FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Mapping von Gesten auf Töne

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Medieninformatik und Visual Computing

eingereicht von

Birgit Chmelar
Matrikelnummer 1227330

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Assoc. Prof. Dr. Dipl.-Ing. Hilda Tellioğlu
Mitwirkung: Univ.Ass. Mag.rer.soc.oec. Roman Ganhör

Wien, 2. Juni 2016

Birgit Chmelar	Hilda Tellioğlu

Erklärung zur Verfassung der Arbeit

Birgit Chmelar
Rienößlgasse 11/9, 1040 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 2. Juni 2016

Birgit Chmelar

Danksagung

Nach monatelanger Arbeit bin ich endlich zu einem der besten Punkte der Bachelorarbeit gekommen, der Danksagung. Nicht nur, weil es bedeutet, dass die Bachelorarbeit fertig ist, sondern auch, weil ich den Menschen, die mir in der Zeit geholfen haben, offiziell danken kann. Danken möchte ich vor allem Roman Ganhör, der sich die Zeit für mich genommen hat und der mir die Möglichkeit gegeben hat an einem Thema zu arbeiten, an dem ich gerne arbeiten konnte. Eine sehr große Hilfe war auch Stefan Beyer, ohne dem ich wahrscheinlich noch länger gebraucht hätte, danke für das Beantworten von massig Fragen zu allen möglichen Themen und die Gesellschaft bei Kaffee und Co in der Zeit. Dank gilt auch meinem Bruder, Markus Chmelar, der mir sowohl bei informatik- als auch musiktechnischen Anliegen weitergeholfen hat. Für Rat und Aufmunterung waren außerdem immer Barbara Werzer, Fabian Schwarzinger, Verena Widhalm und Thomas Muhm da, danke dafür. Unterstützung kam außerdem von Oliver Hödl, danke für das Interview, und meinen Testpersonen, danke für eure Zeit. Und danke an meine Eltern, für alles. Außerdem möchte ich den Leuten der Multidisciplinary Design Group danken, dass sie mich all die Tage bei ihnen erduldet haben. Ich kann mir keinen Ort vorstellen, an dem es schöner gewesen wäre, die Arbeit zu schreiben, als im Institut.

Kurzfassung

Große Touchscreens bieten den Vorteil, dass mehrere Personen zur gleichen Zeit damit interagieren können. Dieser Aspekt fördert die Kooperation. Das Angebot von Anwendungen für Touchscreens deckt Nutzungsgebiete aus Kategorien wie Produktivität, Spiele, Musik, uvm. ab. In dieser Arbeit wird die Entwicklung des Beethoven2016 beschrieben. Ein elektronisches Musikinstrument, das kreatives und kollaboratives Musizieren ermöglicht. Obwohl Musikinstrumente im Normalfall nur von einer Person gespielt werden, soll die Größe des Touchscreen ausgenutzt werden, um zum gemeinsamen Spielen zu motivieren. Das Programm wurde im Rahmen eines User Tests von mehreren Personen getestet und die Ergebnisse wurden analysiert. Daraus sind viele Verbesserungsvorschläge entstanden, die einer weiterführenden Arbeit umgesetzt werden können.

Inhaltsverzeichnis

Kurzfassung **vii**

Inhaltsverzeichnis **ix**

1 Einleitung **1**

2 Referenzprojekte **3**
2.1 Tangible User Interfaces 3
2.2 Anwendungen für mobile Geräte 9
2.3 Zusammenfassung und Gegenüberstellung 15

3 Anforderungen **17**
3.1 Personas 17
3.2 Use Cases 21
3.3 Anwendungsszenarien 21

4 Implementierung **23**
4.1 Allgemeines 23
4.2 Mapping 23
4.3 Komponenten 25

5 User Tests **29**
5.1 Methode 29
5.2 Resultate 31
5.3 Zusammenfassung 38

6 Diskussion **41**

7 Zusammenfassung **45**

Abbildungsverzeichnis **47**

Literaturverzeichnis **49**

KAPITEL 1

Einleitung

Touchscreens können zur Bedienung von Programmen aus unterschiedlichen Anwendungsbereichen verwendet werden. Vorzufinden sind sie beispielsweise auf Fahrkartenautomaten, zur Steuerung von Maschinen in der Industrie oder zur Informations- und Orientierungshilfe bei Messen oder in Einkaufszentren. Die Anwendungsmöglichkeiten enden nicht bei praktischen Zwecken, sondern kommen auch im Bereich Unterhaltung und Spiele zum Einsatz. Im Alltag begegnen uns Touchscreens besonders in Form von Smartphones. Es gibt bereits Anwendungen, die Eingaben eines Touchscreens verwenden, um Musik zu erzeugen. Manche Anwendungen imitieren Musikinstrumente wie Klaviere oder Gitarren, andere nutzen andere Konzepte, die aus Benutzereingaben Musik erzeugen. Das Ziel dieser Arbeit ist es, ein elektronisches Musikinstrument zu entwickeln, das nicht auf die Bildschirmgröße von Smartphones beschränkt ist. Es soll Töne aus Berührungen erzeugen, ohne andere Instrumente nachzuahmen und dabei leicht zu bedienen sein. Gleichzeitig soll die Größe des Touchscreens dafür genutzt werden, um die Kollaboration zwischen Benutzern und Benutzerinnen zu fördern. Die Anwendung wird Beethoven2016 benannt in Anspielung an den Gegensatz zwischen klassischer Musik und der Art, wie Musik in dieser Anwendung produziert wird. Der Prozess, wie aus dieser Idee die Anwendung entstanden ist, wird Schritt für Schritt in dieser Arbeit beschrieben.
Es werden Referenzprojekte vorgestellt, die durch Berührungen und/oder andere Interaktionsmöglichkeiten gesteuert werden, um Musik zu erzeugen. Dadurch wird der Funktionsumfang der eigenen Anwendung abgesteckt, da die hier umzusetzenden Ideen nicht schon existieren sollten. Der dadurch festgelegte Funktionsumfang wird bei den Anforderungen beschrieben. In den Anforderungen wird nur erläutert, was das Programm können soll und nicht wie. Darauf wird im Kapitel Implementierung eingegangen. Das die Architektur der Anwendung beschreibt. Nach dem Programmieren der Anwendung wurde diese von mehreren Personen getestet. Dadurch konnte festgestellt werden, wie die Anwendung bei potenziellen Benutzern und Benutzerinnen ankommt und wo Verbesserungen beziehungsweise Erweiterungen vorgenommen werden können. Die Kritikpunkte werden

zusammengefasst, dadurch können Ideen abgeleitet werden, welche Änderungen und Erweiterungen an der Anwendung vorgenommen werden können. Zum Schluss wird ein Rückblick auf den gesamten Entwicklungsprozess geworfen. In diesem werden relevante Inhalte wiederholt.

KAPITEL 2

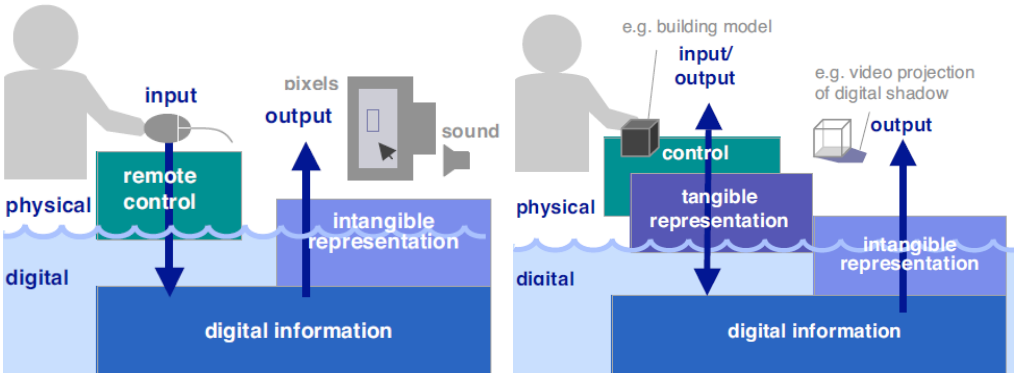
Referenzprojekte

In diesem Kapitel werden Projekte vorgestellt, die der aktuellen Idee der Anwendung ähneln. Dies dient dazu, den Rahmen für diese Arbeit festzulegen, indem in Erfahrung gebracht wird, welche Anwendungen in diesem Bereich bereits existieren. Die vorgestellten Projekte werden in zwei Kategorien unterteilt: Anwendungen für Tangible User Interfaces (TUI) und Anwendungen für mobile Geräte wie Smartphone und Tablets. Nach den Beschreibungen der Projekte werden diese zusammenfasst und gegenübergestellt. Dadurch können Anforderungen für eine neue Anwendung definiert werden. Außerdem sollen Schwachstellen und Pluspunkte erkannt werden, um diese bei der aktuellen Arbeit zu berücksichtigen.

2.1 Tangible User Interfaces

Tangible User Interfaces(TUI)[1] unterscheiden sich von den klassischen Graphical User Interfaces(GUI), die wir von PCs und Ähnlichem gewohnt sind. Bei GUIs erfolgt die Eingabe im Normalfall mit Tastatur und Maus, die graphische Ausgabe sehen wir auf dem Monitor in Form von Pixel. Tangible bedeutet *greifbar* oder *materiell*. Die Prinzipien von GUI und TUI sind in Abbildung 2.1 dargestellt. TUIs verwenden *Tangibles*, diese sind physikalische Objekte, die der Steuerung dienen. Die Formen dieser Objekte können sich von Anwendung zu Anwendung unterscheiden, je nachdem welchem Zweck sie dienen. Die Objekte können einerseits bewegt werden, um mit dem System zu interagieren, und dienen andererseits selbst als Repräsentation der Eingabe. Die Repräsentation kann zusätzlich durch *intangible representation* ergänzt werden, dies sind Projektionen, die nicht zur Steuerung verwendet werden können, sondern nur Information darstellen.

Tangible User Interfaces können in sieben Genres eingeteilt werden. Für die Zwecke dieser Arbeit ist es ausreichend, eines davon vorzustellen - das *Tabletop TUI*. Diese Art von TUIs haben eine Oberfläche, über die die Interaktion abgewickelt wird. Objekte, die auf der Oberfläche liegen oder bewegt werden, werden vom System geortet. Die Daten,



(a) Graphical User Interface (b) Tangible User Interface

Abbildung 2.1: Konzeptionelle Darstellung von GUIs und TUIs[1]

die daraus gewonnen werden, beeinflussen das Programm. Je nach Anwendung haben unterschiedliche Parameter wie Position, Ausrichtung und so weiter einen Einfluss auf das System. Gleichzeitig wird auf die gleiche Fläche anwendungsabhängig zusätzliche Information projiziert.

2.1.1 reacTable

Der reacTable[2][3] ist ein elektronisches Instrument, das sowohl für Anfänger, als auch für Fortgeschrittene ansprechend sein soll. Das Musizieren ist alleine oder mit anderen Personen auf einem oder über mehrere Geräte, die über das Netz verbunden werden, möglich. Die Komponenten des reacTable und deren Zusammenhänge sind in Abbildung 2.2 dargestellt.

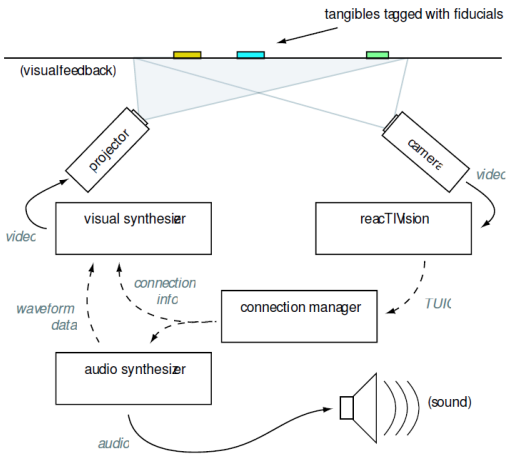


Abbildung 2.2: Komponenten des reacTables[2]

Das Gerät hat ein Tangible User Interface, das auf unterschiedliche Objekte auf der Oberfläche reagiert. Diese Objekte sind entweder Plättchen oder Würfel, auf welchen Symbole abgebildet sind. Diese Abbildungen werden von einer Kamera unter der transluzenten Tischplatte aufgenommen und zur Verarbeitung an die nächste Komponente weitergegeben, welche für die Erkennung des Typs, der Position und des Winkels zuständig ist. Für diese Aufgabe ist das Open Source Framework *reacTIVision* zuständig. In Verbindung mit diesem werden die Symbole auf den Objekten auch *Fiducials* genannt. Es wird zwischen Objekten aus sechs Gruppen unterschieden. Beispiele für diese Gruppen sind Sound Generatoren und Audio Filter. Für jede Gruppe ist festgelegt, wieviele Eingänge und Ausgänge ein Objekt hat. Durch diese Differenzierung und ein paar Regeln wird mit den Informationen aus dem reacTIVision Framework festgestellt, welche Objekte auf der Oberfläche zusammengehören. Dies passiert im *Connection Manager*. Abhängig des Typs eines eingesetzten Objekts können Parameter durch Rotation oder einfachen Fingergesten eingestellt werden. Die gesammelten Daten des Connection Managers werden an einen Visual- und einen Audio Synthesizer weitergeleitet. Letzterer erzeugt die Sounds. Der Visual Synthesizer bezieht Daten sowohl vom Connection Manager als auch vom Audio Synthesizer. Ein Projektor bildet diese Informationen von unten auf die Tischoberfläche ab und erzeugt somit zusätzlich visuelles Feedback, welches die Parameter der Objekte graphisch abbildet und die Verbindungen zwischen zusammengehörigen Objekte verdeutlicht. Abbildung 2.3 zeigt ein Foto, bei dem mehrere Personen mit dem Instrument interagieren. Man erkennt mehrere Objekte und Verbindungen zwischen diesen, welche von unten auf die Oberfläche projiziert werden. Dadurch kann der aktuelle Status des Systems besser erfasst werden.

Abbildung 2.3: Personen in Interaktion mit dem reacTable[4]

Eine weitaus billigere Variante des reacTables ist die App, die für iOS und Android erhältlich ist.

2.1.2 mixiTUI

MixiTUI[5] bedient sich ebenfalls eines Tangible User Interfaces, um elektronische Musik zu erzeugen. MixiTUI ist besonders für Live Performances ausgelegt. Die Entwickler und Entwicklerinnen haben sich beim Design des Systems an Interviews und Umfragen elektronischer Musiker und Musikerinnen orientiert. Dabei kam öfter der Wunsch auf, dass Live Performances interaktiver gestaltet werden können sollten. Bisher war dies schwierig, da die Musiker und Musikerinnen die visuellen Marker zur richtigen Zeit auf den Tisch legen mussten, was Druck auf die Musiker und Musikerinnen ausüben kann. Gegen diesen Druck sollen bestimmte Elemente dagegen wirken. Beim mixiTUI wird, wie beim reacTable, reacTIVision verwendet. Die Objekte haben auf der Unterseite ebenfalls Fiducials und werden beim mixiTUI auch als *Token* bezeichnet. Es wird zwischen *loop token*, *effect token* und *control token* unterschieden. Loop Token sind für die Produktion von Sounds zuständig. Um den Druck von den Livemusikern und Livemusikerinnen zu nehmen, werden mehrere Start- und Stopvarianten zur Verfügung gestellt, mit denen gesteuert werden kann ab und bis wann aus Loop Token Musik erzeugt werden soll. Außerdem gibt es sogenannte *Sessions*. Durch diese bekommen die Musiker und Musikerinnen zusätzliche Kontrolle über die Token. Das System spielt eine von fünf Sessions ab, zwischen denen gewechselt werden kann. Ein Token kann in mehreren Sessions verwendet werden, die Eigenschaften der Token in Sessions können von den Musikern und Musikerinnen variiert werden. Das Projekt wurde mithilfe von Fragebögen des Publikums nach einer Live Performance evaluiert. Dabei ist ein Artist aufgetreten, der sowohl Lieder mit dem Laptop als auch welche mit dem mixiTUI performed hat. Die Performance von letzterem wurde mit großer Mehrheit der Performance mit dem Laptop vorgezogen. In Abbildung 2.4 sieht man den Tisch in Verwendung. Die Token haben auf der Oberseite unterschiedliche Symbole, um sie besser unterscheiden zu können.

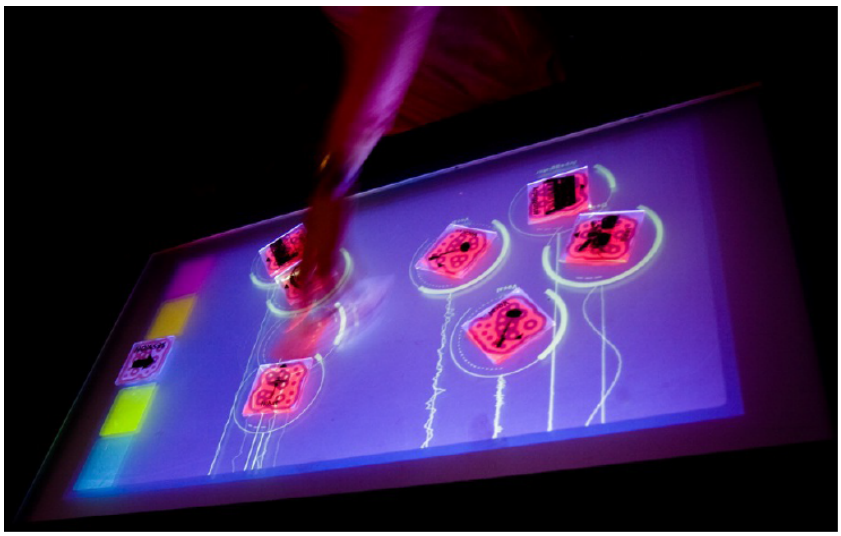


Abbildung 2.4: Interaktion mit mixiTUI[5]

2.1.3 Kollaborative Musik auf einer Multi-Touch Oberfläche

Klügel u.a.[6] haben eine Applikation zur Musikerzeugung für ein Gerät mit Tabletop Interface entwickelt. Sie machen jedoch nur Gebrauch der Touch Interaktion und verzichten auf die Verwendung der Tangibles. Das Ziel bestand darin, die Zusammenarbeit von mehreren Benutzern und Benutzerinnen auf einem Tisch zu ermöglichen. Touchgeräte treffen die Anforderungen und eignen sich gut für die Anwendung, da sie mittlerweile etabliert sind und der Umgang mit ihnen daher alltäglich ist, und nicht erst erlernt werden muss. Sie machen nur Gebrauch der Touch Interaktion und verzichten bewusst auf Tangibles, da diese räumliche Beschränkungen mit sich bringen würden. Das wurde speziell für Zusammenarbeiten in Gruppen als großer Nachteil empfunden. Alle Klänge werden durch Synthesizer erzeugt. Anforderungen an den Prototypen waren, dass er unmittelbar und konstant Feedback liefert und Kollaboration unterstützt. Die Applikation stellt unterschiedliche Elemente zur Verfügung: *Sequences*, die kurze, vom Benutzer oder von der Benutzerin selbst erstellte Musikstücke beinhalten, *Synthesizer*, und *Player*. Diese und andere Elemente sind schematisch in Abbildung 2.5 zu sehen (4a-c). Sie können als Knoten erzeugt und beliebig miteinander verbunden werden. Diese Verbindungen werden in der Anwendung durch Linien zwischen Knoten dargestellt.

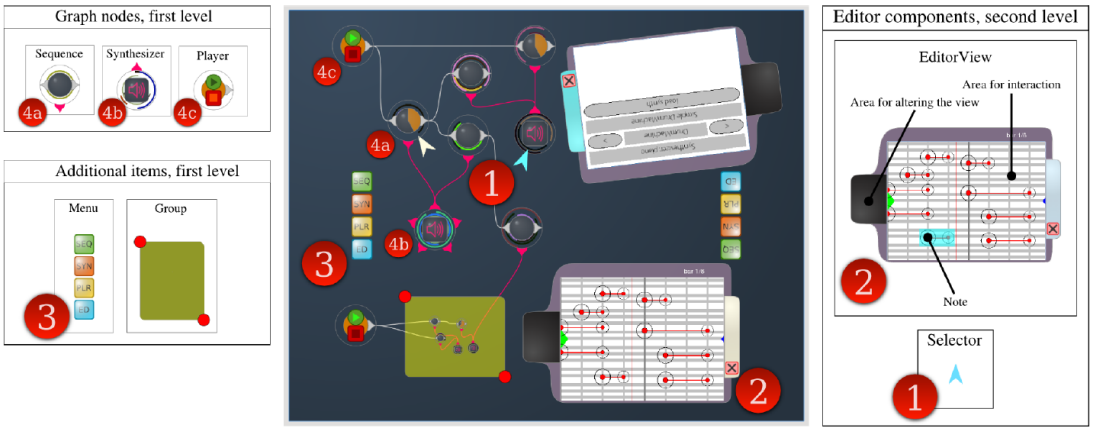


Abbildung 2.5: Konzeptionelle Veranschaulichung der Anwendung für kollaborative Musik von Klügel u. a.[6]

Die Entwickler und Entwicklerinnen haben die Struktur der Applikation auf zwei Level aufgeteilt. Das erste befasst sich mit dem Erstellen von Sequenzen und Anordnen auf der Oberfläche. Elemente können beliebig oft durch Drücken und Wegziehen auf einen der vier, färbigen Buttons erzeugt werden. Für eine bessere Übersicht können Elemente gruppiert und geschrumpft werden. Diese sind durch einen andersfarbigen Hintergrund gut zu erkennen. Das zweite Level dient zum ändern von Eigenschaften der Elemente(2). Dazu wird ein Selektor(1) benötigt, um das zu verändernde Element auszuwählen. Die Synthesizer bieten unterschiedliche Optionen zur Klangerzeugung der Sequenzen, die damit verbunden sind. Der Player dient zum Abspielen der Sequenzen, die mit ihm

verbunden sind. Zusätzlich können die Eigenschaften von vorhandenen Knoten editiert werden. Die Position der Knoten auf dem User Interface hat keinerlei Auswirkung auf deren Eigenschaften, wodurch bei der Zusammenarbeit innerhalb einer Gruppe möglichst viele Freiheiten geboten werden, da sich niemand in die Quere kommt.

2.1.4 Bricktable

Der Bricktable[7] ist ein Tisch mit TUI, der im Zuge der Entwicklung für eine Mehrbenutzeranwendung gebaut wurde. Aus Kostengründen wurde er im Rahmen eines Open-Source Projektes entwickelt. Bauteile und Hardware wurden sparsam zusammengestellt. Auf einer Karte sollten Pfade gemalt werden, die vom Bricktable musikalisch interpretiert werden. Für das Open Source Projekt wurden im Laufe der Zeit mehrere Anwendungen zur Erzeugung von Musik entwickelt. Zwei davon werden in den Unterkapitel Weather Report und Roots vorgestellt.

Weather Report

Bei der Anwendung Weather Report ist auf der Oberfläche des Tisches eine Karte der Vereinigten Staaten von Amerika zu sehen, die mit den aktuellen Oberflächentemperaturen, welche stündlich aktualisiert werden, farbkodiert ist. Objekte können über die Oberfläche bewegt werden, wodurch der Bricktable Töne erzeugt. Wenn die Bewegung des Objekts gestoppt wird, beginnt das Abspielen der Aufnahme in einer Schleife. Die Oberflächentemperatur hat Einfluss auf die Klangfarbe, wodurch sich auch die Wiedergabe der Aufnahme von ein und derselben Bewegung über Zeit verändert. Der Bricktable mit der Anwendung Weather Report ist in Abbildung 2.6 zu sehen.

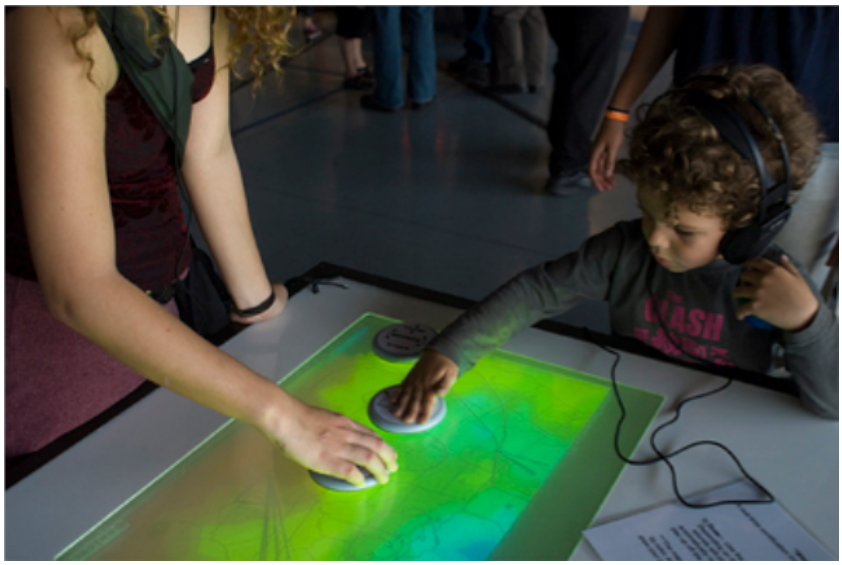


Abbildung 2.6: Bricktable mit der Weather Report Anwendung[7]

Roots

Bei der Interaktion mit der Anwendung Roots[8] kann der Benutzer oder die Benutzerin selbst entscheiden, wie viel Kontrolle er über die Musikerzeugung haben möchte. Bei wenig Kontrolle wachsen bei einer Berührung der Oberfläche zufällig Äste vom Ursprung aus, wobei jeder Punkt, an dem die Oberfläche berührt wird ein Ursprung ist. Im Zuge dessen kommen die wuchernden Äste in Kontakt mit unsichtbaren Zonen, woraufhin Töne erzeugt werden. Wenn der Benutzer gerne mehr Kontrolle hätte, dann hat er die Möglichkeit durch das Platzieren von Gegenständen auf der Oberfläche „Kraftfelder“ zu erzeugen, die das Wachsen der Äste beeinflussen. Am meisten Kontrolle hat der Benutzer, wenn er mit seinem Finger fest auf den Tisch drückt und über die Oberfläche fährt, dadurch werden Töne direkt aus der Bewegung erzeugt. Abbildung 2.7 zeigt diese Anwendung.

Abbildung 2.7: Bricktable mit der Roots Anwendung[8]

2.2 Anwendungen für mobile Geräte

Seit Markteinführung des Apple iPhones 2007 gewannen Smartphones stetig mehr Bedeutung. Dem von Apple verwendeten Betriebssystem iOS folgten bald andere mobile Betriebssysteme. Sie haben Multi-Touch Displays. Smartphones und Tablets sind ständige Begleiter, die durch Applications, kurz Apps, eine Menge Unterhaltungsmöglichkeiten bieten. Auch das Angebot an Apps zur Musikerzeugung kommt nicht zu kurz. Ein paar Erwähnenswerte werden in den folgenden Unterkapiteln vorgestellt.

2.2.1 Magic Fiddle

Magic Fiddle[9] ist eine Anwendung für das iPad, die das Gerät zu einem Instrument werden lässt. Die Entwickler und Entwicklerinnen betonen im Paper, dass sie nicht nur die Touch-Interaktion nutzen wollen, sondern auch die Absicht haben einen materiellen,

greifbaren Aspekt miteinzubeziehen. Das Instrument ist einer Violine nachempfunden, soll dieser jedoch nicht 1:1 entsprechen. Ein Vorgänger der Magic Fiddle war *Smules Ocarina*. Sie war ebenfalls einem echten Instrument nachempfunden und verwendet mehrere Sensoren, die das iPad zur Verfügung stellt. Darunter ist beispielsweise das Mikrofon, in das die Benutzer beziehungsweise Benutzerinnen blasen müssen, damit Ton erzeugt wird. Beim Design für die Magic Fiddle mussten Vereinfachungen vorgenommen werden, um das Spielen auf dem Gerät möglichst praktikabel zu gestalten. Die zwei größten Anpassungen, die gemacht wurden, waren, dass gegenüber einer Violine nur drei statt vier Saiten vorhanden sind, um die Saiten beim Greifen möglichst bequem erreichen zu können. Außerdem wird beim Anschlag nicht zwischen den Saiten unterschieden, stattdessen ist ein gemeinsamer Bereich für alle vorhanden - die sogenannte *bow region*. Bei einer Berührung damit werden alle Saiten angespielt, die zur gleichen Zeit berührt werden.

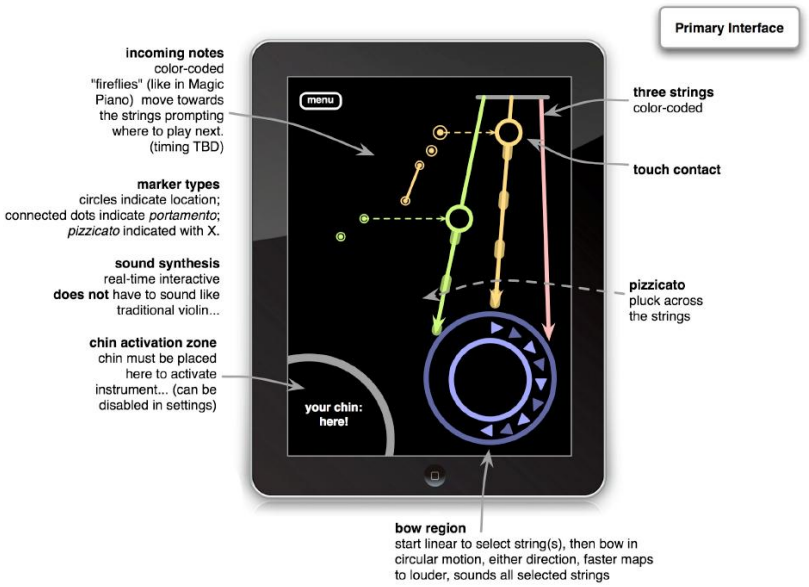


Abbildung 2.8: Konzeptionelles Interface der Magic Fiddle[9]

In Abbildung 2.8 ist das konzeptionelle Interface zu sehen, mit dem der Benutzer oder die Benutzerin interagiert. Ursprünglich war geplant, dass während dem Spielen das Kinn Kontakt mit dem Display des iPads haben muss, dies war technisch jedoch nicht machbar, da zum Beispiel Berührungen durch Bärte nicht korrekt erkannt werden konnten. In Abbildung 2.9 kann man sehen, wie zwei Personen auf der Magic Fiddle spielen.

2.2.2 Musyc

Musyc[10] ist eine iOS App von Fingerlab. Klänge entsteht dadurch, dass unterschiedliche Objekte aneinander stoßen. Objekte sind dabei einfache, geometrische Formen wie

Abbildung 2.9: Zwei Benutzer mit der Magic Fiddle[9]

Dreiecke oder Kreise in unterschiedlicher Größe. Diese Formen können einzeln vom Benutzer beziehungsweise der Benutzerin durch Tippen auf den Bildschirm erzeugt werden, oder durch das Definieren einer „Quelle“, die dann Objekte in regelmäßigen, zeitlichen Abständen automatisch erzeugt. Unterschiedliche Formen erzeugen unterschiedliche Töne. Die Objekte verhalten sich nach physikalischen Gesetzen, also fallen nach unten oder prallen voneinander ab, wenn sie mit anderen Objekten in Berührung kommen. Die Klänge können aus unterschiedlichen Gruppen gewählt werden. In Abbildung 2.10 links sieht man Kreise, die regelmäßig erzeugt werden und dann mit Linien und anderen Formen in Berührung kommen. Jede Berührung zwischen den Formen erzeugt Töne. In der rechten Abbildung sieht man ein Menü mit unterschiedlichen Bedienelementen. In der Spalte rechts sieht man eine Auswahl von Formen, die erzeugt werden können. Der Benutzer beziehungsweise die Benutzerin kann außerdem eingreifen und Objekte manuell verschieben. Dies erzeugt ebenfalls Töne, wenn Formen aneinander stoßen. Optional können Bewegungen aufgenommen und in einer Schleife abgespielt werden.

2.2.3 Pyxis Minor

Pyxis Minor[12][13] ist eine Applikation, zur Erzeugung elektronischer Musik. Sie wurde für iOS Geräte entwickelt. Im Zuge eines iterativen Design Prozesses mit Feedback von potenziellen Nutzern und Nutzerinnen ist die Anwendung entstanden, deren Zielgruppe sowohl musikalisch erfahrene als auch unerfahrene Personen sein sollen. Zu Beginn wurde ein simples, grafisches Interface entwickelt. In Abbildung 2.11 ist ein Screenshot der App zu sehen. Der Benutzer beziehungsweise die Benutzerin hat ein Gitter vor sich, auf dem er Knoten platzieren kann. Die Koordinaten der Knoten bestimmen den Ton und die Geschwindigkeit. Sobald die gewünschte Anzahl an Knoten gesetzt ist, bewegt sich ein Punkt entlang einer geraden Linie von Knoten zu Knoten im Kreis und erzeugt

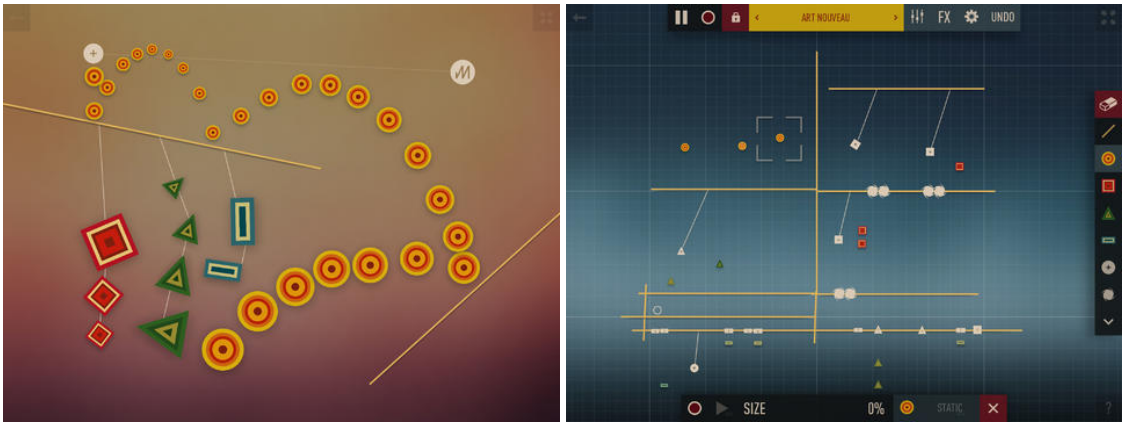


Abbildung 2.10: Screenshots von Musyc[11]

jedesmal einen Ton, wenn er an einem Eckpunkt ankommt. Ein Benutzer beziehungsweise eine Benutzerin kann bis zu vier dieser Schleifen kreieren, diese sind mit vier Symbolen am linken Bildschirmrand assoziiert. In der Abbildung sieht man statt dem obersten Symbol das Menü der orangenen Schleife. Aufgrund des Feedbacks von Testteilnehmern und Testteilnehmerinnen im Zuge der insgesamt drei Design Iterationen wurden mehr Möglichkeiten die Parameter der Musik zu verändern hinzugefügt. Außerdem wurde die Anwendung um ein detaillierteres Tutorial und die paarweise Synchronisierung des Taktes der Anwendung, um zu zweit spielen zu können, erweitert.

2.2.4 TC-11

Der TC-11[14][15][16] ist ein Synthesizer, der für das iPad entwickelt wurde. TC-11 verwendet sowohl Multi-Touch Input des Screens, als auch Messwerte der Bewegungssensoren des Gerätes, um Töne zu erzeugen. Im Gegensatz zur Magic Fiddle, die einer Violine nachempfunden ist, wird im Paper erwähnt, dass die Entwickler von TC-11 absichtlich auf visuelle Metaphern, wie Saiten, die von Instrumenten bekannt sind. Auch Drehknöpfe und andere Bedienelemente, die man aus der realen Welt kennt, wurden bewusst nicht verwendet. Obwohl die App eine Vielzahl an Voreinstellungen bietet, ist sie darauf ausgelegt, dem Benutzer beziehungsweise der Benutzerin die Wahl der Einstellungen selbst zu überlassen. Es können sogenannte *Patches* angelegt werden. Jeder Patch speichert ein Set an Einstellungen, die für den Klang ausschlaggebend sind. Dabei stehen unterschiedliche *Synthesis Objects* zur Verfügung. Synthesis Objects sind unterschiedliche Wavetable Oszillatoren, Filter, Verstärker und Effekte. Die Verarbeitung der Berührungen wird von *Multi-Point Controllern* vorgenommen. Es wird zwischen Single Touch und Group Touch unterschieden. Je nach einer dieser zwei Arten werden steht jede Berührung mit dem Display für sich oder die Berührungen werden zu einer Gruppe zusammengefasst, in der Werte aus allen Berührungen berechnet werden. Bei der Klangsynthese spielen Distanzen, Winkel, Rotation, Geschwindigkeit und Timing der Berührungen eine Rolle. Ein Foto der

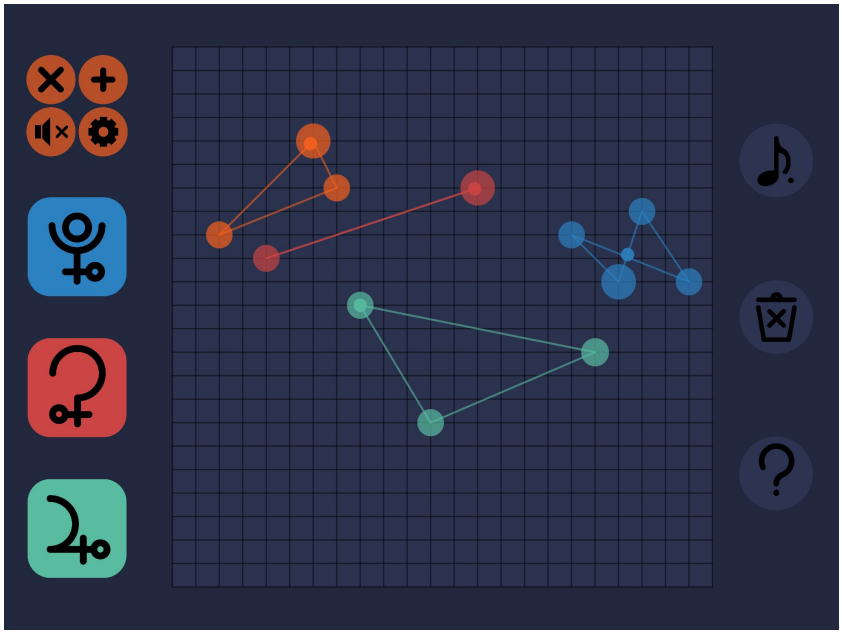


Abbildung 2.11: Screenshot von Pyxis Minor[13]

Anwendung ist in Abbildung 2.12 zu sehen. Die graphische Oberfläche kann aus mehreren Optionen bestimmt werden. Beispiele sind in den Abbildungen 2.13b und 2.13c zu sehen, Abbildung 2.13a zeigt einen Screenshot des Menüs mit Einstellungen der Klangsynthese.

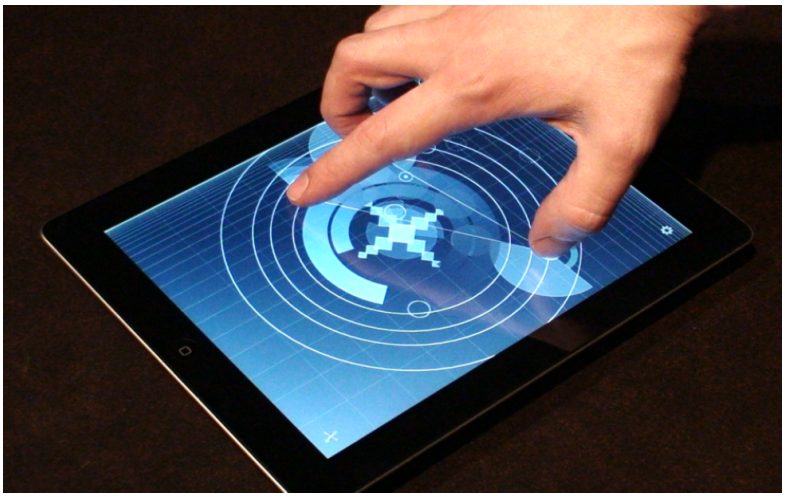
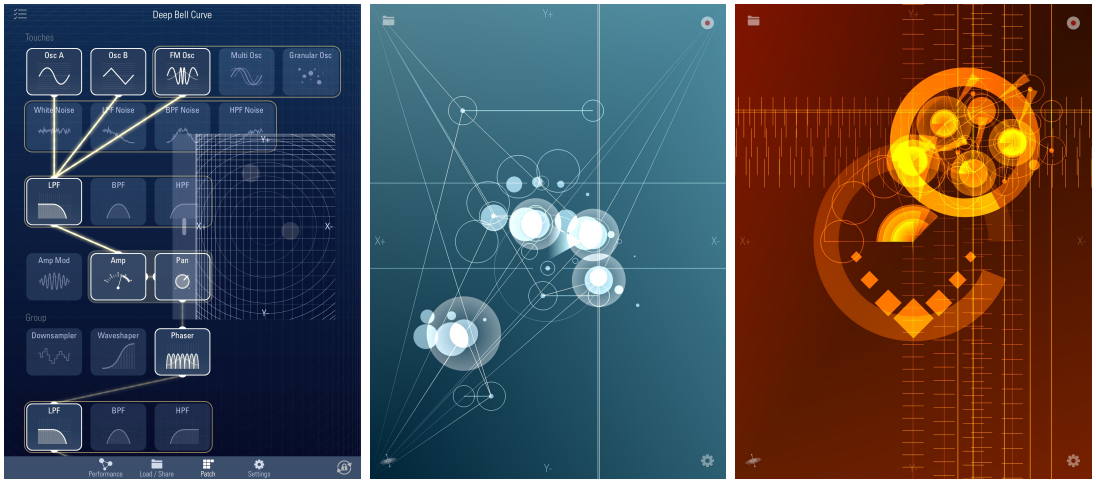


Abbildung 2.12: Interaktion mit TC-11[15]



(a) Menü der Klangsynthese (b) Screenshot von TC-11 (c) Screenshot von TC-11

Abbildung 2.13: Screenshots von TC-11. Links die Einstellungen von Synth Objects, mitte und rechts die Anwendung in Aktion[14]

2.2.5 MetaTravels und MetaLonsdale

Die Applikationen MetaTravels und MetaLonsdale[17] wurden für die Zielgruppe von Perkussionisten entwickelt. Als Plattform wurde das iPad verwendet. Erprobt wurden die Anwendungen von einem Ensemble. Beim Tippen auf das Display werden Töne abhängig von der Position erzeugt. Beim Wischen werden durchgehend Tonaufnahmen abgespielt. Die Geschwindigkeit der Bewegung wird dabei auf die Lautstärke gemapped. Die Anwendungen verfügen über die Möglichkeiten selbst Töne und Geräusche aufzunehmen und diese wieder abzuspielen. MetaTravels und MetaLonsdale sind auf improvisatorische Performances ausgelegt. Kooperation mehrerer Musiker wird durch eine Verbindung über das Netzwerk erreicht. Der Unterschied zwischen den beiden Applikationen ist, dass bei MetaTravels im Gegensatz zu MetaLonsdale auch chromatische Aufnahmen zur Verfügung standen, das heißt, es wurden auch Halbtöne verwendet.

2.2.6 MorphWiz

MorphWiz[18] ist eine Anwendung für Black Berries, Apple und Android Geräte von Wizdom Music. Die Standardoberfläche der App ist in Abbildung 2.14 zu sehen. Bei Berührungen mit dem Display wird abhängig von der Position Musik erzeugt. In der App können Änderungen an Parametern und Einstellungen vorgenommen werden, die die resultierenden Töne beeinflussen. Bewegungen entlang der horizontalen Achse variieren die Tonhöhe. Dabei kann man einstellen, wie viele Oktaven aktuell gespielt werden können, eine bis vier sind möglich. Außerdem lässt sich der festlegen, mit welchem Ton am linken Bildschirmrand begonnen werden soll. Für die vertikale Achse können zwei Wellenformen gewählt werden. Zur Auswahl stehen dabei die Sinus-, Rechteck-, Dreieck-

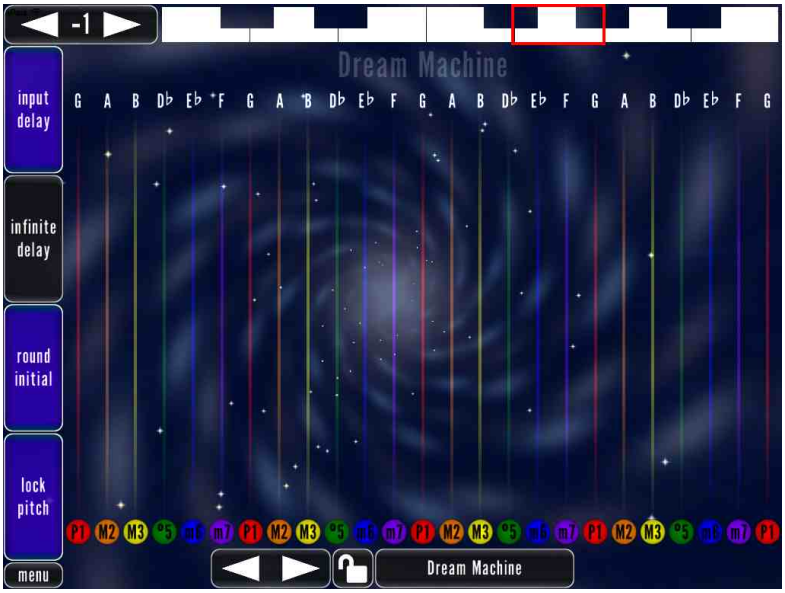


Abbildung 2.14: Screenshot von MorphWiz mit den Default Layout Einstellungen[18]

und Sägezahnwelle. Siehe Abbildung 2.15. Es kann jeweils eine Welle für den unteren (Startwelle) und den oberen (Endwelle) Bildschirmrand gewählt werden. Sind Start- und Endwelle unterschiedlich gibt es einen Übergang zwischen den gewählten Formen entlang der y-Achse. Dies entspricht der Idee des Morphings und ist ein wichtiger Teil der gesamten Anwendung.

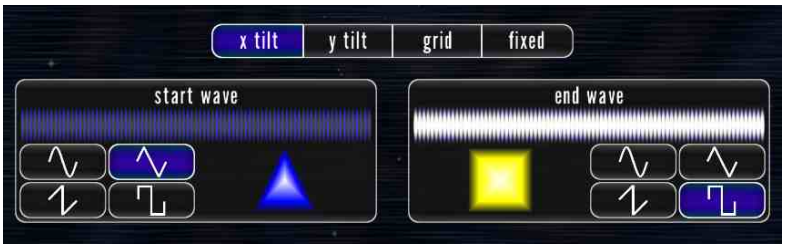


Abbildung 2.15: Einstellungen für Start- und Endwelle in MorphWiz[18]

2.3 Zusammenfassung und Gegenüberstellung

Es wurden Projekte vorgestellt, die Ähnlichkeiten mit der geplanten Anwendung aufweisen. Dabei wurde zwischen Anwendungen für TUIs und welchen für mobile Geräte unterschieden. Erstere werden größtenteils über Objekte auf der Oberfläche gesteuert, Fingergesten dienen bis auf einen Sonderfall als Ergänzung oder werden gar nicht benötigt. Apps für mobile Geräte werden hingegen größtenteils durch Multi-Touch Gesten gesteuert, und durch andere Sensoren ergänzt. Das Projekt von Klügel et. al. in Abschnitt 2.1.3 ist

ein Sonderfall, da es keine Tangibles verwendet aber ebenso keine mobile Anwendung ist.

Die Anwendung Roots für den Bricktable aus Abschnitt 2.1.4 ist das Projekt, das aus der Kategorie der TUIs am meisten Ähnlichkeiten mit der geplanten Anwendung aufweist. Roots lässt jedoch in erster Linie wilde Äste und bietet nur in zweiter Linie ein 1:1 Mapping von Gesten auf Töne. Bis auf den Bricktable sind alle Projekte für TUIs aus diesem Kapitel darauf ausgelegt Musik durch das Zusammensetzen bestimmter Objekte zu produzieren. Die Konzepte der Touch-Applikationen für mobile Geräte unterscheiden sich stärker als die der vorgestellten TUIs. Für diese wurde eine App vorgestellt, die einem echten Instrument nachempfunden ist (Magic Fiddle) und eine App, die genau dies nicht will(TC-11). Außerdem wurden Apps vorgestellt, die abstraktere Funktionsformen aufweisen. Darunter eine Anwendung mit Punkten, die sich auf einer Strecke von Eckpunkt zu Eckpunkt bewegen und dabei Töne erzeugen (Pyxis Minor) und eine Anwendung, die geometrische Formen verwendet, die sich an physikalische Gesetze halten und Töne erzeugen, wenn sie mit anderen Formen zusammenstoßen (Musyc). Nicht zu vergessen ist MorphWiz aus dem Unterkapitel 2.2.6, das abhängig von Parametern Töne aus den Berührungen des Touchscreens erzeugt. Die geplante Anwendung aus dieser Arbeit kommt MorpWiz am nächsten, weshalb darauf geachtet werden sollte, dass sich diese zwei Anwendungen nicht zu sehr ähneln.

Der Fokus der Projekte unterscheidet sich leicht. Den größten Unterschied der TUI Projekte sieht man zwischen dem mixiTUI(2.1.2) und Klügel u. a.(2.1.3). MixiTUI ist speziell für Live Performances ausgelegt. Klügel hat den Fokus auf Kollaboration gelegt. Beim Bricktable stand ebenfalls Kollaboration im Vordergrund. Beim reacTable wurde auf beides Wert gelegt. Die Hintergründe der Apps für mobile Geräte sind teilweise weniger ernsthaft. Der Vorteil an ihnen ist, dass sie aufgrund ihrer Mobilität besser für den Alltag geeignet sind als TUIs. Die Magic Fiddle(2.2.1) und MetaLonsdale(2.2.5) sind gut dafür geeignet in einer Gruppe zu musizieren. Pyxis Minor(2.2.3) bietet explizit die Möglichkeit den Takt der App von zwei Geräten zu synchronisieren. Keine der Apps ist dafür ausgelegt, dass mehrere Personen auf einem Gerät spielen, jeder hat sozusagen sein eigenes Instrument. Aus dem Abschnitt der TUIs ermöglicht hingegen nur der reacTable das Spielen global auf mehreren Geräten, die über ein Netzwerk miteinander verbunden werden können. Die anderen fördern Kooperation durch das Verwenden eines einzigen Gerätes, auf dem mehrere Personen gleichzeitig spielen können.

Die Zielgruppen des reacTable sind sowohl Anfänger und Anfängerinnen als auch Fortgeschrittene. Klügel u. a. fokusiert auf Experten und Expertinnen im Bereich von elektronischer Musik. MixiTUI ist für Live Performances ausgelegt und MetaTravels beziehugsweise MetaLonsdale sind speziell für Perkussionisten.

KAPITEL 3

Anforderungen

Mithilfe der in Kapitel 2 vorgestellten Referenzprojekte wurden Überlegungen angestellt, die für diese Arbeit sinnvoll erscheinen. Sinnvoll in der Hinsicht, dass sie sich von bisherigen Anwendungen abhebt. Bei der Recherche der Referenzprojekte hat sich herausgestellt, dass keine der Touchanwendungen darauf ausgelegt ist, die Kollaboration mehrerer Personen auf einem einzigen Gerät zu unterstützen. Dies unterstützt die ursprüngliche Idee, die Größe des verfügbaren Monitors auszunutzen, um die Benutzung mehrerer Personen gleichzeitig zu fördern. Die Ideen für die Anwendung werden in Use Cases erläutert und anschließend in Anwendungsszenarien in realistische Situationen eingebettet.

3.1 Personas

Personas sind fiktive Personen, die bei der Softwareentwicklung eingesetzt werden. Sie stehen stellvertretend für die Zielgruppe der Software und werden dafür verwendet, realistische Szenarien auszuarbeiten, in denen das jeweilige Programm zur Anwendung kommt. Es werden drei Personas vorgestellt, zwei von ihnen stellen typische Benutzer beziehungsweise Benutzerinnen der Software dar, die dritte Persona ist ein Beispiel für eine Person, die die Anwendung nicht verwenden würde als Kontrast. Bei der Anforderungserhebung kann man sich in die Persona hineinversetzen, um zu ergründen, was sie sich von der Anwendung wünschen oder erwarten würden oder welche Hilfestellungen unter Umständen benötigt werden.

3.1.1 1. Persona - Lisa Klee

Abbildung 3.1: Persona - Lisa Klee[19]

- 22 Jahre alt
- spielt Klavier und singt leidenschaftlich gerne
- studiert Betriebswirtschaft
- macht zwei Mal in der Woche Yoga
- verwendet ein Tastenhandy, hat von ihren Freunden zum Geburtstag ein bebrauchtes Tablet geschenkt bekommen, um sie auf den Geschmack zu bringen
- wohnt in einer Zweier-WG
- geht gerne ins Theater
- fährt gerne Motorrad
- hat zwei ältere Geschwister und einen jüngeren Bruder
- hat eine Ausbildung zur Kosmetikerin angefangen und abgebrochen
- bevorzugt Grün- und Schwarztee, Kaffeeverweigerin
- nicht besonders entscheidungsfreudig
- liest sehr gerne Bücher, besonders Joanne K. Rowling und Harry Potter
- Eltern sind geschieden, kein gutes Verhältnis zu ihrer Mutter, seitdem sie die Kosmetikausbildung abgebrochen hat
- geht vorsichtig mit elektronischen Geräten um

3.1.2 2. Persona - Harald Bauer

Abbildung 3.2: Persona - Harald Bauer[20]

- 44 Jahre alt
- Lehrer für Mathematik an einem Gymnasium
- verheiratet, 2 Töchter, 5 und 8 Jahre alt
- hat eine Schilehrerausbildung
- wohnt in einem Haus mit Garten
- hauptsächlich mit dem Auto unterwegs
- gutes Verhältnis zu seinen Eltern, die in der gleichen Ortschaft wohnen
- befasst sich gerne mit Smartphones und Technik seit ihn seine Halbschwester dafür begeistert hat
- kocht gerne
- hat sich ein bisschen Programmieren selbst beigebracht
- hat zum Leidwesen seiner Frau eine Schwäche für guten Cognac
- spielt leidenschaftlich gerne die Siedler von Catan
- kann nicht gut mit anderen zusammenarbeiten
- schlägt gerne die Zeit mit Quiz und Rätselspielen auf Smartphone und Tablet tot, vor allem mit seinen Töchtern
- verwendet gerne Spotify

3.1.3 Non Persona - Michael Roh

Abbildung 3.3: Persona - Michael Roh[21]

- 54 Jahre alt
- selbstständig
- viel auf Reisen, Wanderurlaube, gerne in der Natur
- schaltet außerhalb der Arbeitszeiten sein Smartphone aus
- hat einen Border Collie
- hört gerne Heavy Metal
- geht hin und wieder gerne Fallschirmspringen
- Vegetarier
- hat eine Ausbildung zum Rettungssanitäter
- war früher oft im Casino und spielsüchtig
- abenteuerlustig und spontan
- sehr geduldig, zuverlässig und ehrgeizig
- hat sich ein Heimkino eingerichtet
- fährt nach Möglichkeit mit dem Fahrrad zur Arbeit
- hat den Segelschein für Binnengewässer

3.2 Use Cases

In diesem Kapitel werden Anwendungsfälle für das Programm vorgestellt. Zur Zeit ist nur der 2. Anwendungsfall mit dem ersten Unterpunkt implementiert.

1. **Tutorial**
   Beim Start der Anwendung soll ein kurzes Schritt für Schritt Tutorial erscheinen, das übersprungen werden kann und währendem die Anwendung läuft nicht wieder angezeigt werden soll. Das Tutorial beschreibt die wichtigsten Funktionen.

2. **Töne durch Berührung erzeugen**
   Der Benutzer kann mit einem oder mehreren Fingern den Touchscreen berühren. Dadurch werden Töne direkt aus der Bewegung erzeugt. Optional:

   - wenn sich Berührungen auf dem Touchscreen näher als einen festgelegten Radius kommen, oder
   - wenn sich Strecken von Berührungen nach kurzer Zeit kreuzen,

   werden die Töne durch zusätzliche Parameter variiert

3. **Aufnehmen von Tönen**
   Gerade gespielte Töne sollen jederzeit abgespeichert werden können, ohne die Aufnahme explizit im Vorhinein starten zu müssen.

4. **Abspielen von Aufnahmen**
   Eine Aufnahme kann abgespielt werden. In Ergänzung dazu können durch weitere Berührungen neue Töne parallel erzeugt werden.

3.3 Anwendungsszenarien

Bei den Anwendungsszenarien werden Use Cases zu größeren Einheiten in einem realistischen Kontext zusammengefasst. Dafür werden die vorgestellten Personas herangezogen.

1. Lisa will sich im Zug die Zeit vertreiben und nimmt das Tablet zur Hand, das sie geschenkt bekommen hat. Sie öffnet die App, die sich noch auf dem Gerät befindet und wischt nach dem Tutorial mit einem Finger über das Display und lässts ich überraschen, was für Töne zustande kommen. Sie probiert ein paar Minuten herum, bevor sie aussteigen muss.

2. Daheim angekommen packt Lisa die App erneut aus. Um eine Mehrstimmigkeit zu erzeugen nimmt sie eine Geste auf und spielt diese anschließend wieder ab. Parallel dazu malt sie mit ihrem Finger eine neue Geste, um neue Töne zu generieren.

3. Im Gymnasium, in dem Harald arbeitet, steht ihm ein Touchscreenmonitor zur Verfügung. Diesen will er seinen Schülern im Musikunterricht zugänglich machen, um mit ihnen die App auszutesten. Mehrere Kinder können gleichzeit auf dem Monitor zeichnen. Infolge dessen werden Töne generiert. Wenn sich Berührungen näher kommen, kann man eine zusätzliche Variation der Töne hören.

KAPITEL 4

Implementierung

Die Anforderungen definieren den Funktionsumfang der Anwendung. In diesem Kapitel wird beschrieben, wie die Anforderungen erreicht werden. Dabei wird ein Blick auf den Zusammenhang zwischen den Bewegungen und den Tönen geworfen, das sogenannte *Mapping* und die Komponenten beziehungsweise die Architektur geworfen.

4.1 Allgemeines

Die Anwendung soll Berührungen auf einem Touchscreen verarbeiten und daraus Töne erzeugen. Die Berührungspunkte zwischen Finger und Touchscreen Monitor werden auch als Touchpoints bezeichnet. Das Ziel ist, ein elektronisches Musikinstrument zu entwickeln. Im Gegensatz zu Geräten mit TUIs soll es keine Objekte geben, die ebenfalls Einfluss auf den Ton haben, es soll ebenfalls auf Tasten verzichtet werden. Außerdem ist es beabsichtigt, dass die Anwendung keinem echten Instrument, beispielsweise Gitarre oder Klavier, gleicht. Die Rahmenbedingungen wurden dabei durch Recherche ähnlicher Projekte festgelegt. Der *MorphWiz* ist ebenfalls ein elektronisches Musikinstrument, das Töne durch Berührungen eines Touchscreen-Bildschirms produziert. Die Bewegungen entlang der x-Achse beeinflussen die Tonhöhe und richten sich dabei nach einer Tonleiter, bei der es diskrete Übergänge zwischen den Tönen gibt. Entlang der y-Achse kann die Wellenform variiert werden. Die MorphWiz Anwendung hat, wie in Abbildung 2.14 zu sehen, einige Menüelemente, um Einstellungen festzulegen. Die Anwendung in dieser Arbeit soll möglichst schlicht sein, damit sich der Benutzer beziehungsweise die Benutzerin nicht überfordert fühlt und einen ästhetischen Eindruck macht.

4.2 Mapping

Das Mapping beschreibt den Zusammenhang zwischen den Berührungen und dem Ton. Sowohl die Berührungen als auch die Töne besitzen Parameter, die variiert werden

können. Beispiele für Parameter der Berührungen sind die Position eines Touchpoints, der Geschwindigkeit, wenn sich die Position der Berührung verändert oder die Richtung der Positionsveränderung. Außerdem gibt es Parameter, die die Relation zwischen zwei oder mehreren Punkten beschreiben, das sind der Abstand zwischen diesen, die Veränderung des Abstands, ob dieser größer oder kleiner wird oder zeitliche Zusammenhänge, in welcher Reihenfolge unterschiedliche Touchpoints erkannt werden. Zu den Parametern, die einen Ton ausmachen gehören unter anderem die Tonhöhe, die Lautstärke, die Klangfarbe und die Dauer. Das Mapping beschreibt, welche und wie sich die Parameter der Touchpoints auf die Parameter des Tons auswirken. Für die Umsetzung wurden mehrere Alternativen ausgearbeitet und diskutiert. Im Zusammenhang des Mappings musste auch die Entscheidung getroffen werden, ob beim Variieren der Tonhöhe der Übergang zwischen diesen diskret oder stetig sein sollte. Das Mapping sollte außerdem den Aspekt der Kollaboration fördern. Das resultierende Mapping bezieht nur einen Teil der möglichen Parameter ein, es berücksichtigt zu Gunsten der Kollaboration die Relationen zwischen Touchpoints.

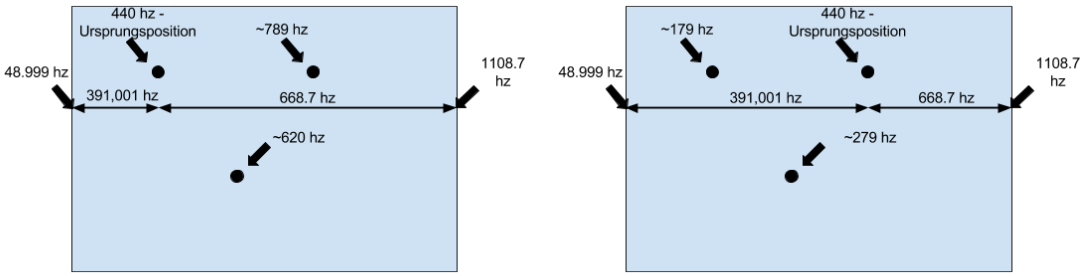


Abbildung 4.1: Beispiele des Mappings mit unterschiedlichen Ursprungspositionen

Das Frequenzspektrum von Tönen, die erzeugt werden können liegt zwischen 48.999 und 1108.7 Hertz. Die Frequenzen der Grenzen entsprechen den Tönen G_1 und cis^3. Sie wurden ausgewählt, weil sie in einem Bereich liegen, in dem der Ton im Normalfall gut wahrnehmbar ist. Bewegungen entlang der x-Achse beeinflussen die Tonhöhe, Bewegungen entlang der y-Achse haben derzeit keinen Einfluss auf den Ton. Die Frequenz des Tones, die eine Berührung erzeugt, ist davon abhängig, ob zur gleichen Zeit noch andere Touchpoints erkannt werden. Jedes Mal, wenn ein Touchpoint erkannt wird und der einzige ist, produziert dieser einen Ton in der Tonhöhe von 440 Hertz, unabhängig von seiner Position auf dem Bildschirm. Solange mindestens eine Berührung erkannt wird, ist die Ursprungsposition dieses ersten Touchpoints ausschlaggebend für die Tonhöhe anderer Berührungen. Wenn zu einem Zeitpunkt keine Berührung erkannt wird, wird die Ursprungsposition zurückgesetzt, bis sie durch eine neue Berührung wieder auf die gleiche Art gesetzt wird. Berührungen, die hinzukommen, erzeugen Töne nach dem gleichen Konzept wie der erste Touchpoint, wobei die Ursprungsposition die gleiche bleibt, wodurch die Position, an der ein Ton mit 440 Hertz erzeugt wird, ebenfalls der gleiche

bleibt.
Jeder Touchpoint ist einer Gruppe zugeteilt, wenn sich Berührungen näher als einen festgelegten Radius kommen, werden diese Gruppen zu einer zusammengefasst. Die Lautstärke ist davon abhängig, wie viele Touchpoints innerhalb einer Gruppe sind. Jede Gruppe produziert nur einen Ton, jedoch wird dieser Ton lauter, wenn mehrere Berührungen enthalten sind.

In Abbildung 4.1 sind zwei Beispiele zu sehen. Sie zeigen jeweils einen Bildschirm und drei Punkte, die sich in beiden Bildern an der gleichen Position des Bildschirms befinden. Die Frequenzen am linken und rechten Bildschirmrand sind immer gleich. Die Ursprungspositionen sind bei den Beispielen unterschiedlich, an dieser Stelle hat der erzeugte Ton eine Frequenz von 440 Hertz. Die anderen Punkte haben in den Beispielen die gleiche Position, jedoch unterscheiden sich deren Frequenzen, da die Ursprungspositionen unterschiedlich sind.

Verschwindet eine Berührung vom Touchscreen hallt der Ton abhängig von der Geschwindigkeit nach, umso langsamer die Geschwindigkeit vor dem Absetzen war umso länger hallt der Ton nach.
Die verwendete Hardware hatte ein Limit von sechs Touchpoints, die zur gleichen Zeit verarbeitet werden können, aus diesem Grund wurde auch die Implementierung für sechs Touchpoints ausgelegt, kann jedoch unproblematisch angepasst werden.

4.3 Komponenten

Das Programm wurde für die Universal Windows Platform (UWP) [22] implementiert, diese bietet eine Auswahl an Programmiersprachen, darunter sind beispielsweise Visual C#, Visual C++ oder JavaScript. Die Bibliotheken, die standardmäßig für UWP zur Verfügung gestellt werden verfügten nicht über die gesuchten Funktionen, weshalb auf andere Libraries zurückgegriffen werden musste. Um die Auswahl der Bibliotheken nicht auf die Programmiersprachen zu beschränken, die UWP zur Verfügung stellt, wurde das Verarbeiten der Eingabe und das Produzieren der Töne entkoppelt. Die Entkopplung sollte auf einer Client-Server Architektur basieren, die über UDP (User Datagram Protocol - ein verbindungsloses Netzwerkprotokoll) miteinander kommunzieren. Die Anwendung wurde in mehrere Komponenten aufgespalten. Jede dieser Komponenten ist ein eigenständiges Programm, das entweder auf einem oder auf unterschiedlichen Rechnern ausgeführt werden kann. Außerdem können Client oder Server auf diese Weise beliebig ausgetauscht werden, sofern sie über die Möglichkeit verfügen UDP zu senden beziehungsweise zu empfangen. Die Architektur des Beethoven2016 ist in Abbildung 4.2 zu sehen. Die erste Komponente bleibt wie ursprünglich geplant eine UWP App und agiert als Client. Sie ist in C# geschrieben, weshalb sie auch *CSBeethoven2016* benannt wurde. Sie nimmt die Eingaben des Touchscreens entgegen und verarbeitet diese soweit wie möglich. Anschließend verpackt sie die benötigten Daten in Packets und schickt diese via UDP an die nächste Komponente, den Server, weiter. Die zweite Komponente wurde, weil sie in Java geschrieben wurde, *JBeethoven2016* benannt. Sie

empfängt die UDP Pakete von SCBeethoven2016 und entscheidet anhand der Daten, welche Töne produziert, verändert oder gestoppt werden sollen und schickt je nachdem OSC Nachrichten an die dritte Komponente weiter. OSC steht für Open Sound Control und ist ein Netzwerkprotokoll für Sound.
Die dritte und letzte Komponente ist der *SuperCollider Server*. Er empfängt die OSC Nachrichten von JBeethoven2016 und generiert die Töne.

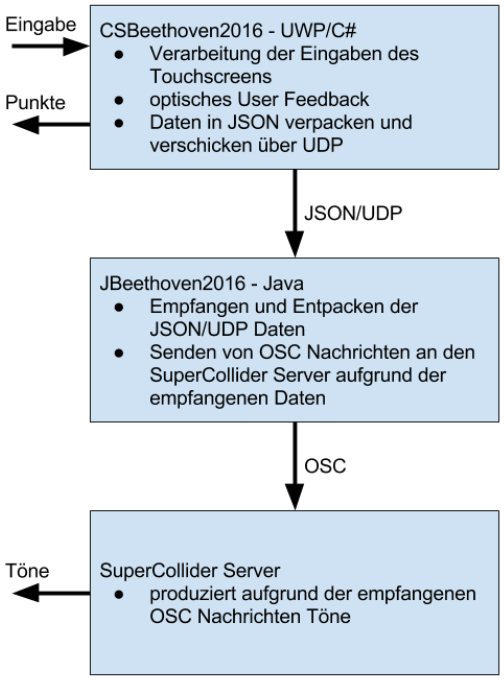


Abbildung 4.2: Architektur der Anwendung

Die Aufgabe des CSBeethoven2016 ist nicht nur das Behandeln von Touchevents und Weitersenden der Daten, sondern auch dem Benutzer optisches Feedback zu geben. Dieses wurde durch ein einfaches Zeichnen mit durchwechselnden Farben an der aktuellen Position, an der gerade eine Berührung stattfindet, erreicht. Die Farben bleiben permanent, bis das Programm neu gestartet wird. Das Malen wurde von fertigen Codestücken aus dem Internet angepasst und wiederverwendet. Die originalen waren für WPF ausgelegt und kamen von einem Artikel von codeproject.com[23] beziehungsweise wurden dort von einer Einführung zu WPF von Jaime Rodriguez vom Microsoft Blog übernommen[24]. Dort ist ein Link zum Download des Codes angeführt. Der Code wurde abgeändert, damit er für UWP geeignet ist, dafür mussten manche Bibliotheken ersetzt werden, der Code wurde außerdem geringfügig refaktorisiert. In weiterer Folge wurde die Dicke der Linien geändert und die eckigen Endpunkte wurden durch runde ersetzt, wodurch Punkte entstehen. Abbildung 4.3 zeigt einen Screenshot der Anwendung, nachdem diese eine

Zeit lang benutzt wurde. Um die Schnittstelle zwischen erster und zweiter Komponente zu vereinheitlichen wurde auf JSON zurückgegriffen, ein Datenformat, das sich gut für Datenaustausch eignet. Ein Pro-Argument ist, dass JSON weit verbreitet ist, und für jede bekanntere Programmiersprache Parser existieren, die Objekte beliebiger Formate in JSON umwandeln können und vice versa. Dadurch wird die Unabhängigkeit von Client und Server nicht maßgeblich beeinflusst.
Die UDP Packets werden in JBeethoven2016 empfangen und geparsed, dadurch können sie in Java wie gewöhnliche Objekte verwendet werden. Der verwendete SuperCollider ist selbst als Client - Server Architektur aufgebaut[25]. Der Server, *scsynth*, kann mit mehreren Clients, *sclang*, gleichzeitig über OSC kommunizieren. SuperCollider verfügt über eine eigene IDE und eine ausführliche Dokumentation. Es gibt außerdem JCollider, eine Java Library, die die Funktionalität besitzt um als Client zu agieren[26][27]. Diese JCollider Library wurde in JBeethoven2016 verwendet, um die OSC Nachrichten an den SuperCollider Server zu senden.

Abbildung 4.3: Screenshot der Anwendung bei Benutzung

KAPITEL 5

User Tests

Nach der Implementation sollte die Anwendung von Testpersonen getestet werden, um beobachten zu können, wie die Leute mit dem System interagieren, ob es einfach zu handhaben ist, was gut und was nicht so gut ankommt. Daraus sollten auch Verbesserungsvorschläge ausgeabeitet werden, um den Wünschen der potenziellen Nutzer entgegenzukommen und die Anwendung attraktiver zu gestalten, damit zukünftige Nutzer und Nutzerinnen diese gerne verwenden. Dieses Kapitel befasst sich mit der Vorbereitung des Tests bis hin zum Ablauf, der Evaluierung und möglicher Verbesserungen.

5.1 Methode

In den folgenden Unterkapiteln wird beschrieben, wie der Test durchgeführt wurde. Dabei wird auf das Setting, die Testpersonen und den Ablauf eingegangen. Im Aufbau wird die technische Ausstattung beschrieben. Die anderen Unterkapitel geben einen Überblick zu den Teilnehmern und wie und welche Informationen durch die Tests von diesen eingeholt wurden. Ein Test mit einer Person wird nachfolgend auch als eine Session bezeichnet.

5.1.1 Aufbau

Der Test fand im Open Lab des Institutes für Gestaltungs- und Wirkungsforschung am Standort Favoritenstraße der TU Wien statt. Als Eingabegerät wurde ein Touchscreen Monitor mit einer Bildschirmdiagonale von 46"verwendet. Dieser kann bis zu 6 Touchpoints gleichzeitig verarbeiten. Die Anwendung wurde auf dem gleichen Laptop ausgeführt, auf dem sie auch entwickelt wurde, einem Lenovo T430s mit Windows 10. Die Daten des Touchinputs vom Monitor wurden über ein USB Kabel zum Laptop übertragen, zusätzlich wurden die Geräte mit einem Cinch Kabel verbunden, um den Ton über die Lautsprecher des Monitors abzuspielen. Außerdem wurde ein seperater Laptop mit Linux verwendet, um die Beobachtungen und Aussagen der Testpersonen

mitzuschreiben. Es wurde darauf geachtet, dass die Teilnehmer genügend Platz haben, um sich vor dem Monitor frei zu bewegen. Der Aufbau ist in Abbildung 5.1 zu sehen. Zwischen dem ersten und letzten Test wurden keine Änderungen am Code vorgenommen.

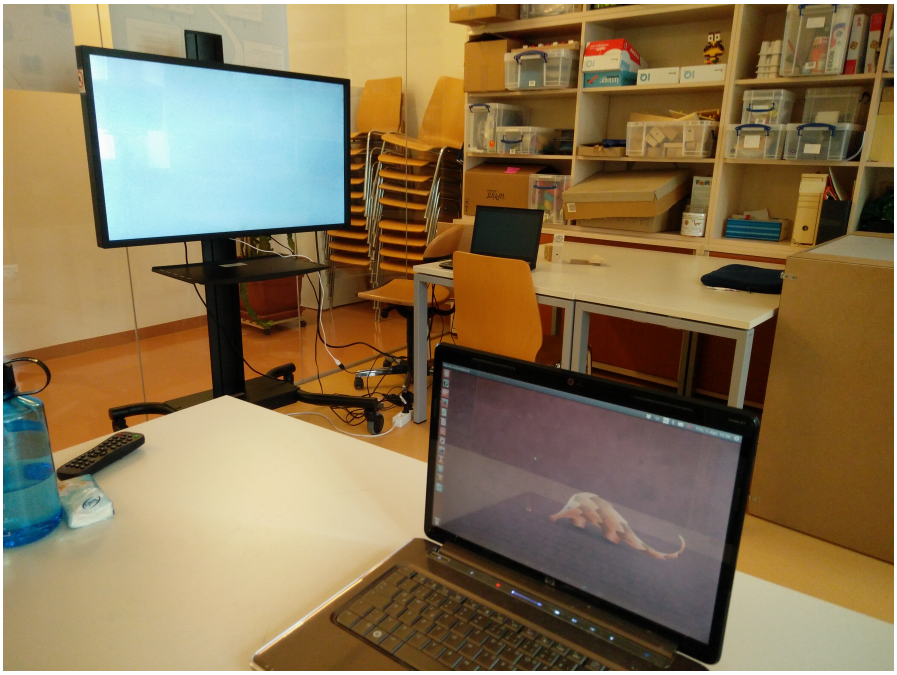


Abbildung 5.1: Setting des User Tests im Open Lab

Dadurch wurde, sofern beeinflussbar, versucht jeder Person den gleichen Ausgangspunkt hat und die Ergebnisse gut miteinander verglichen werden können.

5.1.2 Testpersonen

Der Test wurde mit 14 Personen durchgeführt, diese stammen größtenteils aus dem Bekanntenkreis. Die Personen wurden so ausgewählt, dass sowohl das Geschlecht, das Alter als auch die berufliche Tätigkeit möglichst variieren. Unter den Testern waren sieben Frauen und sieben Männer zwischen 10 und 43 Jahren. Der Altersdurchschnitt liegt insgesamt bei 26.43 Jahren, der der Frauen bei 23, der der Männer bei 29.86. Die Tätigkeiten umfassen unter anderem Wirtschaft, Psychologie, Chemie, Architektur, Informatik, Pharmazeutik, Mathematik und Musikwissenschaften.
Die meisten Testpersonen haben ein Musikinstrument gelernt, spielen dieses mittlerweile jedoch nur noch unregelmäßig oder gar nicht mehr.

5.1.3 Ablauf

Die Testpersonen wurden vor dem Test darauf hingewiesen, dass sie die Anwendung so lange ausprobieren können, wie sie wollen. Sie wurden gebeten, ihre Gedanken laut

mitzusprechen (Thinking Aloud), dadurch soll ihr Handeln leichter nachvollzogen werden können. Ihnen wurde außerdem darüber informiert, dass es sich um einen Prototypen handle, bei dem Fehler auftreten können und wenn sie während des Testens glauben, dass ein solcher aufgetreten ist, dies kommunizieren sollen. Zum Abschluss würden ihnen ein paar Fragen zu ihrer Meinung bezüglich der Anwendung gestellt werden, um herauszufinden, wo bei der Software Nachbesserungsbedarf besteht beziehungsweise Ergänzungen vorgenommen werden sollten. Die Fragen beinhalteten folgenden Punkte und wurden im Zuge eines Interviews besprochen:

- Ist der Zusammenhang zwischen Gesten und Tönen erkennbar, beziehungsweise welche Zusammenhänge waren für die Teilnehmer und Teilnehmerinnen erkennbar
- Wäre ein anderes Mapping wünschenswert, könnten sie sich Alternativen oder Ergänzungen zum jetzigen Mapping vorstellen
- Wie hat ihnen die Visualisierung gefallen, haben sie Verbesserungsvorschläge, fehlt es ihnen an optischem Feedback
- Würden sie die Anwendung wieder verwenden, hat es ihnen gefallen, hatten sie Spaß, damit zu experimentieren
- Sonstige Anmerkungen ihrerseits

Die meisten Sessions haben inklusive dem Ausprobieren und der Besprechung der Fragen zwischen 30 und 45 Minuten gedauert. Die Fragen wurden nicht immer in der gleichen Reihenfolge gestellt.

5.2 Resultate

In diesem Kapitel werden Beobachtungen und Aussagen der Testpersonen zusammengefasst, die für interessant gehalten werden. Darunter fallen Inhalte, die öfter aufgefallen sind oder im Sinne mehrerer Nutzer und Nutzerinnen sein können. Aus diesen werden dann Verbesserungsvorschläge entwickelt. Die Resultate gliedern sich in zwei Teile, der erste beschreibt Vorkommnisse während des Ausprobierens, der zweite Teil beschäftigt sich mit den Fragen, die anschließend in einem Interview besprochen wurden.

5.2.1 Thinking Aloud und Beobachtungen

Die Testpersonen wurden angehalten, ihre Gedanken während des Testens laut auszusprechen, um daraus ableiten zu können, wie intuitiv das System ist und welche Erwartungen die Benutzer und Benutzerinnen gegenüber bestimmter Aktionen haben. Im Falle dieser Anwendung können speziell die Reaktionen der Tester und Testerinnen gegenüber den Zusammenhängen zwischen Bewegungen, Tönen und der Visualisierung beobachtet werden. Es werden auch Hypothesen erwähnt, die von einzelnen Teilnehmern

oder Teilnehmerinnen angemerkt wurden, diese können bei Überlegungen für zukünftige Änderungen miteinbezogen werden.

Nach dem Starten ist die Anwendung im Vollbildmodus und der Hintergrund ist nur weiß(siehe Abbildung 5.1 hinterer Monitor). Für einige Personen war nicht von vorneherein klar, dass die Anwendung über Touchinput, also Berührungen mit dem Laptop, gesteuert wird. Sie dachten zu Beginn, dass das Programm über eine Kamera verfügt und auf ihre Bewegungen reagiert. Fast die Hälfte der Personen dachten, dass die Anwendung noch laden würde und noch nicht bereit wäre.

Auffällig war, dass alle bis auf eine Person zuerst damit angefangen haben, einfach nur mit einem Finger auf den Monitor drauf zu tippen, die Person, die die Ausnahme dargestellt hat, hat gleich zu Beginn mit zwei Fingern auf den Bildschirm getippt.

Es hat unterschiedlich lange gedauert, bis die Teilnehmer und Teilnehmerinnen anfingen, mit dem Finger zu wischen. Die Dauer, bis die Testpersonen mehr als einen Finger zum Experimentieren verwendeten, hat ebenfalls variiert.

Fünf Personen haben auch ausprobiert, wie die Anwendung auf andere Körperteile wie Handfläche oder Ellbogen reagiert.

Ebenso viele Personen haben versucht herauszufinden, wieviele Touchpoints die Anwendung gleichzeitig verarbeiten kann. Nicht alle konnten die tatsächlichen Anzahl feststellen.

Mehr als der Hälfte der Tester und Testerinnen ist aufgefallen, dass der Ton beim Ziehen vom Finger nach rechts höher und nach links tiefer wird. Etwa gleich viele Leute haben explizit erwähnt, dass der Ton bei der ersten Berührung der gleiche ist, egal wo auf dem Monitor sie hintippen. Eine Person hatte die Vermutung, dass die Tonhöhe mit der Dauer der Berührung anstatt der Bewegung zu tun hat. Manche Teilnehmer und Teilnehmerinnen haben zu Beginn geäußert, dass sie sich nicht sicher sind, ob die Tonhöhe etwas mit der Farbe zu tun hat.

Sieben Personen haben geäußert, dass Bewegungen nach oben und unten keinen Einfluss auf den Ton haben.

Zehn Leuten haben im Laufe der Session angemerkt, dass sie nicht nachvollziehen können, warum der Nachhall unterschiedliche lange dauert, obwohl sie nur einzeln immer wieder auf die gleiche Stelle tippen. Einer von ihnen war der Meinung, dass es ein Fehler sei. Zwei Personen haben ihre Hypothese geäußert, dass der Nachhall etwas mit der Position zu tun hat, mussten dies jedoch später wieder verwerfen. Drei andere Personen haben angemerkt, dass es etwas mit der Farbe zu tun haben könnte, mussten dies jedoch nach mehrmaligem Probieren auch wieder verwerfen. Auch die Hypothesen, dass ein längerer Nachhall regelmäßig nach einer bestimmten Anzahl an Berührungen auftritt oder mit der Größe der berührten Fläche zusammenhängt, konnten sich nicht bewahrheiten.

Vier Personen haben geäußert, dass die Farbe nichts mit den Tönen zu tun hat. Wenigen Leuten ist aufgefallen, dass zwei oder mehr Punkte beim Zusammenführen die gleiche

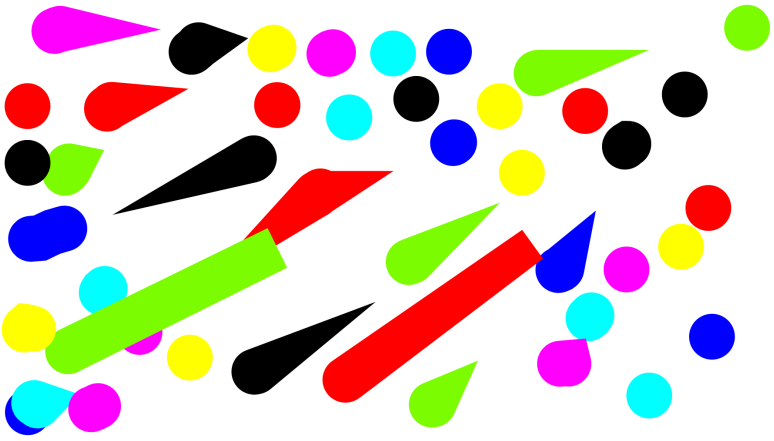


Abbildung 5.2: Screenshot der nachgestellten Spitzen beim Drauftippen

Farbe annehmen. Das Blitzen haben nur wenige Leute erwähnt. Dies ist ein Nebeneffekt, wenn es bei schnellen Bewegungen zu häufigen Farbwechseln kommt.

Wesentlich mehr Leute haben bemerkt, dass sich die Farben beim Zeichnen wiederholen.

Vier Personen ist aufgefallen, dass bei mehreren Berührungen gleichzeitig manche Farbflecken beim Malen im Vordergrund bleiben und es mehrere Ebenen gibt. Erst nach dem Absetzen und neuem Ansetzen der Berührung können auch die vorherigen Flecken übermalt werden.

Mehr als die Hälfte der Teilnehmer und Teilnehmerinnen hat sich erkundigt, ob man das Gezeichnete vom Bildschirm auch wieder löschen kann, damit sie wieder einen weißen Hintergrund bekommen. Etwa die gleiche Anzahl an Personen hat während des Tests versucht, den Bildschirm einheitlich in einer Farbe auszumalen. Dies gestaltete sich für die meisten jedoch eher schwieriger, als anfänglich vermutet, da man nicht absetzen darf und mit dem Monitor nicht aus Versehen mit anderen Fingern in Berührung kommen darf.

Zwölf Leute haben beim Testen ihre Verwirrung über Spitzen der gemalten Punkte zum Ausdruck gebracht, die manchmal auftraten, wenn sie einfach nur auf den Bildschirm tippen. Für eine Veranschaulichung siehe Abbildung 5.2. Manche der Flecken bekommen Spitzen oder werden in eine Richtung lang gezogen, andere nicht. Die Spitzen konnten von den meisten Testern beziehungsweise Testerinnen nicht rekonstruiert werden. Manche Personen haben die Ursache auf die Position oder die Farbe zurückgeführt, bis sie durch Probieren von dieser Idee wieder abkommen mussten. Eine Person hat es sich kurzzeitig so erklärt, dass die Spitze immer dorthin zeigt, wo der vorherige Ton erzeugt wurde. Eine andere Person hat die Spitzen als Fehler in der Software abgestempelt. Eine weitere Vermutung, die von einem Teilnehmer geäußert wurde, war, dass es möglicherweise etwas damit zu tun hat, wie fest man auf den Bildschirm drückt.

Zwei weitere Personen haben während des Testens gefragt, ob der Touchscreen erkennt, wie fest man draufdrückt. Eine andere hat sich zu der Stärke in Verbindung mit dem Knistern geäußert, weil sie dachte, das Knistern würde daher kommen, dass sie zu fest auf den Bildschirm drückt.

Das Problem mit dem Knistern ist bei mehreren Testsessions aufgetreten, es wurde von keinem der Tester oder Testerinnen angesprochen.

Bei drei oder vier Leuten ist das Problem aufgetreten, dass ein Ton endlos lang angedauert hat, obwohl keine Berührung mehr aktiv war. In diesen Fällen konnte der Ton nur beendet werden, indem der Server des SuperColliders, der die Töne erzeugt, neu gestartet wurde.

Ein Tester hat wie bei einem Smartphone versucht,rein und rauszuzoomen und beobachtet, ob dabei etwas passiert.

Eine Person hat während des Tests erwähnt, dass ein Tutorial hilfreich wäre.

5.2.2 Interviews

In diesem Unterkapitel werden Antworten zu den im vorhinein vorbereiteten Fragen zusammengefasst.

Visualisierung

Zu den Farben, mit denen gemalt wird, gab es unterschiedilche Meinungen. Manchen waren die Farben zu grell und sie würden sich eine andere Farbpalette wünschen, andere fanden es gut, da der Bildschirm am Ende schön bunt wäre. Es wurde mehrmals der Wunsch geäußert, dass man den Bildschirm auch wieder löschen kann. Außerdem kam dreimal die Äußerung, dass sie gerne die Farbe selbst auswählen wollen würden, statt dass sich die Farben in einer Reihenfolge immer wiederholen. Dem zehnjährigen Kind hat es gefallen, dass die Farben durcheinander kommen und man sie nicht aussuchen kann. Eine Person war der Meinung, dass die Farben ok wären, weil die Anwendung in erster Linie ohnehin dafür da wäre, dass sie Musik macht, und er hätte spontan keine andere Idee, wie man es sonst visualisieren könnte.
Viele hatten am Anfang die Vermutung, dass die Farben mit den Tönen zu tun hatten, dazu gab es unterschiedliche Hypothesen, wie der Zusammenhang sein könnte. Diese wurden teilweise schon in vorherigen Kapiteln erwähnt. Beispielsweise wurde vermutet, dass die Farbe etwas mit dem Nachhall zu tun hätte, oder dass jede Farbe einen anderen Ton erzeugen würde. In dem Zusammenhang kam auch auf, dass man die Farbe selbst aussuchen könnte, und dadurch auch eine bestimmte Tonhöhe. Die Idee, dass sich die Farbe dem Ton anpassen würde, zum Beispiel dadurch, dass sie dunkler wird, wenn der Ton tiefer wird oder heller, wenn der Ton höher wird, wurde ebenfalls mehrfach geäußert. Dies würde auch vereinfachen, eine bestimmte Melodie zu spielen.
In Verbindung mit den Gruppen kamen zwei Beiträge, zum einen, dass standardmäßig jeder Touchpoint schwarz sein könnte und erst wenn zwei oder mehr Berührungen zu einer Gruppe zusammengefasst werden, diese eine einheitliche Farbe bekommen, bis sich

die Gruppe wieder trennt. Ein anderer Teilnehmer hat gemeint, dass sich die Farben zweier Gruppen annähern könnten, wenn sie sich näher kommen, bis sie die gleiche Farbe bekommen.
Ein paar Personen haben anklingen lassen, dass sie bei dem Nachhall gerne wissen würden, welches Element noch aktiv ist, also von welchem Punkt gerade noch ein Ton kommt.
Dazu, wie man den Nachhall visualisieren könnte, wurden zwei Ideen geäußert. Bei der ersten könnte man die Spitzen nutzen, die bisher unabsichtlich auftraten und des öfteren kommentiert wurden. Umso länger der Nachhall ist, umso länger könnten diese Spitzen sein. Eine zweite Idee war, dass man konzentrische Kreise von dem Punkt ausstrahlen lässt, solange er noch einen Ton erzeugt.
Eine Aussage, die man bei Überlegungen dazu miteinbeziehen könnte war, dass man gar nicht malen würde, der Hintergrund einfärbig wäre und so lange ein Ton erklingt von der Position der Berührung nach oben und unten eine andere Farbe ausgestrahlt werden würde.
Von einer Person, die etwa 8 Jahre lang Klavier gespielt hat, kam der Beitrag, dass man eventuell anzeigen könnte, zwischen welchen Tönen man sich gerade befindet, also beispielsweise zwischen C und D, und wieviel dazwischen fehlen würde. Dies könnte man sich ähnlich wie bei einem Stimmgerät vorstellen. Dies würde ihm zum einen helfen, wenn er eine Melodie nachspielen wollen würde, als auch, wenn er etwas Neues komponieren wollen würde, weil er somit die Töne niederschreiben könnte.
Eine Person hätte anfangs gedacht, dass wenn sie über etwas bereits Gezeichnetes noch einmal mit dem Finger drüberfährt, dass der Ton davon dann noch einmal abgespielt werden würde.

Nachvollziehbarkeit des aktuellen Mappings

Das aktuelle Mapping zwischen den Berührungen und den Tönen hängt von unterschiedlichen Faktoren ab. Ausschlaggebend ist die erste Berührung, die selbst unabhängig von der Position immer einen Ton in der Tonhöhe von 440 Hertz erzeugt. Zusätzliche Berührungen erzeugen Töne abhängig von der ersten Berührung. Links von der Position an der die erste Berührung stattgefunden hat wird der Ton tiefer. Nach rechts wird er höher, umso weiter man sich zum rechten Bildschirmrand bewegt. Der Nachhall ist abhängig von der Geschwindigkeit mit der der Finger über den Monitor bewegt wird.
Die Personen wurden gebeten, ihre Erkenntnisse noch einmal zusammenzufassen, um filtern zu können, welche ihrer Hypothesen wieder verworfen wurden und welche sich bis zum Ende gehalten haben.
Den meisten Testpersonen ist aufgefallen, dass der Ton beim Drauftippen ohne einer weiteren Aktion immer der gleiche ist, und er sich erst dann ändert, wenn man den Finger während der Berührung mit dem Monitor bewegt. Nicht alle haben erkannt, dass der Ton nach rechts höher und nach links tiefer wird. Manche Personen dachten, dass sich der Ton bei Auf- und Abbewegungen, anstatt nach rechts und links, verändert. Es gab auch einen Fall, in dem gar kein Zusammenhang erkannt wurde. Eine Person hat, wie vorher schon erwähnt, gedacht, dass die Veränderung der Tonhöhe nur durch die Dauer des Kontaktes mit dem Bildschirm kommt. Die Tatsache, dass der Touchscreen maximal

sechs Berührungen gleichzeitig verarbeiten kann, ist kaum jemandem aufgefallen. Der Zusammenhang zwischen den Farben und den Gruppen ist den wenigsten aufgefallen. Dass die Lautstärke dabei verändert wird, konnte niemand erkennen. Auch andere Zusammenhänge von Gruppen untereinander und zueinander konnte niemand feststellen. Zwei Testern ist aufgefallen, dass der Daumen anders erkannt wird, wenn man mehrere Finger einer Hand auf den Monitor legt, sie konnten dies aber nicht mit dem Abstand der Berührungen in Verbindung bringen.
Gut die Hälfte der Tester hat bemerkt, dass Bewegungen nach oben und unten keinen Einfluss auf den Ton haben, sofern man den Finger nicht zusätzlich nach rechts oder links bewegt.
Von dem Nachhall waren die meisten verwirrt, niemand konnte rekonstruieren, wann der Ton lange und wann nicht so lange nachklingt.

Vorschläge zum Mapping

Ein paar Leute meinten, dass das Mapping letztendlich nicht so wichtig sei, da man sich darauf einstellen könne, sobald man gemerkt habe, wie es funktioniere.
Sechs Leute haben angemerkt, dass man den vertikalen Platz nutzen könne. Eine Rückmeldung war, dass ein Teilnehmer überfordert sei, wenn es noch eine zweite Dimension gebe.
Von drei Personen kam der Beitrag, dass die Bewegungen nach oben und unten mit der Lautstärke verknüpft werden könnten, dadurch könnte man diese nach oben und unten leiser und lauter stellen.
Ein anderer Teilnehmer hat vorgeschlagen, dass man mit rauf und runter die Oktaven ändern könnte und nach links und rechts die Tonhöhe innerhalb der jeweiligen Oktave. In diesem Zusammenhang kam bei der Testperson auch die Idee auf, dass man unterschiedliche Formen statt nur Punkte verwenden könne, und sich die Töne je nach den Formen unterscheiden könnten.
Sieben Leute haben sich zu diskreten und stetigen Übergängen zwischen Tönen und einem Mapping, bei dem die Töne fix zu bestimmten Positionen gehören, ausgesprochen. Derzeit ist die Tonhöhe bei der ersten Berührung immer die gleiche und erst durch eine Bewegung wird diese variiert, die Frequenz von Tönen, die durch neue Berührungen dazukommen sind abhängig von der ersten Berührung. Alternativ könnte ein Mapping verwendet werden, bei dem die Töne nur von der Position des Touchpunktes auf dem Monitor abhängig sind. Jede Berührung auf einer bestimmten Position würde dann immer den gleichen Ton erzeugen. Dies könnte man sich auch ähnlich einer Klaviatur vorstellen, mit der sich abhängig davon, welche Taste gedrückt wird, ebenfalls nur bestimmte Töne erzeugen lassen. Es können nicht Töne mit beliebigen Frequenzen erzeugt werden, da beispielsweise zwischen C und Cis keine weiteren Tasten sind.. Einige waren der Meinung, dass es einfacher sei, allerdings auch, dass es dadurch eventuell schneller langweilig werden würde und man weniger komplexe Sachen spielen könne. Dafür müsste man nicht einen Finger immer auf dem Bildschirm halten oder ziehen, um unterschiedliche Töne zu erzeugen. Diskrete Töne wären ebenfalls einfacher, würden die Möglichkeiten jedoch auch

wieder einschränken. Zwei Personen haben sich jedoch auch dazu geäußert, dass manche Tone recht nervig klingen, die beim jetzigen stetigen Mapping vorkommen können. Die Meinungen, ob sie einen stetigen oder einen diskreten Übergang zwischen den Tönen bevorzugen würden, war nicht eindeutig. Der Großteil der Tester und Testerinnen hat sich gar nicht dazu geäußert.
Drei Personen haben angemerkt, dass sie es gut fänden, wenn man einen Rythmus beziehungsweise Beat machen könnte. Eine davon hat auch die Möglichkeit angesprochen, Töne aufzunehmen und in einem Loop wieder abzuspielen. Bei sonstigen Anmerkungen kam dieser Vorschlag auch noch von einer anderen Person.
Ein Teilnehmer hat angemerkt, dass er sich vorstellen könnte die Geschwindigkeit oder die Dauer, wie lange man drückt miteinzubeziehen. Er hat jedoch auch gemeint, dass es eine Katastrophe sei, damit etwas Sinnvolles zu spielen.
Zwei Personen haben noch komplett andere Mappings vorgeschlagen. Eines davon war, dass in der Mitte immer der gleiche Ton wäre, und die Tonhöhe von innen nach außen variieren könnte. Der andere Vorschlag war, dass in der Anwendung Buttons verteilt sind, die mit diskreten Tönen, zum Beispiel A oder E, belegt sind, die beim Drauftippen oder Drüberfahren ertönen würden. Die Buttons könnte man verschieben, um sie beliebig am Monitor zu platzieren. Nicht jeder Fleck des Bildschirms müsste mit diesen Buttons belegt sein, es kann auch Freiräume geben, in denen bei einer Berührung kein Ton erzeugt wird.

Wiederverwendung

Der Großteil der Tester hat angegeben, dass sie die Anwendung so, wie sie jetzt ist, nicht wiederverwenden würde. Eine Person würde das Programm benutzen, aber nicht um ernsthaft Musik zu machen. Vor allem das Experimentieren, um die Zusammenhänge zwischen ihren Aktionen und den erzeugten Tönen herauszufinden, hat 11 Testpersonen gut gefallen. Zwei Personen teilen sich die Meinung, dass man jedoch schnell herausgefunden hätte, was alles möglich ist und es dann langweilig werde, weil die Möglichkeiten schnell erschöpft seien. Zwei Teilnehmer gaben an, dass sie nicht der Typ für solche Anwendungen sind, und sie es deswegen nicht öfters verwenden würden.
Drei Personen haben sich dazu geäußert, dass sie es interessanter finden würden, wenn man sinnvolle Melodien spielen könnte, was bei dem derzeitigen Mapping jedoch schwierig sei. Für diesen Zweck wäre es womöglich auch sinngemäßer nur Töne aus einer Tonleiter zu verwenden, anstatt der derzeitigen Implementierung, bei der Töne mit Frequenzen zwischen 48.999 und 1108.7 Hertz erzeugt werden können.
Ein Tester, der sich in seiner Freizeit gerne mit Anwendungen dieser Art beschäftigt, hat geäußert, dass er die Anwendung öfter verwenden würde, wenn es weniger nervige Töne machen würde.
Eine Person würde sich eine andere Visualisierung wünschen, damit es ihr auch leichter fallen würde, bestimmte Melodien zu spielen.

Sonstige Anmerkungen

Die 10-jährige Testperson hat ausgesagt, dass sie sich hauptsächlich mit dem Malen und weniger mit den Tönen beschäftigt habe.
Auch andere Personen haben anklingen lassen, dass ihnen das Malen mindestens genauso Spaß mache, wie das Erzeugen der Töne dadurch.
Eine Person, die solche Anwendungen privat nicht nutzt, hat gemeint, sie könne es sich an öffentlichen Plätzen als Spielerei im Vorbeigehen gut vorstellen.
Einmal wurde erwähnt, dass man eventuell die letzten 10 Minuten automatisch speichern können sollte, um sie wiederzuverwenden.

5.3 Zusammenfassung

Die Anwendung wurde von 14 Leuten im Alter zwischen 10 und 43 Jahren getestet, darunter waren 7 Männer und 7 Frauen. Die Testpersonen konnten sich für das Testen so viel Zeit nehmen, wie sie wollten, danach wurden sie gebeten, ein paar Fragen zu beantworten. Durch die Thinking Aloud Methode, bei der die Testpersonen angehalten werden, ihre Gedanken während der Benutzung der Software laut auszusprechen und durch die anschließenden Interviews konnten Erkenntnisse daraus gezogen werden, was an dem Programm verbessert werden kann.
Der wichtigste Punkt ist, dass der Nachhall bei kurzem Antippen nicht so lange anhalten sollte, dies war für viele Leute verwirrend oder nervig. Dies dürfte auf eine Unachtsamkeit im Code zurückzuführen sein, da die berechnete Geschwindigkeit nicht so niedrig sein sollte, um so einen langen Nachhall zu erzeugen.
Die zweitwichtigste Sache ist, dass die Spitzen, die manchmal beim Malen entstanden sind, beim Großteil der Leute Verwirrung ausgelöst haben. Auch dies ist ein unbeabsichtigter Nebeneffekt, dessen Ursache möglicherweise im verwendeten Codestück aus dem Internet liegt.
Das Knistern, das selten von den Leuten selbst angesprochen wurde, obwohl es bei einigen vorkam, könnte unterschiedliche Ursachen haben. Eine wäre, dass die Performance des Laptops, auf dem die Anwendung getestet wurde, nicht ausreichend gut war. Eine andere Ursache könnte sein, dass der SuperCollider beziehungsweise die verwendete JCollider Library effizienter genutzt werden könnte, was jedoch aufgrund von wenigen Code Samples schwer auszumachen ist.
Das Blitzen der Farben ist ebenfalls bei manchen Leuten aufgetreten und wurde selten erwähnt, dies ist nicht direkt ein Fehler, da Gruppen, wenn sie zusammengelegt werden, eine andere Farbe annehmen. Um das Blitzen zu reduzieren, könnte man eventuell darauf achten, dass Gruppen sich nicht zu oft in kurzer Zeit trennen und wieder zusammengefügt werden dürfen.
Das Feedback zu der Visualisierung und den Farben war zweigeteilt. Ein großer Teil der Tester beziehungsweise Testerinnen hatte im Laufe einer Testsession die Vermutung, die Farben würden in einer Weise mit den Tönen zusammenhängen. Dies sollte für die Zukunft in Erwägung gezogen werden. Ein guter Ansatz wäre die Farbpalette zu ändern,

dabei könnte auch der paarmal aufgetretene Vorschlag eingebaut werden, die Farben der Tonhöhe anzupassen, wenn sich diese verändert. Um ausdrücklicher auf die Gruppen hinzudeuten, könnte auch der Ansatz verwirklicht werden, dass sich die Farben angleichen, wenn sich zwei Gruppen annähern. Optisch hat die Anwendung noch Potenzial, um für Benutzer und Benutzerinnen attraktiver zu werden. Die Änderungen könnt man auch so gestalten, dass mehr Fokus auf den Ton anstatt das Malen gelegt wird. Aufgrund des vielfachen Wunsches, sollte auch die Möglichkeit vorhanden sein, das Gemalte wieder zu löschen. Obwohl der Vorschlag nie in dieser Form aufkam, könnte man auch damit arbeiten, die Linien ebenso wie den Ton wegfaden zu lassen, sprich ihn zunehmend verblassen zu lassen, bis er gar nicht mehr vorhanden ist.
Einige Testpersonen hatten Schwierigkeiten zu erkennen, dass die Anwendung via Touchsceen gesteuert wird, manche haben vermutet, es gäbe eine Kamera, die auf die Bewegungen der Benutzer reagieren würde. Viele hatten auch vermutet, dass die Anwendung noch laden würde, da sie nur einen weißen Bildschirm vor sich hatten. Diese Beobachtungen zeigen, dass die Anwendung Aufforderungscharakter benötigt, der die Leute dazu animiert, den Bildschirm zu berühren und der deutlich macht, dass die Anwendung fertig geladen ist.
Der Endloston, der manchmal aufgetreten ist, ist definitiv unbeabsichtigt und sollte beseitigt werden. Von diesem wurde vermutet, dass er bereits korrigiert wäre.
Was das Mapping der Gesten auf Töne betrifft, gab es mehrere Inputs, die verwirklicht werden könnten. Es wurde zum Beispiel öfter darauf hingewiesen, dass man den vertikalen Platz nutzen könnte. Bezüglich dessen, ob Töne fix zu bestimmten Positionen am Bildschirm gehören sollen oder ob sie relativ zum Startpunkt sein sollen, gab es keine Einigung. Eine Person hat angemerkt, dass man eventuell ein zweites Mapping zur Auswahl stellen könnte.
Ob ein Tutorial benötigt wird, war ebenfalls nicht eindeutig, die Mehrheit hat geäußert, dass sie Spaß am Experimentieren hatten und gerne selbst die Zusammenhänge herausgefunden haben. Ein paar Leute hätten gerne eine Anleitung gehabt, wie welche Töne erzeugt werden können. Ein guter Kompromiss wäre daher wahrscheinlich, ein optionales Tutorial zu machen, um den Leuten zu ermöglichen nachzulesen. Dadurch könnten sie auch im Anschluss an das Experimentieren auch die Sachen nutzen, auf die sie zuvor nicht selbst draufgekommen sind.

KAPITEL 6

Diskussion

Im Folgenden werden Inhalte genannt, die in weiterführender Arbeit umgesetzt werden können. Bei den User Tests sind einige Punkte ans Licht gekommen, bei denen Verbesserungen oder Erweiterungen vorgenommen werden können, dazu kommen außerdem Inhalte, die nicht von den Testern und Testerinnen angesprochen wurden, aber dennoch Potenzial für Verbesserungen haben.
In manchen Testsessions ist das Problem aufgetreten, das ein Ton nicht mehr gestoppt hat. Bei Tests im Zuge der Implementierung sind diese Endlostöne des öfteren aufgetreten, jedoch wurde so lange nach der Ursache gesucht und daran gearbeitet, bis diese nicht mehr aufgetreten sind, weshalb davon ausgegangen wurde, dass dieser Fehler beseitigt wurde. Aufgrund des erneuten Auftretens sollte dieser Fehler gesucht und ausgebessert werden, da er störend ist, wenn er auftritt. Vermutungen dazu sind, dass das Entfernen von Touchpoints in manchen Fällen nicht korrekt erkannt wird oder bei der Übertragung der UDP Packages Fehler auftreten. Je nachdem, welche Daten übertragen werden müssen, könnte man eventuell auf JSON verzichten und die Daten binär schicken, dies könnte zeitlich effizienter sein, jedoch auch unstrukturierter. Die Klasse *TouchData* beinhaltet Informationen zu einzelnen Touchpoints. Sie ist sowohl in C# als auch in Java implementiert, wird jedoch derzeit nur in der C# Komponente verwendet. Die Klasse wurde nicht gelöscht, damit sie bei Bedarf auch im Java Teil verwendet werden kann, jedoch würden die UDP Pakete widerum größer werden, wenn zusätzlich die Daten der einzelnen TouchData Objekte mitgeschickt werden würden. Der Vorteil, dass die Komponenten auf unterschiedlichen Rechnern ausgeführt werden können und durch das Schicken über UDP entkoppelt wurden wird derzeit nicht verwendet, könnte jedoch noch für eine Vielzahl von Zwecken ausgenutzt werden. Komponenten könnten ausgetauscht, hinzugefügt oder kombiniert werden. Dadurch können die Teile auch für andere Absichten wiederverwendet werden. Zwischen den jetzigen Komponenten könnten weitere Komponenten eingefügt werden, beispielsweise um die Eingaben von mehreren Geräten gleichzeitig zu verarbeiten und zu kombinieren, oder um die Interaktion trotz geographischen Entfernungen zwischen

Benutzern und Benutzerinnen zu fördern. Geographische Entfernungen können durch das Benutzen von UDP bereits jetzt überwunden werden, jedoch nur in eine Richtung. Der Nachteil des Komponentenaufbaus ist, dass das Ausführen des Programmes zur jetzigen Zeit umständlich ist, da der SuperCollider Server, die Java Applikation und die C# Applikation gestartet werden müssen. Dies kann möglicherweise durch das Schreiben eines Skriptes umgangen werden, das alle benötigten Anwendungen startet. Der Versuch, die SuperCollider Server direkt mit der JCollider Library zu starten ist gescheitert.
Die Klasse *TouchDataGroup* enthält Informationen über Gruppen, die aus TouchData Objekten berechnet wird. TouchDataGroup Objekte im C# Teil speichern derzeit auch Variablen, die keinen Einfluss auf das Mapping zwischen den Gesten und den Tönen haben. Darunter ist beispielsweise *size*, welche die Größe einer TouchDataGroup repräsentiert. Sie wird durch den Abstand vom arithmetischen Mittel einer Gruppe bis zum Touchpoint, der am weitesten entfernt ist, berechnet und kann zukünftig in das Mapping integriert werden. Außerdem kann die Klasse um weitere Parameter erweitert werden, die sich auf den Ton auswirken können. Die Variable *fingerCount*, also die Anzahl der Finger innerhalb einer Gruppe, wird derzeit verwendet, um die Lautstärke einzelner TouchDataGroups zu regulieren. Wie sich in den User Tests jedoch gezeigt hat, hat sie scheinbar zu wenig Auswirkung, da der Unterschied der Lautstärke zwischen einem und mehreren Fingern niemandem aufgefallen ist. Wenn Finger zu einer Gruppe hinzukommen, kann es außerdem passieren, dass sich die Tonhöhe verändert, da die Position für die Berechnung der Tonhöhe ebenfalls aus dem arithmetischen Mittel der betroffenen TouchData Objekte berechnet wird. Durch das Hinzukommen mehrere Touchpoints kann sich das arithmetische Mittel und somit auch die Tonhöhe verändern. Bei der Recherche über den SuperCollider und den JCollider hat es den Eindruck gemacht, dass die Lautstärke nur für jeden Server reguliert werden kann, anstatt auch für individuelle Gruppen. Da beim Mapping die Lautstärke abhängig von der Anzahl der Finger in einer TouchDataGroup ist, wurde für die maximale Anzahl an TouchDataGroup Objekte jeweils ein eigener SuperCollider Server gestartet. Die verwendete Hardware kann maximal 6 Touchpoints gleichzeitig verarbeiten, deshalb werden auch 6 SuperCollider Server gestartet. Diese Server können über unterschiedliche Ports angesteuert werden. Eine Person, die schon mit dem SuperCollider, aber nicht mit der Java Library JCollider zu tun hatte, hat angedeutet, dass durch das Verwenden mehrerer SuperCollider Server gleichzeitig Probleme auftreten könnten. Die Person war der Meinung, dass es schon möglich sein müsste die Lautstärke für einzelne Gruppen zu ändern anstatt für alle Töne, die von diesem erzeugt werden. Durch das Suchen und Umstellen auf eine Lösung, die nur einen Server statt sechs Server verwendet könnte man sich in Zukunft womöglich Probleme ersparen.
Wenn Berührungen den Kontakt zum Touchscreen Monitor verlieren, wird ein Nachhall erzeugt. Dieser Nachhall ist abhängig von der Geschwindigkeit der Bewegung des Fingers sein, umso langsamer die Bewegung, umso länger sollte der Nachhall andauern. In Fällen, wo der Bildschirm aber nur kurz berührt wird, kommen lange Nachhalle zustande. Dies passiert genau dann, wenn der Finger nur um einen sehr kurzen Abstand bewegt wird und der Kontakt anschließend wieder beendet wird. Dies hat bei den Testpersonen für große Verwirrung gesorgt, weshalb sie viel Zeit damit verbracht haben zu ergründen,

warum dies so ist. Wenn im Zuge einer weiterführenden Arbeit das Problem beseitigt wird und anschließend erneut User Tests durchgeführt werden, können womöglich andere Informationen daraus erhalten werden, da sich die Testpersonen auf andere Faktoren konzentrieren können.
Die Optik der Anwendung wurde zugunsten der Funktionalität stark vernachlässigt. Aus diesem Grund wurden fertige Codestücke verwendet, um die Gestaltung minimal auf provisorische Weise zu implementieren. Am Ende fehlte die Zeit , um diese zu überarbeiten. Die Anforderungen, die die Optik betroffen haben waren, dass sie einfach ist, um den Benutzer nicht abzulenken und ästhetisch zu wirken. Für die Darstellung gab es Kritik und Verbesserungsvorschläge, diese können entweder eingearbeitet werden oder es kann eine komplett neue visuelle Repräsentation erarbeitet werden, die durch Mockups und Befragungen zukünftiger Benutzer und Benutzerinnen evaluiert werden kann, bevor eine Gestaltung umgesetzt wird. Manche Vorschläge können bei einer neuen Darstellung miteinbezogen werden.
Bei den User Tests waren die Testpersonen neben dem Nachhall auch auf die Spitzen der Farpunkte fokusiert. Diese erscheinen in manchen Fällen ungewollt, bei den Tests wurde jedoch nach Gründen für diese Spitzen gesucht. Dieses Problem sollte ebenfalls beseitigt werden, damit die Testpersonen sich mehr auf andere Aspekte konzentrieren können. Dieses Problem und Vorschläge zur Darstellung wurden bereits im Kapitel User Tests erwähnt. Manche Testpersonen haben erwähnt, dass sie beim Nachhall gerne wissen wollen würden, von welcher Berührung dieser Ton noch kommt, eine Anregung zur Visualisierung davon war durch konzentrische Kreise. Außerdem könnten die Farben wie vorgeschlagen mit der Tonhöhe in Verbindung gebracht werden, so, dass sich die Farbe mitverändert, wenn die Tonhöhe verändert wird. Falls die aktuelle Darstellung beibehalten wird, wäre es außerdem wünschenswert, das Löschen von dem Gezeichneten zu ermöglichen, weil es des öfteren bei den User Tests angesprochen wurde. TouchData Objekte gehören zu TouchDataGroups, um die Zugehörigkeit eines Touchpoints zu seiner Gruppe zuverdeutlichen, erhalten alle Touchpoints in der gleichen Gruppe die gleiche Farbe. Wenn Touchpoints ihre derzeitige Gruppe verlassen oder zu einer anderen Gruppe hinzukommen verändern sich deren Farben passend zur aktuellen Gruppe. Wenn dies oft innerhalb von kurzer Zeit passiert, kann es wie ein „Blitzen“ wirken. Eine Vorkehrung gegen dieses Blitzen könnte beispielsweise mithilfe eines Thresholds getroffen werden. Mit diesem könnte die Anzahl limitiert werden, wie oft Touchpoints innerhalb einer bestimmten Zeit die Gruppe wechseln.
Die Anwendung ist so aufgebaut, dass das aktuelle Mapping leicht auszutauschen ist. Die Klasse *BeethovenSCMapping* in Java beinhaltet derzeit die Implementierung des Mappings und steuert somit anhand der Variablen einer TouchDataGroup die Parameter des zugehörigen Tons. Für eine bessere Struktur kann ein Interface hinzufügt werden, das von alternativen, zusätzlichen Mapping Klassen implementiert wird.
Beim derzeitigen Mapping beziehungsweise bei neuen Mappings können auch Bewegungen entlang der y-Achse miteinbezogen werden. Beispielsweise wurde bei User Tests mehrmals genannt, dass man die Lautstärke damit variieren könnte. Die Lautstärke wird im aktuellen Mapping durch die Anzahl der TouchPoints in einer Gruppe bestimmt und

hat, wie schon erwähnt, derzeit nur eine geringe Auswirkung.
Oftmals wurde kritisiert, dass manche Töne unangenehm klingen. Um die Anwendung attraktiver für Benutzer und Benutzerinnen zu gestalten sollten die erzeugten Töne auf jeden Fall angenehm klingen. Dies könnte man unter Umständen dadurch erreichen, dass man von stetigen Übergängen zwischen den Tönen auf diskrete Übergänge wechselt, und sich auf eine bestimmte Tonleiter festlegt. Wenn die stetigen Übergängen beibehalten werden sollen, könnte man Obertöne mit höherer Frequenz als dem Grundton hinzufügen. Außerdem könnte man auch mit den Wellenformen experimentieren, um den Klang der Töne zu verändern.
Während einer User Session hat sich eine Person außerdem danach erkundigt, ob es möglich wäre, einen Rythmus mit Schlagzeug oder Ähnlichem zu erzeugen, beziehungsweise abzuspielen. Die Möglichkeit einen Rythmus zu erzeugen und diesen im Hintergrund in einer Endlosschleife abzuspielen, also *loopen* zu lassen, könnte für zukünftige Benutzer und Benutzerinnen interessant sein, weshalb diese Erweiterung in Erwägung gezogen werden sollte.
Außerdem war von vornherein geplant, die erzeugten Töne aufnehmen und abspielen zu können. Dies ist auch in den Anforderungen aufgeführt, konnte jedoch aus zeitlichen Gründen nicht mehr umgesetzt werden. Die Möglichkeit, Aufnahmen selbst zu erzeugen und diese im Hintergrund loopen zu lassen, könnte die Anwendung ansprechender machen, da dadurch musikalisch mehr Möglichkeiten entstehen.
Besonders bei vielen Touchpoints, die gleichzeitig verarbeitet werden mussten trat öfter ein Knacksen auf. Im Gegensatz zu anderen Kritikpunkten, auf die eine oder mehrere Testpersonen aufmerksam gemacht haben, wurde das Knacksen, das manchmal nicht zu überhören war, von niemandem angesprochen. Eine Vermutung dazu war, dass der Rechner, auf dem die Anwendung entwickelt und getestet wurde, nicht mit der besten Hardware ausgestattet ist und der Laptop womöglich mit vielen Touchpoints überfordert ist. Um diese Vermutung zu testen, wurde die Anwendung auf einem Rechner getestet, der eine bessere Performance hat, wodurch die Bestätigung bestätigt werden konnte, da deutlich weniger Knacksen auftrat. Aus diesem Grund würde es sich auch empfehlen zu versuchen, die Anwendung effizienter zu gestalten.

KAPITEL 7

Zusammenfassung

In dieser Arbeit wurde die Entwicklung des Beethoven2016 beschrieben, ein elekronisches Musikinstrument, das mithilfe eines Touchscreens gespielt wird. Die Anforderungen der Anwendung wurden durch das Recherchieren von Referenzprojekten definiert. Dabei gab es eine Anwendung, die der Idee des direkten Mappings von Berührung und Ton ähnelte, die jedoch nicht für mehr Personen ausgelegt war. Der Aspekt, die Anwendung für mehrere Personen auszulegen, wurde durch den großen Touchscreenmonitor, der zur Verfügung stand, motiviert. Realisiert wurde dies durch die Wahl eines geeigneten Mappings. Im aktuellen Mapping steht nicht jeder Touchpoint für sich sondern steht in Relation zu anderen Touchpoints. Die Anwendung wurde aus drei Komponenten aufgebaut, die unabhängig voneinander sind. Dadurch können sie gegen andere Komponenten ausgetauscht werden, außerdem können sie, dadurch, dass sie über das Netzwerk kommunizieren können, auch auf unterschiedlichen Rechnern ausgeführt werden. Die erste Komponente verarbeitet die Berührungen und schickt die Daten an die zweite Komponente weiter. Diese verwendet die JCollider Library, um mit dem SuperCollider Server zu kommunizieren, die dritte Komponente, die die Töne produziert. Bei den User Tests wurden einige Kritikpunkte dargebracht, die teilweise mit bekannten Problemen zu tun hatten und teilweise neue Erkenntnisse ans Licht brachten. Gedanken, welche Erweiterungen und Verbesserungen an der Anwendung vorgenommen werden können, werden im Kapitel Diskussion festgehalten. Das Fazit dieser Arbeit: die Implementierung bietet eine grundlegende Funktionalität, die einen Teil der Anforderungen erfüllt. Durch den Aufbau aus Komponenten können die Teile wiederverwendet oder ersetzt werden. Durch das Feedback, das bei den User Tests eingeholt wurde, gibt es einige Ansatzpunkte, an denen gearbeitet werden kann, um die Anwendung zu verbessern und zu erweitern.

Abbildungsverzeichnis

2.1 Konzeptionelle Darstellung von GUIs und TUIs[1] 4
2.2 Komponenten des reacTables[2] 4
2.3 Personen in Interaktion mit dem reacTable[4] 5
2.4 Interaktion mit mixiTUI[5] 6
2.5 Konzeptionelle Veranschaulichung der Anwendung für kollaborative Musik von Klügel u. a.[6] 7
2.6 Bricktable mit der Weather Report Anwendung[7] 8
2.7 Bricktable mit der Roots Anwendung[8] 9
2.8 Konzeptionelles Interface der Magic Fiddle[9] 10
2.9 Zwei Benutzer mit der Magic Fiddle[9] 11
2.10 Screenshots von Musyc[11] 12
2.11 Screenshot von Pyxis Minor[13] 13
2.12 Interaktion mit TC-11[15] 13
2.13 Screenshots von TC-11. Links die Einstellungen von Synth Objects, mitte und rechts die Anwendung in Aktion[14] 14
2.14 Screenshot von MorphWiz mit den Default Layout Einstellungen[18] 15
2.15 Einstellungen für Start- und Endwelle in MorphWiz[18] 15

3.1 Persona - Lisa Klee[19] 18
3.2 Persona - Harald Bauer[20] 19
3.3 Persona - Michael Roh[21] 20

4.1 Beispiele des Mappings mit unterschiedlichen Ursprungspositionen 24
4.2 Architektur der Anwendung 26
4.3 Screenshot der Anwendung bei Benutzung 27

5.1 Setting des User Tests im Open Lab 30
5.2 Screenshot der nachgestellten Spitzen beim Drauftippen 33

Literaturverzeichnis

[1] Hiroshi Ishii. Tangible bits: Beyond pixels. In *Proceedings of the 2Nd International Conference on Tangible and Embedded Interaction*, TEI '08, pages xv–xxv, New York, NY, USA, 2008. ACM.

[2] Sergi Jordà, M. Kaltenbrunner, G. Geiger, and R. Bencina. The reactable*. In *Proc.International Computer Music Conference*, 2005.

[3] Sergi Jordà, Günter Geiger, Marcos Alonso, and Martin Kaltenbrunner. The reactable: exploring the synergy between live music performance and tabletop tangible interfaces. In *Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 139–146. ACM, 2007.

[4] Reactable Foto. `https://www.flickr.com/photos/danielwilliams/539568298/in/photostream/`. [Online; letzter Zugriff: 30.05.2016].

[5] Esben Warming Pedersen and Kasper Hornbæk. mixitui: a tangible sequencer for electronic live performances. In Nicolas Villar, Shahram Izadi, Mike Fraser, and Steve Benford, editors, *Tangible and Embedded Interaction*, pages 223–230. ACM, 2009.

[6] Niklas Klügel, Marc R. Frieß, Georg Groh, and Florian Echtler. An approach to collaborative music composition. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 32–35, Oslo, Norway, 2011.

[7] Jordan Hochenbaum and Owen Vallis. Bricktable: A musical tangible multi-touch interface. In *Proceedings of Berlin Open Conference '09. Berlin, Germany*, 2009.

[8] Jordan Hochenbaum, Owen Vallis, Dimitri Diakopoulos, Jim Murphy, and Ajay Kapur. Designing expressive musical interfaces for tabletop surfaces. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 315–318, Sydney, Australia, 2010.

[9] Ge Wang, Jieun Oh, and Tom Lieber. Designing for the ipad : Magic fiddle. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 197–202, Oslo, Norway, 2011.

[10] Musyc. http://fingerlab.net/portfolio/musyc, 2015. [Online; letzter Zugriff: 16.12.2015].

[11] Musyc - iTunes store. https://itunes.apple.com/us/app/musyc-pro/id756468960?mt=8&ign-mpt=uo%3D4. [Online; letzter Zugriff: 30.05.2016].

[12] Timothy J. Barraclough, Dale A. Carnegie, and Ajay Kapur. Musical instrument design process for mobile technology. In Edgar Berdahl and Jesse Allison, editors, *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 289–292, Baton Rouge, Louisiana, USA, May 31 – June 3 2015. Louisiana State University.

[13] Pyxis Minor. http://timothy-j.com/pyxis, 2014. [Online; letzter Zugriff: 16.12.2015].

[14] Bitshapesoftware TC-11. http://www.bitshapesoftware.com/instruments/tc-11/, 2010. [Online; letzter Zugriff: 12.12.2015].

[15] Kevin Schlei. Tc-11: A programmable multi-touch synthesizer for the ipad. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Ann Arbor, Michigan, 2012. University of Michigan.

[16] Bitshapesoftware TC-11. http://www.bitshapesoftware.com/instruments/tc-11/tc-11-user-guide-2.0.pdf, 2010. [Online; letzter Zugriff: 26.12.2015].

[17] Charles Martin, Henry Gardner, and Ben Swift. Metatravels and metalonsdale: ipad apps for percussive improvisation. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*, pages 547–550. ACM, 2014.

[18] Morphwiz. http://www.wizdommusic.com/products/morphwiz.html, 2014. [Online; letzter Zugriff: 17.12.2015].

[19] bewusstkaufen.at/blog. http://blog.bewusstkaufen.at/%C3%BCber\discretionary{-}{}{}diesen\discretionary{-}{}{}blog/die\discretionary{-}{}{}autorinnen/stefanie\discretionary{-}{}{}stolitzka.html. [Online; letzter Zugriff: 09.05.2016].

[20] Potsdam Institut für Klimaforschung. https://www.pik\discretionary{-}{}{}potsdam.de/members/stefan/rahmstorf. [Online; letzter Zugriff: 09.05.2016].

[21] BM Manfred Gratz GmbH Homepage. http://www.baumeister\discretionary{-}{}{}gratz.com/team. [Online; letzter Zugriff: 09.05.2016].

[22] Universal Windows Platform (UWP)-Apps. https://msdn.microsoft.com/de-de/library/windows/apps/dn958439.aspx, 2016. [Online; letzter Zugriff: 26.04.2016].

[23] Ashwin Singh. iPaint - Codeproject.com. http://www.codeproject.com/Articles/630399/iPaint, 2013. [Online; letzter Zugriff: 26.04.2016].

[24] Jaime Rodriguez. WPF Paint Multitouch. https://blogs.msdn.microsoft.com/jaimer/2009/11/04/introduction-to-wpf-4-multitouch/, 2009. [Online; letzter Zugriff: 26.04.2016].

[25] SuperCollider Client vs. Server. http://doc.sccode.org/Guides/ClientVsServer.html, 2016. [Online; letzter Zugriff: 26.04.2016].

[26] JCollider. http://www.sciss.de/jcollider/, 2009. [Online; letzter Zugriff: 26.04.2016].

[27] JCollider GitHub. https://github.com/Sciss/JCollider, 2016. [Online; letzter Zugriff: 26.04.2016].