Bachelorarbeit

Trumpet-Simulator
Eine virtuelle Trompete für Mobile Devices

Im Rahmen des
Computermusik und Multimedia Seminars

durchgeführt am
Institut für Elektronische Musik und Akustik
an der
Universität für Musik und darstellende Kunst, Graz, Österreich

von
Jonas Martin Helm
Matr.-Nr. 1173073

Betreuer
Zmölnig, IOhannes, DI

Graz, 12.09.2015

Abstract

Diese Bachelorarbeit befasst sich mit der Entwicklung einer interaktiven *„Mobile Music Application“* (später nur noch App genannt) für Smartphones, Tablets etc., die die Handhabung, Funktionsweise und den Klang einer Trompete auf ein solches mobiles Gerät überträgt. Es werden dabei verschiedene technische Fähigkeiten der Geräte ausgenutzt, um eine intuitive und realitätsnahe Version einer Trompete im Digitalen zu schaffen und das Gerät somit selbst in ein Instrument zu verwandeln. Die App hat den Namen *„Trumpet-Simulator“* und wird mithilfe der visuell- und datenstromorientierten „open source“ Programmiersprache *Pure Data* [1] auf dem Computer programmiert und kann per App *PdDroidParty* [2] auf Android Betriebssystemen ausgeführt werden. Hierbei war einer der wichtigsten Ansätze, das Starten und Beenden der Töne durch das Blasen ins Mikrophon zu realisieren. Dies wurde bei ähnlichen, getesteten Apps, meist durch ein Drücken veranlasst, was das Spielen auf ihnen sehr unnatürlich machte. Durch eine weitere Verarbeitung des Mikrofonsignals kommt hier außerdem durch die Variation der (Laut-)Stärke des Blasens eine Dynamik zustande, was in dieser Form in keiner anderen Trompeten-App möglich war. Obwohl bei den meisten getesteten Apps anders gelöst, waren die bei der Trompete üblichen drei Ventile in Form von Touch-Feldern auf dem Display einfach umzusetzen und übernehmen eine identische Funktion wie beim Original. Um ein haptisches Feedback zu ersetzen, wird beim Drücken der Ventile zusätzlich eine Vibration des Geräts in Gang gesetzt. Bei der in dieser Arbeit zitierten Umfrage unter Mobile Music App Nutzern, wurde das fehlende haptische Feedback angesprochen, sowie eine möglichst starke Ausnutzung der Sensorik impliziert, worauf mit folgendem Prinzip eingegangen wird. Um den gesamten Tonumfang einer Trompete abdecken zu können, musste auf die Naturtöne „überblasen“ werden, was durch die Neigung des Smartphones bzw. Tablets geschieht. Die Klangerzeugung beruht auf selbst aufgenommenen Samples, welche durch die oben beschriebenen Steuerungsprinzipien kontrolliert werden. Hierbei musste gewährleistet werden, dass sowohl sehr kurze als auch sehr lange Töne spielbar waren und der Ton sowohl angestoßen, als auch gebunden werden können muss. Die Lösungen sind das Loopen des stationären Trompetenklangs und verschiedene Modi zum Starten der Töne. Das Ergebnis ist eine recht natürlich klingende, intuitiv und ähnlich dem Original handhabbare Trompeten-App, die jedoch aufgrund verschiedener Schwachstellen, wie der Latenz und des veränderten Konzepts zum Überblasen, etwas zu behäbig ist. Deshalb ist das Livespiel von komplexeren Stücken leider nicht möglich und die App dient eher als Spielzeug, Übungshilfe für Anfänger oder als Heranführung an das Instrument für Interessierte. Die App ist in die Kategorie App-Instrumente einzuordnen, welche zum Großteil nicht die zukunftsweisenden Umsetzungen von Mobile Music Apps beinhaltet. Neue Ideen der Klangerzeugung und innovative Konzepte zur Bedienung werden von Experten als vielversprechender für die Zukunft und bereichernder für die Musikwelt angesehen.

1 Einleitung

Laut Schätzungen soll es mittlerweile mehr mobile Geräte („mobile devices“) als Menschen auf der Welt geben [3]. Die meisten von ihnen werden Smartphones sein, welche zum überwiegenden Teil auch für das mehr als reichhaltige Angebot von Apps aller Art genutzt werden. Ein Teil dieses Angebots besteht aus etwaigen musikbezogenen Programmen. Trotz der weitreichenden Möglichkeiten, die Smartphones und Tablets heutzutage bieten, sind Apps, welche eher den konsumierenden als den kreativen Umgang mit den Geräten fördern (z.B. Musikstreaming, Musikerkennung etc.), immer noch verbreiteter und beliebter. Jedoch gibt es immer mehr Apps, die sowohl konventionelle Instrumente nachahmen als auch neue, unkonventionelle Wege bieten Musik zu machen. Neben vielen Pianos, Synthesizern, Drum Machines, DJing Apps und Sequencern, welche einen Großteil der interaktiven *„Mobile Music Apps“* ausmachen, gibt es mittlerweile auch mehr Instrumente für Smartphones, welche komplexere Funktionsweisen bzw. Bedienkonzepte besitzen als die oben genannten. Zu den Apps, welche die technischen Möglichkeiten, wie beispielsweise verschiedene, sich in mobilen Geräten befindliche, Sensoren, verstärkt zu ihrem Vorteil nutzen, gehören sowohl Instrumente, welche innovativ in Konzept, Bedienung und Klang sind, als auch herkömmliche Instrumente, welche bei ihrer Umsetzung als App ebenfalls von der Ausnutzung neuer technischer Möglichkeiten profitieren. Im Rahmen dieser Bachelorarbeit wird ein konventionelles Instrument, die Trompete, unter Zuhilfenahme der eingebetteten Systeme im Smartphone möglichst originalgetreu in Spielweise und Klang in eine App umgewandelt.

Hierbei kommen zwangsläufig mehrere Fragestellungen auf, welche die Umsetzung eines herkömmlichen Instruments als App behandeln. Zum einen stellt sich die Frage, wie nah man dem Klang und der Spielweise eines Originals mithilfe einer digitalen Instrumentensimulation auf einem mobilen Gerät kommen kann. Im Zuge dessen ist unter anderem eine Beurteilung des Spielerlebnisses ausschlaggebend, bei dem vor allem die Kompromisse und Restriktionen bei der Umsetzung, aufgrund der technischen Gegebenheiten, eine Rolle spielen. Zum anderen ergibt sich die Frage der Sinnhaftigkeit. Es kann hinterfragt werden, welchen Zweck eine Imitation eines klassischen Instruments haben könnte und welche Art der Nutzung als sinnvoll oder bereichernd gesehen werden kann. Zusätzlich kann das Konzept im Hinblick auf die Zukunft und das Potential von Mobile Music Apps erläutert werden.

In ***Kapitel 2*** wird ein erster Einblick in die Welt der Mobile Music Apps und des aktuellen Angebots von Apps verschiedener Unterkategorien unter dem Oberbegriff Musik geboten. Zusätzlich wird sowohl eine Betrachtung des technischen Potentials von Smartphones und deren Entwicklung behandelt, als auch eine Untersuchung der Umsetzungen und der Musikalität von Mobile Music Apps, speziell im Hinblick auf den sozialen Kontext, musikalische Bildung und Weiterentwicklung durchgeführt. Zum Entwicklungsprozess der eigenen Mobile Music App gehörte zu Anfangs das Betrachten von Referenzprojekten bzw. das Testen und Evaluieren bereits erhältlicher und eventuell vergleichbarer Trompeten-Apps im „Play Store“ um die gesammelten Erkenntnisse in die Entwicklung des eigenen Programms einfließen zu lassen und das Produkt später in das bestehende Angebot

ähnlicher Apps einordnen zu können (***Kapitel 3***). Nachfolgend werden die akustischen Eigenschaften und die Funktionsweise einer Trompete erläutert, um darauf aufbauend die Umsetzung der Tonerzeugung und Bedienung in der Simulation zu schildern und zu begründen (***Kapitel 4***). Abschließend wird die im Zuge dieser Bachelorarbeit entwickelte App, der *Trumpet-Simulator*, dem Angebot von Trompeten-Apps vergleichend gegenübergestellt und in Bezug auf Umsetzung, Musikalität und Anwendbarkeit kritisch hinterfragt (***Kapitel 5***).

2 Mobile Music – Potential und Limits

2.1 Technologische Entwicklung des Smartphones

Im Jahr 1994 kam das erste Telefon auf den Markt, dass in der stetig wachsenden Entwicklungsreihe sogenannter Smartphones als Pionier angesehen werden kann. Der als *Personal Communicator* bezeichnete *„Simon“* war ein von *IBM* und *BellSouth* entwickeltes Telefon, dass als erstes seiner Art unter anderem die Funktionsweisen eines Telefons, E-Mail Programms, Rechners und Kalenders vereinte. Außerdem besaß das Gerät ein druckempfindliches Display als Vorreiter der heutigen Touchscreens. Es konnte zudem mit einer Speicherkarte erweitert werden um weitere Funktionen, wie z.B. eine Musikwiedergabe, zu nutzen [4]. Das Gerät war mit dem System *PEN/GEOS* ausgestattet und konnte optional, wie die später folgenden *Personal Digital Assistants* (*PDAs)* mit einem speziellen Stift bedient werden. Nachdem das System ebenfalls von Nokia für die ersten Geräte der *Communicator Serie* eingesetzt wurde, wurde es bei den späteren Modellen der Serie von dem von *Nokia* mitentwickelten *Symbian* Betriebssystem abgelöst. Mit diesem Betriebssystem waren im Jahr 2006 in etwa 73% der Smartphones ausgestattet [5] [6].

Bereits bei diesen ersten Smartphones war die Richtung der technologischen Entwicklung erkennbar, welche sich derart vollzog, dass die rechenstarken und umfangreichen Telefone eher als Taschencomputer mit Telefonfunktion denn als Telefongerät mit Zusatzfunktionen angesehen werden konnten. In Folge dieser Entwicklung ist das Telefonieren mittlerweile nur noch eine der zahlreichen Funktionen von Smartphones und kein wesentlicher Zweck. Trotzdem ist der Begriff *„SmartPHONE“* im Sprachgebrauch hängen geblieben, wobei das Wort vor allem im Deutschen eine klare Abgrenzung zum einfachen und funktionsärmeren *„Handy“* bezweckt.

Nachdem neben den oben genannten Smartphones unter anderem sogenannte *PDAs* auf den Markt kamen, welche anders als z.B. die *Communicator* Reihe ganz auf eine Software Tastatur setzten, wurde im Jahr 2007 das erste *iPhone* von *Apple* ausgeliefert. Dies kann als ein Meilenstein in der Entwicklung von Smartphones betrachtet werden, da zum ersten Mal eine ausgereifte Variante eines Gerätes verfügbar war, welches mithilfe der ersten Version des von *Apple* entwickelten *iOS* Betriebssystems eine rechenstarke Central Processing Unit (CPU), Graphics Processing Unit (GPU), einen digitalen Signalprozessor (DSP), ein Multitouch-Display, sowie verschiedene Sensoren auf kleinsten Raum nutzbar machte. In den darauf folgenden Jahren kamen neue Betriebssysteme wie *Android*, *Palm webOS* und *Windows Phone 7* auf, welche auf Geräten liefen, die auf eine ähnliche Funktionsweise wie das *iPhone* setzten[14]. Da viele Herstellerfirmen in der Folge auf das *Android* Betriebssystem setzten, erreichte dieses im zweiten Quartal des Jahres 2014 einen Marktanteil von knapp 85%, gefolgt von *Apple iOS* mit ca. 12% [7].

Während seit 2007 bereits 10 Versionen des *iPhones* erschienen und eine riesige Anzahl an *Android* Smartphones auf den Markt kam (neben *Windows*, *BlackBerry OS* und anderen

Betriebssystemen), haben sich die Technologien, welche in den Geräten eingebettet sind vermehrt und verbessert. Gleichzeitig entstanden mehr Ideen und Wege diese Funktionen kreativ zu nutzen. Unten ist eine Liste mit Technologien aufgeführt, welche mittlerweile weit verbreitet sind und durch welche kreative, innovative Ideen von Entwicklern verwirklicht werden können (fettgedruckt sind die beim *Trumpet-Simulator* verwendeten Komponenten).

- **Mikrophon/ Lautsprecher (DSP)**
- **Touchscreen (Multitouch)**
- **schnelle Central Processing Unit (CPU) und Graphics Processing Unit (GPU)**
- **viel Speicher (erweiterbar)**
- kabellose Datenverbindung (Mobile Daten, WLAN, Bluetooth)
- GPS Location
- **Sensoren:**
    - **Beschleunigungssensor**
    - Gyroskopsensor
    - Magnetsensor
    - Nähesensor
    - Lichtsensor
    - Temperatursensor
    - Drucksensor

Ein eher neues Feld bei der Benutzung von Smartphones sind die in den letzten Jahren immer stärker aufkommenden *Smartwatches*. Hierbei handelt es sich um Armbanduhren, welche meist von Handyherstellern speziell für deren Smartphone-Produkte entwickelt werden, wie z.B. die *Apple Watch*, *Sony Smartwatch* oder *Moto 360*. Zusätzlich gibt es unabhängige Ansätze, wie z.B. *Pebble* [8], eine Smartwatch, welche mithilfe von Crowdfunding realisiert wurde. Alle diese Geräte haben gemein, dass sie ähnlich wie Smartphones, über verschiedenste eingebettete Systeme, wie z.B. eigenem Gyroskop- und Beschleunigungssensor verfügen und hauptsächlich zur Nutzung in Verbindung mit einem Smartphone ausgelegt sind. Die Uhren besitzen zum Großteil Touchscreens und können somit, analog zu Smartphones und Tablets, per Finger bedient werden. Die Funktionalität der Smartwatches hat sich in den letzten Jahren derart erweitert und verbessert, dass es mittlerweile unzählige Anwendungsmöglichkeiten für Sportanwendungen, Gesundheitsüberwachung, Kommunikation, Navigation und andere Felder gibt [9]. Teilweise sammeln diese Geräte Informationen über die eingebaute Sensorik, gleichzeitig können die Daten jedoch auch durch die Verbindung zu einem Host, wie einem Smartphone, empfangen werden. Somit dient eine solche Uhr auch als *front end*, also eine Art zusätzlicher Benutzeroberfläche, eines solchen Geräts und kann ebenfalls Informationen senden. Genau wie für Smartphones und Tablets können Dritte Anwendungen für Smartwatches entwickeln und somit eine kreative und innovative Nutzung der mobilen Geräte vorantreiben. Dadurch ergeben sich auch neue Möglichkeiten für Mobile Music Apps, da sich neue Wege auftun Daten mithilfe der Uhren zu generieren und diese in Verbindung

mit dem Smartphone zur Steuerung von Instrumenten-Apps zu nutzen.

2.1.1 Entwicklung der App Märkte

Im Jahr 2014 wurde der *App Store* von *Apple* im Hinblick auf die Anzahl der verfügbaren Apps zum ersten Mal vom *Google Play Store* überholt [10]. 2010 noch fast ein Monopol, wies der *App Store* seitdem außerdem eine geringere Wachstumsrate als der größte Konkurrent auf. Nach dem Aufkommen von konkurrenzfähigeren Android Geräten wuchs der auf deren Nutzer abzielende *Play Store*, in welchem Apps für Android Geräte anboten werden, rasant. Selbst im Jahr 2014 wies dieser eine Verdoppelung der angebotenen Apps auf, sodass mittlerweile über 1,6 Millionen Apps (Stand 26.08.2015 [11]) verfügbar sind und der *App Store* somit überholt wurde. Gleichzeitig gibt es bereits seit 2012 mehr Entwickler für Android Apps als für iOS Anwendungen. Neben dem schnelleren Wachstum des *Play Stores*, liegt außerdem die Kategorie *Music* dort an dritter Stelle der am schnellsten wachsenden Kategorien, während sie es im *App Store* nicht unter die Top 5 schaffte. Die Anzahl der Apps für Android Geräte haben sich in dieser Kategorie im Jahr 2014 mehr als verdoppelt (>125% Wachstum). In Anbetracht dieser Fakten und des Umstands, dass Android Smartphones während dieses Zeitraums, wie auf *S. 6* erwähnt, einen Marktanteil von ca. 85% besaßen, kann schlussgefolgert werden, dass der Großteil der Nutzer mobiler Geräte, als auch der Entwickler, ein gesteigertes Interesse an Apps besitzen, welche in die Kategorie *Mobile Music Apps* eingeordnet werden können. Die obigen Statistiken lassen jedoch keine Bewertung der Qualität der Mobile Music Apps zu, weshalb im Folgenden das Potential, die Umsetzung und die Anwendungsmöglichkeiten dieser App-Kategorie betrachtet werden.

2.1.2 Technisches Potential von mobilen Geräten

Die in Kapitel ***2.1*** aufgezählten Technologien, welche heute recht ausgereift und sogar in den meisten Smartphones der sogenannten Mittel- und Unterklasse vorhanden sind, tragen großes Potential im Hinblick auf innovative Konzepte für musikalische Bedienung und Klangerzeugung in sich.

Laut *Tanaka et al.* ist dabei ein sehr wichtiger Gesichtspunkt das volle Ausnutzen der Möglichkeiten und das Anpassen der Bedienung und des Designs der Benutzeroberfläche an die Handhabung und Größe (Form Faktor) der Geräte [12]. In der Umfrage von *Tanaka et al.*, welche eine Evaluierung der Erfahrungen der Testpersonen mit Mobile Music Apps durchführt und als Ziel eine Art Leitfaden für die Entwicklung von Mobile Music Graphical User Interfaces (GUIs) beinhaltet, wird jedoch der oben genannte Punkt oft bemängelt, weswegen die Instrumenten-Apps zum Großteil nur als Spielzeug betrachtet wurden. Die Gründe dafür seien der oben genannte Form Faktor bzw. die limitierte Größe der Geräte und des Displays (was jedoch auch als Vorteil gesehen werden kann), das Interface Design, sowie Latenzprobleme. Gleichzeitig wurden die vielen unterschiedlichen Sensoren als auch die Mobilität (konträr zur Kritik am Form Faktor) als große Vorteile aufgeführt. Bei dem im Zuge dieser Arbeit entwickelten *Trumpet-Simulator* kann die Anpassung an die Größe des Geräts auch als positiv betrachtet werden. Die Hauptbedienelemente, die virtuellen Ventile, können fast als gleichwertig zum Original angesehen werden, da die Größe der Ventile sehr ähnlich und bei bestimmten Smartphone Modellen sogar deckungsgleich zum Original sind. Auf dem Smartphone, welches zum Testen der App genutzt wurde (4,7" bzw. 11,7 cm Displaydiagonale), sind die Ventile etwa gleich groß und im selben Abstand wie bei der in

Abb. 12 (a) abgebildeten Bb-Trompete mit Perinetventilen. Außerdem besitzen sie dieselbe Funktion, welche eins zu eins ins Digitale übertragbar war.

Bei Tanaka et al. wird kritisiert, dass der Benutzer oft dauerhaft auf den Bildschirm blicken muss, was die musikalische Interaktion mit anderen Musikern einschränkt, bei der üblicherweise auch mit Blickkontakt untereinander kommuniziert wird. Eine verstärkte Ausnutzung der eingebetteten Sensoren in mobilen Geräten führt dazu, dass der Nutzer weniger oder gar nicht auf das Display schauen muss während er die App bedient. Dies verbessert das intuitive und freie Spielen, als auch das Zusammenspiel. Beim *Trumpet-Simulator* kann, wie bei der Trompete, eventuell ohne Benutzung eines optischen Feedbacks gespielt werden. Sollten die Winkelpositionen zum Überblasen soweit erlernt sein, dass das optische Hilfsmittel, die Winkelanzeige (*Abb.8, S. 32 & Abb. 12, S. 38*), nicht mehr benötigt wird, wäre diese Freiheit ein Vorteil des Instruments.

Als eine der Limitierungen wurde bei der erwähnten Umfrage das fehlende haptische Feedback genannt. Bei der Nutzung von Softwarebedienelementen fehlt dieses komplett und ist somit eines der größten Nachteile eines Displays gegenüber dreidimensionalen, physikalischen Knöpfen, Schiebereglern und ähnlichem. Ausgehend von dieser Erkenntnis wurde beim *Trumpet-Simulator* das Auslösen einer Vibration beim Betätigen der virtuellen Ventile hinzugefügt. Dies ist die einzige Möglichkeit die Rückwirkung einer Aktion dem Tastsinn zuzuführen.

2.2 Musizieren mit Mobile Music Apps

2.2.1 Kategorisierung von Mobile Music Apps

Das stetig wachsende Angebot an Apps, die dem Bereich Mobile Music zugeordnet werden können, kann in verschiedene Unterkategorien eingeteilt werden. Dies wird sowohl im *Play Store* als auch beim *Apple App Store* und *Amazon Appstore* nicht getan. Im *Play Store* sind beispielsweise alle Mobile Music Apps unter **Apps → Musik & Audio** aufgelistet. Aufgrund dessen muss ein Smartphone-Nutzer eine genaue Vorstellung haben, welche Arten von Apps in dieser Kategorie vorhanden sind und nach welchen er genau sucht. Das führt dazu, dass viele interessante Apps im riesigen Angebot untergehen, vor allem solche, welche unkonventionelle Konzepte verfolgen und deswegen weder durch vermehrte Suchanfragen, noch durch die Auflistung bei den erfolgreichen Apps zu entdecken sind. Es müsste vielmehr ein spezieller Suchbegriff eingegeben oder sich viel Zeit zum Durchforsten des Angebots an Mobile Music Apps genommen werden. Zielführend wäre hier eine weitere Kategorisierung innerhalb der Kategorie **Musik & Audio**, um den verschiedenen Nutzern mit ihren unterschiedlichen Interessen die passenden Typen der Apps anbieten zu können.

Matthias Krebs nimmt in seinem Artikel *App-Musik – Musizieren mit Smartphones (Seite 16-17 und 19)* [13] eine beispielhafte Unterteilung dieses App-Angebots vor, welche recht einleuchtend und sinnvoll ist. Um einen Überblick über das Angebot an Musik-Apps zu geben und im Folgenden den *Trumpet-Simulator* einer dieser Unterkategorien zuzuordnen, werden hier die von *M. Krebs* beschriebenen Typen von Mobile Music Apps aufgezeigt (*Tabelle 1, S.10*).

In *Tabelle 1* ist die Unterkategorie *App-Instrumente* grau hinterlegt, da die im Zuge dieser Arbeit entwickelte App in diese einzuordnen ist. Beim *Trumpet-Simulator* wird genau das unter der Beschreibung genannte Ziel verfolgt, ein klassisches Instrument originalgetreu

abzubilden. Dieses Konzept gehört jedoch nicht zu den Zukunftsweisenden unter den in der Tabelle aufgelisteten Ansätzen. Die Vorteile von App-Instrumenten sind die bereits bekannten Funktionsprinzipien, sowie die übersichtliche Bedienung und nachvollziehbare Erzeugung vertrauter Klänge. Allerdings wird das Potential der Geräte nur selten ausgeschöpft und keine neuen Ideen entwickelt um das Musizieren mithilfe der technischen Möglichkeiten zu erweitern.

Unterkategorie	Beschreibung	Beispiele
Hilfsmittel und Effektgeräte	Unterstützen oder erweitern das Musizieren	Stimmgeräte, Metronome, Analyse-Tools, Gitarren-Effektgeräte, Recorder, Notendarstellung
App-Instrumente	Bilden Instrumente originalgetreu ab	Synthesizer, Klaviere, Gitarren, Drum Sets
App-Soundtoys	Musikalische Spielzeuge mit leichter Bedienung und fantasievollen Klängen	*SingingFingers, SpectrumGen, Bebot*
Beatmaker- und Sequencer-Apps	Rhythmusmaschinen mit Stepsequencern oder Musikproduktionsprogramme	*FL Studio Mobile, Groove Mixer, My Beat Creator*
DJing-Apps	Virtuelle Turntables mit Mischpult und Effektgeräten	*DJ Studio 5, Cross DJ Free, Looptastic*
Controller-Apps	Multitouch-Fernsteuerung von Studioequipment bzw. Aufnahmesoftware	*TouchOSC, touchAble, Midi Touch*
Kunst-Apps	Interaktive Kunstwerke mit neuartigen, innovativen Konzepten von Klang und Bedienung	*Small Fish, abcdefg, Biophilia, Synse*

Tabelle 1: Unterkategorien von Mobile Music Apps nach M. Krebs [13]

2.2.2 Umsetzung von Mobile Music Apps

Die bereits in Kapitel **2.1.2** erwähnte Umfrage von *Tanaka et al.* [12] hatte den Ansatz iPhone-Nutzer zur Nutzung und ihren Erfahrungen mit Mobile Music Apps zu befragen und sowohl positive als auch negative Kritik als Leitfaden zur Entwicklung von GUIs zukünftiger Apps zu nutzen. Von denjenigen Befragten, welche instrumentenähnliche Apps auf ihrem iPhone nutzten, gab der Großteil (55,1%) an, dass diese sich „nicht wirklich“ oder „gar nicht“ (18,4%) wie ein Instrument anfühlten. Nichtsdestotrotz meinten viele Personen, dass das Smartphone ein „recht gutes“ (28,9%) bzw. ein „sehr gutes“ (30,1%) interaktives musikalisches Erlebnis liefern kann. Anders als die Band *Gorillaz*, welche ihr Album *The Fall*

von 2010 während einer Tour ausschließlich auf dem iPad aufnahm und abmischte [14] und dafür Studio-Apps und verschiedenste App-Instrumente verwendeten, gaben jedoch nur 8,3% der Benutzer von Studio- und Produktions-Apps an, einen Song auf ihrem mobilen Gerät fertiggestellt zu haben. Die Mehrheit der Testpersonen, welche das Smartphone zum Musik machen verwendeten, taten dies in Verbindung mit dem Computer (56,5%) während immerhin 79,3% sich vorstellen konnten, es auch autonom zum Musizieren einzusetzen. Um zu erfahren, welche Technologien bzw. Bedienkonzepte bei Mobile Music Apps am weitesten verbreitet sind, wurde erfragt, welche Teile des Smartphones hauptsächlich genutzt wurden. Dabei nutzten fast alle (91,9%) das Touchscreen, die Mehrheit zudem Beschleunigungssensoren (59,3%) und die Mikrophone (51,2%) und nur wenige die Kameras (19,8%) sowie die Nähesensoren (14%).

Bei oben beschriebener Umfrage ist anzumerken, dass es sich um eine besondere Testgruppe handelte, bei der mehr als ein Drittel Amateurmusiker und ein weiteres Drittel Profimusiker waren und ein sehr großer Anteil Erfahrungen als Programmierer besaßen. Zudem hatten viele der Befragten schon Erfahrungen mit Mobile Music Apps, was natürlich unabdinglich für eine aussagekräftige Umfrage zu diesem Thema ist.

Ausgehend von den Ergebnissen der Umfrage kann festgehalten werden, dass, zumindest bei einer Zielgruppe ähnlich der Testgruppe, ein großes Interesse an der Verbesserung von Mobile Music Apps besteht. Zusätzlich trauen die meisten Personen der Technik von Smartphones viel zu. Ein Großteil glaubt beispielsweise, dass es Potential bei der sozialen Komponente des Musizierens mit Apps gibt und die Technik der mobilen Geräte sich derart verbessert hat, dass man sie auch ohne Computer, also nicht nur als Controller oder Zusatzelement, sondern ebenfalls als eigenständiges Instrument verwenden kann. Das positive Bild wird jedoch getrübt, da die meisten Testpersonen bemängelten, die Apps fühlten sich nicht wie ein Instrument an. Dies kann auch an den nicht ausgiebig genug genutzten technischen Möglichkeiten liegen und den damit verbundenen wenig innovativen Konzepten. Andere Gründe sind beispielsweise die unzureichende Klangqualität und Bedienung bzw. der *„Workflow“*, welche einer ernsthaften und längeren Beschäftigung mit den Apps entgegenwirken. Bezeichnend dafür sind auch die Antworten auf die Frage, welche Sensoren vorwiegend Verwendung finden. Hier liegt zwar überraschenderweise der Beschleunigungssensor auf zweiter Stelle vor dem Mikrophon, jedoch könnten die auf *Seite 7* aufgezeigten Technologien, bzw. weitere Sensoren und vor allem die Vernetzungsmöglichkeiten stärker und ideenreicher genutzt werden. Unter anderem dadurch entsteht der geringe Prozentsatz der Personen, welche ein Projekt auf ihrem Smartphone fertig stellten. Vor allem die Vernetzung verschiedener Geräte und Musiker vor Ort oder über den Globus führt zu einem der interessantesten Punkte in der Entwicklung von Mobile Music Apps: die soziale Komponente. Obwohl 56,8% der Befragten in obiger Umfrage das Smartphone zum gemeinsamen Spielen nutzten, war für die meisten die Interaktion mit anderen Spielern nicht zufriedenstellend. Dieser soziale Aspekt wird in Zukunft eine der größten Herausforderungen beim Konzipieren und entwickeln von Mobile Music Apps sein, welche allerdings mitunter das größte Potential bietet.

Bei den Recherchen über die aktuelle Welt der Mobile Music Apps und Sammeln der Erfahrungen und Erwartungen kristallisierte sich ein Punkt heraus, welcher auch von vielen, sich mit dem Thema beschäftigenden Wissenschaftlern, betont wird. Die Zukunft der Mobile Music Apps liegt nicht im Imitieren von bereits bestehenden, traditionellen Instrumenten (vgl. [12] S.3; [13] S.19), sondern in der Möglichkeit nie dagewesene Konzepte zum Musizieren zu verwirklichen und diese in einen sozialen Kontext zu bringen bzw. Menschen durch die Geräte und Apps zu vernetzen.

2.2.3 Soziale Komponente

Beim Musizieren mit Smartphones und Tablets ist einer der wichtigsten Gesichtspunkte ein sozialer. Wie fälschlicherweise oft angenommen, soll die traditionelle Art Musik zu machen und Klänge zu gestalten nicht durch die neuen, technischen Gegebenheiten ersetzt, sondern bereichert werden, auch im Hinblick auf soziale Vernetzung.

> *„Es geht darum, die traditionelle Musizierpraxis nicht zu ersetzen, sondern sie technisch zu erweitern.“*
>
> *(Krebs, Matthias (2012): App-Musik – Musizieren mit Smartphones, Perspektiven und Potenziale einer neuen musikalischen Form, in: Musikforum 01/2012, Seite 15)*

M. Krebs erwähnt im oben zitierten Artikel, neben den in dieser Thesis später betrachteten Vorteilen von Musik Apps für musikalische Bildung, die direkte soziale Komponente beim Musizieren mit Apps. Die Bedenken, dass beim *„Umgang mit der digitalen Technik […] die Musik […] zu kurz kommt“*, federt er mit dem Argument ab, dass ein positiver Effekt im Hinblick auf eine *„kommunikative Interaktion“*, also ein Zusammenspiel und eine gemeinsame musikalische Erfahrung eintritt. Dies kann sich vorteilhaft auf die Motivation und im Umkehrschluss wiederum auf die musikalische Qualität der Darbietungen auswirken. Analog zum obigen Zitat wird nicht nur das Musizieren an sich, sondern auch das Zusammenspiel durch die Nutzung von Apps erweitert. Während das gemeinsame Spiel mit traditionellen Instrumenten altbekannt ist, können durch verschiedene Arten von Vernetzungen neue Konzepte des kollektiven Musizierens hervorgebracht werden. Als Beispiel sei hier eines der Referenzprojekte für den *Trumpet-Simulator* genannt, die *Smule Ocarina*. Die von *Ge Wang* entwickelte digitale Version des traditionellen Blasinstruments wird als eine Art „soziales Instrument“ beschrieben [15]. Der sogenannte *World Listener* ist ein Teil der App, durch welche man mithilfe von GPS und einer Internetverbindung andere Ocarinaspieler auf der ganzen Welt orten und ihnen beim Musizieren zuhören kann.

Einen traditionelleren Weg mit Mobile Music ein kollektives Erlebnis zu schaffen, geht das an ein Kammerorchester angelehnte *Stanford Mobile Phone Orchestra* [16]. Ähnlich wie in den sogenannten Laptop-Orchestern (beispielsweise *PLOrk* aus Princeton, *SLOrk* aus Stanford, *KULOrk* aus Graz) entwickeln hier die Musiker ihre eigenen Instrumente und suchen neue Wege der Bedienung, Klangerzeugung und letztendlich des Zusammenspiels. Dies ist ein weiteres Beispiel dafür, dass die soziale Komponente durch das Musizieren mit den eher persönlichen Eigentumsgegenständen Smartphone und Tablet nicht verloren geht, sondern verstärkt werden kann. Neben dem kollektiven Spielen auf verschiedenen, selbst entwickelten Instrumenten werden dabei auch Konzepte entwickelt, bei denen z.B. zwei Spieler gleichzeitig an einem Instrument bzw. Gerät agieren können. Maßgeblich dafür ist beispielsweise die bereits vorher erwähnte Multitouchtechnik.

Die soziale Komponente bzw. die Vernetzung von Geräten für eine musikalische Darbietung

ist jedoch keineswegs ein komplett neues Phänomen. Bereits 2001, vor der Verbreitung von internetfähigen Smartphones, entwickelten *G. Levin*, *S. Gibbons et al.* eine Performance, bei der die Handys des Publikums als Instrumente genutzt wurden und ein ausgeklügelter Ablauf verschiedenster Zusammenspiele der an die Geräte gesendeten Klingeltöne und Klänge von den Initiatoren gesteuert wurde. Somit wurden sowohl die von außen angesteuerten Geräte als auch jedes Publikumsmitglied Teil einer musikalischen Darbietung, welche teils mehr und teils weniger geplante melodische Anteile, als auch interessante Klanglandschaften beinhaltete. Uraufgeführt wurde die Performance beim *Ars Electronica Festival 2001* in Linz und war vermutlich der erste ernsthafte Ansatz das musikalische Potential des Alltagsgegenstands Handy aufzuzeigen und die Möglichkeiten des Kollektivs des damals noch sehr bescheidenen musikalischen Potentials der Geräte zu erkunden [17] [18]. Da der Mensch als soziales Wesen dazu neigt, kollektive Erlebnisse zu suchen, und diese Tendenz beim Musizieren nicht weniger stark ist als in anderen Bereichen, spielt ungeachtet des Fortschritts der Technik und der Ausgereiftheit der musikalischen Möglichkeiten der mobilen Geräte und Apps, die soziale Komponente immer eine tragende Rolle.

2.2.4 Musikalische Bildung mit Hilfe neuer Technologien

Ein weiterer sozialer Vorteil neben dem gemeinsamen Spielen und der Vernetzung der Geräte ist die Verfügbarkeit und Allgegenwärtigkeit mobiler Geräte. Dies führt zu einer Art Demokratisierung beim Musizieren, vor allem wenn die neuen Technologien in der Bildung, also beispielsweise im Musikunterricht zu Hilfe genommen werden. Die oben erwähnte Verfügbarkeit der Geräte, sowie deren Preise, sind so gut wie nie und Smartphones sind ohnehin derart verbreitet, dass es beispielsweise im Unterricht keine großen Nachteile für Schüler auftreten, welche keine musikalischen Vorkenntnisse besitzen. Ein Großteil der Schüler aus der Mittelstufe besitzt höchstwahrscheinlich ein solches Gerät oder hat es schon einmal benutzt. Das Gefälle zwischen den Vorkenntnissen bei der Nutzung von Mobile Music Apps ist also geringer als bei traditionellem Musikunterricht, bei dem einige wenige Schüler den Vorteil besitzen bereits ein Instrument privat erlernt bzw. privaten Musikunterricht genossen zu haben, während der Rest keinen Zugang zu diesem Thema hat. Somit können auch Menschen ohne musikalische Vorkenntnisse und Fähigkeiten leichter und schneller Musik machen und auch mit denjenigen interagieren, welche mehr Erfahrung haben und von diesen lernen. Außerdem können die kürzeren Übungszeiten zum Beherrschen eines App-Instruments, im Gegensatz zu klassischen Instrumenten, die Motivation fördern bzw. Frustration beim Erlernen von musikalischen Fähigkeiten aber auch musikalischer Theorie reduzieren. Im Zuge dessen ist es ebenfalls von Vorteil, dass sich bei Mobile Music Apps schneller ein recht professioneller und schöner Klang erreichen lässt als beim Erlernen von traditionellen Instrumenten.

Für den Musikunterricht in der Schule wurden von *Sastre et al.* drei verschiedene Konzepte entwickelt um neue Technologien im Unterricht einzubinden und somit vom größeren Interesse und gesteigerter Motivation der Schüler zu profitieren [19]. Die hohe Verfügbarkeit von Tablets, Smartphones und auch Interfaces wie z.B. *Kinect* [20] können den Unterricht interessanter machen und die Kreativität und entdeckerische Herangehensweise fördern, anstatt von oben herab den Sinn für Ästhetik und den historischen Kontext der Musik zu lehren. Dabei wird die Eigeninitiative der Schüler gefördert, was die Motivation hebt und den Schüler von sich aus antreibt eventuell mehr über das Fach Musik in Erfahrung zu bringen. Da die verbreiteten technischen Geräte für die Schüler zudem greifbarer und schneller zu

verstehen sind als herkömmliche Musikinstrumente, findet eine Gleichstellung der Schüler statt, da beim pragmatischen Ansatz ein kleineres Gefälle der musikalischen Fähigkeiten entsteht, welches ansonsten durch die sehr unterschiedlichen Vorkenntnisse der Schüler beachtlich sein kann. Außerdem wird das kreative und kollaborative Gestalten vereinfacht und in den Vordergrund gerückt.

Neben dem gemeinsamen Gestalten vor Ort sollen auch bei *Sastre et al.* die Möglichkeiten genutzt werden, durch Web 2.0 bzw. Social-Media-Vernetzung, einen Austausch von Projekten sowie Klängen oder entwickelten Soundmodulen zu ermöglichen um abstraktere und dynamischere Konzepte zum Kreieren und Manipulieren von Klängen und Aufführen von Musik zu ermöglichen.

2.2.5 Apps als Übungshilfe für Musiker

Ein weiterer Nutzen von Mobile Music Apps ist, es Musikern zu ermöglichen mithilfe der immer, überall und schnell zugänglichen Programme, zu jeder Zeit, an jedem Ort, spontan und ohne viel Equipment und Vorbereitung üben zu können.

Ein interessantes Beispiel für diesen Ansatz ist das von *Yang Kyu Lim* und *Woon Seung Yeo* entwickelte *vMaestro*, ein Interface für Smartphones, mit dem man das Dirigieren proben kann [21]. Sie entwarfen ein Konzept, um mit Hilfe des Smartphones und dessen Gyroskop die Gesten eines Dirigenten zu erfassen und zu erkennen. Die Gesten werden dabei interpretiert und über *Pure Data* zur Steuerung von in *Ableton Live* [22] importierten Musikclips verwendet, wobei der Nutzer zudem mit bestimmten Gesten zu gewissen Stellen im Stück navigieren kann. Letzteres als auch die Schlichtheit, Erschwinglichkeit und Verfügbarkeit des Systems werden als große Vorteile aufgeführt, vor allem im Hinblick auf bereits existierende Übungssysteme. Mit der Erschwinglichkeit verhält es sich mit allen Apps ähnlich, welche umsonst oder zu einem geringen Preis angeboten werden. Betrachtet man die Anschaffungskosten von Apps, kann oder muss man den Erwerb eines Smartphones außen vor lassen, da ein solches Gerät mittlerweile für die meisten Menschen gewissen Alters und Herkunft als bereits verfügbarer Alltagsgegenstand anzusehen ist.

Bei *vMaestro* war einer der größten Kritikpunkte das Gewicht des Smartphones. Beim Dirigier-Übungssystem tritt der Fall auf, bei dem das Smartphone, welches als *„handheld device“* den Dirigierstock ersetzt, als Substitut um einiges schwerer (und größer) ist als das Original, was, gerade bei den raschen und häufigen Bewegungen beim Dirigieren, ein Problem darstellt. Dieser Unterschied ist noch größer, wenn der Dirigent normalerweise nur die Hände und keinen Stock verwendet. In den meisten Fällen ist jedoch, gerade bei der Umsetzung klassischer Instrumente als Apps, im Hinblick auf Gewicht und Größe, das Gegenteil der Fall. Hierbei wird allerdings meist die Bedienung des Instruments vereinfacht und verändert, da man sie auf die Möglichkeiten des Endgerätes herunterschrauben muss.

Während die Ventilgröße beim *Trumpet-Simulator* ähnlich dem Original ist, es also in dieser Hinsicht keinen Nachteil gibt, ist das in diesem Fall geringere Gewicht des mobilen Geräts ein Vorteil. Die Probleme bei der Umsetzung treten hier eher beim Überblasen auf, welches, wie in Kapitel **4.2.2** beschrieben, konzeptuell anders gestaltet werden musste. Trotzdem kann auch hier ein Vorteil genannt werden, da die von jedem Trompeter erfahrene und gefürchtete Ermüdung der Lippen keine Rolle mehr spielt. Wie das oben beschriebene Übungssystem für Dirigenten, könnte der *Trumpet-Simulator* auch als Übungshilfe für Musiker oder Anfänger genutzt werden. Ohne das Instrument überall mit hinnehmen zu müssen, kann überall und ohne Lärmbelastung der Umgebung (bei leisem Spiel bzw.

Benutzung von Kopfhörern), geprobt werden. Gerade bei der Trompete, ähnlich wie bei anderen Blasinstrumenten, ist die Lärmbelastung für z.B. Nachbarn oder Mitbewohner ein ständiges Hemmnis bzw. ein Grund die Übungszeiten zu minimieren. Mit der App könnten zu allen Tages- und Nachtzeiten, oder nach dem Beenden des Übens wegen Lippenermüdung oder Lärmbelastung, Trompetenstücke vertieft oder auch das Improvisieren spielerisch erkundet werden.

3 Hintergrund und Referenzprojekte

Ein richtungsweisendes Referenzprojekt für die Idee einer Trompeten-App war die beim Thema des Bachelorseminars vorgestellte *Smule's iPhone Ocarina* von *Ge Wang* [15]. Die Umsetzung des traditionellen Blasinstruments in eine App folgt der Inspiration, ein altes und in der Entwicklung schon vor langer Zeit vollendetes System in die Gegenwart zu versetzen. Zudem wird dabei ein spezielles Ziel verfolgt und ein interessanter Prozess durchlaufen, bei dem die Möglichkeiten der Umsetzung expliziter Konzepte von der Natur ins Digitale ausgelotet, gegebenenfalls Kompromisse eingegangen und alternative Entwürfe erarbeitet werden müssen. Da ich selbst seit vielen Jahren Trompeter bin und das Trompetespielen auch im Rahmen meines Bachelorstudiums weiter verfolgen konnte, reizte mich die Vorstellung, dieses Instrument zum Mittelpunkt meines Projekts zu machen.

Die *Smule's Ocarina* als Ideengeber, macht direkt sichtbar, welche Möglichkeiten für die Erschaffung eines App-Instruments vorhanden sind. Vor allem die Bedienung, speziell das Blasen ins Mikrophon und die Fingerlöcher als Tasten auf dem Smartphonedisplay, zeigt auf, dass mobile Geräte heute auf ganz andere Arten verwendet werden können als ausschließlich über das Manipulieren von graphischen Elementen auf dem Display.

Nach dem Entschluss eine Trompete mithilfe von Pure Data und PdDroidParty für Android Smartphones zu entwerfen und während der ersten Ansätzen bei der Umsetzung des Instruments, war eine Recherche, ob und welche Trompeten-Apps im Play Store, dem größten Markt für Android Apps, bereits verfügbar sind, aufschlussreich.

3.1 Recherche

Um einen Überblick über die bereits bestehenden trompetenähnlichen Instrumente im Play Store zu bekommen, sind alle solchen Apps, die über die Suchwörter "Trompete" und "Trumpet" gefunden werden können, begutachtet worden. Die Suche ergab ausschließlich kostenlose Apps, was sicher unter anderem an der geringen Nachfrage, aber auch der angebotenen Qualität lag. Nach dem Herausfiltern einiger Apps, welche nur zum Stimmen oder als Grifftrainer bzw. -tabelle dienten oder ausschließlich Audiofiles abspielten, blieben 9 Anwendungen übrig, die ein Spielerlebnis wie auf der Trompete versprachen.

Nachfolgend ist eine Liste der getesteten Applikationen aufgeführt (angeführt sind die Namen im Play Store, die getestete Version und Entwickler, alphabetisch geordnet):

- *Echte Trompete* *Version 2.6* *Super Hero Games*
- *Paniov Trumpet* *Version 1.1* *Andre Paniov*
- *Pocket Trumpet* *Version 1.4* *SKWorks*
- *Spielen Trompete (1)* *Version 1.0* *gurkansoftware*
- *Spielen Trompete (2)* *Version 1.0* *fiammasoft*
- *Trompete* *Version 1.1* *Tike Develop*
- *Trumpet* *Version 1.7* *memorygamesengine.com*
- *Trumpet blow* *Version 1.2* *J&G*
- *Virtual Trumpet 2* *Version 1.1* *Snowy's Apps*

(Die Namen wurden teilweise im Play Store automatisch ins Deutsche übersetzt, was zu den obigen Namen geführt hat.)

3.1.1 Ergebnisse der Recherche

In der folgenden Tabelle sind die Ergebnisse der Tests auf einige wichtige Eigenschaften beschränkt dargestellt.

Name	Tonerzeugung		Flexible Tonlänge	Originalgetreue Ventilfunktion	Halbtöne spielbar	Tonumfang >1 Oktave
	Blasen	Drücken				
Echte Trompete		✓	✓			✓
Paniov Trumpet		✓	✓	✓*	✓	✓
Pocket Trumpet		✓	✓	✓	✓	✓
Spielen Trompete (1)		✓				
Spielen Trompete (2)		✓				
Trompete		✓	✓			✓
Trumpet	✓		✓			
Trumpet blow	(✓)		(✓)	(✓)	(✓)	(✓)
Virtual Trumpet 2		✓	✓			✓

Tabelle 2: Arbeitsweisen der auf verschiedene Funktionen getesteten Apps
✓ ... Ventilfunktion eingeschränkt aber im Prinzip richtig*
(✓) … Anwendung funktionierte beim Test nicht - Eigenschaften eingeschätzt anhand von Layout.

Das Ziel des konkreten Evaluierens der oben erwähnten Apps war es, die Vorteile und Nachteile der Implementierungen herauszuarbeiten und in die Entwicklung des eigenen Programms einfließen zu lassen. Dabei sind vor allem die vielen Schwachstellen der getesteten Apps auffällig, wobei die Anzahl und Art der ausgewählten Test-Apps repräsentativ für das Angebot an trompetenähnlichen Instrumenten im Play Store war. In *Tabelle 2* sind die groben Eigenschaften und Unterschiede der verschiedenen Apps ersichtlich.

Bei den meisten Tests fiel das Spielerlebnis eher enttäuschend aus. Das hatte verschiedene Gründe. In nur zwei Apps, wurde das Blasen in die Trompete durch Blasen in das Mikrophon des Gerätes substituiert. Dies funktionierte jedoch nur bei einer der beiden Versionen und gestaltete sich sehr behäbig. Ansonsten mussten immer Ventile oder an Ventile erinnernde Felder zur Tonerzeugung berührt werden. Eine Dynamik über die Stärke des Hineinblasens, war bei keiner der Apps möglich. Die Dauer der erklingenden Töne war nicht bei allen Versionen variabel, weshalb selbst bei langem Drücken der Tasten der Ton oft nach kurzer Zeit verstummte. Bei den Apps, bei denen die Dauer des Tons tatsächlich der Länge des Drückens oder Blasens entsprach, gab es trotzdem oft Maximallängen von einigen Sekunden nach denen der Ton verstummte, oder es waren sehr unangenehme Loops ohne Überblendung implementiert, welche knackende Geräusche beim Wiederholen erzeugten.

Des Weiteren fungierten meist wahllos angeordnete 8 "Ventile" als Tasten für die aufeinanderfolgenden Töne einer Dur-Tonleiter, wobei oft nur eine C-Dur Tonleiter spielbar war. Bei Versionen mit 3 Ventilen waren deren Funktionen teilweise falsch (nicht Halbtonventil, Ganztonventil, kl. Terzventil) oder es musste ein zusätzliches Ventil für die Naturtöne (offener Griff) gedrückt werden. Diese Realisierungen entsprechen nicht der natürlichen Funktionsweise einer Trompete und imitieren somit auch nicht das Spielen auf einer solchen. Außerdem deckten einige Apps nur eine Oktave ab, während die anderen höchstens zwei Oktaven beinhalteten.

Die in *Tabelle 2* angeführte Spalte *„Halbtöne spielbar“* ist bei den getesteten Apps, wie oben ersichtlich, mit der originalen Funktionsweise der Trompetenventile verknüpft. Dies zeigt, dass lediglich zwei der getesteten Trompetensimulationen wirklich spieltauglich waren, da diese als einzige alle Halbtöne liefern konnten, was entscheidend für das Spielerlebnis und die musikalische Tauglichkeit ist. Diese beiden Anwendungen waren ähnlich aufgebaut. Auf der jeweils rechten Seite des Displays waren die Ventile abgebildet und auf der linken Seite gab es Felder zum Auslösen der Töne bzw. zum Auf- und Abspielen der Naturtonreihe. Die originalgetreue Funktionalität der Ventile und die Ausnutzung der Naturtonreihe sind zwei wichtige Punkte im Hinblick auf eine möglichst intuitiv spielbare Trompetensimulation.

Während der Recherche wurde eine weitere App namens *Trumpet* getestet, welche zum Zeitpunkt des Schreibens dieses Textes nicht mehr im Play Store zu finden war und sie somit auch nicht in obiger Aufzählung aufgelistet ist. Diese bot als Einzige die Möglichkeit, das Überblasen auf die übereinanderliegenden Naturtöne über die Neigung des Geräts zu steuern. Leider war diese Version ansonsten ähnlich schwer spielbar wie die Anderen (Tonerzeugung durch Drücken, extra Tasten für offenen Griff und Griff aller drei Ventile etc.). Der dort verwendete Neigungswinkel für die Steuerung des Überblasens ist nichtsdestotrotz eine gute Lösung für das Problem der Realisierung der Naturtonreihe und somit des Tonumfangs. Die Notwendigkeit, die Töne durch ein Drücken, anstatt durch ein Blasen zu erzeugen, ist jedoch ungewöhnlich und schwierig zu koordinieren. Dieser Punkt ist also auch zu berücksichtigen, ähnlich wie bei zwei der getesteten Apps, bei denen man durch das Blasen ins Handymikrofon, Töne „erzeugen“ bzw. starten konnte. Ansonsten waren diese Versionen jedoch unzureichend entwickelt. Alles in Allem ist bei der Auswertung zu erkennen, dass vereinzelte Apps jeweils nur gewisse Funktionen gut umsetzten, während sie andere eher vernachlässigten. Somit war keine der getesteten Simulationen wirklich spielbar oder dem Spiel auf einer Trompete ähnlich.

3.1.2 Folgerungen aus der Recherche

Nun werden anhand der oben aufgeführten Erkenntnisse die verschiedenen Funktionsweisen erläutert, welche ausschlaggebend für die angestrebte, möglichst natürliche Spielweise der Trompeten-App sind. Dabei werden vor allem die in *Tabelle 2* erwähnten Funktionen nacheinander begutachtet und somit eine Art Ideengerüst für die Umsetzung erstellt.

Die erste Erkenntnis ist, dass die Tonerzeugung, wie bei den Ergebnissen der Recherche beschrieben, über eine Aufnahme des Schalldrucks am Mikrophon des Geräts vorteilhaft ist. Durch diese Maßnahme ergeben sich mehrere Vorteile. Der Nutzer kann, wie beim zugrundeliegenden Instrument, den Ton durch Blasen erzeugen. Hierbei wird das Trompetenmundstück durch das Mikrophon ersetzt, wobei man den anliegenden Schalldruckpegel als Auslöser zum Starten und Beenden der Töne verwenden kann. In

Wirklichkeit schwingen zwar die Lippen des Trompeters und erzeugen somit den Ton, jedoch kommt diese Lösung dem Original näher, als die Auslösung des Klangs durch Berühren des Displays. Außerdem kann der Pegel auch zum Regeln der Lautstärke der Klänge verwendet werden. Darüber hinaus können, durch die Nutzung des Mikrophons, die Finger des Spielers, wie gewohnt, ausschließlich zum Betätigen der Ventile bzw. zum Variieren der Tonhöhe genutzt werden, was in vielen der getesteten Apps nicht der Fall war. Das Spielerlebnis wird dadurch intuitiver gestaltet und stark verbessert. Ein wichtiger Nebeneffekt dieser Technik ist zudem die flexible Tonlänge, welche durch das Blasen gesteuert bzw. begrenzt werden kann und dem Prinzip einer Trompete gleicht.

Die originalgetreue Funktion der Ventile ist selbstverständlich ein weiterer signifikanter Punkt in der Gestaltung des Programms. Obwohl die Implementierung dieses Prinzips kein Problem darstellen sollte, wird es in nur drei der getesteten Apps verwendet und hat (bei meinem Test) nur in einem Fall funktioniert. Die drei Ventile der Bb- und C-Trompete, an denen sich hier orientiert wird, haben den Zweck, die Gesamtrohrlänge des Instruments zu vergrößern um somit die Naturtöne bzw. Resonanzfrequenzen des Rohres um einen halben, ganzen oder eineinhalb Töne (kleine Terz) zu verschieben. Des Weiteren können diese Ventile miteinander kombiniert werden. Dieses Prinzip kann man bei der App übernehmen um die minimale Anzahl an Touch-Feldern zu erreichen (bei getesteten Apps gab es oft 8 oder mehr Tasten), während man außerdem der Funktionsweise einer Trompete und dem intuitiven Spielen umso näher kommt. Damit wären alle Halbtöne spielbar und jede Tonart möglich, was bei vielen der oben erwähnten Versionen aus dem *Play Store* nicht möglich war.

Der letzte wichtige Punkt ist der Tonumfang des Instruments. Wie aus *Tabelle 2* zu entnehmen, war der Tonumfang nur bei manchen der evaluierten Instrumente größer als eine Oktave, wobei die Implementierung der vorwiegenden Oktavsprünge sehr statisch und kontraproduktiv für das Spielerlebnis war. Eine gute Lösung dafür ist das bereits angesprochene Überblasen auf andere Naturtöne durch die Neigung des Geräts. Die Neigung des Smartphones oder Tablets kann dabei erfasst werden, wobei bestimmte Bereiche des Neigungswinkels bestimmten Naturtönen und deren Modifikationen durch die Ventile zugeordnet werden können. Damit lassen sich eine ausreichende Anzahl von Naturtönen und ein gewisser Tonumfang erreichen.

4 Natürliches Funktionsprinzip & Realisierung in der Simulation

4.1 Natürliches Prinzip

4.1.1 Tonerzeugung

Bei der Trompete, wie bei allen Blechblasinstrumenten, funktioniert die Tonerzeugung nach dem Prinzip der „Polsterpfeife“. Dabei setzt der Spieler seine Lippen am einen Ende des Instruments bzw. des Rohres oder der „Pfeife“ an und erzeugt durch das Blasen eine stehende Welle im zylindrischen Rohr. Die Lippen sind dabei die schwingende Komponente, vergleichbar mit dem Rohrblatt bei Holzblasinstrumenten, und bestimmen durch ihre Schwingungsfrequenz die Tonhöhe.

Es können sich jedoch nur gewisse Frequenzen im Instrument stabilisieren. Dies sind jene, welche einer Resonanzfrequenz des Instruments entsprechen. Die Trompete besitzt zu jedem Zeitpunkt eine bestimmte momentane Rohrlänge und somit festgelegte Resonanzfrequenzen innerhalb des Rohres, welche durch den Bläser angeregt werden können. Diese sind durch das Druckmaximum (Schnelleminimum) am Anfang des Rohres, bzw. bei den Lippen des Bläsers, und durch das Druckminimum (Schnellemaximum) am Ende des Rohres bzw. beim Übergang zur äußeren Atmosphäre bestimmt. Aus diesem Grund können nur stehende Wellen entstehen, bei denen die Rohrlänge einem Viertel der Wellenlänge entspricht bzw. einem ungeradzahligen Vielfachen dessen. Genauer heißt das, die Frequenzen, welche bei gegebener Länge l in einer zylindrischen Polsterpfeife resonieren, sind mit folgender Formel gegeben:

Mit $f_{Res} = \frac{c}{\lambda_{Res}}$ und $l = k \cdot \frac{\lambda_{Res}}{4} \rightarrow \lambda_{Res} = \frac{4}{k} \cdot l$ ergibt sich:

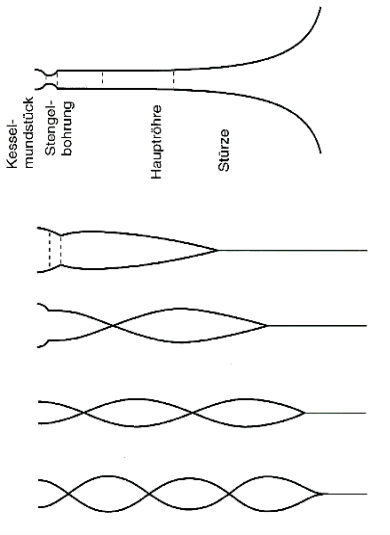
$$f_{Res,k} = k \cdot \frac{c}{4 \cdot l} \quad \text{mit } k = 1,3,5,7, \ldots$$

Diese Frequenzen gelten jedoch nur für das Modell der halboffenen Pfeife. Bei der Trompete werden zusätzlich zwei Effekte genutzt, die einerseits durch einen kleinen konischen Teil am Ende des zylindrischen Rohres und den Schalltrichter, oder zusammen genommen die „Schallstürze“, und andererseits durch das Mundstück zustande kommen.

Der konische Teil des Rohres sorgt für eine Anhebung und Annäherung der unteren Resonanzfrequenzen. Der Trichter verstärkt diesen Effekt aufgrund von verändertem Reflexionsverhalten am offenen Ende. Tiefe Frequenzen bzw. größere Wellenlängen werden fast vollständig reflektiert und erfahren eine effektiv kürzere Rohrlänge, abhängig von den Abmessungen des Trichters. Für kurze Wellenlängen ist die effektive Rohrlänge größer als für lange Wellenlängen, wodurch diese Frequenzen etwas herabgesetzt werden (siehe *Abbildung 1* auf *S.21*). Hohe Frequenzen treten außerdem aufgrund der für sie großen Öffnung leichter durch den Trichter hindurch und werden vermehrt abgestrahlt, was die Klangfarbe, Lautstärke und die Richtcharakteristik der Trompete bei hohen Frequenzen erklärt. Außerdem ist durch die Schallstürze eine stärkere Abstrahlung der Schallenergie an

den Raum möglich. Das Mundstück mit seiner Höhlung und dadurch eingeschlossenen Volumen, senkt vor allem die hohen Resonanzfrequenzen, für die die effektive Rohrlänge ansteigt [23].

Die oben beschriebenen Effekte sorgen gemeinsam dafür, dass die tatsächlichen Resonanzfrequenzen der Trompete genau denen einer harmonischen Reihe, jedoch ohne die Grundfrequenz, entsprechen, also im Verhältnis 2:3:4:5 etc. zueinander stehen (siehe *Abbildung 2)*.



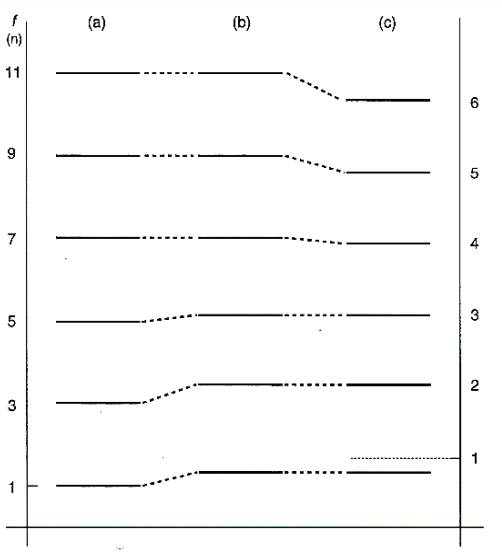


Abbildung 1 (links): *Vereinfachtes Modell einer Trompete mit stehenden Wellen und deren unterschiedliche effektive Rohrlängen*

Abbildung 2 (rechts): ***(a)** Natürliche Eigenschwingungsfrequenzen einer zylindrischen, halboffenen Röhre **(b)** Erhöhung der Frequenzen bei niedrigeren Eigenschwingungszuständen aufgrund von konischem Teil und Schallstürze **(c)** Erniedrigung bei höheren Eigenschwingungszuständen durch das Mundstück (die gestrichelte Linie bei 1 zeigt die gedachte Grundtonfrequenz für die neu entstandene harmonische Reihe)*

(Abb.1 & 2 aus Donald E. Hall - „Musikalische Akustik“ - S.278 [24]*)*

4.1.2 Bedienung

In folgender Abbildung ist die mit einer C-Trompete spielbare Naturtonreihe zu sehen, wobei das tiefste c als Basis der Naturtonreihe nicht dazu gehört, sondern nur der Vollständigkeit halber aufgeführt ist. Der Bläser erreicht durch das Variieren der Lippenspannung die höheren (bzw. tieferen) Naturtöne dieser Reihe, was als sogenanntes Überblasen bekannt ist.

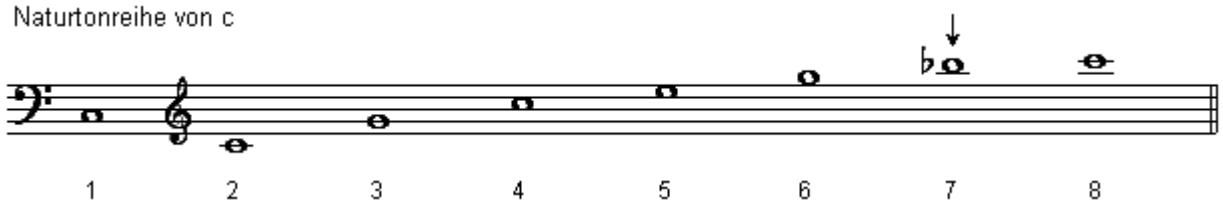


Abbildung 3: Naturtonreihe von c (Das tiefste c ist jedoch keine Resonanzfrequenz der Trompete) (Abbildung aus [25]*)*

Diese Naturtonreihe, erzeugt durch die Resonanzfrequenzen des Instruments, ist die Basis der Tonerzeugung, wobei durch das Betätigen der Ventile der jeweilige Naturton in kleinen und großen Schritten vermindert werden kann. Der Tonumfang erstreckt sich somit bei der C-Trompete vom kleinen fis (185 Hz), erreichbar durch das Verringern des ersten Naturtones mit Hilfe aller drei Ventile, bis zum c''' (dreigestrichenes c, 1046 Hz), wobei geübte Trompeter den Tonumfang sowohl nach unten als auch nach oben erweitern können. Das untere c kann als Pedalton mit etwas Übung erzeugt werden, wobei er keiner Resonanzfrequenz im Instrument entspricht, jedoch die Obertonstruktur durch die höheren Resonanzfrequenzen repräsentiert wird.

Um neben den Naturtönen alle dazwischenliegenden Noten spielen zu können, werden die Ventile der Trompete zum Verlängern der effektiven Rohrlänge verwendet. Da das größte Intervall, zwischen dem zwei Naturtöne chromatisch überbrückt werden müssen, eine reine Quint ist, werden mindestens drei zuschaltbare, unterschiedlich lange Rohrstücke benötigt. Damit sind die maximal 6 Töne zwischen den Naturtönen spielbar. Die drei Ventile bei Bb- und C-Trompete haben folgende Funktionen (Benennung vom Spieler aus Richtung Schallstürze):

- Ventil 1: Absenkung um einen Ganzton
- Ventil 2: Absenkung um einen Halbton
- Ventil 3: Absenkung um eine kleine Terz (bzw. drei Halbtöne)

Es werden sieben Griffmöglichkeiten verwendet um die Naturtöne zu vermindern. Dabei wird üblicherweise für das Absenken der Naturtöne um eine kleine Terz nicht das dritte Ventil, sondern das erste und zweite Ventil gemeinsam verwendet. Des Weiteren klingt der in *Abbildung 3* mit dem Pfeil markierte Ton, das b''', etwas tiefer als notiert, weshalb er in der

Regel nicht über den eigenen Naturton angespielt wird, sondern über den nächsten, das c''', bei Verringerung um einen Ganzton durch das erste Ventil. Dieser Umstand ist bei der Umsetzung der Naturtonreihe in dieser App entgegenkommend.

4.2 Realisierung in der Simulation

4.2.1 Tonerzeugung

Die erste Frage, die sich bei der Planung eines Software-Instruments stellt, ist die Art der Klangerzeugung. Nach längeren Überlegungen und den Versuchen durch eine aufwendige Klangsynthese einen stimmigen Trompetenklang zu synthetisieren, fiel die Entscheidung, die Tonerzeugung über selbst eingespielte Samples zu gestalten. Dies dürfte aufgrund der mittlerweile zur Verfügung stehenden Speicherkapazitäten auf mobilen Geräten kein Problem darstellen. Da die Bb-Trompete vor der C- und Piccolo-Trompete die populärste Gattung darstellt, diente eine solche als Grundlage, mit der der oben beschriebene Tonumfang aufgenommen wurde. Dabei war allerdings zu beachten, dass der tatsächliche Tonumfang aufgrund der Stimmung des Instruments nun um einen Ganzton nach unten verschoben wurde.

Die Erzeugung des Klanges geschieht also durch das Abspielen von Samples, wobei diese flexibel in der Länge und im Ein- und Ausschwingverhalten sein müssen. Die Dauer des einzelnen Tones sollte, wie gewohnt, von der Dauer der Luftzufuhr abhängig sein und muss von kurzem Anblasen bis zu theoretisch unendlicher Länge (vergleichbar mit Zirkularatmung bei Bläsern) variabel sein. Um dies zu bewerkstelligen, wurde folgende Methode beim Sampling der Klänge verwendet:

Jeder Ton, der mit dem *Trumpet-Simulator* spielbar sein sollte, wurde aufgenommen und die (unkomprimierte) *.wav*-Datei in einen eigenen Sampler-Patch in Pure Data importiert. Vorher wurden die Soundfiles normalisiert, damit die verschiedenen Töne später keine ungewollten, größeren Lautstärkeunterschiede aufwiesen. Ein Teil dieses Patches ist das folgende Feld bzw. Array, in welches alle Samples einer Audiodatei gespeichert werden (*Abbildung 4, S.24*).

In diesem Fall handelte es sich um 90672 gespeicherte Werte, was bei der Datenrate von 44100 Hz einer Länge von ca. 2,06 Sekunden entspricht. Bei der Wahl der Länge der Audiodateien waren zwei Punkte zu berücksichtigen. Einerseits sollten diese kurz genug sein, um den benötigten Speicherplatz zu minimieren, andererseits sollte auch eine größere und flexible Dauer der Klänge erreichbar sein, was wiederum für längere Aufnahmen sprach. Dies wäre profitabel, um den Klang bei längerer Wiedergabe seltener wiederholen bzw. „loopen“ zu müssen.

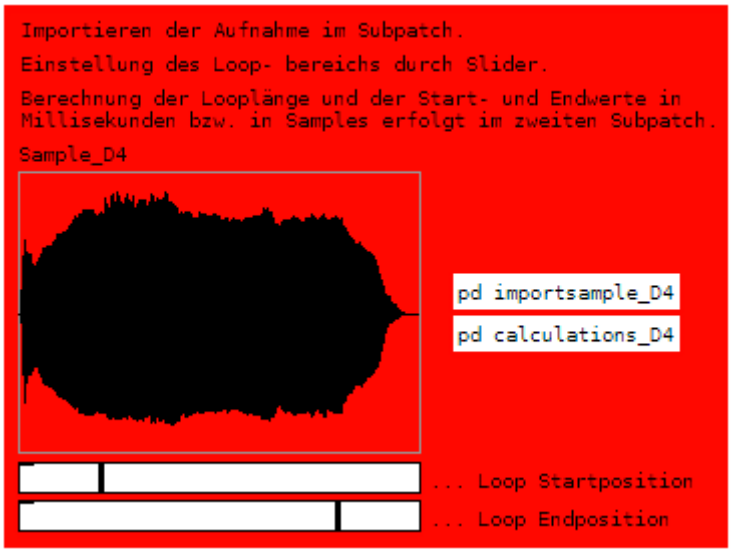


Abbildung 4: *Mit Punkten dargestellte Wellenform der Audiodatei für den Ton D4 der Bb-Trompete (klingend C4, MIDI-Notenbezeichnung) inklusive Loop-Auswahl und zusätzlichem Berechnungspatch aus einem Sampler-Patch*

In obiger Abbildung ist ein wichtiger Teil der Funktionsweise dieses Samplers zu erkennen. Die beiden Schieberegler dienen hier als visuelles Werkzeug, um einen gewissen Bereich der Audiodatei für die Verwendung als Loop einzugrenzen. Dafür eignet sich der Teil des Klanges am besten, welcher annähernd gleichmäßig laut ist, um beim späteren Loopen sich periodisch wiederholende Lautstärkeschwankungen bzw. Artefakte zu vermeiden. Des Weiteren sollte dieser Teil möglichst lang sein um einen natürlicheren Dauerton zu erzeugen.

Der mittlere Teil der Audiodateien wurde direkt nach der Aufnahme bereits durch eine Lautstärkenautomation in der Aufnahmesoftware möglichst konstant gestaltet. Jedoch sind die beweglichen Auswahlwerkzeuge, die bei jedem Ton der App den genauen Zeitbereich zum Loopen eingrenzen können wichtig, da jeder Ton einen etwas anderen Ein- und Ausschwingvorgang besitzt und die Längen der einzelnen Aufnahmen natürlich etwas variieren. Während für den jeweiligen Ton dieser Bereich ausgewählt werden kann, werden die Anfangs- und Endzeiten des Loopbereichs in Millisekunden und Samples durch Nebenrechnungen im *calculations* Subpatch (siehe *Abbildung 4*) ermittelt. Diese werden für die spätere Steuerung des Loops und verschiedener Einhüllender („Envelopes“) benötigt.

Das grundlegende Prinzip für eine flexible Tonlänge im Sampler ist in *Abbildung 5* auf *Seite 25* dargestellt. Dieser Fall beschreibt den Start eines Tons nach Stille bzw. beim konkreten Anblasen eines Tons, ohne, dass das Instrument unmittelbar vorher erklingt. Nur dann erfolgt der Einschwingvorgang wie er am Anfang von *Abb.5 (a)* und *(d)* zu erkennen ist. Der Fall, bei dem zwischen Tönen während des Blasens gebunden wird, wird später aufgegriffen.

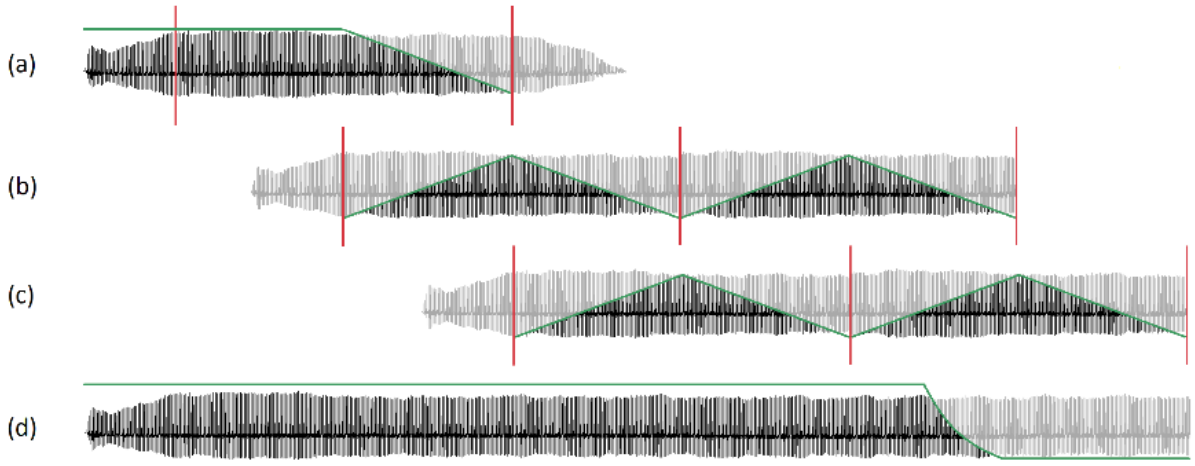


Abbildung 5: *Grundlegendes Prinzip des Wiederholens und Überlappens der Loops mit markiertem Loop-Bereich (senkrechte Striche) und eingezeichneter Einhüllenden* ***(a)*** *Start des Klangs mit normalem Einschwingvorgang und Ausblenden durch Einhüllende* ***(b)*** *Loop 1 mit Einblenden während des Startklangs und Ausblenden bei Beginn von Loop 2* ***(c)*** *Loop 2, welcher mit Loop 1 abwechselnd ein- und ausgeblendet wird* ***(d)*** *resultierender Klang als Summe der ersten drei, welcher am Ende durch eine weitere Einhüllende beendet wird*

Die Einhüllende, welche in *Abbildung 5 (d)* den Ton beendet, kann natürlich bereits früher (oder später) eintreten. Sie wird durch die Beendigung des Blasens ausgelöst, wobei im Fall eines kurzen Tones z.B. Teil *(b)* und *(c)* gar nicht ausgelöst werden. Außerdem wird unmittelbar nach dem Ausklingen des Tons die Wiedergabe aller beteiligten Sequenzen beendet, was in obiger Abbildung nicht zu erkennen ist, damit keine Probleme bei rasch folgendem nochmaligem Anspielen desselben Tons auftreten.

Die eingezeichneten Einhüllenden sind schematische Darstellungen für die Funktionsweise des Samplers. Im Programm sind bei jedem Sampler-Patch eigene Envelopes vorhanden, die andere Kurvenformen besitzen als in der Abbildung. Die Kurvenformen sind, wie die Audiodaten, als Array gespeichert und werden zu gegebener Zeit ausgelesen. Diese Arrays bzw. die genauen Verläufe der Kurven können in den im Patch angezeigten Feldern gezeichnet werden. Somit können sie per Hand modifiziert werden, um die Übergange der Loops für die Aufnahmen, welche sich in Lautstärke und Lautstärkenverlauf etwas unterscheiden, zu optimieren bzw. unauffälliger zu gestalten. Außerdem können dabei für die Abklingkurven leichter spezielle Kurvenformen durch Probieren erreicht werden. Die Abklingkurve der Einhüllenden bei *Abbildung 5 (d)* kann in den Sampler Patches gezeichnet werden und wird sowohl für das Ausblenden bei Beenden des Spielens als auch beim Binden zum Ausklingen, mit unterschiedlichen Auslesezeiten, verwendet. Die Dauer des Ausklingvorgangs der Trompetentöne, welche für alle Töne sehr kurz ist und als „Abreißen des Klangs“ charakterisiert werden kann *(M. Dickreiter - „Handbuch der Tonstudiotechnik – Band 1“ - S.67* [26]*)*, wurde in der globalen Einhüllenden für das Beenden aller Töne auf 80ms festgelegt.

Für das Abspielen der einzelnen Abschnitte mit Pure Data wird das *tabplay~* Objekt verwendet. Bei diesem kann über eine *Liste* innerhalb einer *Message*, das Array, in dem die

Audiodaten gespeichert sind, ausgelesen werden. Die Liste besteht dabei aus zwei Zahlen (*floats*), welche den Startpunkt in Samples und die Auslesedauer in Samples bestimmen. Diese Funktion ist gut geeignet, um die später startenden Loops punktgenau vom Startsample innerhalb des Arrays zum anderen Sample, ebenfalls innerhalb des Arrays abzuspielen. Die genauen Werte werden durch die auf *S.14* erwähnte Einstellung der Loopbereiche und anschließender Berechnung geliefert. Das Auslesen geschieht automatisch mit 44100 Samples pro Sekunde, während am Ausgang des Objekts das Audiosignal anliegt.

Klangabschnitt 1:

In folgendem (vereinfachten) Codeausschnitt ist das Grundprinzip für das Abspielen des in *Abb. 5* zu erkennenden ersten Klangabschnitts dargestellt. Als Beispiel ist hier ein Ausschnitt des Sampler-Patches der Note D4 (der Bb-Trompete) gezeigt, welche einem klingenden C4 in MIDI-Notenbezeichnung entspricht.

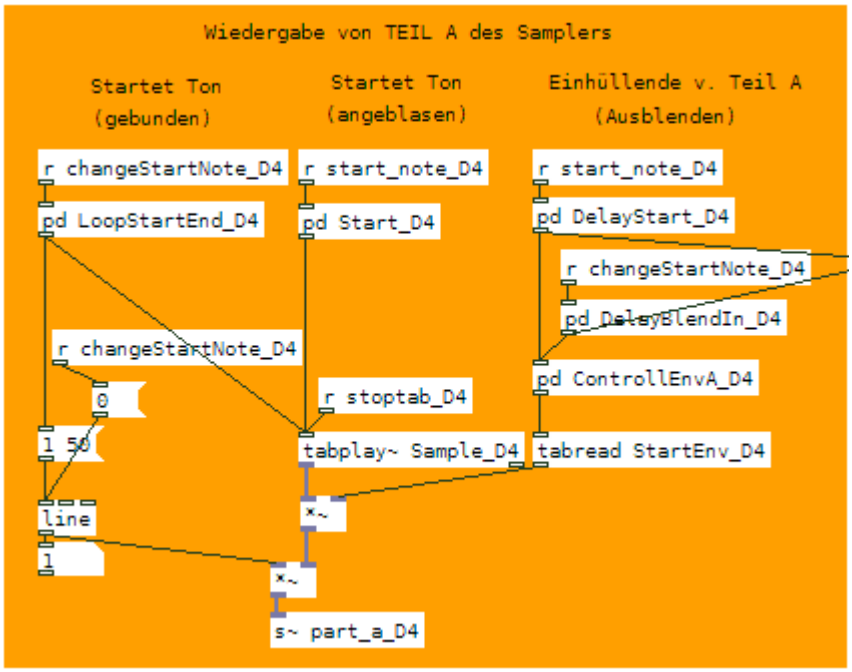


Abbildung 6: *Teil (a) (vgl. Abb. 5) des Sampler-Patches, welcher beim Anblasen ausgelöst wird*

Das Array mit den Audiodaten heißt hier *Sample_D4.* Sobald über das Symbol *start_note_D4* ein *Bang* (vergleichbar mit einem Mausklick) empfangen wird, wird eine im Subpatch generierte *Message* an das *tabplay~* Objekt gesendet und das Auslesen des Arrays wird mit den gewünschten Parametern gestartet. Der erste Abschnitt wird, wie in *Abb.5* zu erkennen, vom Anfang der Aufnahme bis zum Ende des Loopbereichs ausgelesen (angeblasen).

Die Einhüllende, welche den Übergang zwischen dem ersten und zweiten Teil des Klangs (bzw. *(a)* und *(b)* in *Abb. 5*) einleitet, wird über die Objekte im rechten Codeabschnitt

gesteuert (*StartEnv_D4)*. Da die erste Einhüllende erst zum Zeitpunkt des halben Loopbereichs ausgelesen werden soll, wird die *Bang-Message*, die durch das Starten der Note ausgelöst wird, mit genau dieser Verzögerung weitergeleitet. Wenn dies geschieht, wird über die nächste *Message* eine Rampe zum Auslesen des Arrays mit der Einhüllenden generiert. Die 1000 Werte, welche alle Einhüllenden in diesem Code besitzen, werden innerhalb der halben Loopdauer ausgelesen und somit der erste Abschnitt des Klangs ausgeblendet.

Wird das Spielen dieses Tons bereits vorher beendet, wird durch *stoptab_D4*, welches nach komplettem Abklingen des Tons ein *Bang* empfängt, das Auslesen vorzeitig beendet. Außerdem wird in diesem Fall die laufende Verzögerung unterbrochen, da die Einhüllende überflüssig ist bzw. zu Problemen bei raschem Wiederholen des Tons führen kann.

Klangabschnitt 2:

Zu dem Zeitpunkt, bei dem die erste Einhüllende einsetzt um den ersten Teil des Klangs auszublenden, setzt der zweite Teil ein. Dieser beginnt beim ausgewählten Zeitpunkt für den Beginn des Loopbereichs. In folgender Abbildung ist die Implementierung des zweiten Klangabschnitts dargestellt.

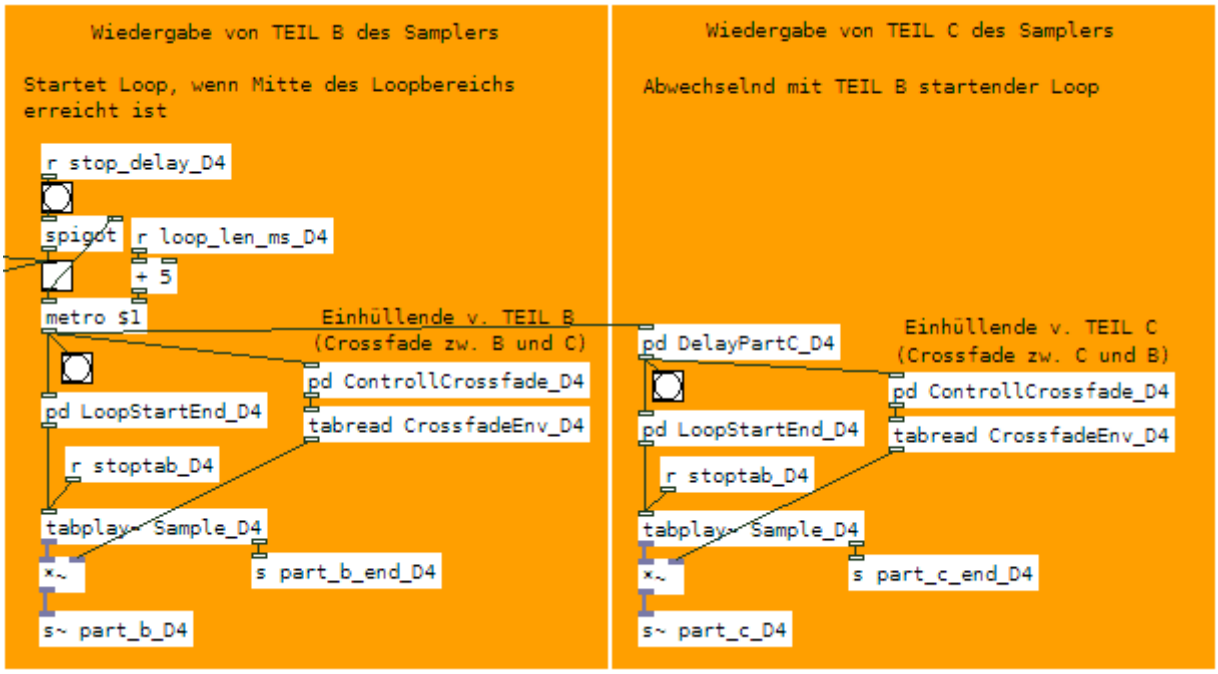


Abbildung 7: *Teil (b) bzw. (c) (vgl. Abb. 5) des Sampler-Patches, welcher das Looping steuert*

Die Funktionsweise ist ähnlich wie bei Abschnitt 1. Allerdings wird hier bereits die *Bang-Message* für das Starten des Auslesevorgangs von *tabplay~* verzögert, da dieser erst gestartet werden soll, wenn der erste Teil langsam ausgeblendet wird. Wenn dies der Fall ist, sendet das Objekt *metro* im Abstand der Looplänge stetig *Bangs* und startet somit periodisch das Abspielen des Loops. Gleichzeitig löst es das Auslesen der Einhüllenden namens *CrossfadeEnv_D4* aus, welche ähnlich wie in *Abb. 5 (b)* bis zur Hälfte des Abschnitts an- und danach absteigt.

Wie bei Abschnitt 1 muss bei (frühzeitiger) Beendigung des Tons das Auslesen durch das Objekt *tabplay~ mit* einer *Stop-Message* (nach Abwarten des Abklingvorgangs, 80ms) abgebrochen und die Verzögerung unterbrochen werden. Außerdem wird, wie jedes Mal bei Erreichen des Loopendes, der Auslesevorgang der Einhüllenden zurückgesetzt. Anders als am Anfang muss hier jedoch noch das *metro* Objekt abgeschaltet werden.

Klangabschnitt 3:

Der zweite und dritte Abschnitt, welche das Wiederholen und Überlappen des mittleren Bereichs des Arrays bilden, sind im Prinzip gleich aufgebaut.

Die Steuerung des letztens Teils des Klangs (der überlappende Loop, vgl. *Abb.5 (c)*), erfolgt analog zu Teil 2. Der gleiche Code wie in *Abbildung 7* wird für das Abspielen des Klangs und das Auslesen der Einhüllenden verwendet, wobei sich ausschließlich die Verzögerungszeiten unterscheiden. Damit erreicht man das abwechselnde, überlappende Abspielen des Loopbereichs und die in *Abb. 5* schematisch dargestellte Klangzusammensetzung.

Die oben beschriebene Funktionsweise deckt jedoch nur den Fall ab, bei dem eine Note direkt angespielt bzw. angeblasen wird und dem der, für die Trompete übliche, impulsartige Einschwingvorgang mit geräuschhaften und hochfrequenten Anteilen vorangeht (Anblasgeräusch). Der andere Fall, bei dem ein bereits stabiler Klang durch das Betätigen der Ventile oder das Variieren der Lippenspannung in seiner Frequenz verändert wird, ist aber ebenfalls essentiell. Hierbei tritt nicht der wie in *Abb. 5 (a)* am Anfang zu erkennende Einschwingvorgang auf. Die Wellenlänge der stehenden Welle im Instrument wird stattdessen rasch entsprechend der Ventilstellung und Lippenspannung auf die neue Wellenlänge verkürzt oder verlängert. Dies geschieht im Prinzip übergangslos, da, wie in ***4.1.1.*** beschrieben, die Resonanzfrequenzen des Rohrapparats verändert werden und sich nur diese zu einer stabilen Stehwelle ausbreiten können. Da es viel Übung erfordert die gewünschte Lippenspannung gezielt und schnell zu erreichen, können in der Praxis bei diesem Wechsel beim Überblasen über mehrere Naturtöne zwar Zwischentöne auftreten, diese sind jedoch im Normalfall nicht erwünscht.

Um diesen Fall zu implementieren benötigt man also einen unmittelbaren Wechsel zwischen zwei Tönen bzw. zwei Klangsamples mit kurzem Übergang, ohne Anblasgeräusch und schneller Stabilisation.

Dafür wurde im oben beschriebenen Sampler-Patch dieser Fall durch separate *Start-* und *Stop-Messages* realisiert, welche beim Wechsel zum neuen Ton hin bzw. vom vorangehenden Ton weg ausgelöst werden. Diese sind Indikatoren für den neuen Fall und lösen die drei Teile des Samplers in anderer Weise aus als beim Anblasen oder vollständigen Beenden.

Der Unterschied liegt hauptsächlich beim ersten Klangabschnitt (vgl. *Abb. 5 (a)*). Dieser startet dieses Mal nicht vom Beginn des aufgenommen Klangs, sondern vom Beginn des Loopbereichs. Damit wird das Anblasgeräusch vermieden und gleichzeitig eine schnelle Stabilisation der Frequenz gewährleistet. Dieser Klangabschnitt startet jedoch nicht sofort, sondern wird, auch zur Vermeidung von Knackgeräuschen, durch eine kurze Einhüllende eingeblendet. Diese Attack-Time, bevor der Klang die maximale Lautstärke erreicht, wurde auf 50 ms festgelegt. Gleichzeitig muss natürlich auch das Ausklingen des vorangehenden Tons kürzer gestaltet werden, weshalb in diesem Fall eine halb so große Release-Time wie

bei Fall 1 benutzt wird (40 ms). Weitere Änderungen bei Auslösen eines Tons durch das Binden zwischen Noten sind die unterschiedlichen Verzögerungszeiten der drei Klangabschnitte, damit bei verändertem ersten Abschnitt die beiden Loops, welche wie gewohnt alternierend den statischen Klang bilden, zur richtigen Zeit einsetzen.

Mit den oben erwähnten Funktionen ist es möglich, eine flexible Tonerzeugung mit kurzen Tonlängen, welche nur einem Anblasen entsprechen, bis zu Tönen uneingeschränkter Dauer zu erzeugen. Außerdem wird auf die unterschiedlichen Prozesse des Ein- und Ausklingens beim Anblasen und beim Binden der Töne Rücksicht genommen.

4.2.2 Bedienung

Der in ***4.2.1*** beschriebene Sampler-Patch, welcher für alle Töne vorhanden ist, wird von den folgenden vier *Messages* und einer Variablen kontrolliert:

- *start_note*
- *changeStartNote*
- *end_note*
- *change_note*
- *midi_note*

Die ersten beiden *Messages* bilden den Auslöser für das Starten des Klangs mit oder ohne Anblasgeräusch. Die dritte und vierte *Message* sorgen für das Beenden des jeweiligen Tons mit normaler oder kurzer Release-Time. Alle vier sind sogenannte *Bangs*, die im jeweiligen Patch die verschiedenen Abspiel- und Auslesevorgänge veranlassen. Die Messages *changeStartNote* und *change_note* sind dabei lediglich Indikatoren für die Änderung der Variablen *midi_note* und decken den Fall des Bindens der Töne ab. Da die Übergabe an die Patches vorwiegend über sogenannte *Sends* abläuft, also innerhalb des Programms ohne direkte, sichtbare Verbindung gesendet und empfangen wird, werden die Symbole bzw. die Bezeichnungen, nach einem Abgleich weiter spezifiziert, damit nur der gewünschte Sampler ausgelöst wird. Dieser Abgleich erfolgt mit der Variablen *midi_note*. Diese wird ebenfalls an das Patch gesendet und durch einen logischen Operator zum Sperren oder Weiterleiten der *Bang-Messages* verwendet, welche dann spezifiziert werden (z.B. *start_note* → *start_note_D4* wenn logische Operation *„wahr“* ist, d.h. *midi_note* übereinstimmt mit der MIDI-Note von D4).

Diese Kontrollelemente werden durch folgende drei Steuerprinzipien, welche die Schnittstelle zwischen Benutzer und Software bilden, beeinflusst.

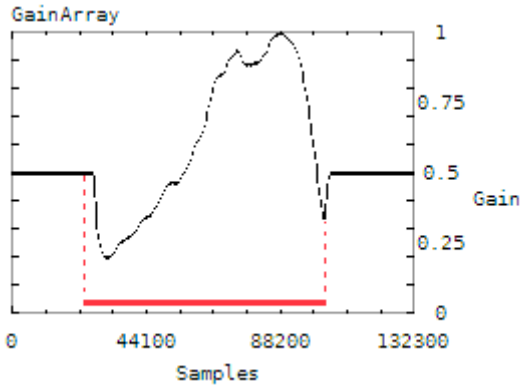
1. Schalldruck am Mikrophon

Zur Tonerzeugung bei der App wird, wie bei einer herkömmlichen Trompete, dem Instrument Luft zugeführt. In diesem Fall wird der am Mikrophon des Geräts anliegende Schalldruckpegel über den Analog-Digital-Umsetzer digitalisiert und einen *Envelope Follower* gemittelt. Dabei wird das quadratische Mittel (*RMS – root mean square*) über ein einstellbares Zeitfenster bzw. eine bestimmte Anzahl von Samples gebildet, wobei eine 1 (Vollaussteuerung) 100dB entspricht. Um ein möglichst reaktionsschnelles System herzustellen wurde die Blockgröße auf 512 Samples eingestellt. Das bedeutet es werden 512 Samples für die Berechnung herangezogen, bevor ein neues Ergebnis zur Verfügung steht. Das entspricht einer Verzögerung von $\frac{512}{44100\frac{1}{s}} \approx 0{,}0116s$ bzw. $11{,}6ms$ von vorherigem zu neuem Wert. Bei weiterer Verkleinerung der Blockgröße schwankten die Werte zu stark und es kam zu Problemen bei folgender Verarbeitung:

Der über die kurze Zeit gemittelte Pegel wird dem Objekt *threshold~* zugeführt, welches den Strom an Zahlen empfängt und ein *Bang* am linken Ausgang ausgibt, sobald ein gewisser Schwellwert überschritten ist. Diese *Bang-Message* wird per oben erwähntem *start_note* versendet und veranlasst den Start der aktuellen Note im jeweiligen Sampler-Patch. Die Beendigung des Blasens wird durch eine Unterschreitung eines zweiten Schwellwerts detektiert. Das Objekt *threshold~* gibt dabei ebenfalls eine *Bang-Message* am rechten Ausgang aus, sobald der untere Schwellwert unterschritten wird. Somit kann eine Art Hysterese genutzt werden, wobei ein größerer Schalldruck zum Anblasen eines Tons überschritten und ein kleinerer Schalldruck zum Beenden eines Tones unterschritten werden muss. Die Differenz der Schwellwerte wurde experimentell auf 10 dB festgelegt. Dieser Umstand ist günstig, da, auch aufgrund der Empfindlichkeit durch recht kleine Blockgrößen bei der Pegelmittelung, das System sonst zu instabil wäre und in ein unkontrollierbares Auslösen der Töne resultieren würde. Außerdem ähnelt diese Funktionsweise dem physikalischen System, bei dem ebenfalls eine etwas stärkere Luftzufuhr zum Anblasen und Stabilisieren der Stehwelle als zum Aufrechterhalten der Töne nötig ist (zumindest bei leisem Spielen der Fall).

Für das Auslösen der Töne wurde also das oben beschriebene quadratische Mittel direkt verwendet. Zur Steuerung der Lautstärke des Tons nach erfolgreichem Anblasen musste das Signal jedoch weiter verarbeitet werden. Im Patch *GainCalc* wird das vom ADC kommende Signal zuerst bei 5000Hz hochpassgefiltert. Dies geschieht aus dem Grund, da durch Aufnahme und Analyse verschieden starken Blasens bzw. Rauschens am Mikrophon des verwendeten Smartphones festgestellt werden konnte, dass bei leiserem Blasen weniger hochfrequente Anteile vorhanden sind, als bei lauterem. Somit ist der nachfolgend errechnete Pegel nach der Filterung des Signals besser zum Erfassen der Stärke des Blasens geeignet. Nach der Berechnung des Pegels in gleicher Weise wie oben beschrieben (Envelope Follower), wird der resultierende Zahlenstrom an Pegelwerten nochmals in ein Signal umgewandelt, um eine Tiefpassfilterung durchzuführen. Durch die Glättung bleiben nur noch die langsameren Lautstärkeschwankungen übrig, welche nach der Umwandlung in die Datenebene zur Berechnung der variablen Lautstärke verwendet werden können. Die Idee dabei ist eine konstante Verstärkung von 0,5 mit der auch der anfängliche Teil, das Anblasgeräusch der Töne, verstärkt wird, welche jedoch direkt nach dem Anblasgeräusch variiert werden kann. Somit wird bei leisem Blasen diese Verstärkung verringert bzw. bei lauterem erhöht, sodass der Spieler kann eine gewisse Dynamik nutzen kann. Diese Dynamik ist, wie beim tatsächlichen Spielen auf der Trompete, nach unten hin

natürlich durch die untere Schwelle zum Beenden des Tons begrenzt. Mit diesem Prinzip entsteht die in folgender Abbildung dargestellte, vom Spieler nutzbare, Variation der Lautstärke.



Graph 1: *Aufnahme der Verstärkung des Tons durch Variation des Blasens. Ein ca. 1,85 s langer Ton wird mit Verstärkung von 0,5 angeblasen, sodass das Anblasgeräusch hörbar ist, dann wird die Verstärkung in Abhängigkeit des Blasens verändert und nach dem Beenden wieder zurückgesetzt.*

Wie im zugrunde liegenden physikalischen System bedingt somit eine Luftzufuhr sowohl die Existenz eines Klangs als auch dessen Lautstärke.

Die beiden folgenden Steuerprinzipien sind beide nicht für das Erzeugen der Töne oder die Dynamik verantwortlich, sondern ausschließlich für das genaue Regulieren der Tonhöhe.

2. Virtuelle Ventile

Um die Funktionsweise der Trompetenventile auf die App zu übertragen, wurden drei Touch-Felder auf der grafischen Oberfläche platziert. Diese entsprechen in Form, Anordnung und letztendlich auch in ihrer Funktion, oder genauer in ihrer Wirkung, den drei Ventilen der Trompete. Wie in Kapitel ***4.1.1.*** beschrieben, dienen diese durch eine Verlängerung der Rohrlänge zum Erniedrigen der Eigenfrequenzen des Instruments. Somit kann man im Original bei einer gewissen Lippenspannung, also bestimmter Tonhöhe, diese nach unten hin verändern. Anders gesagt kann bei Verwendung eines oder mehrerer Ventile und damit größerer Rohrlänge die nun auf einem tieferen Ton aufbauende, zur Verfügung stehende Naturtonreihe durch Überblasen genutzt werden.

Abbildung 8: *Vereinfachte grafische Oberfläche bei offenem Ventilgriff (links) und Drücken des zweiten Ventils (rechts). Am oberen Rand des Displays ist die Anzeige des Neigungswinkels des Geräts zu erkennen. Sie dient als optisches Feedback, auf welcher Stufe der Naturtonreihe man sich befindet.*

Für die drei virtuellen Ventile wurde die auf der offiziellen *PdDroidParty* [2] Website verfügbare Abstraction *touch* verwendet. Diese besteht aus einem Feld gewünschter Größe, welche bei Berührung die Position des Fingers in X- und Y-Richtung sendet, wobei diese über die Kantenlängen des Feldes Werte von 0 bis 1 annehmen können. Wenn keine Berührung stattfindet, wird über beide Koordinaten der Wert -1 gesendet. Damit kann durch eine logische Operation der Zustand abgefragt werden, ob eine Berührung vorliegt oder nicht. Wenn dies der Fall ist, wird die logische 1 mit der Anzahl an Halbtönen, um welche das Ventil den Ton verringern soll, multipliziert und somit eine Neuberechnung der aktuellen MIDI-Note ausgelöst. Diese steuert, wie oben beschrieben, das Starten der richtigen Note

oder, wenn sie während des Spielens verändert wird, den Wechsel zwischen vorheriger und aktueller Note.

Um dem Nutzer eine Art haptischen Feedbacks zu geben, wird bei der Berührung eines virtuellen Ventils eine kurze Vibration des Geräts ausgelöst. Hierbei wird die Länge der Vibration an das System gesendet und die Aktion somit veranlasst. Durch diese Rückwirkung auf das Drücken eines Ventils wird das Spielen etwas natürlicher, da die Information zurückgegeben wird, ob die Finger richtig liegen und das Drücken funktioniert. Außerdem findet eine Art physikalischer Reaktion statt, was bei der Benutzung des Displays normalerweise nicht der Fall ist.

3. Überblasen durch Neigung des Geräts

Das Starten und Beenden der Töne, sowie die Veränderung der Tonhöhe durch die Ventile, sind mit oben beschriebenen Mitteln nun möglich. Ein elementarer Teil der Funktionsweise steht jedoch noch aus: das Überblasen.

Mit dem Überblasen lässt sich die auf der Rohrlänge basierende und somit auf einem gewissen Ton aufbauende Naturtonreihe, gemäß *Abbildung 2* & *3* auf den *Seiten 21 und 22*, ausnutzen. Um einen höheren Ton der Reihe zu erreichen, muss dabei im zugrundeliegenden physikalischen System die Lippenspannung erhöht werden, während man durch eine Entspannung der Lippen wieder auf die darunter liegenden Töne springen kann. Dieser Zusammenhang kann nicht, wie die Funktionsweise der Ventile, einfach auf die App übertragen werden. Es muss eine andersartige, eigene Art der Steuerung geschaffen werden, welche die Wirkung der Lippenspannung nachahmt.

Wie bereits bei den *Folgerungen aus der Recherche* (***3.1.2.***) beschrieben, ist dabei eine elegante Lösung, die Lippenspannung durch den Neigungswinkel des Geräts zu substituieren. Hierbei erscheint es logisch, die niedrigere Lippenspannung dem kleineren Neigungswinkel und die höhere Lippenspannung dem größeren Neigungswinkel zuzuordnen. Dies korreliert außerdem mit der Tonhöhe, welche bei höherer Lippenspannung ebenfalls ansteigt.

Um den angesprochenen Neigungswinkel für die weitere Verwendung im Code zu erfassen, werden die Beschleunigungssensoren genutzt, die bei Smartphones und Tablets mittlerweile zur Standardausstattung gehören.

Exkurs Beschleunigungssensoren:
Beschleunigungssensoren nutzen unterschiedliche Messprinzipien für die Erfassung der auf sie wirkenden Beschleunigung. Dabei basieren sie jedoch meist auf dem gleichen physikalischen Prinzip.
Das erste newtonsche Axiom lautet: *„Ein Körper verharrt im Zustand der Ruhe oder der gleichförmigen Translation, sofern er nicht durch einwirkende Kräfte zur Änderung seines Zustands gezwungen wird.“* Dies wird auch als Trägheitsgesetz bezeichnet. Das zweite newtonsche Axiom besagt: *„Die Änderung der Bewegung ist der Einwirkung der bewegenden Kraft proportional und geschieht nach Richtung derjenigen geraden Linie, nach welcher jene Kraft wirkt“* [27]. Von Euler wurde dieses Prinzip später in die mathematische Form $\vec{F} = m \cdot \vec{a}$ übertragen, wobei $\vec{a}$ dabei den Beschleunigungsvektor bezeichnet, welcher der im zweiten Axiom beschriebenen Änderung der Bewegung in Richtung der Kraft entspricht.
In den Beschleunigungssensoren wird eine beweglich gelagerte sogenannte seismische

Masse bei Beschleunigung in eine Richtung, aufgrund ihrer Trägheit, gegenüber der unbeweglichen Teile des Sensors in die entgegengesetzte Richtung ausgelenkt. Die Sensoren in Smartphones nutzen dabei den festen und beweglich gelagerten Teil (seismische Masse) als Kondensatorelektroden. Wenn eine Auslenkung aufgrund einer Krafteinwirkung auftritt, verändern sich die Abstände zwischen den Elektroden. Somit ändert sich die Kapazität und bei konstanter, auf dem Kondensator befindlicher Ladung die Spannung. Diese wird erfasst und es kann die Richtung und Größe der Auslenkung bzw. der zu ihr proportionalen Beschleunigung berechnet werden. Bei den in Smartphones befindlichen MEMS (Micro-Electro-Mechanical System) Bausteinen, wird dieses elektromechanische Prinzip in Kleinstbauweise für alle drei Raumrichtungen genutzt, wobei die Auswertung in einem integrierten Schaltkreis stattfindet. Mit Hilfe dieses miniaturisierten Bauteils kann somit die Beschleunigung in allen modernen Smartphones, Tablets, aber auch anderen elektronischen Geräten, durch Kombination dieser 3 Richtungen in alle Richtungen gemessen werden [28] [29].

Um den Neigungswinkel der Geräte zu bestimmen, wird eine Kraft gemessen, die ständig auf die Massen innerhalb der Sensoren wirkt, die Gravitationskraft. Sie sorgt für die sogenannte Erdbeschleunigung, welche, abhängig vom Ort, um den Wert 9,81 m/s² schwankt. Ausgehend davon, dass bei der Verwendung der App nur diese Kraft wirkt und keine vom Benutzer hervorgerufene Beschleunigung auftritt, kann über die Verteilung dieser Erdbeschleunigung auf die Y- und Z-Achse der Neigungswinkel bestimmt werden. Dabei ist anzumerken, dass die Gewichtskraft der seismischen Massen der Sensoren in Achsenrichtung gemessen wird und somit die Werte invertiert dargestellt werden, da die Massen durch die Kraft nicht, wie oben beschrieben, aufgrund ihrer Trägheit in die entgegengesetzte Richtung ausgelenkt werden, sondern in Richtung der Gravitationskraft.

Der Bezugspunkt bzw. Nullwinkel ist das flach liegende Gerät, wobei die Erdbeschleunigung in diesem Fall ausschließlich in Z-Achsenrichtung wirksam ist. Die seitliche Neigung des Geräts sollte bei der Nutzung minimal bleiben, sodass sich die Erdbeschleunigung bei einer Neigung vergleichbar mit *Abb. 9 (b)* nur auf die Y- und Z-Achse vektoriell aufteilt.

In *Abbildung 9 (b)* auf *Seite 35* ist die Aufteilung der Erdbeschleunigung bei einem Neigungswinkel von 45° ersichtlich. Anhand dieser Darstellung erfolgt hier eine Beispielberechnung des Winkels, wie sie im Programm bei Empfang bzw. Änderung der Werte für Y und Z abläuft.

$$y = 6{,}94\,\frac{m}{s^2} \qquad z = 6{,}94\,\frac{m}{s^2}$$

$$\theta = \arctan\left(\frac{y}{z}\right) = \arctan\left(\frac{6{,}94}{6{,}94}\right) = \arctan(1) \approx 0{,}785398rad$$

$$\theta = 0{,}785398 \cdot \frac{180°}{\pi} = 45°$$

Um den Neigungswinkel beim *Trumpet-Simulator* zum Überblasen zu nutzen, wird er, nach seiner Berechnung, mit den gewählten Winkelstellungen für die verschiedenen Naturtöne verglichen. Somit wird die Stufe der Naturtonreihe detektiert, auf welcher man sich momentan befindet. Gemäß Intervallzusammensetzung der Naturtonreihe, ausgehend vom

zweiten Naturton (da der Grundton nicht im Tonumfang auftritt), wird dann die MIDI-Note um das passende Intervall angehoben oder abgesenkt.

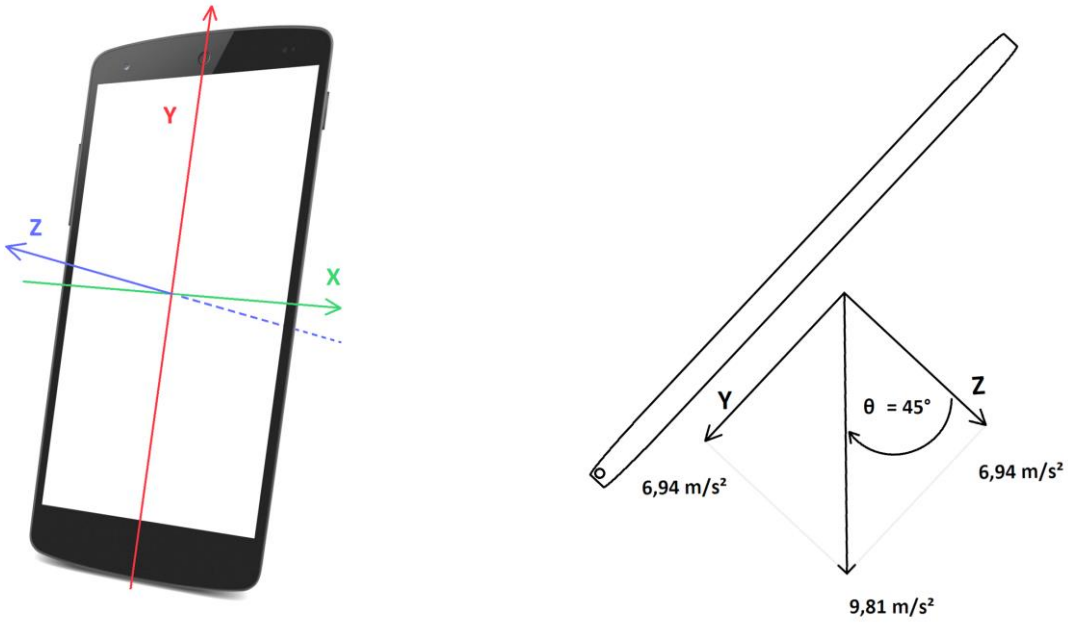


Abbildung 9: ***(a, links)*** *Einteilung der Achsen, in deren Richtungen die Beschleunigung gemessen wird* ***(b, rechts)*** *Aufteilung der Erdbeschleunigung auf zwei Achsen bei einem Neigungswinkel von 45° (Ausrichtung der Achsen wie oben beschrieben in entgegengesetzter Richtung als bei (a))*

Wie in *Abbildung 10* auf *Seite 36* zu erkennen, wurde dafür das Objekt *moses* verwendet, welches in diesem Fall die obere Schwelle des Winkelbereichs festlegt und den Eingangswert am linken Ausgang ausgibt, wenn er unter dem Nennwert liegt und am rechten Ausgang ausgibt, wenn er diesen überschreitet. Dabei wird der Wert am Ausgang für den jeweiligen Bereich nicht als solcher verwendet, sondern die Ausgabe an sich als *Bang* zum Auslösen der folgenden Berechnung herangezogen. Durch Reihenschaltung des *moses* Objekts sind somit die Winkelbereiche abgesteckt und die nach den Objekten befindlichen *Messages*, inklusive der gewünschten Intervallerhöhungen in Halbtönen, werden zum richtigen Zeitpunkt an die nachfolgende Berechnung weitergeleitet. Durch Beseitigung der Redundanz im Zahlenstrom geschieht dies nur, wenn ein, dem bestimmten Teilton zugeordneter, Winkelbereich verlassen bzw. betreten wird. Die MIDI-Note kann anschließend durch die Ventilstellung weiter modifiziert werden bevor sie zur Steuerung der Tonerzeugung bzw. Tonhöhenveränderung genutzt wird.

Um das Überblasen bzw. die Einschätzung des Winkels beim Spielen zu erleichtern, wurde das in *Abbildung 8* auf *S.32* am oberen Rand des Displays ersichtliche Feedback in Form einer horizontalen Winkelanzeige mit eingezeichneten Winkelbereichen für die Stufen der Naturtonreihe hinzugefügt. Dadurch hat der Nutzer jederzeit Erkenntnis darüber, auf welchem Teilton und wie weit entfernt er sich vom darüber oder darunter liegenden Teilton

befindet.

Der oberste Winkelbereich, der der 8. Stufe der Naturtonreihe entspricht (bei imaginärem Grundton der Reihe, welcher jedoch nicht spielbar ist), beinhaltet theoretisch den 7. und 8. Teilton. Der 7. Teilton wird nämlich als Naturton umgangen, da er wie in *Abb. 3* auf *S. 22* zu erkennen, etwas zu tief klingt und in der Praxis mit Hilfe des nächsten Teiltons bei Verringerung über die Ventile gespielt wird. Dies ist bei der Realisierung in der Simulation vorteilhaft, da somit ein Winkelsegment bzw. eine Stufe der Naturtonreihe eingespart wird und der Bereich für die oberen Töne ausreichend groß gewählt werden kann.

Im Codeausschnitt in *Abb. 10* befindet sich rechts ein Objekt namens *r changerange*, welches zum Begrenzen des Tonumfangs dient. Auf der grafischen Oberfläche der App lässt sich die Grenze dieses Tonumfangs bei g''' oder c''' wählen (siehe *Abb. 12* auf *S.38*), was durch oben genanntes Objekt empfangen wird. Damit kann der oberste Winkelbereich (der des c''') nach Wunsch umgangen werden, sodass durch Abriegelung beim g''' kein Abrutschen der Note nach oben geschehen kann.

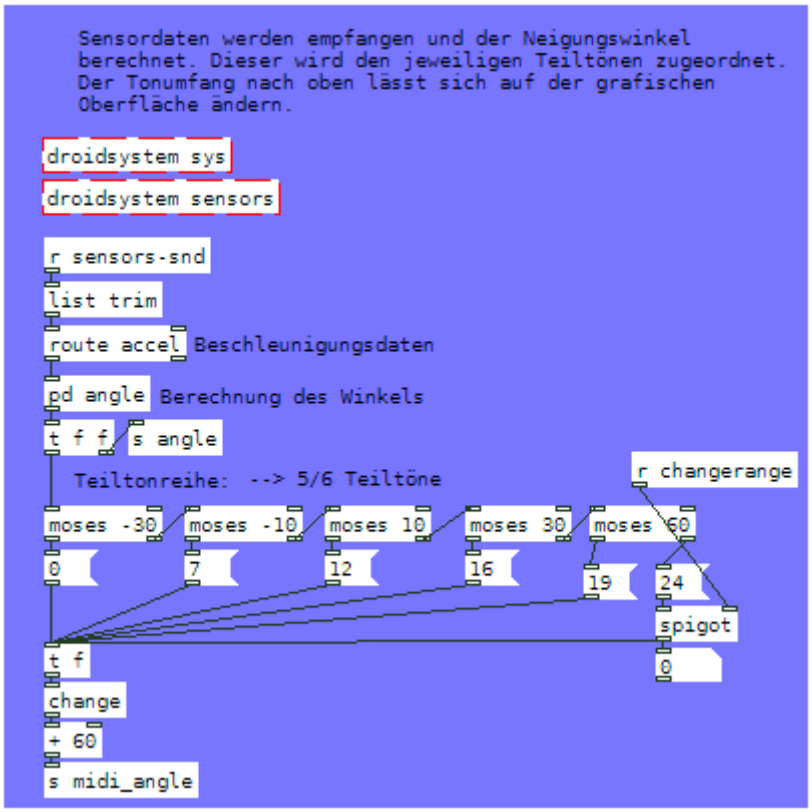


Abbildung 10: *Empfang der Sensordaten und Detektion der Winkel bzw. Stufen der Naturtonreihe. Zusätzlich: Begrenzung des Tonumfangs nach oben.*

4.2.3 Zusammenfassung der Simulationsprinzipien

Durch die in diesem Kapitel beschriebenen Steuerprinzipien interagiert der Benutzer des *Trumpet-Simulators* mit der Software bzw. dem Instrument, wobei sich das in *Abbildung 11* dargestellte (Daten-)Flussdiagramm vom schematischen Ablauf ergibt.

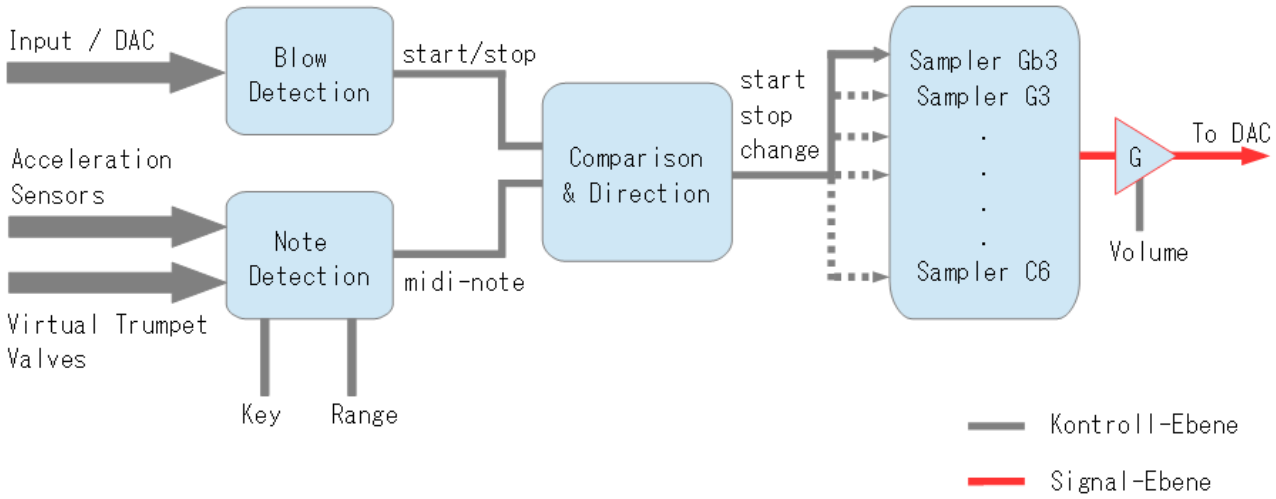


Abbildung 11: *Flussdiagramm der Verarbeitung der Inputs zur Tonerzeugung & Steuerung der Tonhöhe*

In diesem Diagramm ist der Datenfluss, beginnend mit den drei Bedienungsprinzipien am linken Rand und endend bei der Wiedergabe der richtigen Samples und dem Zuführen zum Digital-Analog-Umsetzer, schematisch dargestellt. Dabei sind die einzelnen Blöcke Codeabschnitte, welche am Eingang gewisse Argumente empfangen und in einen neuen Ausgangswert umrechnen. Die drei Argumente *Key*, *Range* und *Volume* sind dabei Kalibrierungen, die auf der grafischen Oberfläche der App gewählt werden können und das Verhalten der Blöcke beeinflussen. Der mittlere Block ist hier eine Verarbeitungsstufe, welche eigentlich im Sampler-Patch selbst beinhaltet ist. Da dies aber in allen Sampler-Patches gleichzeitig geschieht und somit die Kontrollelemente quasi geroutet werden, kann das Ablaufschema wie oben dargestellt werden.

4.2.4 Grafische Benutzeroberfläche

Die grafische Benutzeroberfläche besteht, neben den bereits in *Abb. 8* auf *S.32* ersichtlichen virtuellen Ventilen und der Winkelanzeige, aus mehreren Widgets bzw. Steuerelementen zum Wählen der in ***4.2.3*** erwähnten Einstellungen.

Da die Mikrophone von mobilen Geräten bei einem nahen Pusten oder Blasen recht schnell voll ausgesteuert sind und somit beim Blasen keine allzu große Dynamik erreichbar ist, hat es sich bei der Entwicklung der App als vorteilhaft herausgestellt eine „globale“ Lautstärke per Schieberegler festzulegen und um diesen Punkt herum die in Kap. ***4.2.2*** auf *S. 30 und 31* beschriebene Dynamik zu verwenden. Dafür gibt es einen Schieberegler an der

Benutzeroberfläche, durch den die ansonsten recht konstant lauten, normalisierten Töne des *Trumpet-Simulators* verstärkt und abgeschwächt werden können.

Des Weiteren befindet sich auf der grafischen Oberfläche die Option, die Tonart zu wechseln. Es kann gewissermaßen zwischen einer Bb-Trompete und einer C-Trompete gewählt werden, was das Instrument um einen Ganzton transponiert. Dies kann beim Begleiten anderer Instrumente bzw. Musiker oder beim Spielen zu Musik von Vorteil sein.

Der Tonumfang kann sich ebenfalls, wie auf *S.36* beschrieben, mittels eines Schalters auf der Benutzeroberfläche einstellen lassen um die Nutzung des oberen Winkelbereichs zu erleichtern.

Da bei der App die Anzeige eines Hilfetextes zur Erklärung der Funktionsweise ungünstig ist, ist ein Feld mit einem Fragezeichen auf der Oberfläche platziert. Bei dessen Berührung wird zur Starthilfe eine Audiodatei abgespielt, auf der die Funktionsweise kurz erklärt wird.

Abbildung 12: *(**a**, links) Foto einer Bb-Trompete (**b**, mittig) grafische Benutzeroberfläche des Trumpet-Simulators (**c**, rechts) mit veränderten Einstellungen*

Die oben beschriebenen vier Einstellungsmöglichkeiten sind, wie in *Abbildung 12* erkennbar, in der linken Reihe von unten nach oben angeordnet. Die rechte Reihe an Steuerelementen dient ausschließlich zum Abspielen zwei verschiedener, 12-taktiger Bluesstücke, die als Begleitung zum Improvisieren mit dem *Trumpet-Simulator* gewählt werden können. Dabei sind ein Bb- und ein C-Blues wählbar, welche nach dem 12 -taktigen Bluesschema aufgebaut sind und so lange wiederholt werden, bis man die Wiedergabe stoppt. Beide Stücke wurden selbst mit Software Instrumenten eingespielt und in die App importiert. Diese sollen vor allem für den Anfang einen Anreiz zum Spielen und Ausprobieren der Mobile Music App geben.

5 Konklusion

5.1 Zusammenfassung der Arbeit

Im Zuge dieser Bachelorarbeit wurde die Entwicklung einer interaktiven Mobile Music App von der Motivation über die Idee und Umsetzung bis zu einer Einordnung in den aktuellen App Markt und eventuellen Einsatzmöglichkeiten beschrieben. Im zweiten Kapitel wurde auf die technische Entwicklung von mobilen Geräten und deren Potential, vor allem im Hinblick auf Mobile Music Apps, eingegangen und ein kurzer Überblick über die Entwicklung der wichtigsten App Märkte gegeben. Anschließend wurde speziell auf das Musizieren mit Mobile Music Apps eingegangen, wobei zuerst eine Kategorisierung der verschiedenen Ansätze innerhalb dieses Feldes nach *M. Krebs* beschrieben wird. Außerdem wurde die Umsetzung dieser Konzepte, unter anderem anhand einer Benutzerumfrage, diskutiert und potentielle Verbesserungen bzw. Schwerpunkte genannt. Einer dieser Schwerpunkte ist die darauf folgend erläuterte soziale Komponente. Es werden zwei akademische Anwendungsmöglichkeiten von Mobile Music Apps betrachtet. Dabei wird auf die musikalische Bildung, beispielsweise im Musikunterricht, und die Verwendung von Apps als Übungshilfe bzw. zur Übungserweiterung für Musiker eingegangen. Die anfängliche Hintergrundrecherche lieferte zum einen eine Richtung für den eigenen Ansatz als auch Erkenntnisse über die Umsetzungen ähnlicher Apps mit deren Stärken und Schwächen. Somit konnten nach dem Entschluss, eine Trompete mithilfe der Programmiersprache Pure Data und der mit der Library-Version von Pure Data (*libpd)* laufenden App *PdDroidParty* zu simulieren, die ersten eigenen Ansätze hinterfragt und mit ähnlichen Ansätzen von bereits verfügbaren Trompeten-Apps verglichen werden. Anhand dessen konnte ein Ideengerüst entstehen, welches die brauchbarsten Konzepte für Bedienung, Klangerzeugung und Funktionen des Instruments beinhaltete. Im Anschluss wurden die akustischen Eigenschaften der Trompete und deren Tonerzeugung und Bedienung erklärt um eine Basis für die folgende Beschreibung der Umsetzung bei der Simulation zu schaffen. Es wurde die auf Samples basierende Tonerzeugung innerhalb von Pure Data, sowie die Arten der Interaktion des Nutzers mit dem Instrument, bzw. die Steuerung der Klänge beschrieben. Daraufhin wurde der so entstandene Datenfluss zusammenfassend dargestellt und das Design der grafischen Benutzeroberfläche aufgezeigt und begründet. Zum Ende werden hier die Vor- und Nachteile der in dieser Arbeit beschriebenen Umsetzung des *Trumpet-Simulators* diskutiert, sowie Anwendungsmöglichkeiten dargelegt und das Konzept im Hinblick auf die Zukunft von Mobile Music Apps erörtert.

5.2 Diskussion

Der *Trumpet-Simulator* ist laut Kapitel ***2.2.1*** auf *S.10* in der weniger innovativen Kategorie *App-Instrumente* einzuordnen und dient daher eher dazu, die Möglichkeiten der mobilen Geräte und deren technischen Gegebenheiten auszunutzen und möglichst genau ein bereits entwickeltes mechanisch-akustisches System nachzubilden, als die allgemeine Musizierpraxis zu erweitern. Dabei ist das theoretische Ziel die Spielbarkeit bzw. das Spielerlebnis vom physikalischen Instrument auf die digitale Simulation zu übertragen, sodass man auf ihm ähnlich virtuos wie auf dem Original musizieren kann. Aufgrund einiger Limitationen ist jedoch das resultierende Spielerlebnis nur ähnlich und nicht mit dem Original

vergleichbar. Dies liegt vor allem an der Reaktivität (Latenz) und dem zwangsläufig gegenüber dem Original veränderten Konzept zum Überblasen.

Die drei in Kapitel ***4.2.2*** beschriebenen Bedienkonzepte sind für das Spielerlebnis auf dem *Trumpet-Simulator* entscheidend. Diese Zusammenarbeit verschiedener Interaktionsebenen zwischen Spieler und Instrument wirkt sich positiv auf ein intuitives und natürliches Spiel aus. Die Verwendung verschiedener Systeme zur Steuerung dient als Beispiel für den positiven Effekt bei verstärkter Ausnutzung der technischen Möglichkeiten, welche, wie vorher erwähnt, von vielen als essentiell für die Weiterentwicklung von Mobile Music Apps gesehen wird. Die drei Bedienkonzepte besitzen jedoch alle sowohl Vor- als auch Nachteile.

Der Schalldruckpegel am Mikrophon, welcher zum Auslösen der Töne verwendet wird, zählt zum wichtigsten Punkt im Hinblick auf originalgetreues Spielen. Jedoch ist die Pegelberechnung (s. ***4.2.2*** *S.30/31*) zwangsläufig mit einer Verzögerung verbunden, welche sich zusätzlich zur globalen Latenz, die bei Android als großes Problem im Hinblick auf Echtzeitprozesse gesehen wird, addiert. Dies macht das Gesamtsystem recht träge und die deutlich spürbare Verzögerung zwischen Blasen und Tonerzeugung trübt das Spielerlebnis. Diese Funktion ist bei einer Trompeten Simulation jedoch unerlässlich und stellt den *Trumpet-Simulator* den meisten in Kap. ***3.1.1*** getesteten Apps voran. Eine weitere nützliche Funktion und ein Vorteil gegenüber den in *Tab.2* aufgelisteten Apps ist, dass zwischen angeblasenen bzw. angestoßenen und gebundenen Tönen unterschieden wird. Dies trägt vor allem zur Natürlichkeit des Klangs bei.

Die virtuellen Ventile besitzen nur den kleinen Nachteil, kein haptisches Feedback zu erzeugen. Dadurch fühlen sie sich etwas unnatürlich an, was jedoch beim Spielen nicht sehr stört. Um dieses Problem ein wenig zu entschärfen, gibt es ein Vibrationsfeedback beim Drücken der Ventile. Ansonsten kann diese Bedienung als gleichwertig mit dem Original angesehen werden, da gleichzeitig die Latenz über die Touchscreen-Steuerung weniger zum Tragen kommt. Trotz des recht simplen Konzepts der Funktionsweise der Ventile, ist dieser Teil der Umsetzung außerdem den in Kap. ***3.1.1*** getesteten Varianten überlegen, da diese oft gar keine richtigen Ventile bzw. Ventilfunktionen beinhalteten und somit keine Halbtöne spielbar waren.

Das Überblasen über den Neigungswinkel des Geräts ist einerseits eine gute Lösung für das zwangsläufig veränderte Konzept, besitzt andererseits jedoch ein paar Nachteile. Der Vorteil ist, dass, im Gegensatz zu den meisten getesteten Apps, ohne ein Umschalten oder eine Auswahl der Oktave, der gesamte Tonumfang des zugrundeliegenden Instruments recht intuitiv und flexibel ausgenutzt werden kann. Nachteile entstehen dadurch, dass kein Binden der Töne über mehrere Teiltöne möglich ist, da der dazwischenliegende Ton zwangsläufig kurz erklingt und ein optisches Feedback in Form einer Winkelanzeige nötig ist, um, vor allem zu Beginn, das Spielen zu erleichtern. Nach Gewöhnung an dieses Verhalten kann jedoch auch ohne optische Hilfe gespielt werden, was beispielsweise beim gemeinsamen Spielen nützlich sein kann.

Den Klang über Samples zu erzeugen ist ein weiterer Punkt, der die Simulation dem Instrument näher bringt. Samplebasierte Tonerzeugung ist nicht nur bei Blasinstrumenten der Klangsynthese vorzuziehen, da einfacher ein natürlicher Klang zu erreichen ist. Dabei spielt der erhöhte Speicherbedarf heute keine allzu große Rolle mehr, da selbst bei Smartphones und Tablets ein ausreichend großer und erweiterbarer Speicher zur Verfügung steht.

Der *Trumpet-Simulator* besitzt keine besondere soziale Komponente, sondern ist ein

eigenständiges Solo-Instrument. Der soziale Aspekt kann jedoch im Hinblick auf die musikalische Bildung aufgeführt werden. Durch die bereits in Kapitel ***2.2.4*** beschriebene hohe Verfügbarkeit und Allgegenwärtigkeit von mobilen Geräten kann die Instrumenten-Simulation jungen Menschen einen Zugang zur Musik und zum zugrundeliegenden Instrument verschaffen. Die Vorteile beim Musizieren mit App-Instrumenten liegen unter anderem beim erhöhten Interesse junger Menschen an der modernen Technik, beim schnelleren Verständnis der vereinfachten Instrumenten-Simulationen, bei sofortiger Möglichkeit des Erzeugens ästhetisch ansprechender Klänge und beim rascheren Fortschritt durch Üben. Selbst wenn ein App-Instrument anfangs als Spielzeug angesehen und genutzt wird, besteht die Möglichkeit das Interesse des Nutzers zu wecken und ihm eine Möglichkeit zu verschaffen das Instrument und dessen Funktionsweise näherzubringen. In Anbetracht der oben betrachteten Vor- und Nachteile der Umsetzung und der kritischen Betrachtung des Spielerlebnisses, ist dies eine Hauptanwendungsmöglichkeit neben der im Folgenden vorgeschlagenen Nutzung als Übungserweiterung.

Neben den Nutzern, welche ausschließlich die App und keine echte Trompete besitzen und diese entweder als musikalisches Spielzeug oder zum Ausprobieren und Heranführen an das richtige Instrument benutzen, können Musiker, die bereits über das Instrument verfügen und dessen Grundfunktionen beherrschen, mithilfe der App überall und jederzeit Improvisieren, Melodien austesten und aktuell geprobte Stücke üben oder die Trompetengriffe vertiefen. Der *Trumpet-Simulator* kann somit einem Trompeter oder einer Trompeterin als Übungshilfe dienen, wobei das Gewicht und die Größe des mobilen Geräts und die dadurch entstehende Mobilität große Vorteile sind. Da die Lippen beim Spielen des App-Instruments nicht verwendet werden, können diese auch direkt nach dem eigentlichen Üben bei Ermüdung der Lippenmuskulatur die App als Übungshilfe zur Hand nehmen.

Der *Trumpet-Simulator* ist den getesteten vergleichbaren Trompeten-Apps in den meisten Punkten überlegen. Für eine Verwendung als Instrumentenersatz oder zum Spielen von komplexerer und temporeicher Musik ist der *Trumpet-Simulator* aufgrund der oben aufgeführten Nachteile wie der Verzögerungszeit und der Veränderung des Konzepts zum Überblasen jedoch weniger zu gebrauchen, weswegen er letztendlich, neben den Überlegungen zur Verwendung als Übungshilfe und Bildungsgegenstand, nicht wirklich über die Nutzung als App-Spielzeug hinauskommt.

5.3 Ausblick

Die in dieser Arbeit behandelte Umsetzung eines traditionellen Instruments in eine Mobile Music App, zeigt mehrere Gründe auf, warum dieses Konzept von App-Instrumenten in Zukunft eine kleinere Rolle spielen wird, als innovative, kreative und neue Methoden mobile Geräte zu Musikinstrumenten zu machen. Zum einen gibt es keine wirkliche Bereicherung der musikalischen Möglichkeiten bzw. *„Erweiterung der Musizierpraxis“ (M. Krebs, vgl. Kap. **2.2.3**)*, zum anderen müssen bei der Simulationen von physikalischen Prinzipien von Instrumenten meist Kompromisse eingegangen werden, die Bedienung, Klang und letztendlich Spielpraxis beeinträchtigen.

Die in Kapitel ***2.1.2*** als essentiell beschriebene volle Ausnutzung der technischen Möglichkeiten der Systeme, Sensoren und Ein- und Ausgabegeräte kann außerdem nur gewährleistet werden, wenn die Entwickler von Mobile Music Apps alle Freiheiten besitzen das vorhandene technische Potential nach ihren Wünschen zu nutzen. Während die Simulation eines vorliegenden Instruments den Entwickler einschränkt, da er klare Vorgaben

umsetzt und kein oder nur wenig Raum für kreative Erweiterungen vorhanden ist, kann bei einem umgekehrten Ansatz der Fantasie freien Lauf gelassen werden. Es kann selber entschieden werden, welche der vielen technologischen Möglichkeiten verwendet werden und wie komplex oder simpel die Bedienung und Klangerzeugung ist. Mit umgekehrtem Ansatz ist hierbei die Argumentationsreihenfolge beim Konzipieren der App gemeint. Beim weniger innovativen Ansatz dient ein Instrument als Vorlage und soll in eine App verwandelt werden. Im optimalen Fall ist diese App dann natürlich wieder als ein Instrument zu bezeichnen. Der andere Ansatz ist der, eine Idee mithilfe einer App zu realisieren um diese, bzw. das Gerät, somit in ein Instrument zu verwandeln. Dieser zweite Ansatz wäre laut *Tabelle 1* auf *S.10* in die Kategorie *Kunst-Apps* bzw. *App-Soundtoys* einzuordnen und birgt, vor den *Controller-Apps* und den *Hilfsmitteln und Effektgeräten*, wohl das größte Potential für die vielversprechende Zukunft von Mobile Music Apps.

Im Anhang befindet sich:

- Eine .zip-Datei (Der verpackte Ordner “Trumpet-Simulator”, welcher die Pure Data Patches, SVG-Dateien und eine TrueType-Schriftartendatei beinhaltet)
- Eine .pdf-Datei der Bachelorarbeit

Die PdDroidParty APK-Datei zum Ausführen des Programms, sowie die Abstractions und eine Erklärung finden sich auf http://droidparty.net .

Ein Demonstrationsvideo der Verwendung der App findet sich unter:
https://www.youtube.com/watch?v=TTvKpnsBqFQ

Literaturverzeichnis

[1] Wikipedia, „Pure Data,“ 2015. [Online]. Available: https://de.wikipedia.org/wiki/Pure_Data. [Zugriff am 26 08 2015].

[2] C. McCormick, „PdDroidParty,“ 2013. [Online]. Available: http://droidparty.net. [Zugriff am 26 08 2015].

[3] Z. D. Boren, „There are officially more mobile devices than people in the world,“ 2014. [Online]. Available: http://www.independent.co.uk/life-style/gadgets-and-tech/news/there-are-officially-more-mobile-devices-than-people-in-the-world-9780518.html. [Zugriff am 26 08 2015].

[4] I. Sager, „Before IPhone and Android Came Simon, the First Smartphone,“ 2012. [Online]. Available: http://www.bloomberg.com/bw/articles/2012-06-29/before-iphone-and-android-came-simon-the-first-smartphone. [Zugriff am 26 08 2015].

[5] Wikipedia, „Nokia Communicator,“ 2015. [Online]. Available: https://de.wikipedia.org/wiki/Nokia_Communicator. [Zugriff am 26 08 2015].

[6] Wikipedia, „Smartphone,“ 2015. [Online]. Available: https://de.wikipedia.org/wiki/Smartphone. [Zugriff am 26 08 2015].

[7] T. Költsch und S. Analytics, „Android läuft auf fast 85 Prozent aller Smartphones,“ 2014. [Online]. Available: http://www.golem.de/news/mobile-betriebssysteme-android-laeuft-auf-fast-85-prozent-aller-smartphones-1408-108290.html. [Zugriff am 26 08 2015].

[8] Wikipedia, „Pebble (Armbanduhr),“ 2015. [Online]. Available: https://de.wikipedia.org/wiki/Pebble_(Armbanduhr). [Zugriff am 26 08 2015].

[9] Wikipedia, „Smartwatch,“ 2015. [Online]. Available: https://en.wikipedia.org/wiki/Smartwatch. [Zugriff am 26 08 2015].

[10] A. Appfigures.com, „App Stores Growth Accelerates in 2014,“ 2015. [Online]. Available: http://blog.appfigures.com/app-stores-growth-accelerates-in-2014/. [Zugriff am 26 08 2015].

[11] Appbrain, „Number of Android applications,“ 2015. [Online]. Available: http://www.appbrain.com/stats/number-of-android-apps. [Zugriff am 26 08 2015].

[12] A. Tanaka, Parkinson, A., Settel, Z. und Tahiroglu, K., „A Survey and Thematic Analysis Approach as Input to the Design of Mobile Music GUIs,“ in *NIME'12*, University of Michigan, Ann Arbor, 2012.

[13] M. Krebs, „App-Musik - Musizieren mit Smartphones,“ *Musikforum 1/12,* pp. 14 - 19, 2012.

[14] Wikipedia, „The Fall (Album),“ 2015. [Online]. Available: https://de.wikipedia.org/wiki/The_Fall_(Album). [Zugriff am 26 08 2015].

[15] G. Wang, „Designing Smule's iPhone Ocarina,“ in *NIME09*, Pittsburgh, 2009.

[16] J. Oh, J. Herrera, N. J. Bryan, L. Dahl und G. Wang, „Evolving The Mobile Phone Orchestra,“ in *NIME2010*, Sydney, 2010.

[17] G. Levin, „Mobile Phone Concert @ Ars Electronica 2001,“ 2007. [Online]. Available: https://www.youtube.com/watch?v=g1G-YesiBB8. [Zugriff am 26 08 2015].

[18] G. Levin, „Dialtones (A Telesymphony) Final Report,“ 2001.

[19] J. Sastre und e. al., „New Technologies for Music Education,“ in *Second International Conference on e-Learning and e-Technologies in Education*, Lodz (Poland), 2013.

[20] Wikipedia, „Kinect,“ 2015. [Online]. Available: https://de.wikipedia.org/wiki/Kinect. [Zugriff am 26 08 2015].

[21] Y. Lim und W. Yeo, „Smartphones-based Music Conducting," in *NIME'14*, Goldsmiths, University of London, 2014.

[22] Wikipedia, „Ableton Live," 2015. [Online]. Available: https://de.wikipedia.org/wiki/Ableton_Live. [Zugriff am 26 08 2015].

[23] J. Wolfe, „Brass instrument (lip reed) acoustics: an introduction," [Online]. Available: http://newt.phys.unsw.edu.au/jw/brassacoustics.html. [Zugriff am 26 08 2015].

[24] D. E. Hall, Musikalische Akustik, Mainz: Schott, 2008.

[25] V. S. L. GmBH, „Trompete in C - Tonerzeugung," 2015. [Online]. Available: https://vsl.co.at/de/Trumpet_in_C/Sound_Production. [Zugriff am 26 08 2015].

[26] M. Dickreiter, Handbuch der Tonstudiotechnik Band 1, München: K. G. Saur, 2008.

[27] Wikipedia, „Newtonsche Gesetze," 2015. [Online]. Available: https://de.wikipedia.org/wiki/Newtonsche_Gesetze. [Zugriff am 26 08 2015].

[28] Wikipedia, „Beschleunigungssensor," 2015. [Online]. Available: https://de.wikipedia.org/wiki/Beschleunigungssensor. [Zugriff am 26 08 2015].

[29] M. Dadafshar und Rinortner, K., „MEMS-Sensoren," 2015. [Online]. Available: http://www.elektronikpraxis.vogel.de/analogtechnik/articles/458045/. [Zugriff am 26 08 2015].