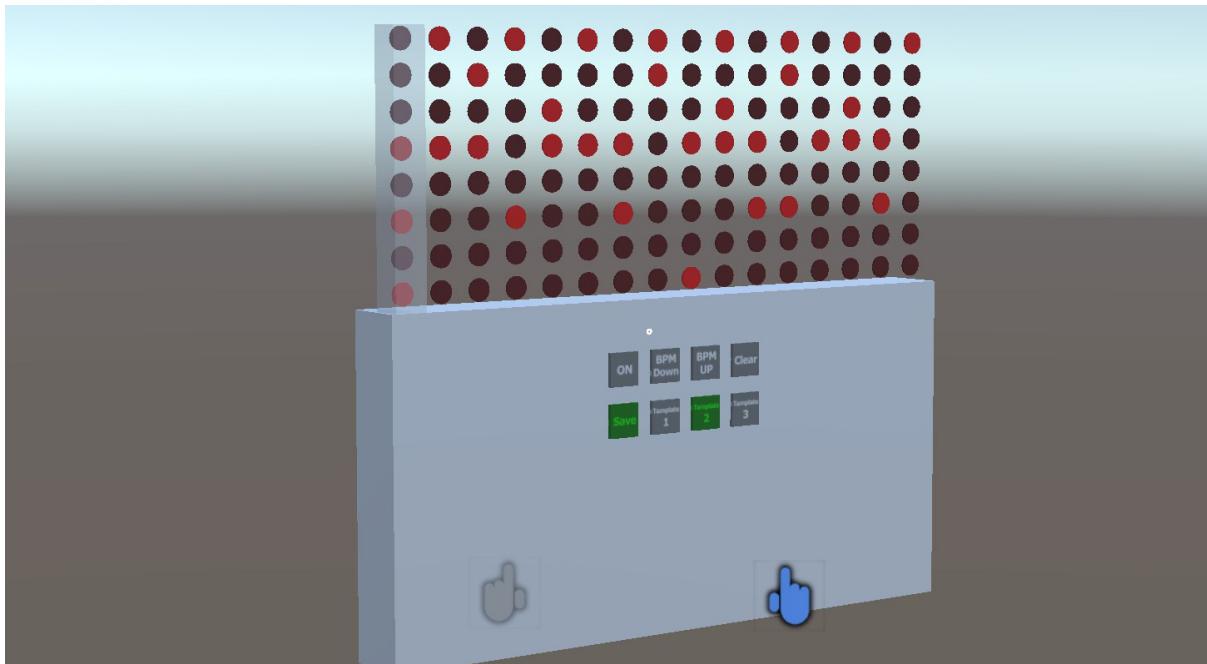


Abbildung 18: Sequencer der zweiten Iteration (eigene Darstellung)

Da es bei der Bedienung des Sequencers nur zu wenigen Problemen seitens der ProbandInnen kam, gibt es neben der neu implementierten Speicherfunktion (beschrieben in 7.1.2) zur Speicherung unterschiedlicher Partituren nur wenige Anpassungen. Der Balken zur Anzeige des aktuellen Partitionsschritts wird direkt auf die Partition gelegt. Zuvor befand er sich hinter der Partition. Dies soll NutzerInnen die Erkennung des aktuellen Partitionsschritts erleichtern. Da der Balken nun direkt auf der Partitur liegt wird er optisch leicht durchsichtig gestaltet, sodass der aktuelle Partitionsschritt weiterhin erkennbar bleibt. Eine weitere Anpassung, die neben der erleichterten Bedienung auch eine Vereinheitlichung in den Bedienkonzepten der Instrumente verursachen soll, ist das Wegfallen des "Off" Tasters. Das Ein- und Ausschalten von Funktionen geschieht sowohl beim "Sequencer" Schalter des Keypads, als auch beim "On" Schalter des Drone Generators immer über ein schaltbares Bedienelement. Beim Sequencer aber gab es bisher einen "On" und einen "Off" Taster zum Ein- und Ausschalten. Dieser wird ähnlich wie beim Drone Generator in der zweiten Iteration durch einen simplen "On" Schalter ersetzt, welcher im eingeschalteten Zustand grün aufleuchtet.

7.1.4 Drone Generator

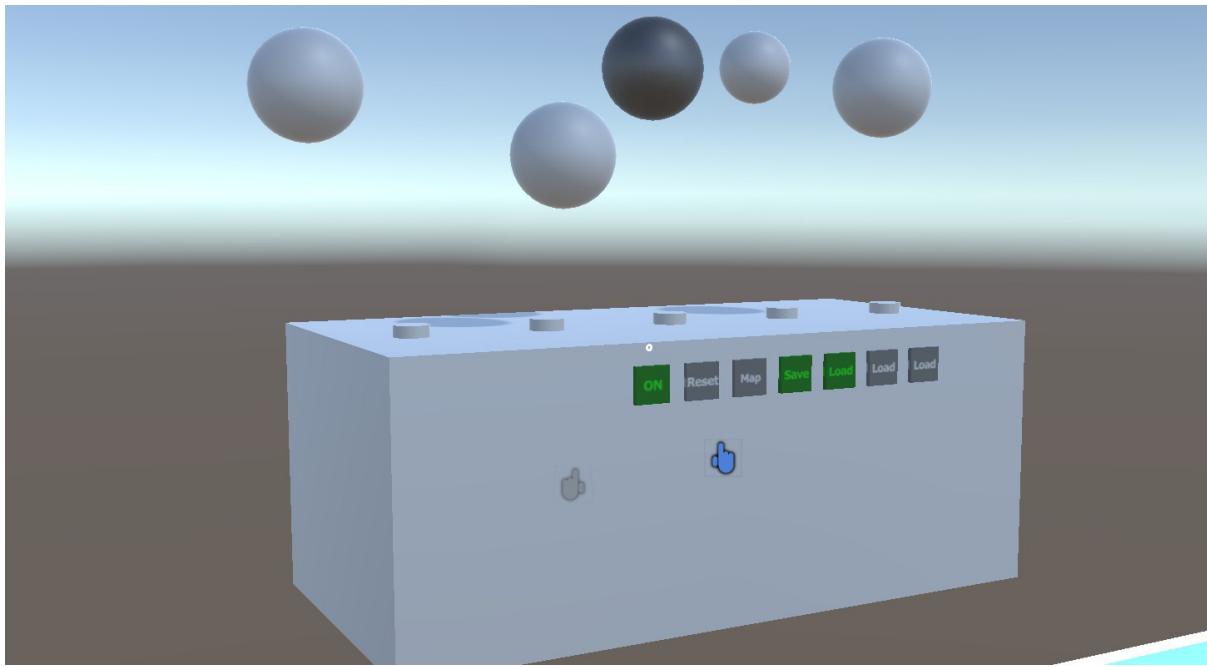


Abbildung 19: Drone Generator der zweiten Iteration (eigene Darstellung)

Beim Drone Generator gab es seitens der ProbandInnen Kritik an der Klangerzeugung. Diese wurde als zu eintönig und unabwechslungsreich empfunden. Aufgrund dessen wird hier eine Anpassung vorgenommen. Da aber der Charakter eines Drone Instruments, das im Grunde aus stehenden Klängen besteht, erhalten bleiben soll, wird die Klangerzeugung nicht verändert, sondern lediglich ein Zugriff auf die Effektparame ter ermöglicht. Hierzu wird einem der "Sphere" Objekte eine spezielle Funktion zugewiesen, die sich von der Funktion der anderen Objekte unterscheidet. Dadurch fällt ein Klangerzeuger weg, da so die Grundfunktionalität und Erscheinung der fünf bewegbaren Objekte erhalten bleibt. Das "Sphere" Objekt mit den neuen Funktionen verändert genau wie die anderen Objekte, durch Positionieren im Raum, anhand der X, Z und Y Koordinat werte, Parameter der Klangerzeugung. Bei dem Objekt sind diese Parameter allerdings nicht Tonhöhen und Lautstärke eines Klangerzeugers, sondern die Verzögerungszeit eines Delay-Effekts(X Koordinate), Filter Frequenz eines Low-Pass Filters (Y Koordinate) und der Grad der Verzerrung eines Overdrive Effekts (Z Koordinate). Der Delay Effekt war bereits in der ersten Iteration schon in der Klangerzeugung enthalten, wird nun aber von NutzerInnen anpassbar sein. Filter und Verzerrung sollen die klanglichen Möglichkeiten nun noch erweitern. Durch dieses neue Objekt, dass sich auch optisch von den anderen unterscheidet (Abb.), soll auch das Spielerlebnis erweitert werden, da die Positionierung dieses Objekts deutlich andere Ergebnisse erzeugt als es bei den anderen Objekten der Fall ist. Auf Grund dieses neuen

Objekts und der Anpassung in der Klangerzeugung. ist der "FX" Taster in seiner Funktion nun nicht mehr zu gebrauchen und wird entfernt, da die Effekte nun permanent eingeschaltet sind. Die Speicherfunktion wird auch beim Drone Generator angepasst und um zwei Speicherplätze erweitert (siehe 7.1.2). Da die ProbandInnen während der Tests oft von den unsichtbaren virtuellen Raumgrenzen des Drone Generators verwirrt waren, soll in der zweiten Iteration die Möglichkeit bestehen die Raumgrenzen des echten Raums in dem sich NutzerInnen befinden, mit in den virtuellen Raum einzubinden. Durch Spatial Mapping der Hololens soll der echte Raum getrackt und in einen Virtuellen verwandelt werden, sodass die Bewegung der "Sphere" Objekte durch diesen begrenzt werden.

7.2 Implementierung der Anpassungen

7.2.1 Gestensteuerung

Die Anpassung der Interpolationsgeschwindigkeit gestaltet sich aufgrund der gewählten Objektarchitektur in Unity als sehr simpel. Jeder Schieberegler des Keypads enthält ein Script `SliderScript.cs` welches wiederum ein public Feld zur Anpassung der Interpolationsgeschwindigkeit enthält. Diese kann somit in Unity pro Objekt über den Inspector angepasst werden. Gleiches gilt für die "Sphere" Objekte des Drone Generators.

7.2.2 Keypad

Die Umschaltung der Wellenformen erfolgt nun über einen einzelnen Schalter. Auf diesem wird immer die aktuell verwendete Wellenform in Text angegeben (Sine, Saw, Triangle). Bei der Verwendung wird eine Nachricht mit der Adresse "soundcontrol" und dem zur Wellenform zugehörigen Wert (1-3) an den Klangerzeuger gesendet. In der Klangerzeugung wird diese Nachricht genau wie zuvor Verarbeitet. Durch ein `selector~ 3` Objekt kann bei Erhalt eines Wertes zwischen 1 und 3 zwischen drei verschiedenen Signal Outputs unterschieden werden. An diesen Outputs hängt jeweils einer der drei Wellenformoszillatoren, die so durch ein Umschalten des `selector~ 3` Objekts ausgewählt werden. Das Wechseln der angezeigten Schrift erfolgt durch ein Austauschen des verwendeten Materials. Dem Taster Objekt ist ein Skript `MaterialSwitcher` angehangen, welches bei Erhalt einer Air-Tap Geste zwischen Materials in einer festgelegten Reihenfolge wechselt. Das X / Y Pad besteht auf der User Interface Seite aus einem skalierten Würfel, der die Bedienfläche darstellt und einem skalierten Zylinder

Objekt an dem ein Skript, welches die Gertensteuerung, die Bewegung, die Begrenzung des Bewegungsbereiches, und die Versendung der angepassten X und Y Werte realisiert. Dieses `FxDotScript.cs` ist erneut eine abgewandelte Version des `HandDraggable.cs` Skripts des Hololens Toolkits. Die X und Y Werte werden erneut auf Werte zwischen 1 und 256 skaliert um eine Weiterverarbeitung in Max/MSP zu erleichtern. Auf der Klangerzeuger Seite wurde ein neues Effekt Modul realisiert, dass ein in der Verzögerungszeit und Rückspeisungsintensität anpassbares Delay darstellt. Dieses wird über die Objekte `tapin~` und `tapout~` realisiert, zwischen denen Signale mit einer Zeitverzögerung versendet werden können. Das von der Klangerzeugung kommende Signal wird in das `tapin~` Objekt gesendet und mit einer Zeitverzögerung aus dem `tapout~` Objekt weitergeführt. Da diese Objekte das Signal nicht über einen "signal cord"⁴⁵ übertragen, ist es möglich, die aus `tapout~` empfangenen Signale zurück in das `tapin~` Objekt zu senden ohne dabei Rekursionsprobleme hervorzurufen. Hierdurch wird die Rückspeisungsintensität über eine Multiplikation des in `tapin~` rücklaufenden Signals realisiert ($0 < x < 1$). Es ist nun möglich, die Verzögerungszeit beim `tapout~` Objekt anzupassen. Das Empfangen des Signals wird selbst während einer Anpassung der Zeit nicht abgebrochen, sodass Delay-Effekte von analogen Tape-Delays durch die Veränderung der Verzögerungszeit während des Abspielens simuliert werden können. Durch die Rückspeisung und die Anpassung der Rückspeisungsintensität wird das empfangene Signal immer wieder leiser werdend abgespielt, was den typischen Delay-Effekt erzeugt (siehe Abb. 20).

45 Anm.: Spezielle Max/MSP Verbinder zur Audiosignalübertragung

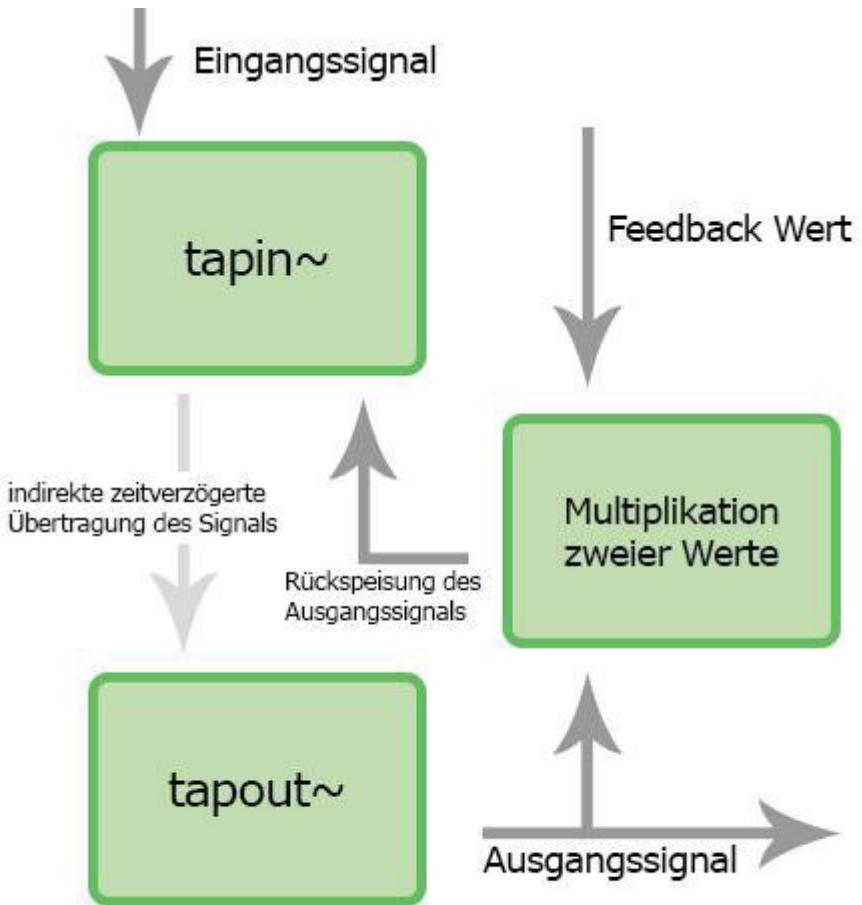


Abbildung 20: Funktionsweise des Delay-Effekts (eigene Darstellung)

Die neugestaltete Speicherfunktion funktioniert bei jedem Instrument im Grunde gleich. Die Save Taste enthält immer ein simples Skript, das beim Auslösen drei Gameobjects abwechselnd sichtbar und unsichtbar macht. Diese drei Gameobjects enthalten nun jeweils ein Skript mit zwei unterschiedlichen Funktionen. Ist der "Save" Schalter aktiv, wird in diesem Skript eine Speicherung wie in 5.1.1.1 beschrieben ausgeführt. Ist ein Speicherplatz belegt und der "Save" Schalter inaktiv, wird eine Ladefunktion ausgelöst. In der Klangerzeugung werden lediglich zwei weitere Speicherplätze erzeugt, welche durch ein simples routing mithilfe von gswitch2 Elementen einzeln angesprochen werden können. Beim Auslösen des "Save" Schalters, schalten diese Elemente um, sodass ein Auslösen eines Speicherplatz Schalters nun eine Speicherung auslöst. Bei inaktiver "Save" Taste stehen diese Elemente so, dass ein Laden ausgelöst werden kann.

7.2.3 Sequencer

Bei dem Sequencer Instrument ist die Implementierung der Anpassungen wesentlich simpler. Die durchsichtigkeit und Position des Partiturbalkens wird über den Unity Inspector angepasst. Die An- und Aus-Funktion in einem Schalter wird durch ein simples Script, das den Aktiv Zustand des `MoverScript.cs` pro Air-Tap Geste auf `True` oder `False` setzt realisiert. Bei der Speicherung mussten einige Anpassungen vorgenommen werden, welche aber im Grunde in 7.2.3 beschrieben wurden. Die Speicherfunktion ist identisch zu der des Keypads. Lediglich die zu Speichernden Werte unterscheiden sich. Pro Speicherung wird der gesamte Inhalt der 16x8 Partitur Matrix in ein Array gespeichert, dass beim Laden dann wieder zurück geschrieben wird(siehe 5.1.1.1). Die Klangerzeugung wurde nicht verändert.

7.2.4 Drone Generator

Die Veränderte Klangerzeugung des Drone Generators wird lediglich in Max/MSP zu Anpassungen führen. Einer der fünf implementierten Synthesizer wird entfernt, dafür kommt ein Delay-Modul zum Einsatz. Das frei gewordene "Sphere" Objekt sendet nach wie vor seine Positionsreihenfolge skaliert auf Werte zwischen 1 und 256 an die Klangerzeugung. Bei dem Delay-Modul handelt es sich um das gleiche Modul wie in 7.2.2 beschrieben. Die X Koordinate greift dann mit skalierten Werten von 10 bis 2560 auf die Verzögerungszeit des Moduls zu. Die Y Koordinate des frei gewordenen "Sphere" Objekts greift mit skalierten Werten auf ein `lores~` Objekt zu und verändert hierbei die Filterfrequenz mit Werten zwischen 30 und 7680. Die Z Koordinatenwerte werden auf Werte zwischen 1 und 10 skaliert. Damit kann der Verzerrungsgrad eines `overdrive~` Objekts eingestellt werden. Im User-Interface wird das "Sphere" Objekt optisch durch eine andere Farbgebung als die anderen Objekte hervorgehoben, sodass Nutzende dieses besondere Objekt von den anderen unterscheiden können. Da die Effekt Sektion des Drone Generators nun permanent in Gebrauch ist, wird der "FX" Schalter entfernt, da er keine Funktion mehr bedient. Anstelle des "FX" Schalters kommt nun ein Schalter zum Ein- und Ausschalten des Spatial Mappings.

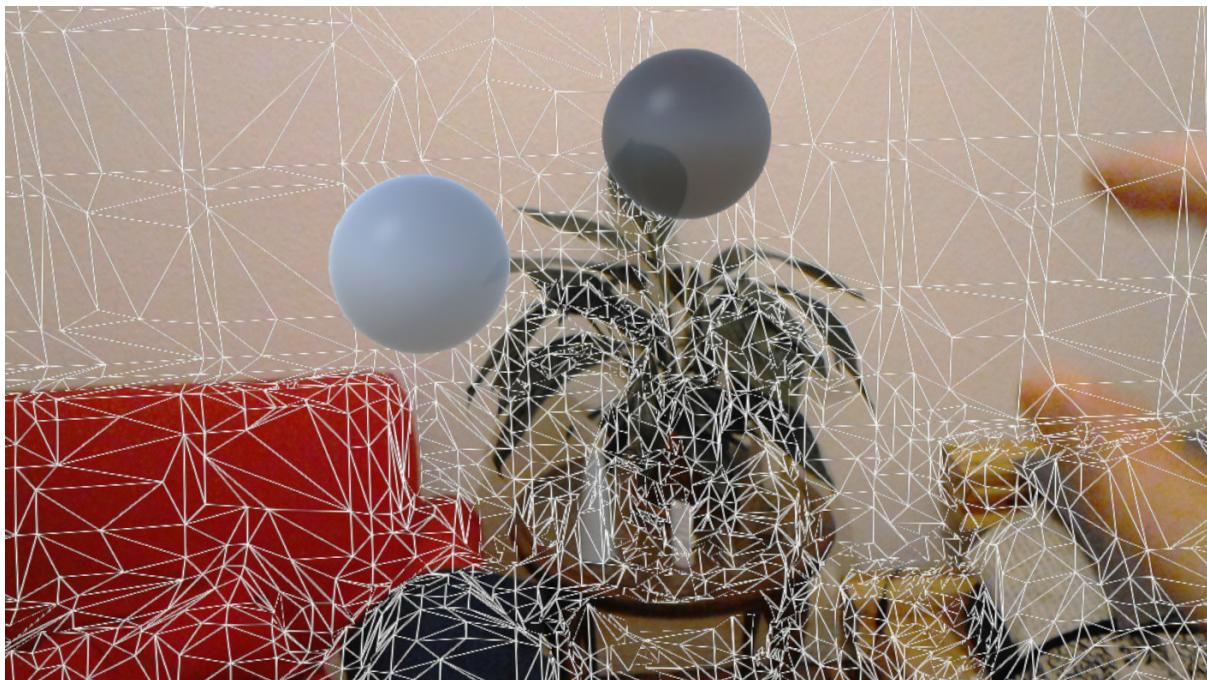


Abbildung 21: Screenshot der Hololens bei Spatial Mapping (eigene Darstellung)

Diese Funktion wird durch einen Schalter ein- und ausschaltbar gemacht, da es gerade in engeren Räumen schnell dazu kommen kann, dass sich "Sphere" Objekte plötzlich hinter dem Mapping befinden und nicht mehr erreichbar sind, oder der Raum grundsätzlich zu klein ist und von vornherein Teile des Instruments verdeckt werden. Das Spatial Mapping wird von einem Prefab Gameobject des Hololens Toolkits realisiert und besteht aus drei Skripten `SpatialMappingObserver.cs`, `SpatialMappingManager.cs` und `ObjectSurfaceObserver`, welche in Zusammenarbeit ein einfach zu integrierendes System bilden, dass den echten Raum in ein dreidimensionales Mesh mit anpassbarer Polygongröße wandelt (siehe Abb.). Dieses Mesh wird in variablen Zeitabständen aktualisiert, um Änderungen im Raum wahrnehmen zu können. Des Weiteren erzeugt das Prefab einen Mesh-Collider, mit dem andere Gameobjects interagieren können und es ist über den Unity-Inspector möglich, eine visuelle Repräsentation des Meshs anzeigen zu lassen. Dies wird in der Anpassung des "Drone Generators" verwendet, sodass Nutzende sehen können sobald das Mesh aktiv ist. Die Speicherfunktion wird auch beim "Drone Generator" angepasst und um zwei Speicherplätze erweitert (siehe 7.1.2). Hierbei erhält das `SphereScript.cs` weitere Vector3 Felder zur Speicherung unterschiedlicher Positionen.

8 Zweite Testphase

8.1 Aufbau der Tests

Die Testmethoden der zweiten Tests sind deckungsgleich mit denen aus 6.2 und 6.3 da ein Vergleich sonst nicht möglich wäre. Allerdings werden die Aufgaben der Tests zu Teilen leicht abgewandelt, da die Instrumente neue Funktionalitäten aufweisen, die sonst nicht geprüft würden. Die Fragebögen des Keypads und des "Drone Generators" werden um eine Frage bezüglich der bewegbaren Objekte erweitert. Die Ergebnisse dieser Frage werden bei dem Vergleich mit den Tests der ersten Iteration aber nicht mit eingebunden, sondern separat aufgeführt und ausgewertet. Anhand dieser soll ermittelt werden, welches Verhalten bei bewegbaren Objekten von den ProbandInnen am positivsten bewertet wird. Die Tests werden mit den gleichen ProbandInnen der ersten Tests durchgeführt. Bei allumfänglichen Tests hätte eine zweite Kontrollgruppe ohne Vorkenntnisse in der Nutzung des Systems objektive Vergleichsdaten erbracht. Dies aber würde den Rahmen dieser Arbeit überschreiten und bei Tests mit den gleichen ProbandInnen ist davon auszugehen, dass diese annähernd gleichwertige Bewertungen wie in den ersten Tests abgeben, sodass ein Vergleich eher möglich ist. Es muss aber bedacht werden, dass die ProbandInnen bereits vor den Tests mit dem System vertraut sind und deshalb manche Fragen wahrscheinlich besser bewerten.

8.2 Ablauf der Tests

Der Ablauf der Tests unterscheidet sich bis auf die Befragung zu den bewegbaren Objekten und den angepassten Aufgaben nicht von dem in 6.2 beschriebenen Ablauf. Die Befragung zu bewegbaren Objekten findet am Ende der Befragung zu den einzelnen Instrumenten statt. Hierzu können die ProbandInnen die Hololens erneut aufsetzen um die bewegbaren Objekte noch einmal miteinander zu vergleichen. Bei der zweiten Iteration der Tests wurden Bemerkungen für die Verbesserung nicht mehr dokumentiert, da die Entwicklungsphase nach der zweiten Iteration endete. Die neuen Aufgaben lauten wie folgt:

Keypad:

- Spiele 4 vorab ausgewählte Töne deiner Wahl in regelmäßigen Zeitabständen
- Nutze den Sequenz Button um anschließend einige Töne in eine Sequenz zu speichern
- Stelle dir nun den Klang des Instrumentes so ein, dass er dir gefällt
- Stelle das Delay über das X/Y Pad so ein, dass es dir gefällt
- Speichere deine Einstellungen, Verändere Sie und lade Sie anschließend wieder
- Verändere deine Einstellungen und Speichere Sie in einen zweiten Slot

Sequencer:

- Stelle eine Sequenz deiner Wahl ein und spiele sie ab
- Verstelle die Geschwindigkeit nach deinen Vorstellungen
- Speichere zwei unterschiedliche Partituren in zwei Speicherplätzen
- Lade die zuerst gespeicherte Partitur und lösche sie

Drone Generator:

- Stelle dir einen Ton nach deinen Vorstellungen ein
- Speichere einen Ton und setze die Kugeln in die Ausgangsposition
- Stelle das Spatial Mapping ein und bewege die Kugeln durch den Raum
- Speichere einen weiteren Ton
- Lade den zuerst gespeicherten Ton erneut

Hierbei hat das Keypad mehrere Aufgaben bekommen, da hier die komplexere Speicherfunktion ausgiebig getestet werden soll.

8.3 Auswertung der Testdaten

Die Auswertung der Testdaten der zweiten Iteration dient vor allem dazu, die numerischen Ergebnisse zu erheben, da diese nun mit denen aus der ersten Iteration verglichen werden. Hierbei soll gezeigt werden, ob die Anpassungen an das System aus der Sicht der ProbandInnen eine Verbesserung darstellen.

Während der Beobachtung der ProbandInnen in der zweiten Iteration der Tests konnten kaum Schwierigkeiten bei der Anwendung der Air-Tap Geste festgestellt werden. Beim Agieren mit den bewegbaren Objekten fiel einzelnen ProbandInnen schon vor der expliziten Befragung zu diesen auf, dass sie sich unterschiedlich verhalten. Hierbei

wurden aber noch keine Bewertungen abgegeben. Das laute Denken zeigte, dass die neue Speicherfunktion wenig Probleme verursachte. Es kam hierbei kein einziges Mal zu Fehleingaben oder Ratlosigkeit. Dies ist wahrscheinlich auch mit dem Vorwissen zu Begründen, das die ProbandInnen bei den Tests der ersten Iteration während der Verwendung der Speicherfunktion des Sequencers sammeln konnten. Dennoch wurde die neue Speicherfunktion, die sich doch stark von der alten Funktion des Sequencers unterscheidet, gut angenommen. Die Auswertung der Instrumentenfragebögen zeigt, dass es eine Verbesserung in der wahrgenommenen Bedienbarkeit bei den ProbandInnen gab. Die Ergebnisse der einzelnen Fragen ergeben sich hierbei aus der Addition aller Antwortwerte. Wenn die Antwort einer Frage bspw. "Stimme zu" ist, so wird sie mit 5 gewichtet. Pro Frage werden anschließend alle Antworten addiert. Jedes einzelne Instrument wurde in der zweiten Iteration besser bewertet als in der Ersten (Siehe Tabelle 9).

	Erste Iteration	Zweite Iteration
Keypad	Ergebnis	Ergebnis
Die allgemeine Erscheinung des Instruments spricht mich an	24	25
Die Bedienung der einzelnen Elemente ist intuitiv	28	33
Die Anzahl der Bedienelemente ist ausreichend	31	32
Die Steuerung des Instruments funktioniert gut	34	32
Mehrere Gesten würden die Bedienung erleichtern (n)	22	21
Die Möglichkeiten der Klangerzeugung sprechen mich an	30	33
	169	176

Sequencer	Ergebnis	Ergebnis
Die allgemeine Erscheinung des Instruments spricht mich an	26	29
Die Bedienung der einzelnen Elemente ist intuitiv	25	31
Die Anzahl der Bedienelemente ist ausreichend	29	26
Die Steuerung des Instruments funktioniert gut	33	34
Mehrere Gesten würden die Bedienung erleichtern (n)	27	30
Die Möglichkeiten die Partitur zu Bedienen sind ausreichend	22	21
	162	171

Drone	Ergebnis	Ergebnis
Die allgemeine Erscheinung des Instruments spricht mich an	26	30
Die Bedienung der einzelnen Elemente ist intuitiv	24	30
Die Anzahl der Bedienelemente ist ausreichend	25	26
Die Steuerung des Instruments funktioniert gut	30	30
Mehrere Gesten würden die Bedienung erleichtern (n)	22	18
Die Möglichkeiten der Klangerzeugung sprechen mich an	21	26
	148	160

Tabelle 9: Ergebnisse der Instrumenten-Fragebögen

Der Score von 80 aus den ersten Tests wurde in der zweiten Iteration mit einem Score von 86 deutlich verbessert und ist nun im Bereich "Excellent" (Siehe Tabelle 10)

System Usability Scale					
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.		1	1	3	
Ich empfinde das System als unnötig komplex.	5				
Ich empfinde das System als einfach zu nutzen.				1	4
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	3	1	1		
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.				2	3
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	5				
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.			1	3	1
Ich empfinde die Bedienung als sehr umständlich.	2	3			
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.			1		4
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	2	2	1		
insgesamt 172 Punkte					
Score: $172 \times 2.5 / 5 = 86$					

Tabelle 10: Ergebnisse des zweiten SUS Fragebogens

Die Befragung zu bewegbaren Objekten zeigten, dass die Mehrzahl der ProbandInnen bei den Schieberegbern eine langsamere Interpolation(zwischen 0.01 und 0.1) bevorzugen. Bei den "Sphere" Objekten hingegen wurde ein Wert zwischen 0.05 und 0.1 als am besten Bedienbar empfunden. Diese Werte werden den Objekten nachträglich noch zugewiesen, sodass nun alle Schieberegler einen Wert von 0.05 und alle "Sphere" Objekte den Wert 0.075 erhalten.

8.4 Fazit

Die Tests sollten vor allem dazu dienen, die Qualität der Instrumente allgemein zu verbessern. Sowohl in ihrer Bedienbarkeit, als auch in der Funktionsweise der Bedienelemente. Durch das Beobachten der ProbandInnen und das laute Denken konnte festgestellt werden wie sicher ProbandInnen mit der Steuerung umgehen können und wo sie auf Probleme treffen. Bei den Einzelgesprächen konnten die ProbandInnen Verbesserungsvorschläge äußern, Kritik üben und gerade bei Steuerelementen die ihnen Probleme bereiteten genau formulieren, was diese sind und wie diese verändert werden sollten. Hier konnte außerdem auch Kritik an der Klangerzeugung geäußert werden. Die Fragebögen sollten vor allem dazu dienen, eine Verbesserung zwischen den Iterationen feststellbar zu machen. Dies ist nur bedingt gelungen, da die Ergebnisse der beiden Iterationen doch sehr nah aneinander lagen und die Verbesserung durchaus auch mit den Vorkenntnissen der ProbandInnen während der zweiten Testphase begründbar sind.

9 Fazit

Im Folgenden werden die gesammelten Ergebnisse, die diese Arbeit erbracht hat, zusammengetragen und vorgestellt. Abschließend gibt es noch einen Ausblick, der mögliche Erweiterungen des realisierten Systems aufzeigt.

9.1 Zusammenfassung

Diese Arbeit diente in erster Linie dazu, neue Steuerkonzepte für digitale Musikinstrumente in einer Augmented Reality Umgebung zu entwickeln, umzusetzen und mithilfe von Usability-Methoden auf ihre Benutzbarkeit zu prüfen.

Hierzu wurden verschiedene Konzepte digitaler Musikinstrumente betrachtet und anschließend dazu verwendet, auf Basis der Microsoft Hololens, eigens entwickelte digitale Musikinstrumenten zu realisieren. Durch die Erarbeitung verschiedener Nutzungsszenarien und Tests an Prototypen mit potentiellen NutzerInnen, wurden diese zuerst konzeptionell entwickelt, um daraus Anforderungen für eine Umsetzung eines lauffähigen Systems zu extrahieren.

Anhand dieser Anforderungen wurden verschiedene mögliche Designs für das umzusetzende System abgewogen und verglichen, das letztendlich verwendete Design erarbeitet und festgelegt. Dieses trennt die beiden Komponenten Klangerzeugung und User-Interface nicht nur software-seitig in Max/MSP und Unity sondern auch hardware-seitig durch Hinzuziehen eines Windows PCs für den Stand-Alone-Betrieb in Max/MSP. Sowohl die Klangerzeugung in Max/MSP als auch das User-Interface in Unity wurden dabei objektorientiert entworfen und jeweils strukturell in die drei unterschiedlichen Instrumente aufgeteilt. In Unity geschieht das durch die Verwendung dreier Szenen, in Max/MSP durch eine Aufteilung in drei Sektionen, anhand der Nachrichten die das User-Interface sendet. Die Verbindung wird per UDP unter Verwendung einer OSC Datenstruktur realisiert.

Das System wurde anschließend entsprechend dem entworfenen Design realisiert. Dies geschah mithilfe von iterativer Entwicklung. In der ersten Iteration wurde eine Grundfunktionalität des Systems implementiert und die Anforderungen aus drittens weitestgehend umgesetzt.

Der aus der Realisierung hervorgegangene lauffähige Prototyp wurde daraufhin in einer Testphase verwendet um die entwickelten Instrumente mithilfe einiger Usability-Methoden auf ihre Bedienbarkeit zu prüfen und Verbesserungsvorschläge der ProbandInnen zu erarbeiten. Hier wurden Probleme mit einigen Bedienelementen

festgestellt. Diese Bedienelemente wurden in der darauf folgenden, zweiten Entwicklungsphase angepasst und teils neu konzipiert. Nach Beendigung der zweiten Entwicklungsphase kam es zu einer abschließenden zweiten Testphase, bei der erneut ProbandInnen mit verschiedenen Usability-Methoden getestet wurden. Diese zweite Testphase diente vor allem dazu, Vergleichsdaten zu erhalten, um die beiden Iterationen des Systems auf ihre Bedienbarkeit hin vergleichen zu können. Die Auswertung der Tests ergab, dass die ProbandInnen die zweite Iteration des Systems besser in ihrer Benutzbarkeit empfanden.

Die Testphasen der iterativen Entwicklung ergaben einige Ergebnisse. Anhand dieser könnte das existierende System durch weitere Iterationsphasen noch verbessert werden.

9.2 Ausblick

Die Entwicklung und anschließende Analyse der Musikinstrumente zeigte, dass ein System dieser Art nicht nur technisch realisierbar ist, sondern auch von potentiellen NutzerInnen als durchaus positiv sowohl in der Konzeption, als auch in der Bedienbarkeit bewertet wurde und durch iteratives Testen sogar noch verbessert werden konnte.

Das System ist im jetzigen Zustand nach wie vor ein Prototyp, doch könnte durch umfangreiches iteratives Testen, vor allem mit verschiedenen Nutzergruppen durchaus zu einem fertigen Produkt entwickelt werden. Gerade die optische Erscheinung des User-Interface müsste hierfür noch wesentlich verbessert werden, aber auch die Gestensteuerung ist noch lange nicht voll ausgereift und könnte noch erweitert werden. Die Netzwerkverbindung der beiden Komponenten ist ein weiterer Punkt. Auf der User-Interface Seite fehlt hier noch die Möglichkeit IP Adresse und Port des Klangerzeuger PCs über das Interface selbst einzugeben. Als weitere Entwicklung wäre beispielsweise auch eine Portierung der Klangerzeugung in eine DAW über „Max For Live“ denkbar.

Glossar

Stand-Alone Hard-/Software, die eigenständig, ohne weitere Hard-/Software funktioniert

Waveguide-Display Durch Brechung von Unidirektionalem Licht mit einer vielzahl kleiner Prismen auf einer Linse wird diese zu einem durchsichtigen Display

Spacial Mapping Erstellung einer virtuellen Repräsentation der reellen Umgebung

Prototyp Vereinfachte Abbildung des Zielsystems

Mock-up Kopie eines Systems mit eingeschränktem Funktionsumfang

cross platform Entwicklung auf unterschiedlichen Plattformen (hier Betriebssystemen)

multiplatform build Kombilieren eines Projekts auf unterschiedliche Plattformen (bspw. Windows, iOS, Android, etc.) möglich

Asset In Unity entwickelte Elemente die von Entwicklern oder Unity selbst bereitgestellt werden um in anderen Unity Projekten eingesetzt zu werden

Prefab Zusammenstellung verschiedener Unity Software Elementen

Gaze Mittelpunkt des Sichtfeldes der Hololens, dass meist durch einen Weißen Punkt dargestellt wird

Immersion das Wahrnehmen der virtuellen Umgebung als real

Abbildungsverzeichnis

Abbildung 1: Schaubild des iterativen Testings (eigene Darstellung).....	11
Abbildung 2: Erhebungsmethoden der Usability (Eller 2009).....	12
Abbildung 3: Prototyp 1 (eigene Darstellung).....	18
Abbildung 4: Prototyp 2 (eigene Darstellung).....	19
Abbildung 5: Prototyp 3 (eigene Darstellung).....	20
Abbildung 6: Prototyp 4 (eigene Darstellung).....	21
Abbildung 7: Prototyp 5 (eigene Darstellung).....	22
Abbildung 8: Prototyp 6 (eigene Darstellung).....	23
Abbildung 9: Darstellung der einzelnen Komponenten (eigene Darstellung).....	39
Abbildung 10: Schematische Darstellung der Architektur von Max/MSP (eigene Darstellung)	
.....	41
Abbildung 11: Keypad der ersten Iteration (eigene Darstellung).....	44
Abbildung 12: Sequencer der ersten Iteration (eigene Darstellung).....	45
Abbildung 13: Drone Generator der ersten Iteration (eigene Darstellung).....	46
Abbildung 14: Probandin während Test (eigene Fotografie).....	59
Abbildung 15: Darstellung der SUS Skale (Brooke 2011)	61
Abbildung 16: Keypad der zweiten Iteration (eigene Darstellung).....	65
Abbildung 17: Darstellung der Speicherfunktion (eigene Darstellung).....	66
Abbildung 18: Sequencer der zweiten Iteration (eigene Darstellung).....	67
Abbildung 19: Drone Generator der zweiten Iteration (eigene Darstellung).....	68
Abbildung 20: Funktionsweise des Delay-Effekts (eigene Darstellung).....	71
Abbildung 21: Screenshot der HoloLens bei Spatial Mapping (eigene Darstellung).....	73

Tabellenverzeichnis

Tabelle 1: Auswertung Prototyp 1.....	24
Tabelle 2: Auswertung Prototyp 2.....	25
Tabelle 3: Auswertung Prototyp 3.....	26
Tabelle 4: Auswertung Prototyp 4.....	26
Tabelle 5: Auswertung Prototyp 5.....	27
Tabelle 6: Auswertung Prototyp 6.....	27
Tabelle 7: Auswertung der Ergebnisse der ersten Iteration.....	62
Tabelle 8: Auswertung SUS Test.....	63
Tabelle 9: Ergebnisse der Instrumenten-Fragebögen.....	77
Tabelle 10: Ergebnisse des zweiten SUS Fragebogens.....	78

Quellenverzeichnis

VintageSynth (2019): Yamaha DX7. Zuletzt aufgerufen am 09.03.2019 unter:
<http://www.vintagesynth.com/yamaha/dx7.php>

Nunzio, A.-D. (2013): Sequential Drum. Zuletzt aufgerufen am 09.03.2019 unter:
<http://www.musicainformatica.org/topics/sequential-drum.php>

Broll, W., Dörner, R., Grimm, P., Jung, B. (2013): Virtual und Augmented Reality (VR/AR).
Springer Verlag. Berlin Heidelberg. 351

Richter, M., Flückiger, M. (2010): Usability Engineering kompakt. Spektrum akademischer
Verlag. Zweite Auflage. Heidelberg. 139

Malinowski, S. (2019) The Magical Number Seven, Plus or Minus Two: Some Limits on Our
Capacity for Processing Information. Zuletzt aufgerufen am 09.03.2019 unter:
<http://www.musanim.com/miller1956/>

Coughlan, C. (2016): Iterative User Testing. Zuletzt aufgerufen am 09.03.2019 unter:
<https://atendesigngroup.com/blog/iterative-user-testing>

Eller, B. (2009): Usability Engineering in der Anwendungsentwicklung. Gabler. Wiesbaden. 261

Brosius, H.-B., Koschel, F., Haas, A. (2009): Methoden der empirischen
Kommunikationsforschung. 5. Auflage. VS Verlag. Wiesbaden

Microsoft (2019a): Development overview. Zuletzt aufgerufen am 09.03.2019 unter:
<https://docs.microsoft.com/en-us/windows/mixed-reality/development-overview>

Microsoft (2019b): Install the tools. Zuletzt aufgerufen am 09.03.2019 unter:
<https://docs.microsoft.com/en-us/windows/mixed-reality/install-the-tools>

Microsoft (2018): MR Basics 100: Getting started with Unity. Zuletzt aufgerufen am
09.03.2019 unter: <https://docs.microsoft.com/en-us/windows/mixed-reality/holograms-100>

UnityOSC (2019): UnityOSC Zuletzt aufgerufen am 09.03.2019 unter:
<http://thomasfredericks.github.io/UnityOSC/>

Microsoft Mixed Reality Developer Forum (2017): UDP-Communication [SOLVED] Zuletzt

aufgerufen am 09.03.2019 unter: <https://forums.hololens.com/discussion/7980/udp-communication-solved>

Hunsucker, A. (2017): Usability Testing In Augmented Reality. Zuletzt aufgerufen am 09.03.2019 unter: <https://medium.com/@AndrewJHCI/usability-testing-in-augmented-reality-df8f6c8d0d71>

Yom, M., Wilhem, T., Gauert, S. (2007) Protokolle Lauten Denkens und Site Covering In Renate Buber & Hartmut H. Holzmüller (Hrsg.), Qualitative Marktforschung Konzepte – Methoden – Analysen. Hogrefe. Göttingen. 637-652

Brooke, J. (2011): Smart Phone applications for people with Brain Injury, Zuletzt aufgerufen am 09.03.2019 unter:

http://www.tbistafftraining.info/smartphones/documents/b5_during_the_trial_usability_scale_v1_09aug11.pdf

Miranda, E. R., Wanderley, M. M. (2006): New Digital Musical instruments: Control and Interaction beyond the Keyboard. Vol. 21. A-R Editions. Wisconsin. 287

Eigenständigkeitserklärung:

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel verfasst habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Ort, Datum, Unterschrift