



Fachhochschule Nordwestschweiz
Hochschule für Technik

Sonic Classroom

Bachelor Thesis, FHNW, Studiengang Informatik

Auftraggeber	Pädagogische Hochschule FHNW Daniel Hug, Professur für Musikpädagogik
Autoren	Florian Rüegg Florin Tiefenauer
Betreuende	Prof. Dr. Doris Agotai, Institut für 4D-Technologien Roman Bolzern, Institut für 4D-Technologien
Expertin	Claudia Denz
Projektnummer	i4Ds25
Abgabetermin	14. August 2015

Inhaltsverzeichnis

1	<u>Zusammenfassung</u>	1
1.1	Problemstellung	1
1.2	Vorgehen	1
1.3	Hauptergebnisse	1
2	<u>Einleitung</u>	2
2.1	Ausgangslage	2
2.2	Ziele	2
2.3	Methodik	3
2.3.1	Machbarkeit / Recherche	3
2.3.2	Anforderungsanalyse	3
2.3.3	Vorgehensmodell Softwareentwicklung	3
2.4	Anforderungsanalyse	4
2.4.1	Stakeholder	4
2.4.2	Expertengespräche	5
2.4.3	Kontextszenarien	6
2.4.4	User Stories	7
2.4.5	Use Cases	8
2.5	Konzept	10
2.5.1	Recherche	10
2.5.2	Sensing Devices	11
2.5.3	Umsetzungsvarianten	12
2.5.4	Spezifikation Kinect V2	14
2.5.5	Lösungsansätze	15
2.5.6	Positionierung Kinect V2	20
2.5.7	Gesten	20
2.5.8	Klangmodifikationen	24
2.5.9	Spielabläufe	27
3	<u>Hauptteil</u>	28
3.1	Technologien	28
3.2	Frameworks / Libraries	28
3.2.1	Three.js	28
3.2.2	Web Audio API (Webaudiox)	29
3.2.3	Tuna.js	29
3.2.4	jQuery UI	29
3.2.5	Bootstrap	29
3.2.6	Kinect for Windows SDK 2.0	30
3.2.7	Alchemy Websockets	30
3.3	Sonic Classroom Controller	31
3.3.1	Übersicht	31
3.3.2	Gestenerkennung	34
3.3.3	Positionserkennung	39

3.4 Datenübermittlung	40
3.4.1 JSON	40
3.4.2 WebSocket Server Implementierung C#	43
3.4.3 WebSocket Client Implementierung JavaScript	43
3.5 Ludosonica	45
3.5.1 Grafische Oberfläche	45
3.5.2 Implementierungen	48
3.6 Fieldtest	53
3.7 Erweiterbarkeit	54
<u>4 Schluss</u>	<u>55</u>
4.1 Resultate	55
4.2 Ausblick	55
4.3 Reflexion	56
<u>5 Verdankung</u>	<u>57</u>
<u>6 Ehrlichkeitserklärung</u>	<u>58</u>
<u>7 Glossar</u>	<u>59</u>
<u>8 Abbildungsverzeichnis</u>	<u>61</u>
<u>9 Tabellenverzeichnis</u>	<u>63</u>
<u>10 Quellenverzeichnis</u>	<u>64</u>
<u>11 Anhang</u>	<u>IV</u>
11.1 Anhang 1 – Planung	IV
11.2 Anhang 2 – Leitfaden Expertengespräche	V
11.3 Anhang 3 – Skizze A3 Erklärung für Expertengespräch	VII
11.4 Anhang 4 – Auswertung Expertengespräche	VIII
11.5 Anhang 5 – Testmatrix Gesten	XII
11.6 Anhang 6 – Gestengruppierung	XIII
11.7 Anhang 7 – Codeabgrenzung Ludosonica	XIV
11.8 Anhang 8 – Vorgehen Test mit Schulklasse	XV
11.9 Anhang 9 – Auswertung Test mit Schulklasse	XVII

1 Zusammenfassung

1.1 Problemstellung

Mit Hilfe einer Sensing Device soll die „Augmented Reality“ in die Klassenzimmer Einzug halten. Die technische Affinität soll mit den klassischen Unterrichtsformen verbunden werden, um so neue Wege der Interaktion zwischen Mensch und Maschine zu finden. Ziel dieses Projektes ist es herauszufinden, welche Möglichkeiten die heutigen Technologien im Musikschulunterricht bieten und wo die Grenzen liegen.

1.2 Vorgehen

Durch Gespräche mit Experten der Musikpädagogik konnte festgestellt werden, welche konkreten Anwendungsgebiete die Software haben könnte. Auf dieser Basis wurde ein Konzept ausgearbeitet, sowie eine Softwarekomponente zur Ansteuerung einer Microsoft Kinect V2 entwickelt und das bestehende Projekt „Ludosonica“ des Instituts für 4D-Technologien erweitert und angepasst.

1.3 Hauptergebnisse

Nebst dem Konzept liegt ein Prototyp in Form von Software vor. Dieser Prototyp ermöglicht es, mit beinahe 30 Körpergesten verschiedene Klangobjekte im Raum zu bedienen. Klänge können mit bis zu 15 verschiedenen Modifikationen gestartet, verändert und beendet werden. Damit die entstehenden Klangkompositionen für das menschliche Gehör angenehm klingen, wird ein Synchronisationsmechanismus eingesetzt. Der Prototyp wurde durch eine Schulklasse im realen Umfeld getestet wobei die Resultate keinen abschliessenden Charakter haben.

2 Einleitung

2.1 Ausgangslage

Es gibt heute auf dem Markt diverse Geräte die es ermöglichen, die Bewegungen des menschlichen Körpers mittels Kamera und/oder Infrarot Sensor zu erfassen. Diese Interaktion des Menschen mit dem Computer und die Erweiterung der Realität werden in der Literatur oft als „Augmented Reality“ beschrieben. Das Institut für 4D-Technologien hat unabhängig von diesem Projekt in Zusammenarbeit mit der Hochschule für Musik und der Hochschule für Gestaltung und Kunst einen Prototyp mit dem Namen Ludosonica entwickelt, der es ermöglicht, in einem dreidimensionalen Raum Gesten zu erkennen und diese mit dem Abspielen von Musik zu koppeln. Dieser Prototyp und die Technologie der heutigen Sensing Devices bilden die Basis des Projektes.

In der Schweizer Volksschule wird der Unterricht heute schon in grossen Teilen mit Informatikmitteln aktiv mitgestaltet. Das Interagieren mit dem Computer und allgemein technischen Geräten ist für die heutigen Kinder alltäglich. Um die technische Affinität mit den klassischen Unterrichtsformen zu verbinden, werden neue Wege der Interaktion zwischen Mensch und Maschine gesucht. Je nach Einsatzzweck kann es eine grosse Bereicherung des Schulalltags sein, wenn die Kinder mit neuen Arten des Zusammenspiels zwischen ihnen und dem Computer in Berührung kommen.

Verschiedenste Forschungsprojekte befassen sich aktuell mit Themen der Interaktion zwischen räumlicher Bewegung und Klang. Die Pädagogische Hochschule der Fachhochschule Nordwestschweiz interessiert sich dabei für die konkreten musikpädagogischen Anwendungsgebiete, wie mit Kindern im Musikzimmer mit diesen Technologien interagiert werden kann.

2.2 Ziele

In erster Linie soll aufgezeigt werden, was für Möglichkeiten die heutigen Technologien bieten und was die Grenzen sind. Weiter soll ein Prototyp entwickelt werden, der es ermöglicht, eine Umgebung in Ludosonica zu definieren und die Gesten auf Klänge zu verknüpfen, diese mit einer Sensing Device aufzuzeichnen und den Klang abspielen zu lassen.

2.3 Methodik

2.3.1 Machbarkeit / Recherche

Als ersten Schritt des methodischen Vorgehens wurde recherchiert, was für technologische Ansätze verfolgt werden könnten und wie sich diese auf die Machbarkeit des Projektes auswirken. Auf dem Markt sind verschiedenste Sensing Devices verfügbar. Die einen wurden für den Spielkonsolenmarkt entwickelt und andere für das Motion Tracking einer Hand. Einige Geräte arbeiten nur mit Color Cameras, die meisten jedoch mit einer Color Camera und einer Infrarot (Depth) Camera. Ein wichtiges Kriterium waren die unterstützten Programmiersprachen und deren SDK's welche bereits mitgeliefert werden. Hinzu kommt die Tatsache, dass gewisse Geräte bereits die Skelette von Menschen erkennen können. Um eine Entscheidungsgrundlage über die Machbarkeit zu schaffen wurde eine Entscheidungsmatrix für die recherchierten Informationen erstellt (siehe Kapitel 2.5.2 Sensing Devices). Im Zusammenhang mit diesen Technologien bestehen schon einige Referenz Projekte mit verschiedensten Ansätzen. Die konkreten Probleme welche durch diese Projekte verfolgt wurden, entsprachen aber nicht dem in dieser Arbeit zu untersuchenden Problem.

2.3.2 Anforderungsanalyse

Am Anfang wurde vor allem der Kontakt zu Herrn Hug, dem Verantwortlichen auf Seiten des Auftraggebers, gepflegt. Er gab das Projekt im Auftrag der Professur für Musikpädagogik bei der Hochschule für Technik ein. Basierend auf seinen Schilderungen und Beschreibungen konnte ein grobes Bild der Anforderungen an dieses Projekt gemacht werden. In einem weiteren Schritt wurden Expertengespräche durchgeführt mit Mitarbeitenden aus der Professur. Da alle diese Personen einen unterschiedlichen Hintergrund mit sich brachten, konnten sehr viele interessante Aspekte für das Projekt erkannt werden. Diese Expertengespräche waren sehr wertvoll um die Kontextszenarien daraus ab zu leiten und die Prozesse verstehen zu können.

2.3.3 Vorgehensmodell Softwareentwicklung

Aus den Expertengesprächen mit Mitarbeitenden der Professur für Musikpädagogik haben sich gewisse Kontextszenarien herauskristallisiert. Weiter wurde versucht, die wichtigsten Szenarien in User Stories zu unterteilen. An Hand dieser User Stories konnten verschiedene Tasks, zum Erreichen dieser Ziele definiert werden. Grundlage für die Use Cases sind ebenfalls die User Stories. In einer ersten Phase wurde ein Prototyp angestrebt, der mit Grundfunktionen ausgestattet ist, um in einer Schulklasse, in einem realen Umfeld, einen Test machen zu können. Die Erkenntnisse dieses Testes sollen in den zweiten Prototypen einfließen. Auch um die noch nicht implementierten Funktionen soll der zweite Prototyp erweitert werden. Nach weiteren Tests durch das Projektteam wird alles noch fehlende und nicht korrekt funktionierende, in die Final Version einfließen.

Als Vorgehensmodell wird Scrum verwendet. Scrum eignet sich durch sein schlankes und einfach zu organisierendes Projektmanagement Framework für ein kleines Projektteam sehr gut. Die Entwicklung wurde in die drei Sprints „Prototyp 1“, „Prototyp 2“ und „Final“ unterteilt. Jedem Sprint wurden die dazugehörigen User Stories zugewiesen, aus welchen danach die effektiven Tasks entstanden sind. Für die Verwaltung dieser Komponenten wird Trello verwendet.

Für die Übersicht über das Gesamtprojekt wurde eine Grobplanung mit Hilfe des Tools Office Timeline [1] erstellt und fortlaufend geführt (siehe Anhang 1 - Planung).

2.4 Anforderungsanalyse

2.4.1 Stakeholder

Als Stakeholder dieses Produktes werden vor allem die Lehrpersonen und im speziellen Lehrpersonen mit musikpädagogischem Hintergrund verstanden. Im Zuge der Anforderungsanalyse wurden vier Expertengespräche durchgeführt. Für die Weiterarbeit werden diese Experten als Stakeholder definiert.

Daniel Hug

Die Kontaktperson zum Auftraggeber (Pädagogische Hochschule) ist Daniel Hug. Er ist wissenschaftlicher Mitarbeiter an der Professur für Musikpädagogik. Er betreut das Projekt, bringt Ideen ein und stellt das Bindeglied zur Pädagogischen Hochschule dar.

Prof. Markus Clovjecsek

Herr Cslovjecsek ist Leiter der „Professur für Musikpädagogik im Jugendalter“. Er hat die Projektidee mitentwickelt und verfügt über ein breites Spektrum an Erfahrung in unterschiedlichsten Unterrichtssettings.

Thomi Christ

Herr Christ arbeitet am Institut Sekundarstufe I und II der Pädagogischen Hochschule. Er ist tätig als Dozent der Musikpädagogik und hat auch ein technisches Interesse an der Materie.

Gabriele Noppeney

Als stellvertretende Professur Leiterin verfügt Frau Noppeney über einen starken methodisch/didaktischen Hintergrund. Sie begleitet Studierende während Praktika und in Reflexionsseminaren.

Boris Lanz

Boris Lanz ist Lehrbeauftragter für Musikpädagogik und beschäftigt sich mit Instrumenten-Eigenbau. Weiter interessiert er sich besonders für Anwendungen in Bezug auf Menschen mit Behinderung.

2.4.2 Expertengespräche

Um die geeigneten Kontextszenarien zu ermitteln, wurden im Vorfeld Gespräche mit den oben aufgeführten Experten der Professur für Musikpädagogik durchgeführt. In einem ersten Schritt wurden Leitfragen für das Interview vom Projektteam ausgearbeitet (siehe Anhang 2 – Leitfaden Expertengespräche). Weiter wurden die Fragen mit dem Auftraggeber und den Projektbetreuern im Vorfeld besprochen. Im Gesamten wurden vier verschiedene Experten befragt. Das Ziel dieser Gespräche sollte sein, dass die verschiedenen Kontexte, in welchen diese Software zum Einsatz kommen könnte, besser verstanden werden. Der Einstieg ins Gespräch fand anhand einer kurzen, durch ein grafisches Schema (siehe Anhang 3 – Skizze A3 Erklärung für Expertengespräch) unterstützten Erklärung statt. Es sollte den Interviewpartnern aufzeigen, um was es geht und was die Vorbedingungen und Möglichkeiten im Rahmen dieses Projektes sind. Spannend war, dass die Befragten alle mit einem anderen Hintergrund zu diesen Gesprächen kamen. Eine Person erhofft sich von einem solchen System eventuell die Möglichkeit, dass man damit auf eine Art dirigieren könnte. Ein anderer der Befragten interessierte sich sehr für eine Verwendung im Zusammenhang mit Personen mit eingeschränkter Beweglichkeit. Durch diese Durchmischung konnte eine ganze Reihe von Anwendungsfällen festgehalten werden. Aus Sicht des Projektteams war es wichtig, dass diese Erkenntnisse in den weiteren Projektverlauf einfließen. Auch wenn keine exakte Anforderungsliste besteht, sollte es doch durch diese Gespräche ermöglicht werden, eine Tendenz zu erkennen, in welche Richtung das Projekt gehen soll.

Mit der ersten Frage wurde versucht, herauszufinden, wie eine konkrete musikalische Aktivität aussehen könnte. Fast ausschliesslich alle Befragten waren sich einig, dass eine Aktivität mit kleineren Gruppen am sinnvollsten wäre. Das heisst eher in einer Gruppe, anstatt in einer gesamten Schulklass. Weiter wäre eventuell interessant, musikalische Aktivitäten für Personen mit eingeschränkter Bewegungsfreiheit zu realisieren. Beispielsweise die Möglichkeit, wenn eine Person durch ihre Behinderung gar keinen lauten Ton mit einer Pauke erzeugen kann und somit ohne Kraft und nur durch eine kleine Bewegung, sehr laute Töne oder musikalische Klänge erzeugen könnte.

Die Klangmodifikationen sind in dieser Arbeit sehr zentral. Welche Klangmodifikationen in einer solchen Umgebung sinnvoll wären, wurde mit der zweiten Frage herauszufinden versucht. Nebst den bereits vorgängig definierten Modifikationen, wie zum Beispiel „Musik Ein/Aus“, „Musik Lauter/Leiser“ kamen weitere wertvolle Inputs hinzu. In erster Linie sollten gemäss den Experten nicht nur Töne/Klänge abgespielt werden, sondern auch Geräusche (z.B. Vogelgezwitscher, vorbei fahrende Autos, usw.). Verschiedene Klanghöhen, Halleffekte, Tremolo Effekt und Low/High Pass Filter sind weitere Modifikationen die aufgenommen wurden. Ein Input erfolgte zum Thema MIDI Steuerung. Die Meinung war, dass eine „professionelle“ Erzeugung von Musik nur über das weitersenden der Signale an einen MIDI-Controller denkbar wäre.

In einem nächsten Schritt wurde mit den Experten darüber diskutiert, welche Gesten/Bewegungen sinnvoll sind und welche nicht. Nebst den intuitiv bereits definierten, standen zum Beispiel fließende Gesten zur Diskussion. Der Bedarf für fließende Gesten wäre beispielsweise beim Verstellen der Lautstärke denkbar.

Weiter wurden die Experten gefragt, ob eine Konfiguration des Tools durch die Lehrperson gewünscht würde oder eher vordefinierte Settings Sinn machen. Diese Frage beantworteten die Befragten geschlossen, indem sie vordefinierte Settings mit einer eventuellen Möglichkeit zur Bearbeitung bevorzugen würden. Bei einer konkreten Nutzung müsste sicher eine Schulung im Vorfeld durchgeführt werden.

Es wurde diskutiert, ob eine Live-Recording Funktion nötig ist und unter welchen Umständen diese sinnvoll wäre. Alle Experten meinten, dass eine damit realisierte „Looping“ Funktion sehr interessant wäre.

Auch bei der letzten Frage, ob die Installation fix oder mobil sein soll, waren sich die Befragten einig. Es sollte fix sein damit so wenig Aufwand wie möglich nötig ist. Aufwand und komplizierte Installationen schrecken die Benutzer ab.

Die zusammengefassten Ergebnisse der Expertengespräche sind im Anhang 4 – Auswertung Expertengespräche zu finden.

2.4.3 Kontextszenarien

Die folgenden Kontextszenarien sollen auf Basis der Expertengespräche darlegen, wie der Kontext des Projektes aussieht.

Kontextszenario (KS-1) – Definition eines Spielablaufs	
Hauptaktivitäten	Die Lehrperson oder der Schüler definiert ein Unterrichtssetup anhand seiner Überlegungen. Danach kann das Setup mit einer Gruppe für eine musikalische Aktivität verwendet werden.
Umfeld	Nebst der Idee wird ein Gerät (Laptop/Computer) benötigt auf welchem die Sonic Classroom Software installiert ist.
Dauer	Je nach Komplexität des Setups.
Endergebnis	Ein realistisches Unterrichtssetup welches mit einer Gruppe erfolgreich durchgeführt werden kann.

Tabelle 1 Kontextszenario 1 - Definition eines Spielablaufs

Kontextszenario (KS-2) – Autonome Gruppenaktivität	
Hauptaktivitäten	Eine Gruppe (2-6) von Schülern nutzt die Installation um eine musikalische Aktivität durchzuführen. Es sollen bestimmte Bewegungen koordiniert oder unkoordiniert ausgeführt werden, um damit ein Klangmuster zu erzeugen.
Umfeld	Diese Aktivität kann in einem Klassen- oder Musikzimmer durchgeführt werden und das ganz ohne Lehrperson.
Dauer	Dauer einer Aktivität kann frei gewählt werden. Je nach Bedürfnis kann es nur einige Minuten oder auch länger dauern.
Endergebnis	Die Gruppe kann den Raum als musikalisches Element nutzen. Gruppenzusammenhalt und Musikgehör / Taktgefühl können gestärkt werden.

Tabelle 2 Kontextszenario 2 - Autonome Gruppenaktivität

Kontextszenario (KS-3) – Geführte Gruppenaktivität	
Hauptaktivitäten	Eine Gruppe (2-6) von Schülern nutzt die Installation unter der Führung einer Lehrperson um eine musikalische Aktivität durchzuführen. Es sollen bestimmte Bewegungen koordiniert oder unkoordiniert ausgeführt werden, um damit ein Klangmuster zu erzeugen.
Umfeld	Diese Aktivität kann in einem Klassen- oder Musikzimmer durchgeführt werden und das ganze unter der Leitung einer Lehrperson.
Dauer	Dauer einer Aktivität kann frei gewählt werden. Je nach Bedürfnis kann es nur einige Minuten oder auch länger dauern.
Endergebnis	Die Gruppe kann den Raum als musikalisches Element nutzen und vorgegebene Abläufe durchführen. Gruppenzusammenhalt und Musikgehör / Taktgefühl können gestärkt werden. Erreichen eines Ziels gemäss konkreter Vorgabe der Lehrperson.

Tabelle 3 Kontextszenario 3 - Geführte Gruppenaktivität

2.4.4 User Stories

Für die spätere Definition der Tasks wurden folgende User Stories notiert. Sie sollen die konkreten Prozesse des Systems aufzeigen.

User Stories	
US-1	Die <i>Lehrperson</i> möchte mit Hilfe der Software Ludosonica Objekte im Raum positionieren können.
US-2	Die <i>Lehrperson</i> möchte auf die im Raum positionierten Objekte Gesten verknüpfen.
US-3	Auf die verknüpften Gesten soll die <i>Lehrperson</i> Klänge oder Klangmodifikationen definieren können.
US-4	Der <i>Schüler</i> soll eine Rückmeldung bekommen, ob er an dem von der <i>Lehrperson</i> definierten Ort steht und vom System erkannt wurde.
US-5	Der <i>Schüler</i> möchte anhand einer bestimmten Geste an einem bestimmten Ort einen bestimmten Klang oder eine Modifikation auslösen können.
US-6	Eine Gruppe von <i>Schülern</i> möchte anhand deren konkreten Gesten und der dazu abgespielten Klängen eine Klangkomposition realisieren können.
US-7	Die <i>Lehrperson</i> will die Software über einen definierten Ablauf in Betrieb nehmen können.

Tabelle 4 Beschreibung der User Stories

2.4.5 Use Cases

Nachfolgende Anwendungsfälle haben sich durch die Anforderungsanalyse ergeben. Da es keine konkrete Anforderungsliste gibt, sind auch die Use Cases nicht abschliessend.

Use Case 1 (UC-1)	
Titel	Installieren der Software Sonic Classroom Controller und gegebenenfalls lokale Ludosonica Webapplikation
Beschreibung	Die Lehrperson installiert den Sonic Classroom Controller und falls nötig die dazugehörige Webapplikation Ludosonica auf dem dafür vorgesehenen System.
Akteure	Lehrperson / Administrator
Vorbedingungen	Es ist ein System vorhanden das den minimalen Anforderungen an die Software genügt (min. Windows 8.0 Betriebssystem, Microsoft Kinect Minimalanforderungen, Chrome/Firefox Browser).
Nachbedingungen	Es kann eine funktionierende Szene erstellt werden.
Szenario	<ol style="list-style-type: none"> 1. Die Controllersoftware wird auf dem System, welches für die Verbindung zur Kinect vorgesehen ist, installiert. 2. Durch anhängen der Kinect wird automatisch die Treibersoftware installiert. 3. Über den Browser kann auf Ludosonica zugegriffen werden.
Alternativen	<ol style="list-style-type: none"> 3. Falls das Gerät autonom und ohne Internet Anbindung betrieben werden soll, kann ein lokaler Webserver mit Ludosonica installiert werden, welcher über localhost verfügbar ist.

Tabelle 5 Use Case 1

Use Case 2 (UC-2)	
Titel	Personen / Objekte im Raum positionieren
Beschreibung	Die Lehrperson positioniert in einem ersten Schritt für jeden Schüler ein Objekt im Raum.
Akteure	Lehrperson
Vorbedingungen	Die Anzahl Personen welche am Spielablauf teilnehmen ist bekannt.
Nachbedingungen	Die gewünschte Anzahl Objekte (Personen) sind am richtigen Ort positioniert und mit Gesten und deren Aktionen versehen.
Szenario	<ol style="list-style-type: none"> 1. Öffnen von Ludosonica im Editor Modus. 2. Über „New File“ eine neue leere Szene öffnen. 3. Das gewünschte Objekt anwählen und mit der Maus im Raum positionieren. 4. Den Körper anwählen und definieren bei welcher Geste welche Aktion ausgeführt werden soll. 5. Falls Musik oder Loop Ein/Aus gewählt wurde, eine Musikdatei hinterlegen. 6. Das Objekt über „Advanced Settings“ in der Erscheinung oder Physik verändern. 7. Schritt 3 - 6 beliebig oft wiederholen. 8. Um die Szene zu speichern auf „Export“ klicken. 9. Speicherort wählen und speichern.

Tabelle 6 Use Case 2

Use Case 3 (UC-3)	
Titel	Spielablauf durchführen
Beschreibung	Die Lehrperson startet den Spielablauf.
Akteure	Lehrperson
Vorbedingungen	Eine vorgängig definierte Szene mit verschiedenen Objekten und darauf verknüpften Gesten mit deren Aktionen.
Nachbedingungen	Die Verbindung zum Controller wurde aufgebaut und der Spielablauf kann durchgeführt werden.
Szenario	<ol style="list-style-type: none"> 1. Mit einem Klick auf den „Play“ Button wird in den Play Modus gewechselt. 2. Weiter können sich die Teilnehmer im Raum positionieren und bekommen in Form eines farbigen „Characters“ Rückmeldung über die Position und Geste.

Tabelle 7 Use Case 3

2.5 Konzept

2.5.1 Recherche

In der ersten Phase des Projektes wurde nach bereits bestehenden Projekten gesucht. Eine der interessantesten Anlaufstellen für Projekte dieser Art ist die NIME (New Interfaces for Musical Expression). Unter diesem Namen findet jährlich eine Konferenz statt, auf welcher die verschiedenen Forschungsgruppen auf der ganzen Welt ihre Resultate präsentieren. Die Resultate werden in den meisten Fällen in Form eines Papers publiziert. Diese finden sich auch auf der Webseite dieser NIME (www.nime.org).

Ein Projekt aus dem Jahre 2011 beschreibt die Verwendung von Bewegungen im Raum zum Ansteuern eines MIDI Gerätes [2]. Verwendet wurde die Microsoft Kinect Version 1. Unter der Verwendung des dort entwickelten Kinect-to-MIDI Programms werden die Kinect Daten in MIDI Signale umgewandelt.

Auf dem KINECT for Windows Blog von Microsoft wurde ein Artikel [3] publiziert der zeigt, wie ein Artist ein Klavier vollständig über eine Kinect steuert. Es werden Gesten erkannt, am Computer verarbeitet und danach mechanisch die Töne am Klavier getriggert. Diese Art von Verwendung ist sehr speziell und gibt einen Mix aus neuer Technologie und dem bereits Bekannten. Es werden keine digitalen Klänge ausgegeben, sondern schlussendlich nichts anderes als das, was das Klavier schon immer erzeugt hat.

Weiter wurde vom Auftraggeber auf einen Artikel in der Fachzeitschrift c't hingewiesen [4]. Dieser zeigt auf, wie gut das neue SDK (Software Development Kit) 2.0 von Microsoft ist. Dort wird eine einfache Methode gezeigt, wie ein „Luftpiano“ programmiert und benutzt werden kann. Es wird aufgezeigt werden, wie einfach es ist, mit der neuen SDK eine einfache Anwendung zu erstellen.

2.5.2 Sensing Devices

Folgende Entscheidungsmatrix soll aufzeigen, was für Sensing Devices zum jetzigen Zeitpunkt auf dem Markt verfügbar sind und über welche technischen Spezifikationen diese verfügen.

	Microsoft Kinect V2	PlayStation Camera	Asus XTION Pro Live	Structure.io	Creative Sens3d	Intel RealSense 3D-Kamera F200	duo3d	Project tango	Infineon
Hersteller	Microsoft	Sony	Asus	Occipital	Creative	Intel	Code Laboratories	Google	Infineon
Distanz	Min: 50cm Max: 4.5m	Min: 30cm Max: Infinity	Min: 80cm Max: 3.5m	Min: 40cm Max: 3.5m	Min: 15cm Max: 1m	Min: 20cm Max: 1.2m	nicht bekannt	-	Min: 20cm Max: 1m
Öffnungswinkel	Horizontal: 70° Vertikal: 60°	85° Field of View	Horizontal: 58° Vertikal: 45°	Horizontal: 58° Vertikal: 45°	Horizontal: 74° Vertikal: 74°	nicht bekannt	170° Wide Angle	-	Horizontal: 82° Vertikal: 66°
Depth Camera (IR) Auflösung	512 x 424	-	VGA (640 x 480): 30 fps QVGA (320 x 240): 60 fps	nicht bekannt	320 x 240	nicht bekannt	56 FPS @ 752x480 62 FPS @ 640x480 123 FPS @ 640x240 240 FPS @ 640x120 93 FPS @ 320x480 184 FPS @ 320x240 360 FPS @ 320x120	-	160 x 120 / 45fps
Color Camera (RGB) Auflösung	1920 x 1080 / 30 fps	1280x800 pixel @ 60fps 640x400 pixel @ 120fps 320x192 pixel @ 240fps	1280 x 1024	VGA (640 x 480) / QVGA (320 x 240)	1280 x 720 / 30fps	1920 x 1080	-	-	-
Multiuser	Ja, bis zu 6 Personen gleichzeitig	Ja, bis zu 4 Personen gleichzeitig	nicht bekannt	Nein	Nein	Nein	Nein	-	Nein
Skeletal Tracking	Ja, 25 Gelenke werden erkannt	Ja, 21 Gelenke werden erkannt	Ja	Nein	Nein, nur Gesichts- Gestenerkennung	Nein, nur Hand und Gesichtserkennung	Nein	-	Nein
Depth Tracking	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja	Ja
SDK's	Kinect for Windows SDK 2.0, OpenNI	HTLib & IISU Middleware von SoftKinetic	OpenNI	OpenNI, Structure SDK for iOS	Intel Perceptual Computing SDK, Unity, Havok	Intel RealSense SDK	DUO API/SDK	-	C/C++ & Matlab-SDK
Unterstützte OS	Windows 8	Windows 7,8	Windows XP, Vista, 7, 8 Ubuntu 10.10	iOS	Windows 7, 8	Windows 8.1	Windows, OS X, Linux	-	Windows, OS X, Linux
Unterstützte Programmiersprachen	C++, C#, Visual Basic, other .NET languages	C++	C++, C#, Java	Objective-C	C#, Java, Processing, Unity	C++, JavaScript, C#, Java, Processing	C++, C#, Python, MATLAB	Java, C/C++, Unity	C/C++, Matlab
Mehrere Geräte koppeln	Nein	nicht bekannt	nicht bekannt	-	-	-	nicht bekannt	-	-
Preis	ca. 200 \$	59 \$ (+ IISU 1499 \$)	140 \$	379 \$	ca. 130 \$	99 \$	1000 \$	-	nicht bekannt
Besonderheiten	Sehr viele OpenSource Libraries vorhanden	Funktioniert ohne IR, die Position wird mit zwei Kameras durch Triangulation berechnet. IISU mit Multiuser ist für die ersten 3 Monate gratis, danach nur noch eingeschränkt brauchbar.		Wird benutzt um Räume in 3D zu scannen.	Webcam mit Depth Sensing	Webcam mit Depth Sensing. Bestellung in die Schweiz nicht möglich	Leap Motion Alternative	Es wurde noch kein Gerät des Projekts Tango released	Referenzimplementierung von PMD (CamBoard pico) erhältlich

Nach der Auswertung der Matrix wurde schnell klar, dass nur die PlayStation Camera von Sony oder die Kinect V2 der Microsoft in Frage kommt. Hauptgrund ist das Skeletal Tracking welches bei diesen zwei Devices das Detektieren von menschlichen Körper ermöglicht. Diese Funktion ist für das Projekt Sonic Classroom absolut fundamental. Bei genauer Betrachtung der beiden Devices stach die starke Unterstützung für verschiedene Programmiersprachen, wie das bereits vorhandene SDK von Microsoft, heraus. Am Anfang des Projektes war noch nicht sicher, mit wie vielen Personen die Software genutzt werden soll. Wird aber angenommen, dass man von einer Schulklasse ausgeht, dann ist die Anzahl Personen die das Device erkennen kann, von hoher Bedeutung. Auch hier ist die Kinect V2 mit bis zu 6 Personen am besten aufgestellt. Dies führte ziemlich klar zum Entscheid für die Kinect V2 von Microsoft.

2.5.3 Umsetzungsvarianten

Die Programmierung des Kinect Sensors ist in verschiedenen Programmiersprachen möglich. Microsoft bietet für die Programmierung der Kinect ein eigens dafür entwickeltes Software Development Kit an. Dieses Kinect for Windows SDK 2.0 bietet Unterstützung für .NET Sprachen, wie C#, C++ und VB.NET, sowie bedingt für JavaScript an. [5]

Aus Sicht des Projekts Sonic Classroom gibt es zwei verschiedene Umsetzungsvarianten.

2.5.3.1 Variante Windows Desktop Applikation & Web Applikation

Die Gesten- und Positionserkennung wird mit einer in C# geschriebenen Windows Desktop Applikation (Controller) gelöst. In dieser App werden die Kinect Sensoren angesprochen und aus den erhaltenen Daten sollen Gesten und Positionen von Personen ermittelt werden. Mittels einer Schnittstelle werden diese Daten zu Position und Geste an die Ludosonica Web Applikation gesendet.

Eine eventuelle Erweiterung der Gesten- und Positionserkennung mittels Bildverarbeitung könnte durch den Einsatz von OpenCV erreicht werden. OpenCV kann mit C# angesprochen werden.

Ludosonica soll soweit erweitert werden, dass der Empfang der Daten, sowie das Verknüpfen von Position und Geste mit verschiedenen Klängen und Klangmodifikationen ermöglicht werden. Diese Erweiterungen werden in JavaScript umgesetzt.

Schnittstellen

Für die Schnittstelle zwischen Ludosonica und Controller würde in dieser Variante WebSockets verwendet werden. WebSockets ermöglichen eine asynchrone Verbindung über TCP zwischen Client- und Serverapplikation herzustellen. Die Messages können bidirektional und unabhängig voneinander verschickt werden. Im Falle dieses Projekts wäre in einer Message die Geste und Position einer detektierten Aktion des Kinect Sensors und eventuell weitere nützliche Informationen vorhanden.

Es gibt eine WebSockets API für JavaScript und verschiedene Third Party Libraries für C#.

Vorteile	Nachteile
Anschluss mehrerer Geräte an Ludosonica über ein Netzwerk wird ermöglicht.	Zusätzlich zur Installation der Desktop Applikation muss Ludosonica auf einem externen oder lokalen Webserver installiert sein.
Sehr gute Dokumentation und viele Tutorials für die Kinect SDK sind vorhanden.	Eventuelle Verzögerung durch Schnittstelle zwischen Desktop Applikation und Ludosonica
State of the Art für Kinect Programmierung	-
Klare Empfehlung von Simon Felix (Dozent C# Module).	-

Tabelle 8 Vor- und Nachteile Windows Desktop Applikation & Web Applikation

2.5.3.2 Variante Windows Store Applikation mit JavaScript

Eine andere Variante ist die Erstellung einer Windows Store App, welche direkt in JavaScript geschrieben wird. Eine Anbindung an das Kinect for Windows SDK 2.0 ist möglich. In dieser Variante würde die bestehende Ludosonica Applikation in die neue Windows Store App migriert werden. Somit wird ermöglicht, dass nur noch eine Applikation benötigt wird, welche sowohl die Gesten- und Positionserkennung, wie auch die Verknüpfung mit den Klangelementen beinhaltet.

Vorteile	Nachteile
Nur eine Installation der Windows Store App	Unklar ob eine Portierung von Ludosonica mit allen angebundenen JavaScript Libraries möglich ist.
Keine Client Server Applikation	Praktisch keine Dokumentation seitens Microsoft vorhanden
Nur eine Programmiersprache benötigt.	Store App Lifecycle muss beachtet werden (Daten müssen beim Start einer anderen App gespeichert und danach wieder neu geladen werden).

Tabelle 9 Vor- und Nachteile Windows Store Applikation mit JavaScript

2.5.3.3 Fazit

Aufgrund der starken Unterstützung von C# durch Microsoft im Zusammenhang mit der Entwicklung für Kinect V2 Applikationen und den wenigen Informationen zu Windows Store Applikationen, wird die erste Variante Windows Desktop Applikation & Web Applikation weiter verfolgt. Die zusätzliche Empfehlung durch Herr Simon Felix aus dem Institut für 4D-Technologien hat die Entscheidung zusätzlich positiv beeinflusst.

2.5.4 Spezifikation Kinect V2

Um die verschiedenen Lösungsansätze zu verstehen, werden in folgendem Abschnitt die physikalischen Grenzen der Kinect V2 von Microsoft [6] erläutert.

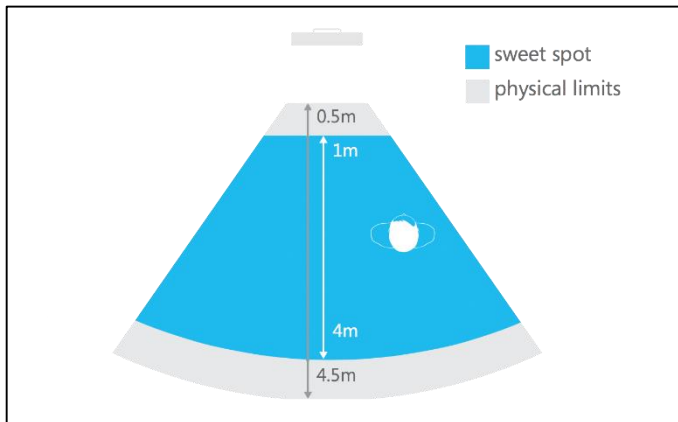


Abbildung 1 Kinect V2 Erfassungsbereich Vogelperspektive

Der Erfassungsbereich erstreckt sich über eine Distanz von 4 Metern in einem horizontalen Winkel von 70 Grad.

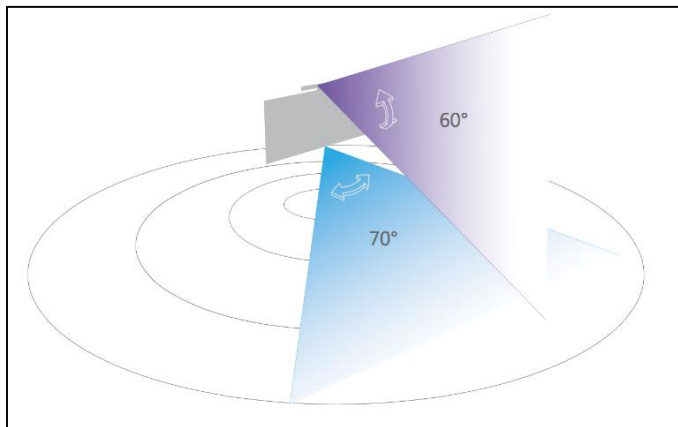


Abbildung 2 Kinect V2 Erfassungswinkel Horizontal / Vertikal

Nebst dem horizontalen Winkel ist der Erfassungsbereich in der vertikalen Ebene 60 Grad.

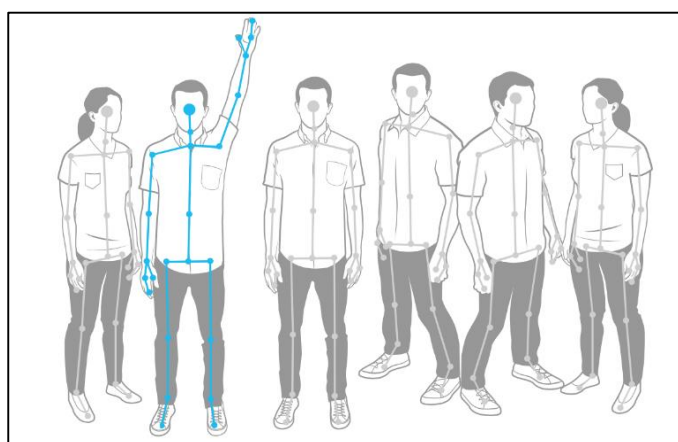


Abbildung 3 Skelett Tracking

Grundsätzlich kann eine Gruppe von bis zu 6 Personen mittels Skeletttracking (Infrarot Sensor) und deren Gesten erkannt werden.

Pro Person können bis zu 25 sogenannte Joints (Gelenke) erkannt werden. Diese sind auf dem ganzen Körper verteilt.

Mindestanforderungen

Zum Betreiben einer Microsoft Kinect V2 muss das System gewisse Mindestanforderungen [7] erfüllen. Es gibt ein „Kinect Configuration Verifier tool“ [8] welches die Systemanforderungen überprüft und eventuelle Lücken aufzeigt.

Folgende Systemanforderungen gelten für die Kinect V2 [7]:

- 64 bit Prozessor
- Physischer Dual-core 3.1 GHz Prozessor
- USB 3.0 Controller
- 4GB RAM
- Grafiksupport für DirectX 11
- Windows 8, 8.1 oder Windows Embedded 8

2.5.5 Lösungsansätze

Da das Sensing Device auf verschiedenste Arten im Raum positioniert werden kann, werden verschiedene Lösungsansätze untersucht. Es soll herausgefunden werden, wie die physikalische Anordnung aussehen könnte und welche Vor- und Nachteile eines einzigen Sensing Device gegenüber mehreren in einem Raum sein können.

2.5.5.1 Single Sensing Device

Eine Möglichkeit wäre die ganze Umgebung mit einem einzigen Sensing Device aufzubauen. Vorteil dieser Variante wäre, dass mit einem Sensor die Komplexität kleiner gehalten werden könnte. Als erster Schritt wird eine solche Umsetzung angestrebt um das Zusammenspiel zwischen den verschiedenen Komponenten zu testen.

Eine Umgebung mit einem Sensor in einem Raum der zirka 15m x 15m gross ist, würde ungefähr wie folgt aussehen:

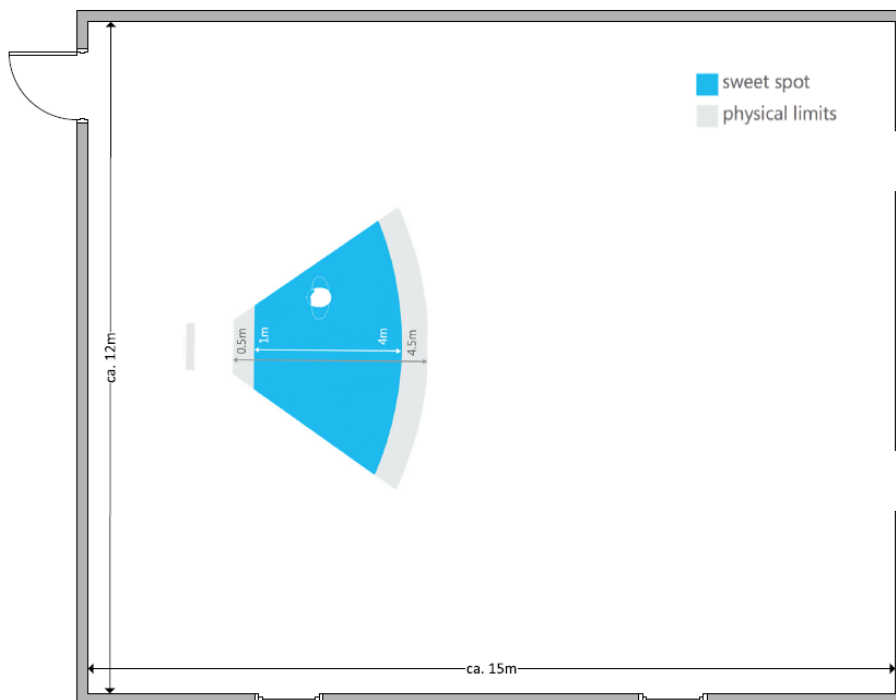


Abbildung 4 Darstellung Lösungsansatz Single Sensing Device

Vorteile	Nachteile
Kleine Latenzzeit, da Controller und Ludo-sonica an einem Gerät sind und keine Kommunikation über ein externes Netzwerk stattfindet	Es können maximal 6 Personen gleichzeitig erkannt werden.
Einfacher Aufbau der Umgebung, da nicht mehrere Sensing Devices aufeinander abgestimmt werden müssen.	Platz an welchem Personen und Gesten erkannt werden, ist kleiner als bei den weiteren Varianten
Es wird weniger Platz im Raum benötigt.	-
Nur eine Kinect und ein Computer werden benötigt	-
Komplexität ist gegenüber den Varianten mit mehreren Geräten oder mit Bildverarbeitung kleiner	-

Tabelle 10 Vor- und Nachteile Single Sensing Device

2.5.5.2 Multiple Sensing Devices

Da mit einem Sensor nur maximal 6 Personen erkannt werden können und der horizontale Winkel auf 70 Grad beschränkt ist, sollen auch Konzeptvarianten mit mehreren Sensing Devices geprüft werden. Die technische Voraussetzung, um mit einem Gerät mehrere Kinect V2 anzusteuern, ist vom Hersteller (Microsoft) nicht gegeben. Mit dem momentanen SDK ist es nur möglich, ein Sensing Device anzusteuern.

Variante 1

Mit 6 Devices könnte ein nachfolgender Ansatz verfolgt werden. Es wäre so eventuell möglich, bis zu 36 ($6 \cdot 6$) Skelette zu erkennen. Auch könnte ein grösserer Bereich abgedeckt werden. Ob sich die Bereiche überschneiden dürfen und wie das System reagiert, ist rein hypothetisch und müsste auf Machbarkeit in der Implementationsphase geprüft werden. Physisch würde diese Variante etwa wie folgt aussehen:



Abbildung 5 Darstellung Lösungsansatz 1 Multiple Sensing Device

Vorteile	Nachteile
Es können bis zu 36 Personen maximal erkannt werden.	Alle Computer müssen mit Ludosonica verbunden sein → Latenzzeit wird erhöht
Grundsätzlich keine Limitierung auf maximale Anzahl Geräte	Für jede Kinect muss ein Computer zur Verfügung stehen.

Tabelle 11 Vor- und Nachteile Multiple Sensing Devices Variante 1

Variante 2

Als zweite Variante eines „Multiple Device“ Ansatzes wird eine kreisartige Anordnung mit 5 Geräten in Betracht gezogen. Wenn die 5 Devices alle zentral und voneinander weggerichtet aufgestellt würden, dann könnte um den Mittelpunkt ein ringförmiges Bewegungsfeld mit einem 350 Grad Winkel und einem 4.5m Radius erreicht werden. Dies würde dann in etwa wie folgt aussehen:

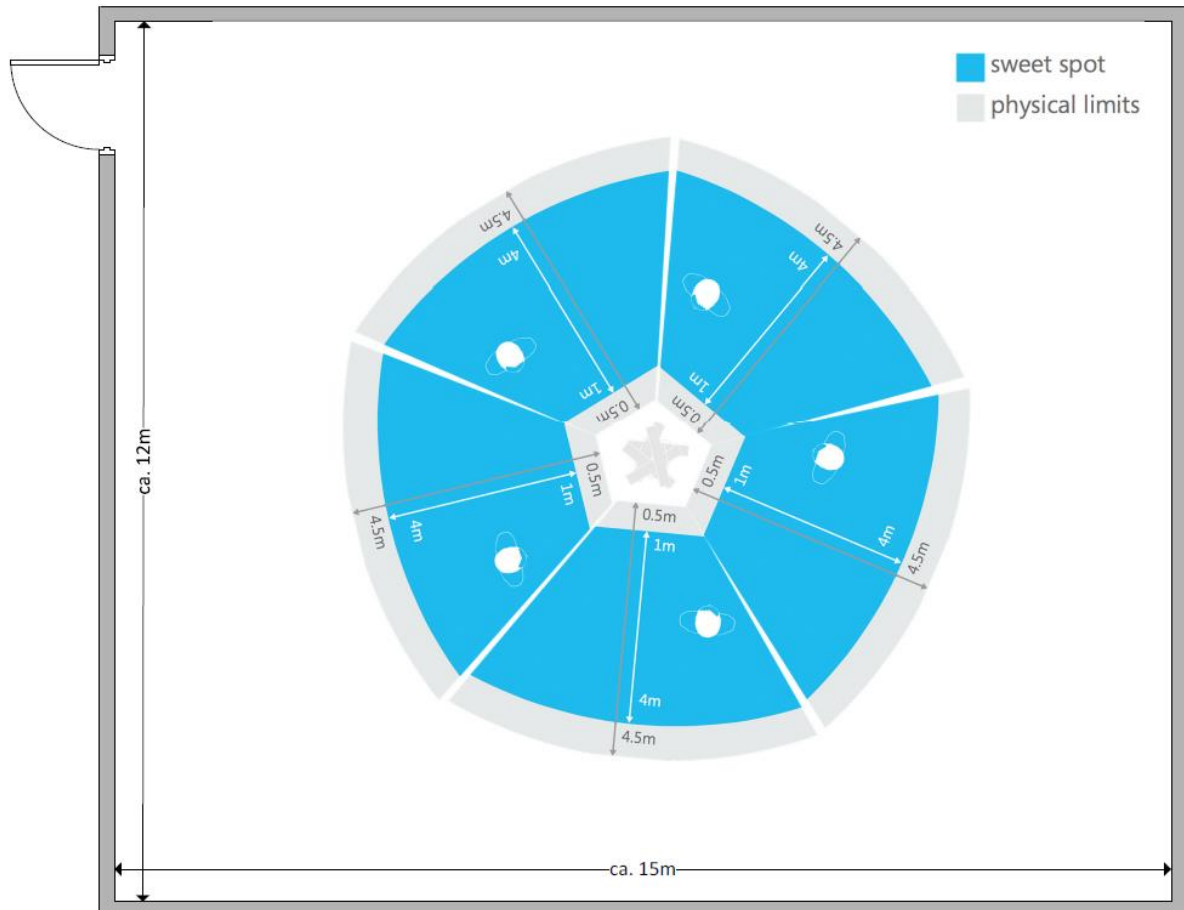


Abbildung 6 Darstellung Lösungsansatz 2 Multiple Sensing Device

Vorteile	Nachteile
Alle Kinects können an einer Installation befestigt werden → Nur einmalige Platzierung und Kalibrierung nötig	Alle Computer müssen mit Ludosonica verbunden sein → Latenzzeit wird erhöht
Es können bis zu maximal 30 Personen gleichzeitig erkannt werden.	Für jede Kinect muss ein Computer zur Verfügung stehen.
-	Limitierung auf maximal 5 Geräte

Tabelle 12 Vor- und Nachteile Multiple Sensing Devices Variante 2

2.5.5.3 Bewegungstracking mit Bildverarbeitung

Ein weiterer Ansatz ist die Möglichkeit, die RGB Kamera der Kinect mittels Bildverarbeitung zu nutzen, um die Reichweite der Gesten- und Positionserkennung zu erweitern. Dazu sollen die Personen im Raum sowie deren Bewegungen mittels Bildverarbeitungs-Libraries, wie beispielsweise OpenCV, erkannt werden. Die Erkennung von unterschiedlichen Bewegungen ist durch dieses Verfahren jedoch nur bedingt möglich. Da die Position der Personen wegen begrenzter Reichweite nicht mehr mit den Infrarot Sensoren ermittelt werden kann, müsste ein anderer Ansatz gewählt werden. Eine Möglichkeit wäre die Nutzung von Referenzkörpergrößen, anhand derer die Distanz zu den im Bild vorhandener Personen ermittelt werden könnte.

In dieser Variante muss die Kinect so positioniert werden, dass möglichst der ganze Raum überblickt und alle Personen erfasst werden können. Dazu würde sich eine möglichst hohe Position in einer Raum Ecke eignen.

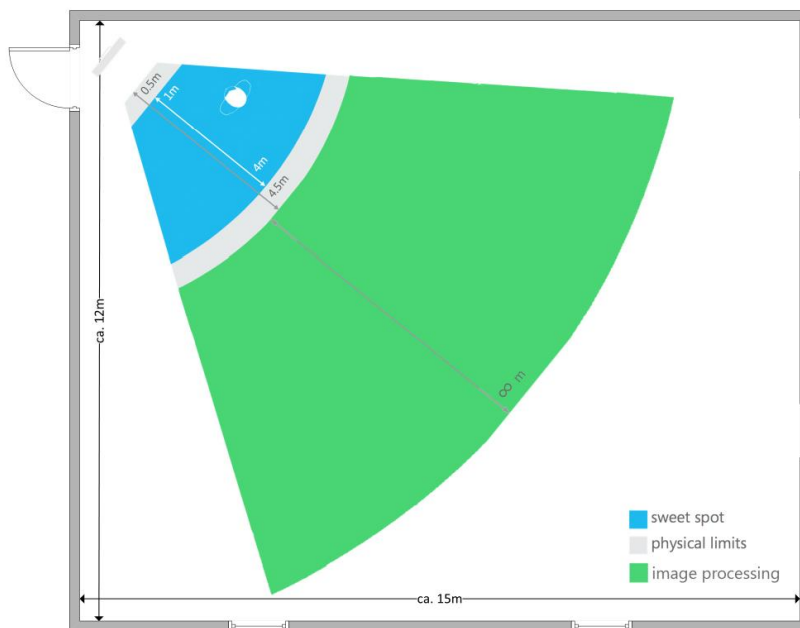


Abbildung 7 Darstellung Variante mittels Bildverarbeitung

Vorteile	Nachteile
Theoretisch unbegrenzter Radius	Komplexe Implementierung mit Bildverarbeitung
Mehr als 6 Personen könnten erkannt werden.	Zuverlässigkeit der Erkennung

Tabelle 13 Vor- und Nachteile Bewegungstracking mit Bildverarbeitung

2.5.5.4 Fazit

In Zuge dieses Projektes wird in der Realisierungsphase ausschliesslich die erste Variante verfolgt. Die Umsetzung mit mehreren Devices könnte in einem weiteren Schritt geprüft werden, jedoch ist dies nicht Bestandteil dieses Projektes. Weiter ist auch die Bildverarbeitung ein eigenes Gebiet das sehr interessante Ansätze mit sich bringt, jedoch in diesem Projekt keine Anforderung ist.

2.5.6 Positionierung Kinect V2

Für die Platzierung der Kinect werden die folgenden Einstellungen empfohlen:

- Die Kinect sollte auf einer Höhe von 1.20 Meter aufgestellt werden
- Der Neigungswinkel der Kamera sollte 90 Grad betragen
- Vor der Kinect sollte ein Feld von zirka 5x5 Meter frei sein, was ungefähr dem Bereich entspricht, worin Körper erkannt werden können

Bei den oben genannten Einstellungen erzielt die Kinect für dieses Projekt die besten Resultate. Zwar wird von Microsoft [9] eine Höhe von 1.80 Meter als Optimum angegeben. Dies schränkt jedoch den Erfassungsbereich für unseren Anwendungsfall massgeblich ein. Einen Neigungswinkel von 90 Grad ist einerseits optimal für die Erfassung von Körpern, andererseits kann dieser ohne Winkelmessung eingestellt werden.















Abbildung 8 Kinect Einstellungen

2.5.7 Gesten

Es wird zwischen statischen (discrete) und fliessenden (progress) Gesten unterschieden. Statische Gesten sind solche, die entweder erkannt oder nicht erkannt werden. Hingegen wird bei fliessenden Gesten zusätzlich einen Verlauf detektiert der erkennt, in welcher Position die Geste aktuell ist. Um ein Musikstück ein und aus zu schalten eignet sich demnach klar eine statische Geste. Jedoch um die Lautstärke eines Stücks zu verändern, bietet sich eine fliessende Geste an. So kann grundsätzlich stufenlos ein Wert verändert werden.

Die nachfolgenden Gestenbilder, welche auch in Ludosonica zur visuellen Rückmeldung verwendet werden, wurden von Fiona Nüesch illustriert.

Die folgenden statischen Gesten werden umgesetzt:

Statische Gesten (realisierbar)	
Linker Arm nach links ausstrecken stretch left arm left	
Linker Arm nach oben ausstrecken stretch left arm up	
Linker Arm nach vorne ausstrecken stretch left arm front	
Rechter Arm nach rechts ausstrecken stretch right arm right	
Rechter Arm nach oben ausstrecken stretch right arm up	
Rechter Arm nach vorne ausstrecken stretch right arm front	
Beide Arme seitlich ausstrecken stretch both arms sideways	
Beide Arme nach vorne ausstrecken stretch both arms front	
Linkes Bein ausstrecken stretch left leg	
Rechtes Bein ausstrecken stretch right leg	
Kick mit dem linken Bein kick left	
Kick mit dem rechten Bein kick right	






Hüpfen jump	
Kniebeuge squat	
Bücken stoop	
Klatschen vorne clap front	
Klatschen über Kopf clap up	

Tabelle 14 Realisierbare statische Gesten

Einige der angedachten Gesten haben sich als nicht sinnvoll erwiesen. Stampfen ist eine Geste die nicht umgesetzt werden kann, da die Haltung des Beins beim Stampfen die gleiche ist wie in der stehenden Grundposition der Person. Beide Arme nach oben ausstrecken wird weggelassen, da diese Geste nicht zusammen mit Klatschen über dem Kopf funktioniert. Statische Gesten dürfen nicht gleichzeitig ausgelöst werden, da ansonsten jede Geste mehrere Male ausgeführt wird.

Statische Gesten (nicht realisierbar)
Stampfen links stamp left
Stampfen rechts stamp right
Beide Arme nach oben ausstrecken stretch both arms up

Tabelle 15 Nicht realisierbare statische Gesten

Die folgenden fließenden Gesten werden umgesetzt:









Fließende Gesten (realisierbar)	
Oberkörper nach vorne beugen bend upper body front	
Kopf nach vorne beugen bend head front	
Linker Arm ausgestreckt seitlich heben lift stretched left arm	
Rechter Arm ausgestreckt seitlich heben lift stretched right arm	
Linker Arm ausgestreckt vorne heben lift stretched left arm front	
Rechter Arm ausgestreckt vorne heben lift stretched right arm front	
Linker Arm gegen oben heben lift left arm up	
Rechter Arm gegen oben heben lift right arm up	
Linkes Knie anziehen pull left knee up	
Rechtes Knie anziehen pull right knee up	

Tabelle 16 Realisierbare fließende Gesten

Fließende Gesten mit Armen können nur mit der geschlossenen Hand des jeweiligen Arms ausgelöst, beziehungsweise fixiert werden. Für die weiteren fließenden Gesten muss für deren Auslösung mindestens eine Hand geschlossen sein.

In der Testphase hat sich gezeigt, dass das seitliche Beugen des Oberkörpers wie auch des Kopfes Gesten sind die von der Kinect nicht genug zuverlässig erkannt werden können. Die Geste „Arme von Mitte gegen aussen strecken“ wirkt sich beim Erkennen sehr störend auf die anderen mit den Armen ausgeführten Gesten aus.

Fließende Gesten (nicht realisierbar)
Oberkörper seitlich beugen bend upper body sideways
Kopf seitlich beugen bend head sideways
Arme von Mitte gegen aussen strecken stretch arms middle to side

Tabelle 17 Nicht realisierbare fließende Gesten

2.5.8 Klangmodifikationen

Nachdem durch eine bestimmte Geste ein Sound gestartet wurde, sollen gewisse Modifikationen möglich sein. Unter Klangmodifikationen werden nebst dem Ein- und Ausschalten von Musik und Loops, zum einen das Setzen von Filtern und deren Parameter und zum anderen das Verändern der Lautstärke, Geschwindigkeit und der Verzögerung zwischen den Tönen bei einem Loop, verstanden.

Globale Soundeinstellungen

Beim Erzeugen von rhythmischen Kompositionen ist es von hoher Wichtigkeit, dass die Töne in einem definierten Abstand aufeinander folgen. Das Ausführen von Gesten um Klänge zu starten, ist in diesem Fall sehr zeitkritisch. Dies wird durch einen Mechanismus bei welchem die Anschläge pro Minute (BPM) eingestellt werden können, erreicht. Beim Auslösen eines Sounds durch eine Geste wird dabei auf den nächsten Schlag aufgerundet. Somit entsteht ein für das menschliche Gehör rhythmischer Klang.

Werden jedoch ganze Samples, die 10-20 Sekunden oder länger dauern, gestartet, so ist es absolut notwendig, dass auch hier eine Funktion der Software hilft, den neu gestarteten Sample auf die vorherigen, bereits laufenden, zu synchronisieren. Dies wird mittels einer globalen Synchronisation realisiert.

Klangmodifikationen für statische Gesten

Um folgende Klangmodifikationen zu verwenden eignen sich statische Gesten. Ein Musikstück kann einmalig oder in einem Loop gestartet werden. Einen Hall-, Tremolo, oder WahWah-Effekt kann auf den Klängen mit statischen Gesten ein und wieder ausgeschaltet werden.

Klangmodifikationen	
Musik Ein / Aus Play sound / Stop sound	Starten und Stoppen eines Sounds.
Loop Ein / Aus Play loop / Stop loop	Starten eines Sound in einer Endlosschleife. Loop Aus stoppt den Sound.
Hall Effekt Reverb [10]	Dieser Effekt erzeugt einen künstlichen Nachhall des Sounds.
Tremolo Effekt [11]	Durch den Tremolo Effekt wird in kurzen Zeitabständen die Lautstärke des musikalischen Signals verändert.
Wah Wah Effekt [12]	Das Wort „Wah Wah“ beschreibt den Klang welcher durch diesen Effekt hervorgerufen wird.

Tabelle 18 Beschreibung von Klangmodifikationen für statische Gesten

Klangmodifikationen für fließende Gesten

Bei Klangmodifikationen, bei welchen beispielsweise die Frequenz als Parameter verändert werden muss, sind fließende Gesten sinnvoll. Da mit einer Geste nur ein Parameter verändert werden kann, wird in Ludosonica bei Filtern, welche mehrere Parameter enthalten, nur der Hauptparameter verändert. Die weiteren Parameter sind fix eingestellt.

Klangmodifikationen	
Loop Verzögerung Loop delay	Durch die Loop Verzögerung kann der Abstand zwischen dem Ende und dem nächsten Beginn eines Sounds in einem Loop eingestellt werden.
Lautstärke Volume	Verändert die Lautstärke der Sounds.
Geschwindigkeit / Pitch Speed [13]	Durch verändern der Abspielgeschwindigkeit wird die Tonhöhe (Pitch) verändert.
Lowpass [14]	Alle Frequenzen unterhalb der eingestellten Grenzfrequenz können den Filter ungehindert passieren, Frequenzen oberhalb dieser Grenze werden gedämpft.

Highpass [14]	Alle Frequenzen oberhalb der eingestellten Grenzfrequenz können den Filter ungehindert passieren, Frequenzen unterhalb dieser Grenze werden gedämpft.
Bandpass [14]	Nur Frequenzen eines Frequenzbands können den Filter ungehindert passieren, Frequenzen darunter und darüber werden gedämpft. Die gesetzte Frequenz entspricht der Mitte des Frequenzbands. Die Breite des Bands wird durch einen Q Wert [15] gesetzt, welcher fix auf 2 gesetzt ist. Dadurch ergibt sich eine Bandbreite von der Resonanzfrequenz / 2.
Lowshelf [14]	Bei den Frequenzen unterhalb der eingestellten Grenzfrequenz wird die Lautstärke erhöht. Frequenzen oberhalb des Grenzwerts werden nicht beeinflusst. Neben der Grenzfrequenz des Filters, kann auch die Lautstärke, um welche die Frequenzen verstärkt werden sollen, eingestellt werden. Diese wird in Ludosonica auf 25 dB fixiert.
Highshelf [14]	Bei den Frequenzen oberhalb der eingestellten Grenzfrequenz wird die Lautstärke erhöht. Frequenzen unterhalb des Grenzwerts werden nicht beeinflusst. Neben der Grenzfrequenz des Filters, kann auch die Lautstärke, um welche die Frequenzen verstärkt werden sollen, eingestellt werden. Diese wird in Ludosonica auf 25 dB fixiert.
Peaking [14]	Bei Frequenzen eines Frequenzbands wird die Lautstärke erhöht, Frequenzen oberhalb oder unterhalb werden nicht beeinflusst. Die gesetzte Frequenz entspricht der Mitte des Frequenzbands. Die Breite des Bands wird durch einen Q Wert [15] gesetzt, welcher fix auf 2 gesetzt ist. Dadurch ergibt sich eine Bandbreite von der Resonanzfrequenz / 2. Auch kann die Lautstärke eingestellt werden, um welche die Frequenzen verstärkt werden sollen. Diese ist in Ludosonica auf 25 dB fixiert.
Notch [14]	Der Notch Filter ist das Gegenteil des Bandpass Filters. Nur Frequenzen ausserhalb eines Frequenzbands können den Filter ungehindert passieren, Frequenzen innerhalb werden gedämpft. Die gesetzte Frequenz entspricht der Mitte des Frequenzbands. Die Breite des Bands wird durch einen Q Wert [15] gesetzt, welcher fix auf 2 gesetzt ist. Dadurch ergibt sich eine Bandbreite von der Resonanzfrequenz / 2.
Allpass [14]	Lässt alle Frequenzen passieren, ändert aber die Phasenbeziehungen zwischen den verschiedenen Frequenzen. Es entsteht ein Phaser Effekt [11].

Tabelle 19 Beschreibung von Klangmodifikationen für fließende Gesten

2.5.9 Spielabläufe

Das entwickelte System beinhaltet grundsätzlich keine grossen Einschränkungen in Bezug auf die Verwendung. Welche Sounddateien verwendet werden, ist frei wählbar. Weiter ist auch die Loop Funktion für alle Audiodateien verfügbar. Obwohl darauf geachtet wurde, dass das System offen ausgelegt ist und beliebig verwendet werden kann, haben sich ein paar interessante Spielabläufe herauskristallisiert.

Durch die Möglichkeit, dass die Beats per Minute eingestellt werden können, ist es interessant, wenn verschiedene rhythmische Loops übereinander gelegt werden. Dadurch können verschiedenste Kompositionen entstehen die eine für den Menschen angenehme Wahrnehmung haben. Die Soundmodifikation „Loop Verzögerung“ wird in diesem Kontext spannend. Dadurch wird ermöglicht, dass durch eine fließende Geste die Abstände zwischen den Tönen verändert werden kann. Dies bietet ein weiteres Mittel um die Töne unterschiedlich aufeinander abzustimmen und so können verschiedene Klangkompositionen entstehen.

Eine weitere zentrale Funktion ist das Synchronisieren von Sounds. Diese Synchronisation erlaubt es, dass mehrere, gleichlange Samples (Tonspuren) zu unterschiedlicher Zeit gestartet werden können. Der Synchronisationsmechanismus synchronisiert jeden neu gestarteten Sound auf den zuletzt gestarteten. Somit werden die Samples aufeinander abgestimmt und es entsteht eine Art Musikstück. Beim Abspielen von Samples ist es interessant wenn die verschiedenen Filter und Effekte auf die verschiedenen Tonspuren angewendet werden. Durch diese Kombination können mehrere Personen zusammen ein Musikstück spielen und es nach Belieben modifizieren.

3 Hauptteil

3.1 Technologien

Durch die Weiterverwendung von Ludosonica als Basis für unser Projekt war von Anfang an klar, dass mit den üblichen Webtechnologien wie HTML, JavaScript und CSS gearbeitet werden soll. Der „Sonic Classroom Controller“ ist in C# geschrieben. Das GUI ist mit dem Windows Presentation Foundation (WPF) Framework umgesetzt. Zum Ansprechen der Kinect V2 wird das Kinect for Windows SDK 2.0 verwendet. Die Kommunikation zwischen den beiden Komponenten wird über einen bidirektionalen WebSocket realisiert.

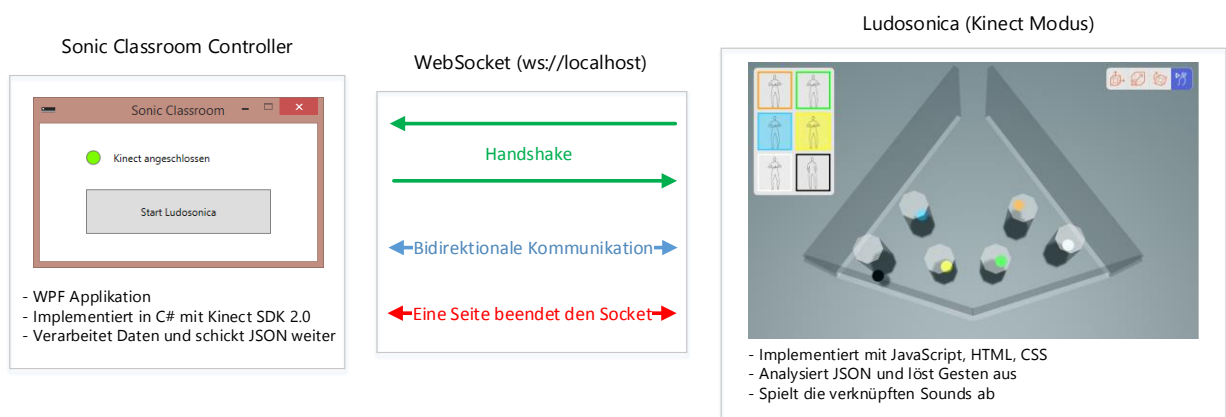


Abbildung 9 Übersicht der Komponenten

3.2 Frameworks / Libraries

3.2.1 Three.js

Three.js ist eine auf WebGL basierende JavaScript Library die es ermöglicht, animierte 3D Objekte im Browser zu erstellen und darzustellen. Im Zuge dieses Projektes wurden lediglich ein paar bereits verwendete Three.js Objekte angepasst. Die Library wurde jedoch nicht vertieft behandelt.



Abbildung 10 Three.js Beispiele (Wikipedia)

Version: 69dev (<https://github.com/mrdoob/three.js/releases/tag/r69>)
 Website: <http://threejs.org/>
 Doku: <http://threejs.org/docs/>

3.2.2 Web Audio API (Webaudiox)

Web Audio API ist eine von der W3C spezifizierte JavaScript API, welche die Arbeit mit Audio in Web Applikationen ermöglicht. Sie bietet den Entwicklern Funktionen für die Auswahl von Audio Quellen, die Bearbeitung von Sounds durch Effekte und Filter, sowie Visualisierungsmöglichkeiten von Sounds.

Website: <http://webaudio.github.io/web-audio-api/>

Doku: https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API

Webaudiox.js ist eine Ansammlung von Hilfsfunktionen für die Web Audio API. Sie bietet Funktionen an, welche die Anwendung der Web Audio API vereinfachen sollen.

Version: v1.0.1

Website/Doku: <https://github.com/jeromeetienne/webaudiox>

3.2.3 Tuna.js

Tuna.js ist eine Audio Effekt Library von Dinahmo. Sie stellt eine Reihe von Effekten für die Web Audio API zur Verfügung, welche teilweise in diesem Projekt benutzt werden.

Website/Doku: <https://github.com/Theodeus/tuna>

3.2.4 jQuery UI

jQuery UI ist eine Erweiterung der Open Source JavaScript Library JQuery. Sie enthält eine Vielzahl von User Interface Interaktionen, Widgets und Effekten. Die jQuery UI Library wird für die visuelle Rückmeldung der Gesten benötigt.



Abbildung 11 Logo jQuery UI (Wikipedia)

Version: 1.11.4

Website: <http://jqueryui.com/>

Doku: <http://api.jqueryui.com/>

3.2.5 Bootstrap

Bootstrap ist ein HTML, CSS und JavaScript Framework von Twitter. Das Framework enthält verschiedene Gestaltungsvorlagen für Buttons, Tabellen und weitere GUI Elemente. Bootstrap wird zur Gestaltung der Startseite von Ludosonica verwendet.



Abbildung 12 Logo Bootstrap (Wikipedia)

Version: v3.3.5

Website: <http://getbootstrap.com/>

Doku: <http://getbootstrap.com/getting-started/>

3.2.6 Kinect for Windows SDK 2.0

Die Kinect for Windows SDK 2.0 bietet verschiedene Tools, Treiber, API's und Beispielcode für Applikationen, welche mit der Kinect V2 arbeiten. In diesem Projekt werden zur Programmierung die API's „Windows.Kinect“ und „Windows.Kinect.VisualGestureBuilder“ benutzt. Die Aufnahme der Gesten Clips wird mit dem Kinect Studio v2.0 gemacht. Für die Erstellung der Gesten wird der Visual Gesture Builder verwendet.

Version: 2.0.1410.19000
Website: <https://www.microsoft.com/en-us/kinectforwindows/develop/>
Doku: <https://msdn.microsoft.com/en-us/library/dn799271.aspx>

3.2.7 Alchemy Websockets

Die Alchemy Websockets Library von Olivine Labs bietet eine WebSocket Server Implementierung für C# Applikationen. Die Library unterstützt die gängigsten Versionen des WebSocket Protokolls.

Da die aktuellste Version der Alchemy Websockets Library kein Trennen der Verbindung von Seite des Servers zur Verfügung stellt, wird der aktuellste GitHub Fork des Users SteForster verwendet. Aufgrund eines Bugs in dieser Version der Library beim Schliessen der Applikation, wurden einige Anpassungen vorgenommen. Dabei wurden alle von der Library gestarteten Threads in Background Threads umgewandelt. Somit werden die beim Schliessen bestehenden Threads geschlossen und die Applikation kann beendet werden.

Version: Fork von SteForster (31.12.2014) der offiziellen Version 2.2.1.238
Fork: <https://github.com/steforster/Alchemy-Websockets>
Website: <http://olivilabs.com/Alchemy-Websockets/>
Doku: <http://olivilabs.com/Alchemy-Websockets/docs/>

3.3 Sonic Classroom Controller

3.3.1 Übersicht

Der Sonic Classroom Controller ist auf den Grundlagen des Beispiels „DiscreteGestureBasics-WPF“ der Kinect for Windows SDK 2.0 aufgebaut. Der Controller dient zum Verarbeiten des Kinect Datenstroms und zum Weiterleiten der aufbereiteten Daten an Ludosonica.

Das GUI der Applikation ist mit dem Windows Presentation Foundation Framework (WPF) gestaltet. Das Design wird bewusst schlicht gehalten, da es einzig zur Darstellung des Status der angeschlossenen Kinect und zum Starten von Ludosonica dient. Der Link zu Ludosonica kann in den Resources der Applikation geändert werden.

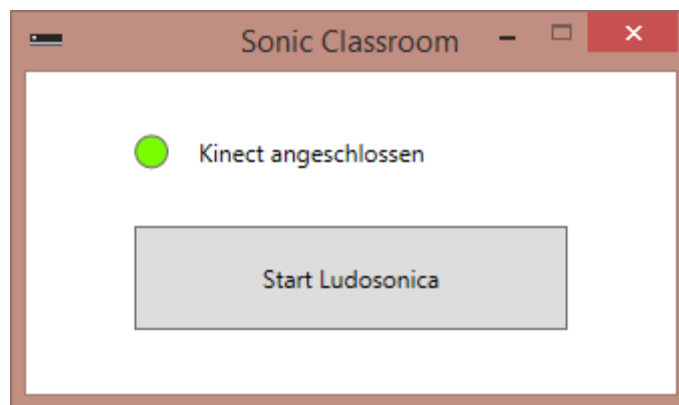


Abbildung 13 GUI Sonic Classroom Controller

Die nachfolgende Tabelle zeigt eine Übersicht über die Klassen des Sonic Classroom Controllers und ihre Aufgabenbereiche.

Klasse	Hauptaufgaben
MainWindow.xaml MainWindow.xaml.cs	<ul style="list-style-type: none"> - Aufbau und Abbau der Applikation (Verbindung zur Kinect, Start / Stop des KinectWebSocketServers, BodyFrameReader) - Darstellung des GUIs mit Binding zum Kinect Status - Empfangen von Body Frames, welche Informationen zu den erkannten Körpern liefert - Instanzieren und aktualisieren (Bodies, Tracking IDs) der Gesture Detectors (Ein Gesture Detector pro Körper)
Gestures/ GestureDetector.cs	<ul style="list-style-type: none"> - Verbindung zur Gesten Datenbank (Gestures/SonicClassroomGestures.gbd) - Empfangen von Gesture Frames und Erkennen von Gesten - Broadcast von Gesten mit KinectWebSocketServer und JSONParser
Server/ KinectWebSocketServer.cs	<ul style="list-style-type: none"> - Verbindungen aufbauen, schliessen - Senden, Broadcast, empfangen von Mitteilungen
JSONParser.cs	<ul style="list-style-type: none"> - Parsen von Gestendaten in JSON Format

Tabelle 20 Übersicht über die Hauptaufgaben aller Klassen des Sonic Classroom Controllers

Das Klassendiagramm des Sonic Classroom Controllers zeigt die Einzelheiten der Klassen und deren Beziehungen. Da es sich bei der Klasse „JSONParser.cs“ um eine statische Klasse handelt, ist die Verbindung mit der Klasse „GestureDetector.cs“ nicht ersichtlich.

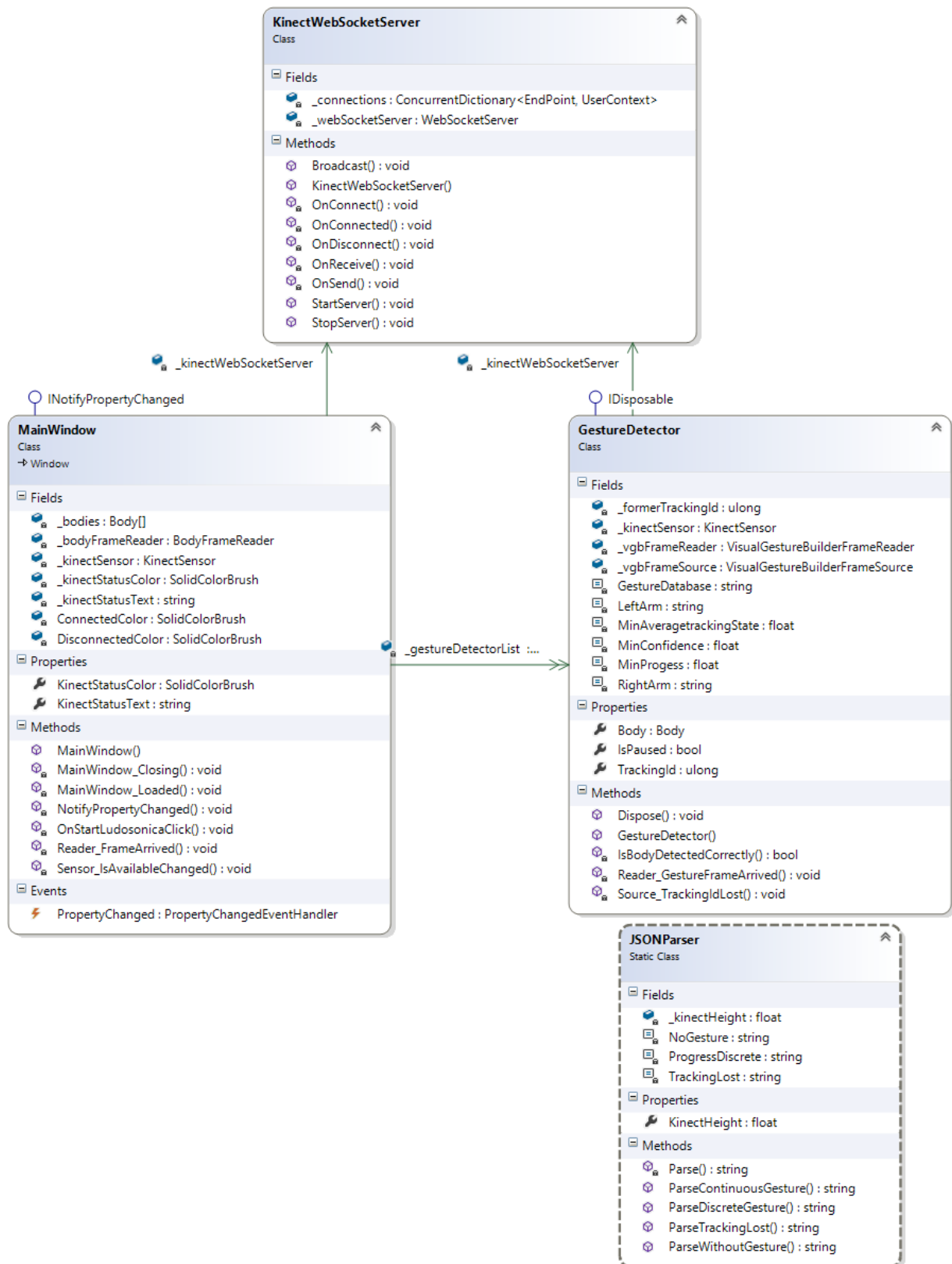


Abbildung 14 Klassendiagramm Sonic Classroom Controller

Das folgende Flussdiagramm stellt den groben Ablauf vom Empfang von Frames, welche von der Kinect gesendet werden, bis zum Versenden der Gesten- und Positionsdaten an Ludosonica dar. Die genaueren Abläufe zur Erkennung von Gesten und Position sowie dem parsen und versenden von Daten ist in den nächsten Kapiteln erklärt.

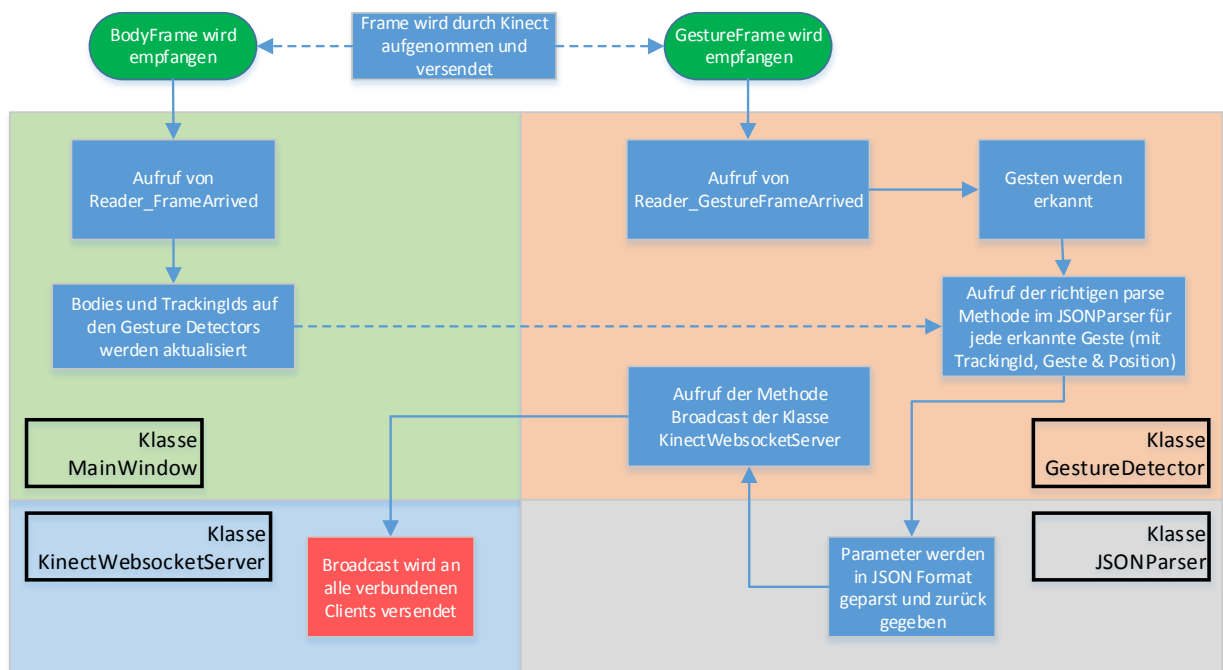


Abbildung 15 Flussdiagramm Grobablauf Sonic Classroom Controller

Detailliertere Klassen- und Sequenzdiagramme des Sonic Classroom Controllers sind im Visual Studio Projekt unter den Ordnern „SequenceDiagrams“ und „ClassDiagrams“ zu finden.

3.3.2 Gestenerkennung

Für die Erkennung von Gesten stellt das Kinect for Windows SDK 2.0 mit dem Visual Gesture Builder ein mächtiges Tool zur Verfügung. Das Tool ermöglicht das Erstellen von Gesten mit Hilfe von aufgenommenen Clips dieser Gesten. Durch Markierungen in den Clips kann, mittels Machine Learning Algorithmen bestimmt werden, wie eine Geste auszusehen hat. Dies vereinfacht die Erkennung von Gesten gegenüber der heuristischen Methode ungemein. Beispielsweise wird laut Microsoft [16, p. 3] für die Geste Winken, um zuverlässig erkannt zu werden, zirka 550 Zeilen Code benötigt. Mit dem Visual Gesture Builder wird sowohl der Aufwand als auch die Komplexität zur Erstellung der Gesten markant verringert.

Folgende Grafik zeigt den groben Ablauf vom Aufnehmen der Clips bis hin zur Datenbank, welche in die Applikation integriert werden kann.

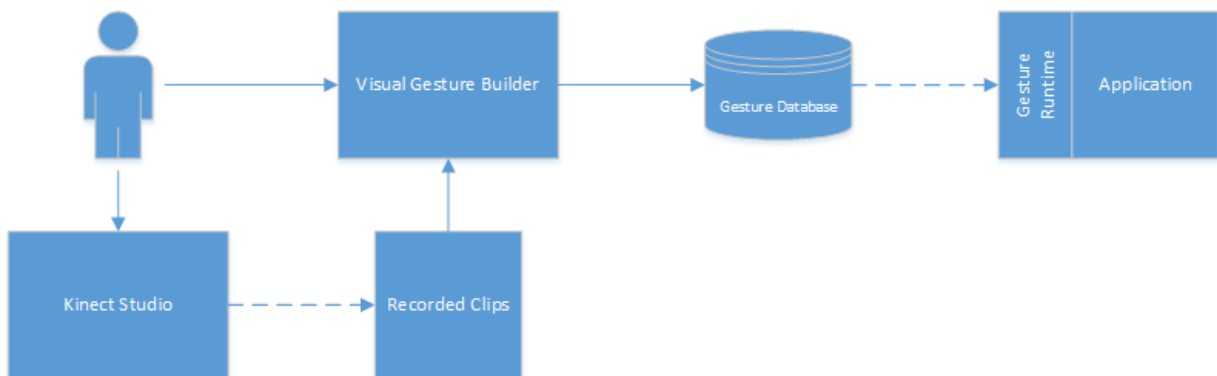


Abbildung 16 Visual Gesture Builder im Kontext [27]

Für dieses Projekt wurden für alle Gesten, Clips mit mehreren Personen aufgenommen. Damit wird eine gewisse Bandbreite über verschiedene Körpergrößen und Alter erreicht. In der nachfolgenden Tabelle sind die Personen aufgelistet.

Name	Alter [Jahre]	Grösse [Meter]
Delia Tiefenauer	10	1.35
Florian Rüegg	23	1.75
Florin Tiefenauer	26	1.90
Jonas Tiefenauer	11	1.45
Simone Gossweiler	25	1.60

Tabelle 21 Übersicht über Personen für die Aufnahme der Gesten

Bei vereinzelt Gesten mussten für eine verbesserte Erkennung zusätzliche Clips mit weiteren Personen aufgenommen werden. Eine hundertprozentige Zuverlässigkeit ist mit dieser Menge an Clips nicht möglich. Da es sich bei diesem Projekt um einen Prototypen handelt, ist dies jedoch auch nicht gefordert. Eine abschliessende Antwort auf die Frage, wie viele Clips für eine zuverlässige Geste benötigt werden, gibt es laut Microsoft [16, p. 8] nicht.

Aufnahme von Clips mit Kinect Studio

Die Clips werden mit dem ebenfalls in der Kinect for Windows SDK 2.0 enthaltenen Kinect Studio v2.0 aufgezeichnet. Um von zukünftigen Weiterentwicklungen des Depth / Skeleton Tracking von Microsoft profitieren zu können, wird empfohlen, [16, p. 6] die Clips im 11bit Raw Format (XRF, eXtended Raw File) aufzuzeichnen.

Die folgende Abbildung zeigt die Einstellungen zur Aufzeichnung von Raw Clips.

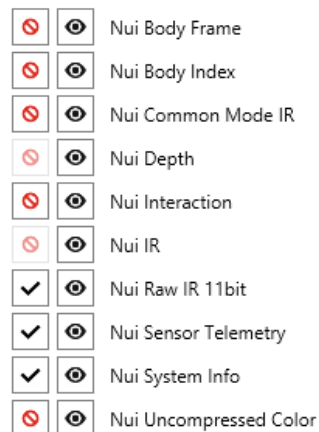


Abbildung 17 Einstellungen für
Aufnahmen im Raw Format (XRF)

Die Clips im XRF Format müssen für die Verarbeitung mit dem Visual Gesture Builder in XEF (eXtended Event File) Clips konvertiert werden. Dazu wird das Kommandozeilen Befehlstool KSConvert.exe verwendet, welches ebenfalls im SDK enthalten ist. Der folgende Befehl konvertiert alle XRF Files im „raw“ Ordner und kopiert die XEF Files in den Ordner „xrf“. Der Parameter StripColor zeigt an, dass die Farbinformationen nicht konvertiert werden. Diese werden für die Gestenerkennung nicht benötigt und würden nur unnötigen Speicherplatz verbrauchen.

```
KSConvert.exe /s "path\raw" "path\xrf" /StripColor
```

Gestenerkennung mit Visual Gesture Builder

Zur Erkennung von Gesten werden vom Visual Gesture Builder zwei Algorithmen angeboten. Mit dem AdaBoostTrigger Algorithmus [17] können statische Gesten erkannt werden. Diese Gesten werden entweder erkannt oder nicht. Es gibt keinen Verlauf der Geste, einzig die Zuverlässigkeit wird mit einem Wert zwischen 0 und 1 angegeben. Der RFRProgress Algorithmus [18] ist für die fließenden Gesten geeignet, da er den Verlauf der Gesten erkennt.

Für die Erstellung einer Gestendatenbank wird im Visual Gesture Builder eine sogenannte Solution erstellt, welche mehrere Projects enthält. Pro Geste gibt es ein Project worin alle dafür benötigten Clips enthalten sind.

Bei den statischen Gesten mit dem AdaBoostTrigger Algorithmus werden alle Frames, welche die gewünschte Geste beinhalten, markiert. Im folgenden Beispiel, erkennt man im unteren Teil „Control“ die Frames, welche markiert sind. Bei diesen wird der Wert der Geste bei den Properties rechts oben auf „True“ gesetzt.

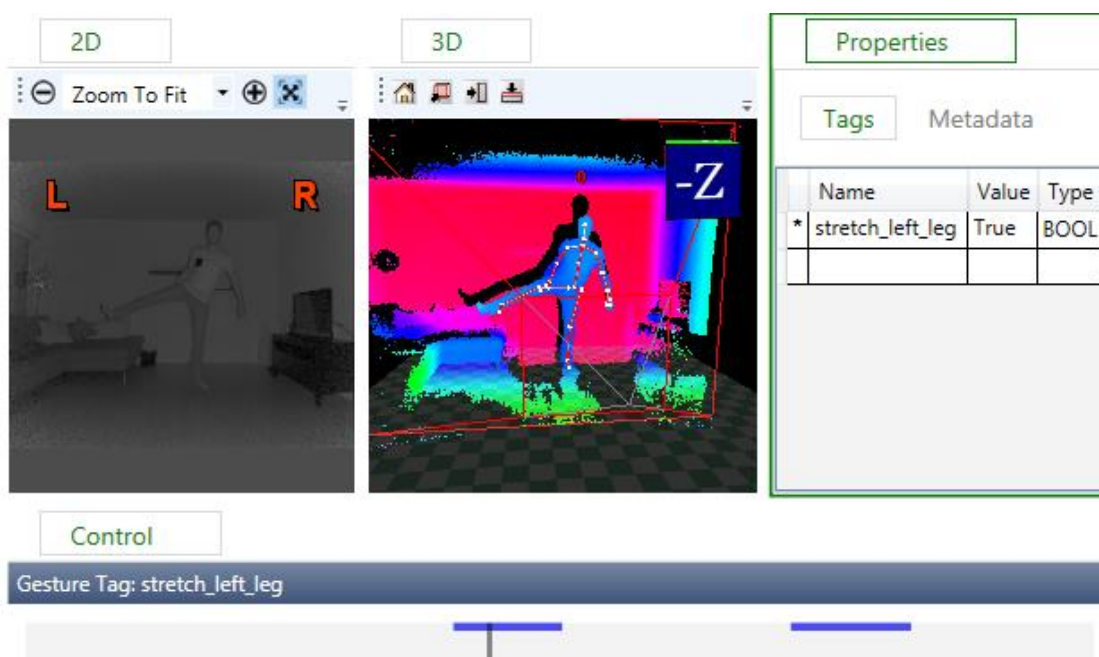


Abbildung 18 Visual Gesture Builder Markierungen für statische Gesten

Falls eine statische Geste auch bei einer Bewegung ausgelöst wird, welche nicht Teil der eigentlichen Geste ist, kann für diese Bewegung ein Clip hinzugefügt werden. Dieser Clip wird nicht markiert und somit erkennt der Algorithmus, dass diese Bewegung nicht Teil der Geste ist.

Bei der Markierung von fließenden Gesten für den RFRProgress Algorithmus, kann rein theoretisch jedem Frame ein Wert für den Fortschritt im Verlauf der Geste gesetzt werden. Bei der praktischen Umsetzung wird jedoch meist nur der tiefste und höchste Fortschritt markiert. Die weiteren Werte dazwischen werden berechnet. Im nachfolgenden Beispiel ist die Verlaufskurve mit den einzelnen Markierungen der tiefsten und höchsten Werte ersichtlich.

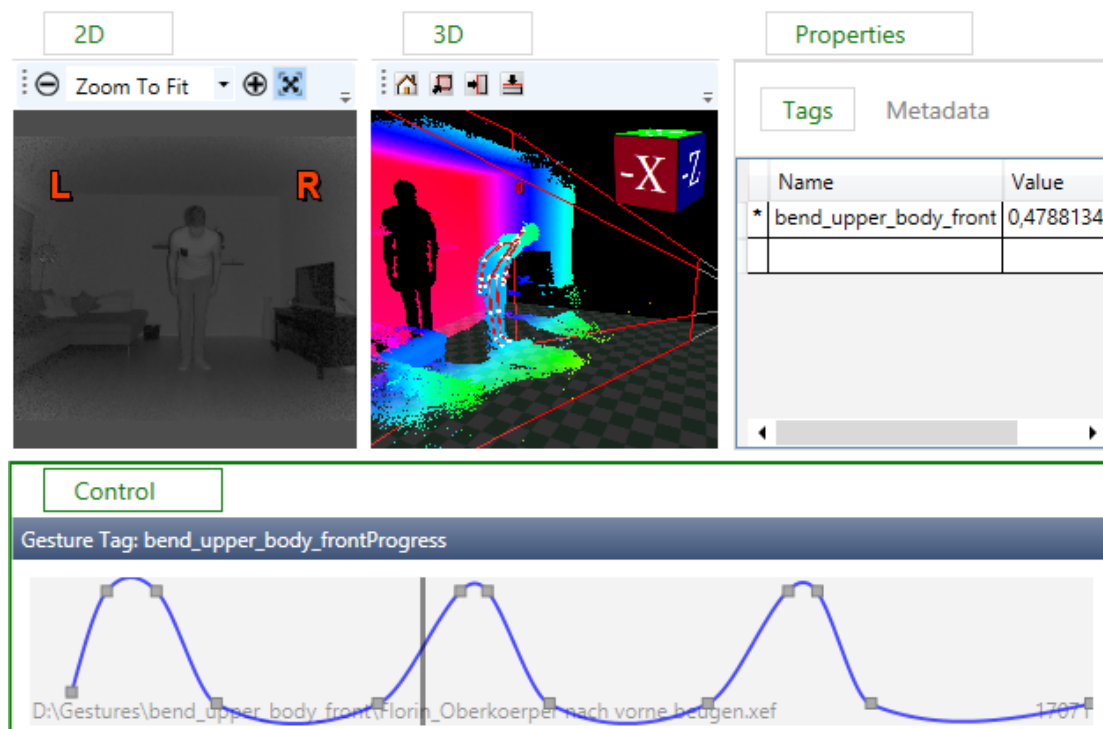


Abbildung 19 Visual Gesture Builder Markierungen für fließende Gesten

Auch für fließende Gesten können negativ Beispiele hinzugefügt werden. Da der RFRProgress Algorithmus nur Frames analysiert, welche mit einem Wert versehen sind, müssen bei diesen Clips alle Frames mit dem Wert 0 markiert werden.

Verarbeitung der Gesten im Sonic Classroom Controller

Damit die Gestenerkennung funktioniert, müssen für beide Gestenerkennungsalgorithmen Libraries [19] hinzugefügt werden. Die Libraries „AdaBoostTech.dll“ und „RFRProgressTech.dll“ liegen im Ordner vgbtechs und werden beim Build in den „bin“ Ordner kopiert.

Die API stellt zur Erkennung der Gesten die Klassen „VisualGestureBuilderFrameSource“ (Frame Source) und „VisualGestureBuilderFrameReader“ (Frame Reader) zur Verfügung. Die Frame Source ist mit dem Kinect Sensor und der Gestendatenbank verbunden und enthält die Tracking ID eines Körpers. Der Frame Reader wird über die Frame Source gestartet und empfängt die Gesture Frames, welche mit einer Callback Methode verarbeitet werden. Über das „IsPaused“ Flag des Frame Readers kann das Empfangen von Gesture Frames für inaktive Körper pausiert werden.

Folgendes Flussdiagramm stellt den Ablauf der Callback Methode „Reader_GestureFrameArrived“ dar, welche bei jedem ankommenden Gesture Frame aufgerufen wird.

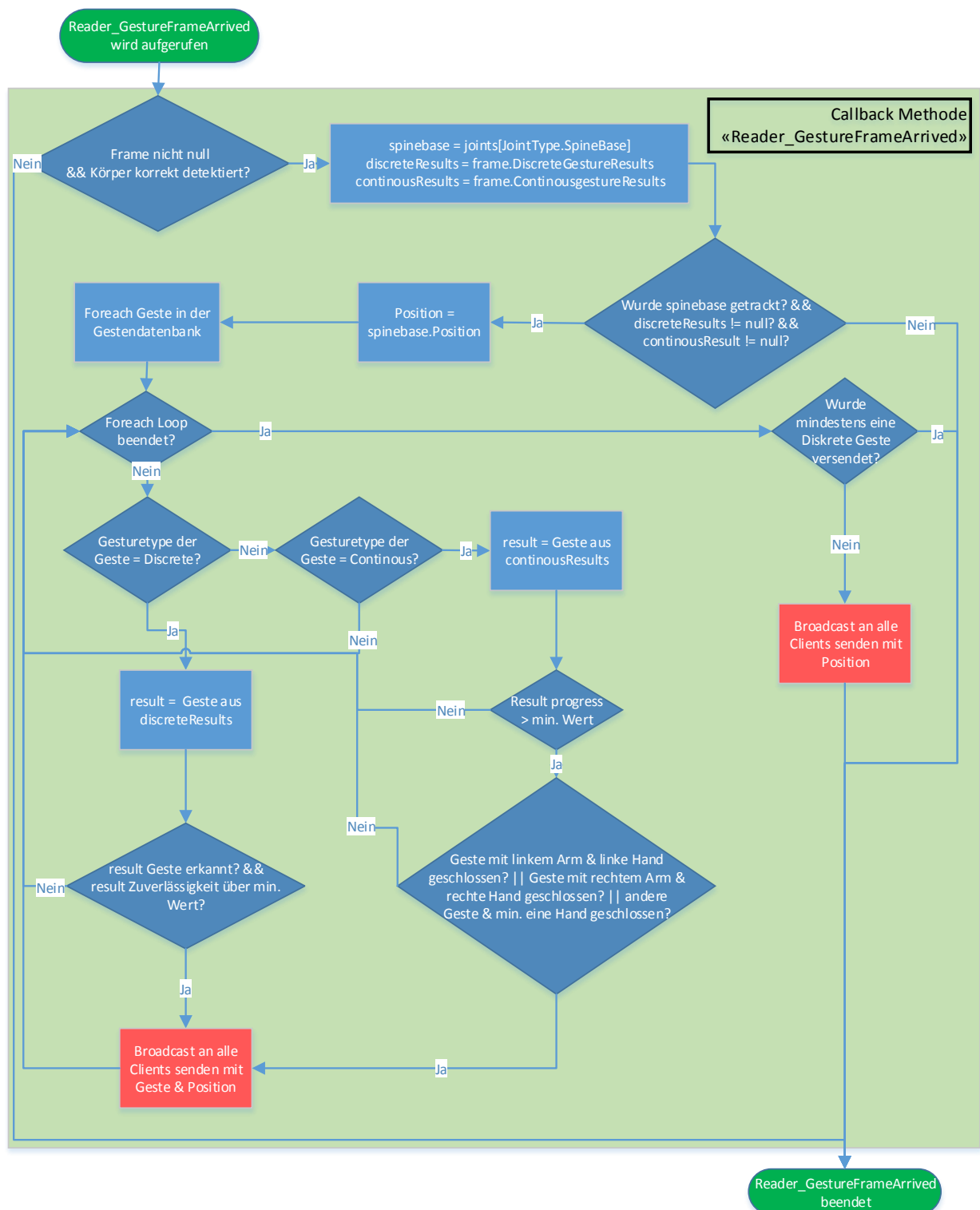


Abbildung 20 Flussdiagramm der Gestenerkennung im Sonic Classroom Controller

3.3.3 Positionserkennung

Die Position der Person wird anhand des „spine_base“ Gelenks ermittelt. Dabei handelt es sich um den Mittelpunkt der Hüfte. Dieser Punkt wird gewählt, weil es sich dabei bei den meisten Bewegungen um den stabilsten Punkt des Körpers handelt. Somit wird verhindert, dass sich durch das Ausführen einer Geste die Position der Person ändert. Das nachfolgende Bild zeigt eine Übersicht über alle Gelenktypen (JointTypes), welche von der Kinect erkannt und verarbeitet werden.

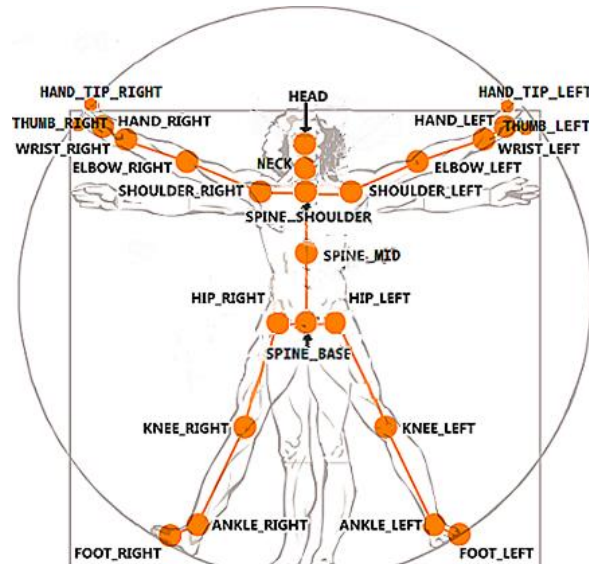


Abbildung 21 Übersicht über alle JointTypes [28]

Die Position der Gelenke kann abgefragt werden und wird als dreidimensionaler Punkt im Koordinatensystem zurückgegeben. Das Koordinatensystem hat folgende Eigenschaften [20]:

- Der Nullpunkt ($x=0$, $y=0$, $z=0$) liegt in der Mitte des Infrarot Sensors der Kinect
- X wächst vom Sensor aus gesehen in die Linke Richtung
- Y wächst gegen oben (Bemerkung: Die Richtung ist abhängig von der Neigung des Sensors)
- Z wächst in die Richtung in welche der Sensor zeigt
- 1 Einheit = 1 Meter



Abbildung 22 Koordinaten System der Kinect V2 [20]

3.4 Datenübermittlung

3.4.1 JSON

Für die Übertragung der Gesten- und Positionsdaten an Ludosonica wird JSON als Datenformat gewählt. Die zur Übermittlung definierte Datenstruktur besteht aus einem Objekt, welches verschiedene Key-Value-Paare beinhaltet, die in der nachfolgenden Tabelle kurz definiert und erläutert werden.

Key	Value
trackingId	Tracking ID des Körpers (wird von der Kinect definiert)
x	x-Koordinate der Position des Körpers
y	y-Koordinate der Position des Körpers
z	z-Koordinate der Position des Körpers
gesture	Name der zu versendenden Geste Ausnahmen: <ul style="list-style-type: none"> - Bei Übermittlung der Position ohne Geste: „gesture“=“none“ - Wenn ein Körper von der Kinect nicht mehr erkannt wird (Tracking Verlust): „gesture“=“tracking_lost“
progress	Progress Wert der fliessenden Geste Ausnahmen: <ul style="list-style-type: none"> - Bei statischen Gesten: „progress“=“discrete“ - Bei Tracking Verlust: „progress“=“none“

Tabelle 22 Beschreibung JSON Key-Value-Paare

Zusammengesetzt ergibt sich aus den Key-Value-Paaren die folgende Datenstruktur:

```
{
  "trackingId":trackingId,
  "x":x,
  "y":y,
  "z":z,
  "gesture":gesture,
  "progress":progress
}
```

In der folgenden Tabelle werden die verschiedenen für die Applikation benötigten Nachrichtentypen aufgelistet und mit einem Beispiel illustriert.

Typen	Beispiel JSON
Statische Geste	<pre>{ "trackingId": "72057594037927973", "x": "0.1210797", "y": "0.8740084", "z": "3.269177", "gesture": "stretch_both_arms_sideways", "progress": "discrete" }</pre>
Fließende Geste	<pre>{ "trackingId": "72057594037927974", "x": "0.1084615", "y": "0.8629642", "z": "3.263785", "gesture": "lift_stretched_right_arm", "progress": "0.8309187" }</pre>
Ohne Geste / Nur Position	<pre>{ "trackingId": "72057594037927973", "x": "1.255873", "y": "0.7958131", "z": "2.107728", "gesture": "none", "progress": "discrete" }</pre>
Tracking Verlust	<pre>{ "trackingId": "72057594037927973", "x": "0", "y": "1.093064", "z": "0", "gesture": "tracking_lost", "progress": "none" }</pre>

Tabelle 23 JSON Nachrichtentypen mit Beispielen

Die JSON Daten werden über die statische Klasse „JSONParser.cs“ erstellt. Für alle JSON Typen steht eine Methode zur Verfügung, welche jeweils die veränderbaren Werte als Übergabeparameter anbietet. Zusätzlich enthält die Klasse ein Property „KinectHeight“ welches die Höhe des Sensors beinhaltet. Die Höhe wird zum Y Wert hinzugezählt, da die Kinect als Ursprungspunkt seines Koordinatensystems den Sensor und nicht den Boden wählt. In der privaten Methode „Parse“ wird das JSON mit Hilfe eines StringBuilders zusammengebaut. Somit kann der Aufwand zur Erstellung der JSON Nachrichten minimiert werden, da keine zusätzlichen Objekte für das JSON Parsing erstellt werden müssen.

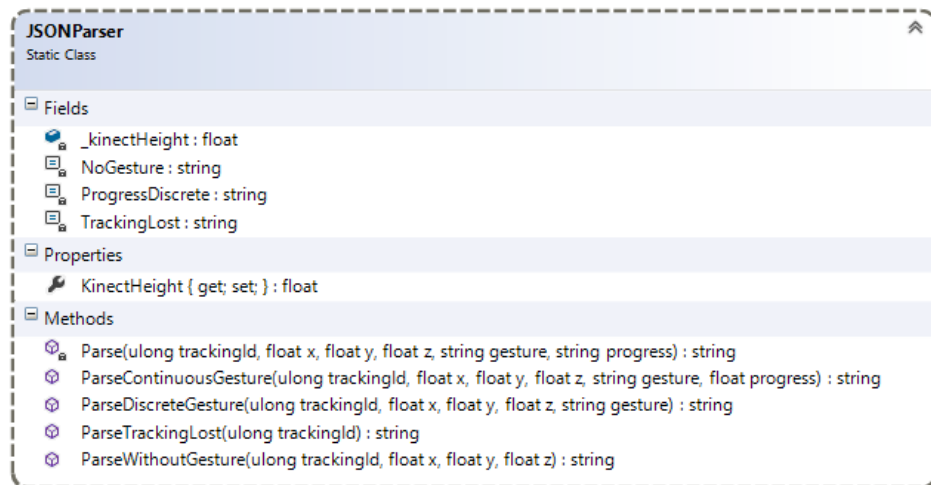


Abbildung 23 Klasse JSONParser

3.4.2 WebSocket Server Implementierung C#

Die WebSocket Implementierung wird mit der im Kapitel 3.2.7 Alchemy Websockets erwähnten Alchemy WebSockets Library von Olivine Labs realisiert. Die Library ist unter „Libraries/Alchemy.dll“ zu finden. Die für den WebSocket Server erstellte „KinectWebSocketServer.cs“ Klasse wurde bewusst offen gestaltet, um die Wiederverwendbarkeit sicherzustellen. Der WebSocket ist unter dem Port 6789 zu erreichen.

Über die „Broadcast“ Methode kann allen verbundenen Clients ein beliebiger String zugesendet werden. Dazu wird eine Liste mit allen Verbindungen verwaltet, welche in den Methoden „OnConnected“ und „OnDisconnect“ auf- beziehungsweise abgebaut wird. Über die „StopServer“ Methode werden alle Verbindungen getrennt und der Server beendet.

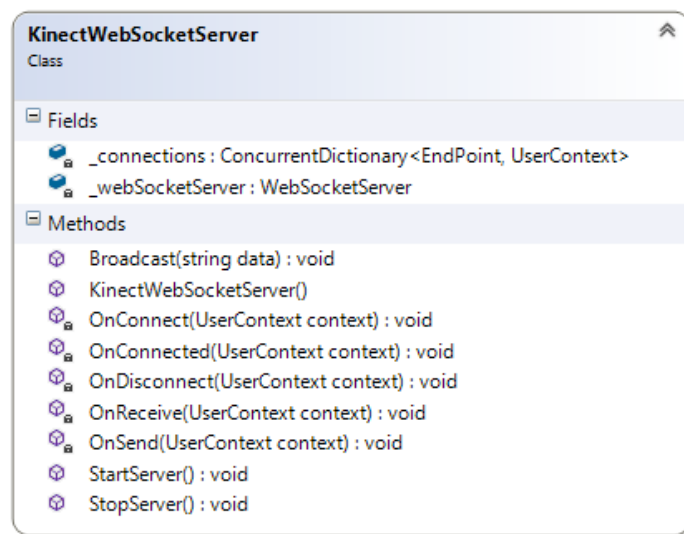


Abbildung 24 Klasse KinectWebSocketServer

3.4.3 WebSocket Client Implementierung JavaScript

In HTML5 ist eine WebSocket API von der W3C spezifiziert [21]. Unter der Verwendung dieser API wird beim Starten eines Spielablaufs von Ludosonica im Kinect Modus eine Verbindung mit dem WebSocket Server auf der anderen Seite aufgebaut. Nach erfolgreichem Aufbau der Verbindungen empfängt der Client alle Daten, welche von der Kinect über den WebSocket Server geschickt werden, in seiner „socket.onmessage“ Callback Methode. Von dort wird der empfangene Datensatz an „handleData“ weitergegeben. Die „handleData“ Methode ist zentraler Bestandteil der Applikation. Sie analysiert den Datensatz nach dem Aufruf mit „JSON.parse(data)“ und speichert die verschiedenen Attribute in lokalen Variablen.

Der Ablauf dieser Methode soll in folgendem Flussdiagramm dargestellt werden:

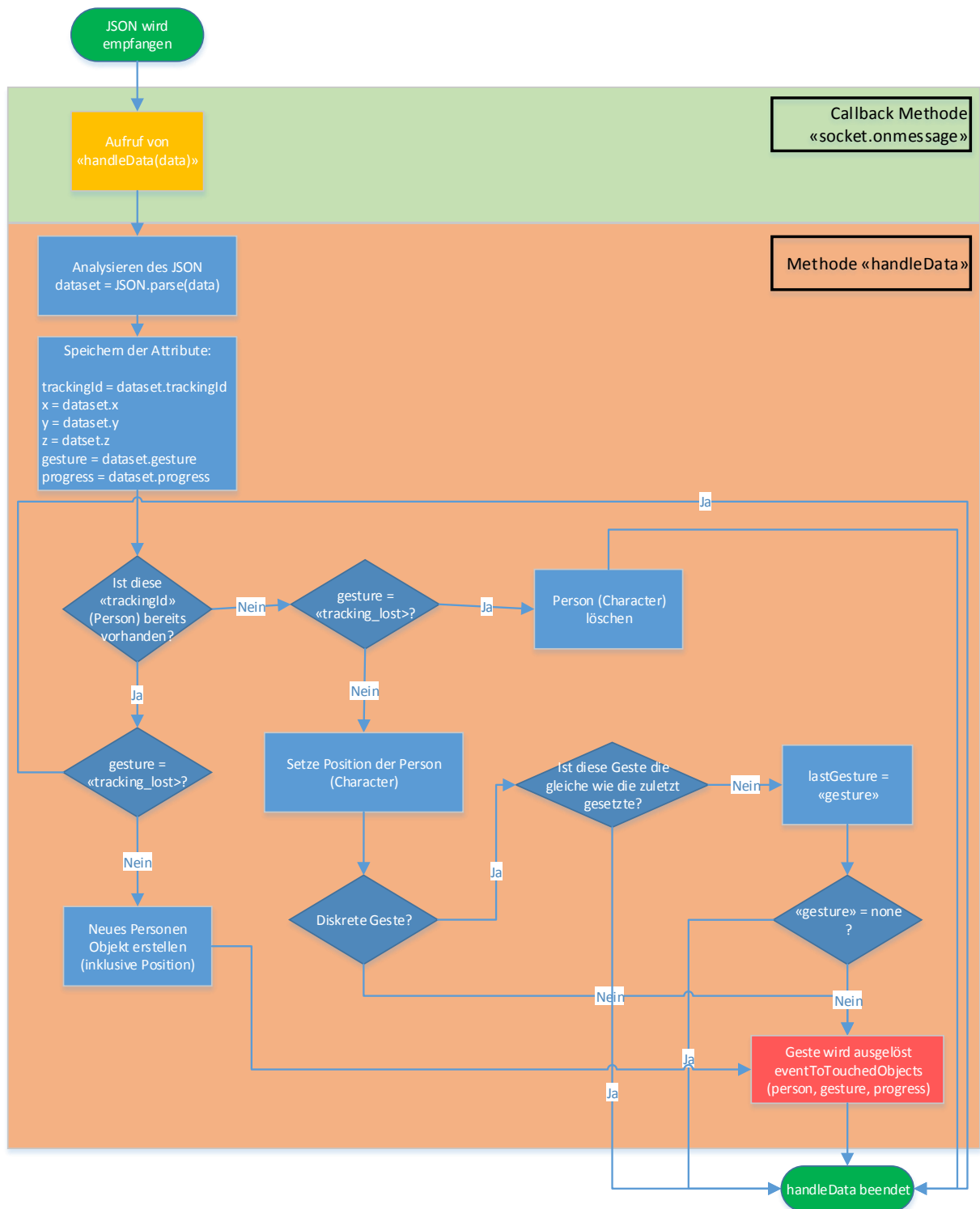


Abbildung 25 Flussdiagramm der JSON Verarbeitung in Ludosonica

3.5 Ludosonica

Wie in Kapitel 2.1 Ausgangslage erwähnt, dient das Projekt Ludosonica als Basis von Sonic Classroom. In diesem Projekt ist es Teil des Auftrags, die Software Ludosonica um eine Steuerung mittels Kinect zu erweitern. Um dies zu erreichen, mussten in der Programmlogik, wie auch an der grafischen Oberfläche, gewisse Veränderung vorgenommen werden. Damit der Benutzer klar die Unterschiede feststellt, wurde eine Art Startseite erstellt, die den Benutzer zu Beginn vor die Wahl der beiden Modi stellt.

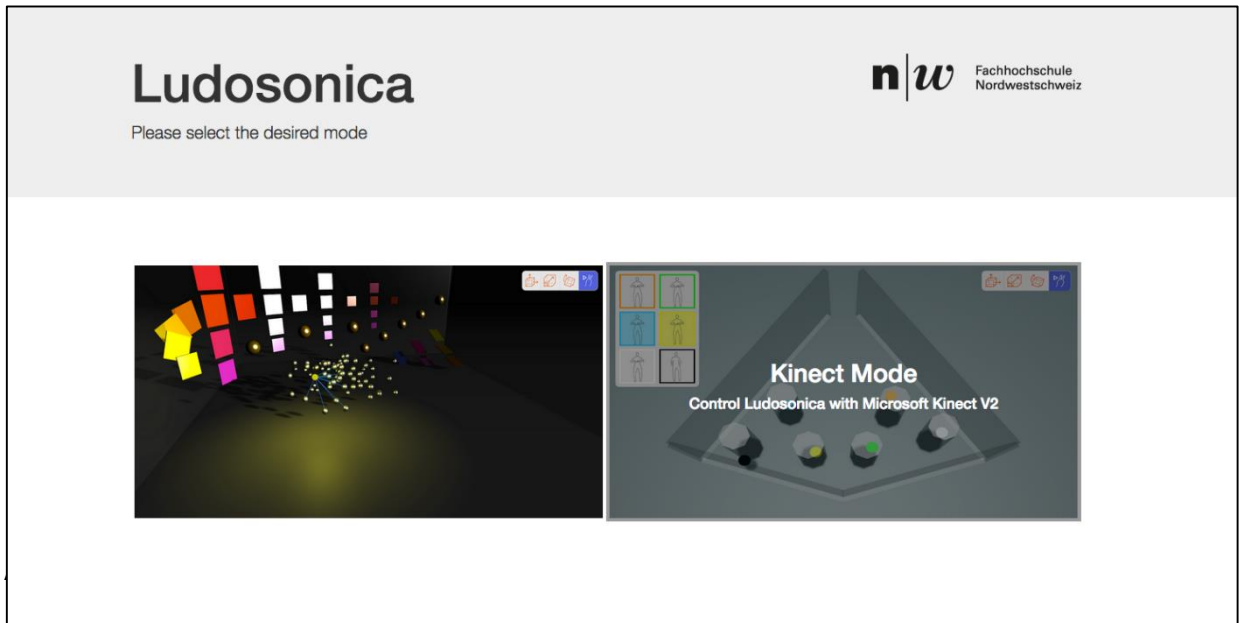


Abbildung 26 Ludosonica Startseite

Auf die Bedienung des LEAP Modus wird hier bewusst nicht weiter eingegangen, da dies nicht Teil dieses Projekts ist.

3.5.1 Grafische Oberfläche

Editor Modus

Die Software Ludosonica ist in zwei Modi unterteilt. Einen Editor Modus, in welchem die Objekte im Raum positioniert und deren Gesten und Soundmodifikationen definiert werden und einen Play Modus, auf den das System durch Klicken auf den „Play“ Button oben rechts, wechselt. Anschliessend öffnet sich der Socket zum Sonic Classroom Controller und die Daten der Kinect werden empfangen. Im Editor Modus wurden Änderungen am GUI vor allem bei der Definition der Events vorgenommen.



Abbildung 29 Dropdowns zur Definition der Events

Durch ausgiebiges Testen konnte festgestellt werden, welche Gesten miteinander funktionieren und welche nicht. Es gibt durchaus viele Gesten, die sich durch ihre Ähnlichkeit gegenseitig negativ beeinflussen und so ungewünschte Effekte hervorrufen, wenn sie gleichzeitig verwendet würden. Auf Basis dieser Testmatrix (siehe Anhang 5 – Testmatrix Gesten) wurden die Gesten gruppiert. Anhand dieser Gruppen wird jedes Mal, wenn eine neue Geste ausgewählt wird, eine Überprüfung durchgeführt, welche Gesten auf diesem Objekt schon gesetzt sind und so die Auswahl eingeschränkt. Dieser Mechanismus ermöglicht es, eine gewisse Zuverlässigkeit der Software zu erreichen. Auf den Dropdown Elementen wurde anhand einer „exclusionId“ die Gruppe angegeben (siehe Anhang 6 - Gestengruppierung). Bei jedem „change Event“ wird die Methode „triggerDropdownExclusions“ aufgerufen und dabei werden die Dropdownlisten aktualisiert.

Ausschnitt aus den definierten Dropdown Elementen mit zugehörigen Attributen:

```

1. <option exclusionId="1" type="discrete">Stretch left arm left</option>\
2. <option exclusionId="2" type="discrete">Stretch left arm up</option>\
3. <option exclusionId="3" type="discrete">Stretch left arm front</option>\
4. <option exclusionId="4" type="discrete">Stretch right arm right</option>\
5. <option exclusionId="5" type="discrete">Stretch right arm up</option>\
6. <option exclusionId="6" type="discrete">Stretch right arm front</option>\
7. <option exclusionId="7/8" type="discrete">Stretch both arms sideways</option>\
8. <option exclusionId="9/10" type="discrete">Stretch both arms front</option>\

```

Zwei weitere GUI Änderungen sind die bereits unter Kapitel 2.5.9 Spielabläufe beschriebenen „Beats per Minute (BPM)“ und „Synchronize Sounds“.

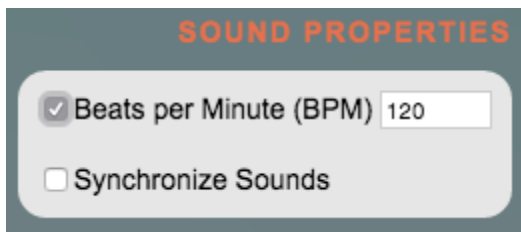


Abbildung 30 Sound Properties im Editor Modus

Es kann jeweils nur eine Funktion von beiden aktiviert werden. Diese Einstellungen werden auf globalen Variablen im Script „SoundCollection.Kinect.js“ gesetzt. Da diese Einstellungen für spezifische Spielabläufe von Bedeutung sind, werden bei einem Export die neu hinzugefügten „Sound Properties“ auch gespeichert. Beim Import wiederum werden sie wieder gesetzt.

Play Modus

Auch der Play Modus hat ein paar grafische Veränderungen erhalten. In einem ersten Schritt wurden den einzelnen Personen (Character) verschiedene Farben zugeteilt. Es besteht eine Map mit 6 verschiedenen Farben. Jedes Mal wenn sich eine Person in den erkannten Bereich bewegt, wird der nächste nicht gebrauchte Color aus der Map genommen und diesem Character zugeteilt. Beim Verlassen des Bereichs wird die Farbe wieder freigegeben indem das Attribut „used“ wieder auf „false“ gesetzt wird.

Die Character Color Map mit den gespeicherten Farben und dem Attribut ob die Farbe bereits verwendet wird oder nicht:

```
1. this.characterColorMap = {  
2.     Orange : {hex:0xFF9900, used:false},  
3.     Lime   : {hex:0x00FF00, used:false},  
4.     Blue   : {hex:0x33CCFF, used:false},  
5.     Yellow : {hex:0xFFFF00, used:false},  
6.     White  : {hex:0xFFFFFFFF, used:false},  
7.     Black  : {hex:0x000000, used:false}  
8. };
```

Im Zusammenhang mit der farblichen Erkennung der einzelnen Personen war eine visuelle Rückmeldung gewünscht, welche dem Benutzer anzeigt, ob die ausgeführte Geste vom System erkannt wurde oder nicht. Um die visuelle Wirkung zu verstärken, wird bei jedem Detektieren einer Geste das Bild der Geste angezeigt und der Hintergrund des Feldes mit der gleichen Farbe wie der Character eingefärbt. Mit einer Fade-Out Animation wird der Hintergrund innerhalb von 1.5 Sekunden wieder fließend ausgeblendet.

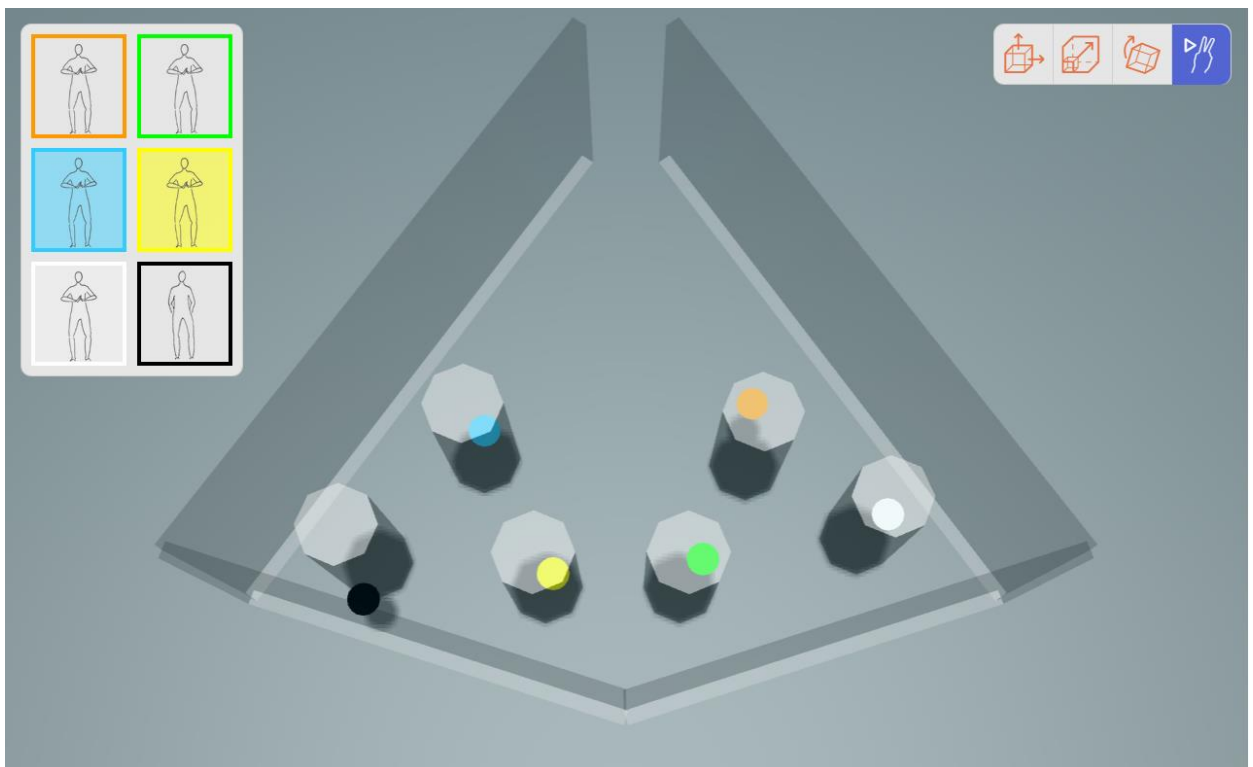


Abbildung 31 Visuelles Feedback im Play Modus

3.5.2 Implementierungen

Der in diesem Projekt neu geschriebene Code in Bezug auf Ludosonica ist teilweise schwer abzugrenzen gegenüber dem Bestehenden. Einige Scripts wurden praktisch komplett neu geschrieben, bei anderen wurden nur kleine Änderungen durchgeführt (siehe Anhang 7 – Codeabgrenzung Ludosonica). Die Konzepte die bereits bestehend waren, wurden so weit möglich übernommen. Der Code ist wie bei jedem Prototyp nicht abschliessend. Folge dessen ist in der bestehenden Implementierung einiger Code doppelt vorhanden. Eine abschliessende Version bedürfte noch vertieftem Refactoring des Source Codes. Da der Fokus dieser Arbeit darauf lag, einen möglichst repräsentativen Prototyp zu erstellen, wurde der Code in eine lauffähige, aber nicht abschliessende Form gebracht.

Play.Kinect.js

Sobald der Benutzer in den Play Modus wechselt wird das Script „Play.Kinect.js“ aktiv. Sie ist wie in Kapitel 3.4.3 WebSocket Client Implementierung JavaScript beschrieben, für die Kommunikation über den WebSocket zuständig. Weiter sind diverse Methoden zum Auslösen der definierten Aktionen auf den Objekten vorhanden. Die „handleData“ Methode ruft beim Eintreffen eines Datensatzes „eventToTouchedObjects“ auf. Da ein Event nur ausgelöst wird, wenn eine Person im Bereich des Objektes steht, werden anschliessend alle aktuellen „Touches“ aus der Szene überprüft. Ist auf einem Objekt, das gerade von einem Character einer Person berührt wird, die ausgeführte Geste definiert, wird deren Aktion ausgelöst. Die Auslösung der Aktion wird über eine Callback Methode im „Editor.Kinect.js“ gemacht.

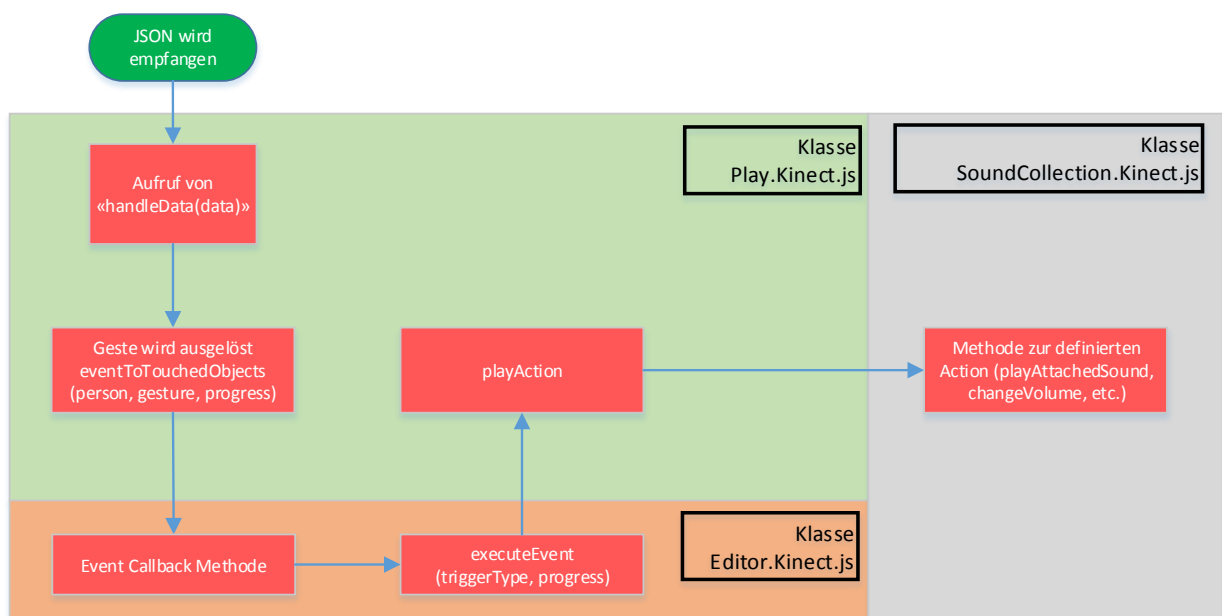


Abbildung 32 Ablauf beim Auslösen einer Action

Editor.Kinect.js

Im „Editor.Kinect.js“ Script wurden einzig die Event Callback Methoden der Gesten angepasst und zentral über eine „executeEvent“ Methode ausgeführt.

Die Callback Methoden wurden auf folgende Art auf „executeEvent“ verknüpft:

```
1. if (hasTrigger(clone.events, triggerType = 'Jump'))
2.     clone.jump = executeEvent.bind(clone,triggerType);
3.
4. var executeEvent = function (triggerType, progress) {
5.     if (this.events != undefined) {
6.         for (var i = 0; i < this.events.length; i++) {
7.             // we want to execute the event
8.             if (this.events[i].trigger.type == triggerType) {
9.                 editor.play.playAction(this, i, progress);
10.            }
11.        }
12.    }
13. };
```

Tools.Kinect.Modes.js

Das folgende Flussdiagramm zeigt den Ablauf des Verbindungsaufbaus, beziehungsweise – Abbaus zum Sonic Classroom Controller sowie deren Fehlerbehandlung beim Starten und Stoppen des Play Modus. Diese Aktionen werden über den Play Button ausgelöst.

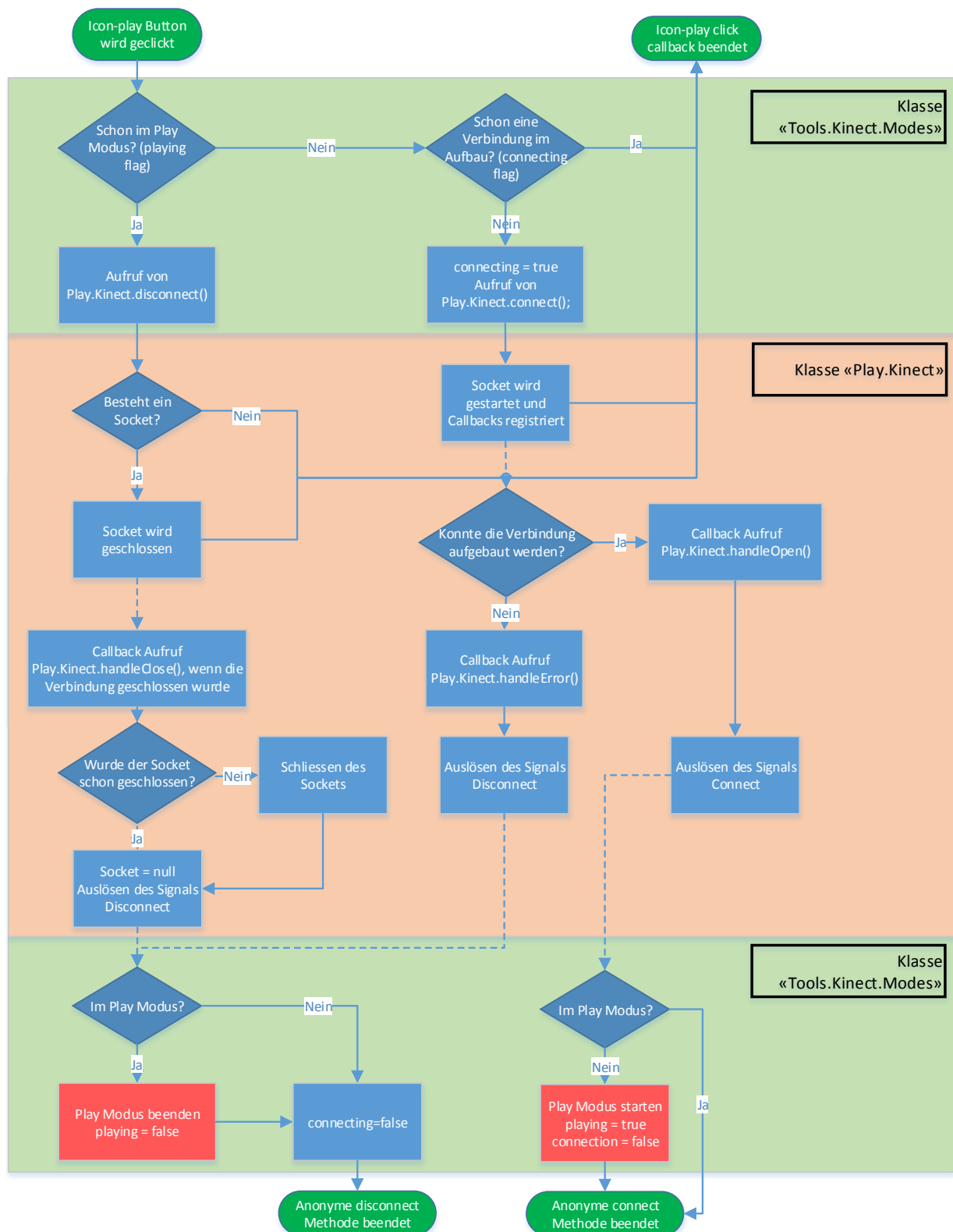


Abbildung 33 Workflow beim Klicken des Play Buttons

Viewport.Kinect.js

Im Editor Modus wird eine graue Fläche auf dem Boden aufgetragen die genau dem durch den Kinect V2 Sensor abgedeckten Bereich entspricht. So ist es für den Benutzer klar wo er seine Objekte platzieren kann und wo nicht.

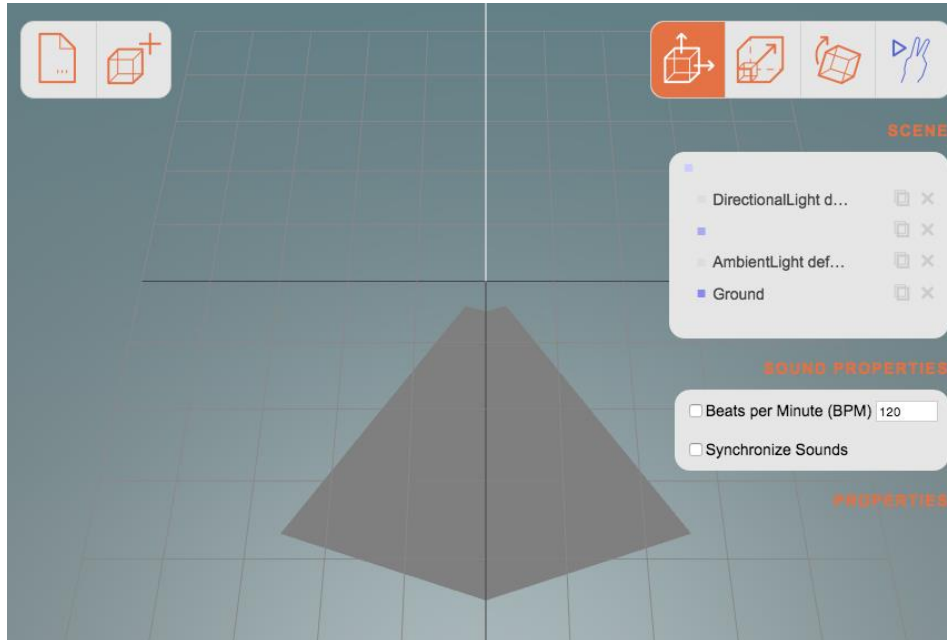


Abbildung 34 Kinect Erfassungsbereich Editor Modus

Der gleiche Effekt ist im Play Modus abgebildet. Dort werden sogenannte „Kinect Boundary Walls“ angezeigt, die den Erfassungsbereich markieren.

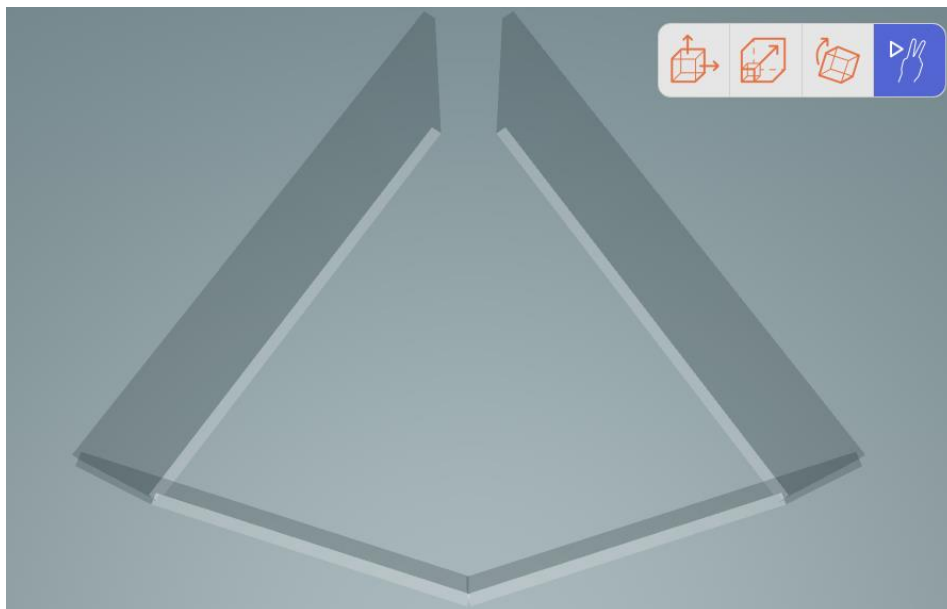


Abbildung 35 Kinect Erfassungsbereich Play Modus

SoundCollection.Kinect.js

Das Abspielen und vor allem auch Manipulieren von Sounds gehört zum Kern dieses Projektes. Demnach ist auch das Script „SoundCollection.Kinect.js“ zentral. Hier wurde ausser dem Hinzufügen, Stoppen und Abspielen von Sounddateien einiges hinzugefügt und abgeändert. Unter der Verwendung der im Kapitel 3.2.2 Web Audio API (Webaudiox) beschriebenen Schnittstelle für Web Audio ist der sogenannte „audio routing graph [22]“ zentral. Bei jedem Starten eines Sounds, Hinzufügen oder Entfernen eines Filters oder Effekts, wird der Audio Graph über die Methode „_updateAudioGraph“ aktualisiert. Zuerst werden alle sogenannten „Nodes“ entfernt und danach der Reihe nach dem Graph hinzugefügt. Beim Starten eines weiteren Sounds auf einem gleichen Objekt werden jeweils immer die zuletzt eingestellten Frequenzen der Filter übernommen. So hat jeder neue Sound auf dem gleichen Objekt den gleichen Stand. Das folgende Beispiel zeigt einen einfachen Audio Graph, der eine Source (Soundfile) über eine Destination (Ausgabekanal des Computers) ausgibt. Dazwischen sind ein „Lowpass“ Filter und ein „GainNode“ geschaltet. Der Filter kann über das Attribut „frequency.value“ verändert werden und der GainNode über das Attribut „gain.value“, zum Beispiel um die Lautstärke der Ausgabe zu verändern.

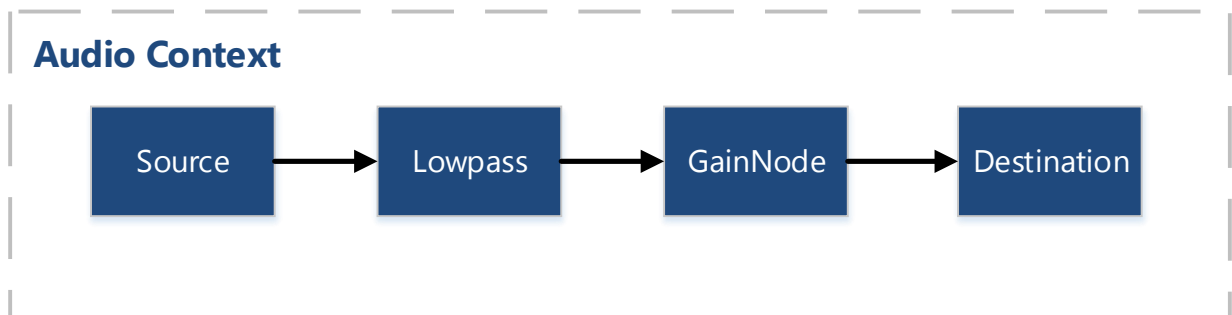


Abbildung 36 Beispiel eines Audio Graph

Wegen der Art der Tonhöhenempfindung des menschlichen Gehörs wird die Frequenz generell in einem logarithmischen Massstab angegeben [23]. Aus diesem Grund wird für jede Frequenzveränderung eines Filters eine „_changeFrequency“ Methode aufgerufen. Diese berechnet aus dem eingetroffenen „value“ eine für den Menschen gut erkennbare Veränderung der Frequenz welche danach auf dem Filter gesetzt wird.

```

1. _calculateFrequency: function (value) {
2.     // Clamp the frequency between the minimum value (40 Hz) and half of the
3.     // sampling rate.
4.     var faktor = Math.pow(10, 2);
5.     var input = Math.round(value * faktor) / faktor;
6.     var minValue = 40;
7.     var maxValue = this._context.sampleRate / 2;
8.     // Logarithm (base 2) to compute how many octaves fall in the range.
9.     var numberOfOctaves = Math.log(maxValue / minValue) / Math.LN2;
10.    // Compute a multiplier from 0 to 1 based on an exponential scale.
11.    var multiplier = Math.pow(2, numberOfOctaves * (input - 1.0));
12.    // Get back to the frequency value between min and max.
13.    return maxValue * multiplier;
14. },

```

Diese Berechnung stammt von „Boris Smus“, der das Buch „Web Audio API“ vom O'REILLY Verlag geschrieben hat. Er verwendet diese Berechnung in seinen Beispielen die unter der „Creative Commons license“ auf seinem GitHub Repository [24] verfügbar sind.

ui.js

Wie im Kapitel 3.5.1 Grafische Oberfläche beschrieben, wurden an der grafischen Oberfläche vor allem die „Kinect Event List“ und das „Visual Feedback“ hinzugefügt. Diese Implementierungen finden sich im Script „ui.js“.

3.6 Fieldtest

Einen Prototyp des Projektes in einem realen Umfeld zu testen, war ein geforderter Bestandteil der Umsetzung. Gabriel Imthurn, Dozent an der Professur für Musikpädagogik, ermöglichte einen Test des ersten Prototyps mit seiner Musikschulklasse. Im Vorfeld wurden Überlegungen zu den verschiedenen Spielabläufen gemacht, die getestet werden sollten (siehe Anhang 8 – Vorgehen Test mit Schulklasse). Ein paar grundsätzliche Szenarien galt es zu testen und herauszufinden, wie Jugendliche darauf reagieren. Zum einen wurden Szenarien mit kurzen rhythmischen Klängen, mit zwei bis sechs Personen pro Ablauf, getestet. Ein weiterer Test hatte als Grundlage ein bekanntes Musikstück welches gestartet, durch Filter verändert und wieder gestoppt wurde. Dies zuerst mit drei Personen gleichzeitig und danach mit einer einzelnen Person. Auch die Loop Funktion, wo Klänge immer wieder ausgelöst werden, wurde von den Schülern ausprobiert. Zum Schluss des Tests wurde ein Szenario mit Umgebungsklängen aufgesetzt. Sechs verschiedene Personen konnten vom Erklingen einer Kirchenglocke, über das Vorbeifahren eines Ambulanzfahrzeugs mit Sirene, bis hin zum Vogelgezwitscher allerlei Umgebungsklänge starten und stoppen. Für das Projekt war es zum einen wichtig herauszufinden, wie die verschiedenen Tests auf die Schüler wirken und wie viele Personen gleichzeitig im Erfassungsbereich der Kinect zusammen interagieren können. Die Interaktion mit dem Computer durch Körpergesten war eine spezielle Erfahrung für viele der Probanden.



Abbildung 37 Szenario mit 5 Personen



Abbildung 38 Szenario mit 3 Personen

Anschliessend an den Test füllten die Schüler einen kurzen Fragebogen aus und bewerteten die verschiedenen Szenarien (siehe Anhang 9 – Auswertung Test mit Schulklasse). Zusammengefasst kann ein sehr gutes Fazit gezogen werden. Die jungen Schüler hatten Spass an dieser Abwechslung und sie machten sehr gut mit. Es brauchte keine motivierenden Worte damit sie mitmachten. Die Koordination zwischen den verschiedenen Personen ist sicher etwas, dass geübt werden muss. Das funktioniert nicht von Anfang an so reibungslos.

Fazit des Tests ist sicher, dass bis und mit vier Personen im Erkennungsbereich, ein angenehmes Bewegen und Ausführen der Gesten möglich ist. Sind es mehr als vier Personen, kommt man sich in den Weg. Zum Beispiel das Ausstrecken des Arms wird dabei von einer anderen Person verdeckt und wird so nicht mehr erkannt. Ein weiteres, sehr interessantes Element sind die fließenden Gesten beim Verändern von Lautstärke oder der Parameter eines Filters. Diese werden sehr zuverlässig erkannt und stossen so auf viel Begeisterung beim Verändern eines Samples. Da in diesem Projekt, nach dem Test mit der Schulklasse, noch sehr viel an der Implementation verbessert wurde, müsste in einem weiteren Schritt nochmals getestet werden um die Software zu verfeinern. Aus organisatorischen Gründen war es im Zusammenhang mit diesem Projekt nicht möglich, nochmals Live zu testen.

3.7 Erweiterbarkeit

Als Erweiterung könnten dem System neue Gesten hinzugefügt werden. Grundsätzlich ergaben sich aus den Expertengesprächen die ungefähr 30 Gesten die umgesetzt wurden. Eventuell bei einer neuen Version der Kinect die sogar die Bewegungen der Finger genau erkennen könnte würden sich neue Gesten ergeben. Das Set an Soundmodifikationen könnte gut noch um einige ergänzt werden. Im Gespräch mit dem einen Experten stellte sich heraus, dass eine Schnittstelle zu MIDI sehr interessant wäre. Aus zeitlichen Gründen konnte diese im Zuge dieses Projektes nicht realisiert werden. Die Gestenerkennung ist gut aber sicher noch nicht perfekt. Es könnten noch mehr Gestenvideos von verschiedenen Personen aufgenommen werden, um die Gestendatenbank zu optimieren. Bei der Bedienung von Ludosonica fällt es einer nicht technisch versierten Person schwer es auf Anhieb zu verstehen. Wenn eine Softwarekomponente so umfangreich ist wie Ludosonica wird die Bedienung schnell unübersichtlich. An diesem Punkt könnte man ansetzen und sich Überlegungen machen zu Bedienung im Editor Modus. Momentan besteht in Ludosonica eine Galerie die mit vordefinierten Settings für die Bedienung mit der LEAP Motion ausgestattet ist. Die vordefinierten Umgebungen für Kinect Spielabläufe könnten noch besser ausgearbeitet und auch in die Galerie integriert werden.

4 Schluss

4.1 Resultate

Auf der einen Seite ist es gelungen, die Daten der Microsoft Kinect V2 in einer Softwarekomponente zu empfangen, zu analysieren und für den in diesem Projekt notwendigen Umfang aufzubereiten. Weiter ist es möglich die Daten über eine Webschnittstelle ohne merkbare Verzögerung an Ludosonica zu übertragen. Schlussendlich wird dieser Input in der Webapplikation verarbeitet und dem Benutzer verständlich angezeigt. Alle diese Teilresultate führen zu einem System, dass es ermöglicht, durch Körpergesten im Erfassungsbereich des Sensors, den Ablauf von Klängen zu starten, deren Erscheinung durch Setzen von Filtern und Effekten zu verändern und schlussendlich auch die Klänge wieder zu stoppen. So entstand als Resultat eine Software die Grundfunktionen bietet, aber den Benutzer in der Gestaltung der Klangkompositionen nicht einschränken soll. In Bezug auf die Effizienz des Systems können folgende allgemeine Aussagen getätigt werden. Die Kinect V2 sendet 30 Frames pro Sekunde pro erkannter Person an das Zielsystem. Das Verarbeiten dieser Datenmenge und Weitersenden über einen lokalen WebSocket findet bei aktueller Hardware ohne signifikante Verzögerung statt. Das verwendete Testgerät (HP EliteBook 8470w) hat mit sechs Personen im Erfassungsbereich keine höhere CPU Auslastung als 40% (Grundauslastung ohne erkannte Personen ca. 10%) und benötigt nicht mehr als 500 MB Arbeitsspeicher.

4.2 Ausblick

Da beim Musizieren die Reaktionszeit der Instrumente aller Art ein sehr kritischer Faktor darstellt, muss die Erkennung der Gesten von der Sensing Device sehr genau sein. Wenn zum Beispiel vier Mal über dem Kopf geklatscht wird und das System aus Genauigkeitsgründen nur drei Mal ein Klatschen erkennt, ist der Benutzer verunsichert. In Zukunft sollte daher der Zuverlässigkeit im Erkennen von Gesten immer grosse Aufmerksamkeit geschenkt werden. Bei einer Softwarelösung, die so stark von der Benutzerinteraktion abhängig ist wie diese hier, sollte immer stark mit den potenziellen Nutzergruppen zusammengearbeitet werden. Ausgiebiges Testen und erforschen der Möglichkeiten und Grenzen des Systems sollten eine zentrale Rolle spielen. Da sich in der Konzeptphase herausgestellt hat, dass die Anwendung von solchen Systemen grundsätzlich nur mit kleineren Gruppen Sinn machen, ist eine Weiterverfolgung der erweiterten Erkennung durch Bildverarbeitung nicht sinnvoll. Aufgrund dieser Erkenntnis sind auch Umsetzungsvarianten mit mehreren Sensorgeräten auf ein System nicht weiter zu verfolgen.

4.3 Reflexion

Bei der Evaluation von verschiedenen Gesten sollte auch immer die Machbarkeit und der Sinn der einzelnen Bewegung in Betracht gezogen werden. Einige Bewegungen könnten anstössig wirken oder in Bezug auf die technischen Spezifikationen des Sensors keinen Sinn ergeben. Die Entscheidung, den Sonic Classroom Controller mit der von Microsoft entwickelten Programmiersprache C# umzusetzen, stellte sich als eine richtige Entscheidung heraus. Die Unterstützung einer solch neuen Technologie im Zusammenhang mit der Kinect V2 ist enorm wichtig um Zeit und Aufwand für nicht projektrelevante Arbeiten einzusparen. Die gesamte Gestenaufnahme- und Erstellung wird durch die von Microsoft gebotenen Tools stark vereinfacht.

Die Einarbeitung in bestehenden Sourcecode wurde anfänglich unterschätzt. In Zukunft würde das Einlesen in bereits vorhandene Software schon während der Konzeptphase parallel betrieben werden umso später bei der Umsetzung Zeit zu gewinnen.

5 Verdankung

In erster Linie möchten wir unserer Betreuerin und unserem Betreuer für die Unterstützung während dem ganzen Projekt danken.

Vor allem zu Beginn unserer Arbeit konnten wir wichtige Erkenntnisse aus den Gesprächen mit den Experten der Professur für Musikpädagogik gewinnen. Für die Flexibilität und Zusammenarbeit möchten wir uns bei allen Experten bedanken.

Der durchgeführte Test im realen Umfeld ermöglichte uns Herr Imthurn mit seiner Sekundarschulklasse in Safenwil. Für diese Möglichkeit und das gute Mitmachen der Schüler möchten wir uns bedanken.

Auch möchten wir uns bei unseren Gegenlesern bedanken, die viel Zeit in die Korrektur dieser Arbeit investiert haben.

Einen speziellen Dank gilt unserer Kommilitonin, Fiona Nüesch, die uns für die visuelle Rückmeldung diverse Gestenbilder illustriert hat.

6 Ehrlichkeitserklärung

Das eingereichte Projekt ist das Resultat der persönlichen und selbstständigen Arbeit der Autoren. Hiermit erklären die Autoren, die Arbeit ohne unzulässige Hilfe Dritter und nur unter Benutzung der im Verzeichnis aufgeführten Quellen verfasst zu haben.

Florian Rüegg

Florin Tiefenauer

Ort, Datum, Unterschrift

Ort, Datum, Unterschrift

7 Glossar

Glossar	
API (Application Programming Interface)	Schnittstellendefinition einer Software für den Zugriff von ausserhalb.
BPM (Beats per Minute)	Mit Beats per Minute (Schläge pro Minute) kann man das Tempo eines Musikstücks messen. Meistens wird als Zählleinheit eine Viertelnote verwendet [25, p. 196].
Frame	Einzelnes Bild aus einer Filmsequenz.
GitHub Repository	Ein GitHub Repository ist eine Plattform, auf welcher Entwickler ihre Software veröffentlichen können.
GUI (Graphical User Interface)	Die grafische Oberfläche die der Benutzer zum Interagieren mit der Software sieht.
LEAP Motion	Die LEAP Motion ist ein Sensing Device, welches das Erkennen beider Hände und deren Tracking ermöglicht. Die LEAP Motion erkennt sowohl das Skelet der Hand sowie verschiedene vorgegebene Gesten.
Ludosonica	Vom Institut für 4D-Technologien entwickelte Webapplikation (http://playfulmedia.cs.technik.fhnw.ch/).
Machine Learning	Ein System das durch eine Lernphase Informationen verallgemeinern kann.
MIDI [26]	Kurzform für Musical Instrument Digital Interface. MIDI ist ein Standard der den elektronischen Austausch von musikalische Steuerelemente zwischen Instrumenten beschreibt.
OpenCV	Open CV ist eine freie verfügbare Library für Bildverarbeitung.
Parser	Ein Parser wandelt eine Eingabe in ein definiertes und für die Weiterverarbeitung brauchbares Format um.
Scrum	Scrum ist ein schlankes, iteratives Prozessmodell für Softwareentwicklungsprojekte.
SDK (Software Development Kit)	Eine Ansammlung von verschiedenen Tools und Dokumentationen zum Erstellen einer Software.
Sensing Device	Ein Sensing Device ist ein Gerät, welches das Erkennen von Körpern oder Körperteilen und deren Bewegungen ermöglicht.
Third Party Library	Eine Softwarekomponente die nicht von der ursprünglichen Plattform entwickelt wurde.
Tracking	Tracking bezeichnet das Verfolgen von bewegbaren Objekten und deren Abbildung in technischen Anwendungen.

Trello	Trello ist eine web-basierte Projektmanagementsoftware die es ermöglicht, mit verschiedenen Benutzern auf sogenannten „Boards“ die Projekttasks zu verwalten.
W3C	Kurzform für World Wide Web Consortium. W3C ist ein Gremium, welches die Techniken des World Wide Web standardisiert.
WebGL (Web Graphics Library)	Grafik Bibliothek für die Hardwarebeschleunigte Darstellung von 3D Objekten im Browser.
WPF (Windows Presentation Foundation)	Framework von Microsoft zum Erstellen von grafischen Oberflächen mit C#.

8 Abbildungsverzeichnis

Abbildung 1 Kinect V2 Erfassungsbereich Vogelperspektive	14
Abbildung 2 Kinect V2 Erfassungswinkel Horizontal / Vertikal.....	14
Abbildung 3 Sekelett Tracking	14
Abbildung 4 Darstellung Lösungsansatz Single Sensing Device.....	15
Abbildung 5 Darstellung Lösungsansatz 1 Multiple Sensing Device.....	17
Abbildung 6 Darstellung Lösungsansatz 2 Multiple Sensing Device.....	18
Abbildung 7 Darstellung Variante mittels Bildverarbeitung	19
Abbildung 8 Kinect Einstellungen	20
Abbildung 9 Übersicht der Komponenten.....	28
Abbildung 10 Three.js Beispiele (Wikipedia).....	28
Abbildung 11 Logo jQuery UI (Wikipedia)	29
Abbildung 12 Logo Bootstrap (Wikipedia)	29
Abbildung 13 GUI Sonic Classroom Controller.....	31
Abbildung 14 Klassendiagramm Sonic Classroom Controller.....	32
Abbildung 15 Flussdiagramm Grobablauf Sonic Classroom Controller	33
Abbildung 16 Visual Gesture Builder im Kontext [27]	34
Abbildung 17 Einstellungen für Aufnahmen im Raw Format (XRF).....	35
Abbildung 18 Visual Gesture Builder Markierungen für statische Gesten.....	36
Abbildung 19 Visual Gesture Builder Markierungen für fließende Gesten	37
Abbildung 20 Flussdiagramm der Gestenerkennung im Sonic Classroom Controller.....	38
Abbildung 21 Übersicht über alle JointTypes [28]	39
Abbildung 22 Koordinaten System der Kinect V2 [20].....	39
Abbildung 23 Klasse JSONParser	42
Abbildung 24 Klasse KinectWebSocketServer.....	43
Abbildung 25 Flussdiagramm der JSON Verarbeitung in Ludosonica	44
Abbildung 26 Ludosonica Startseite	45
Abbildung 27 Startseite von Ludosonica.....	45
Abbildung 28 Startseite von Ludosonica / Auswahl der Modi.....	45
Abbildung 29 Dropdowns zur Definition der Events	45
Abbildung 30 Sound Properties im Editor Modus.....	46
Abbildung 31 Visuelles Feedback im Play Modus.....	47
Abbildung 32 Ablauf beim Auslösen einer Action.....	48
Abbildung 33 Workflow beim Klicken des Play Buttons.....	50

Abbildung 34 Kinect Erfassungsbereich Editor Modus.....	51
Abbildung 35 Kinect Erfassungsbereich Play Modus.....	51
Abbildung 36 Beispiel eines Audio Graph	52
Abbildung 37 Szenario mit 5 Personen.....	53
Abbildung 38 Szenario mit 3 Personen.....	53

9 Tabellenverzeichnis

Tabelle 1 Kontextszenario 1 - Definition eines Spielablaufs	6
Tabelle 2 Kontextszenario 2 - Autonome Gruppenaktivität.....	6
Tabelle 3 Kontextszenario 3 - Geführte Gruppenaktivität.....	7
Tabelle 4 Beschreibung der User Stories.....	7
Tabelle 5 Use Case 1	8
Tabelle 6 Use Case 2.....	9
Tabelle 7 Use Case 3.....	9
Tabelle 8 Vor- und Nachteile Windows Desktop Applikation & Web Applikation	13
Tabelle 9 Vor- und Nachteile Windows Store Applikation mit JavaScript.....	13
Tabelle 10 Vor- und Nachteile Single Sensing Device	16
Tabelle 11 Vor- und Nachteile Multiple Sensing Devices Variante 1	17
Tabelle 12 Vor- und Nachteile Multiple Sensing Devices Variante 2	18
Tabelle 13 Vor- und Nachteile Bewegungstracking mit Bildverarbeitung.....	19
Tabelle 14 Realisierbare statische Gesten	22
Tabelle 15 Nicht realisierbare statische Gesten	22
Tabelle 16 Realisierbare fließende Gesten.....	23
Tabelle 17 Nicht realisierbare fließende Gesten.....	24
Tabelle 18 Beschreibung von Klangmodifikationen für statische Gesten.....	25
Tabelle 19 Beschreibung von Klangmodifikationen für fließende Gesten	26
Tabelle 20 Übersicht über die Hauptaufgaben aller Klassen des Sonic Classroom Controllers	31
Tabelle 21 Übersicht über Personen für die Aufnahme der Gesten.....	34
Tabelle 22 Beschreibung JSON Key-Value-Paare	40
Tabelle 23 JSON Nachrichtentypen mit Beispielen.....	41

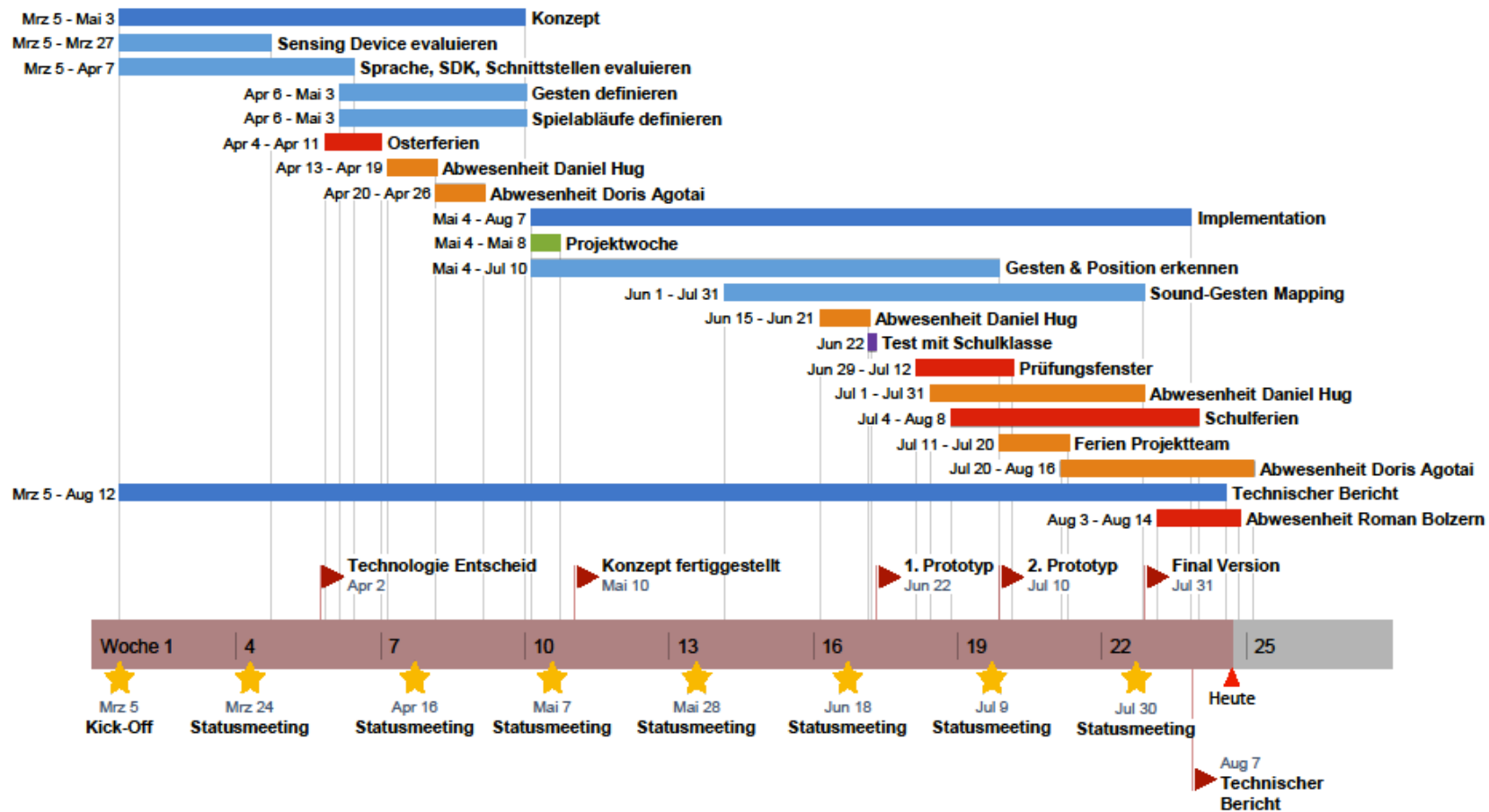
10 Quellenverzeichnis

- [1] Office Timeline, «Die kostenlose Software Nr.1 zur Erstellung von Zeitachsen für PowerPoint - Office Timeline,» [Online]. Available: <https://www.officetimeline.com/de>. [Zugriff am 7 März 2015].
- [2] M.-J. Yoo, J.-W. Beak und I.-K. Lee, «Creating Musical Expression using Kinect,» in *International Conference on New Interfaces for Musical Expression*, Oslo, Norway, 2011.
- [3] Kinect for Windows Team, «KINECT for Windows Blog,» Microsoft, 28 Mai 2015. [Online]. Available: <http://blogs.msdn.com/b/kinectforwindows/archive/2015/05/28/quot-fortissimo-quot-via-kinect.aspx>. [Zugriff am 10 Juli 2015].
- [4] G. Tim, «c't,» heise, 14 November 2014. [Online]. Available: <http://www.heise.de/ct/ausgabe/2014-25-Mit-der-Kinect-V2-MIDI-Geraete-steuern-2450350.html>. [Zugriff am 10 Juli 2015].
- [5] Microsoft Corporation, «Kinect for Windows - Learn the basics,» Microsoft Corporation, [Online]. Available: <https://www.microsoft.com/en-us/kinectforwindows/develop/learn.aspx>. [Zugriff am 15 April 2014].
- [6] Microsoft Corporation, «Human Interface Guidelines v2.0,» 2014. [Online]. Available: <http://go.microsoft.com/fwlink/?LinkId=403900>. [Zugriff am 15 April 2015].
- [7] Microsoft Corporation, «System Requirements,» [Online]. Available: <https://msdn.microsoft.com/en-us/library/dn782036.aspx>. [Zugriff am 12 August 2015].
- [8] Microsoft Corporation, «Technical documentation and tools,» [Online]. Available: <https://www.microsoft.com/en-us/kinectforwindows/develop/downloads-docs.aspx>. [Zugriff am 12 August 2015].
- [9] Microsoft Corporation, «Platzieren des Kinect-Sensors der Xbox One,» [Online]. Available: <http://support.xbox.com/de-DE/xbox-one/kinect/positioning-kinect-sensor>. [Zugriff am 8 August 2015].
- [10] stompboxland.com, «Guitar pedals explained (Part 4): Delay and Reverb,» 5 February 2015. [Online]. Available: <http://stompboxland.com/guitar-pedals-explained-delay-and-reverb/>. [Zugriff am 12 August 2015].
- [11] stompboxland.com, «Guitar pedals explained (Part 3): Modulation,» 5 February 2015. [Online]. Available: <http://stompboxland.com/guitar-pedals-explained-modulation/>. [Zugriff am 12 August 2015].
- [12] stompboxland.com, «Guitar pedals explained (Part 5): Filtering,» 5 February 2015. [Online]. Available: <http://stompboxland.com/guitar-pedals-explained-filtering/>. [Zugriff am 12 August 2015].
- [13] B. Smus, «Chapter 4. Pitch and the Frequency Domain,» 2013. [Online]. Available: <http://chimera.labs.oreilly.com/books/1234000001552/ch04.html>. [Zugriff am 12 August 2015].

- [14] Mozilla Developer Network, «BiquadFilterNode,» [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/API/BiquadFilterNode>. [Zugriff am 12 August 2015].
- [15] sengpielaudio.com, «Berechnung Equalizer Bandpass Filter,» [Online]. Available: <http://www.sengpielaudio.com/Rechner-grenzfrequenzen.htm>. [Zugriff am 13 August 2015].
- [16] Microsoft Corporation, «Visual Gesture Builder: A Data-Driven Solution to Gesture Detection,» [Online]. Available: <https://onedrive.live.com/view.aspx?resid=1A0C78068E0550B5!77743&app=WordPdf>. [Zugriff am 8 August 2015].
- [17] Microsoft Corporation, «AdaBoostTrigger,» [Online]. Available: <https://msdn.microsoft.com/en-us/library/dn785522.aspx>. [Zugriff am 8 August 2015].
- [18] Microsoft Corporation, «RFRProgress,» [Online]. Available: <https://msdn.microsoft.com/en-us/library/dn785524.aspx>. [Zugriff am 8 August 2015].
- [19] M. Taulty, «Kinect for Windows V2 SDK: Hello 'Custom Gesture' World Part 2,» 21 October 2014. [Online]. [Zugriff am 6 May 2015].
- [20] Microsoft Corporation, «Coordinate mapping,» [Online]. Available: <https://msdn.microsoft.com/en-us/library/dn785530.aspx>. [Zugriff am 9 August 2015].
- [21] W3C, «The WebSocket API,» 20 September 2012. [Online]. Available: <http://www.w3.org/TR/2012/CR-websockets-20120920/>. [Zugriff am 12 August 2015].
- [22] W3C, «Web Audio API Modular Routing,» W3C, 22 Juli 2015. [Online]. Available: <http://webaudio.github.io/web-audio-api/#modular-routing>. [Zugriff am 11 August 2015].
- [23] Lärmorama, «Hörempfindungen,» [Online]. Available: http://www.laermorama.ch/m2_hoeren/empfindung_w.html. [Zugriff am 11 August 2015].
- [24] B. Smus, «Basic Web Audio API samples,» 2013. [Online]. Available: <https://github.com/borismus/webaudioapi.com/blob/master/content/posts/filter/filter-sample.js>. [Zugriff am 11 August 2015].
- [25] M. Pilhofer und H. Day, Musiktheorie für Dummies, Wiley-VCH Verlag GmbH & Co., 2012.
- [26] MIDI Manufacturers Association, «MIDI Manufacturers Association - the official source of information about MIDI,» [Online]. Available: <http://www.midi.org/index.php>. [Zugriff am 10 August 2015].
- [27] Microsoft Corporation, «Visual Gesture Builder: Overview,» [Online]. Available: <https://msdn.microsoft.com/en-us/library/dn785529.aspx>. [Zugriff am 8 August 2015].
- [28] Microsoft Corporation, «JointType Enumeration,» [Online]. Available: <https://msdn.microsoft.com/en-us/library/windowspreview.kinect.jointtype.aspx>. [Zugriff am 9 August 2015].

11 Anhang

11.1 Anhang 1 – Planung



11.2 Anhang 2 – Leitfaden Expertengespräche

Im Zuge unserer Bachelor Arbeit (Sonic Classroom) werden mit Experten, die über ein musikpädagogisches Hintergrundwissen verfügen, Gespräche durchgeführt. Ziel dieser Expertenrunde ist, dass wichtige Erkenntnisse über die konkreten Kontextszenarien für die Weiterführung des Projektes gewonnen werden können.

Aufbau

1. Die Ziele des Gespräches werden vom Projektteam dargelegt
2. Das Projektteam stellt die Rahmenbedingungen und konkreten Ideen des Projektes vor
3. Das Gespräch wird mit konkreten Leitfragen geführt
4. Zum Schluss sollte noch Zeit für weitere Anregungen zum Projekt zur Verfügung stehen

Ziele des Gesprächs

Das Projektteam möchte herausfinden welche Kontext- und Nutzungsszenarien auftreten und was die Anforderungen an ein solches System im musikpädagogischen Kontext sind.

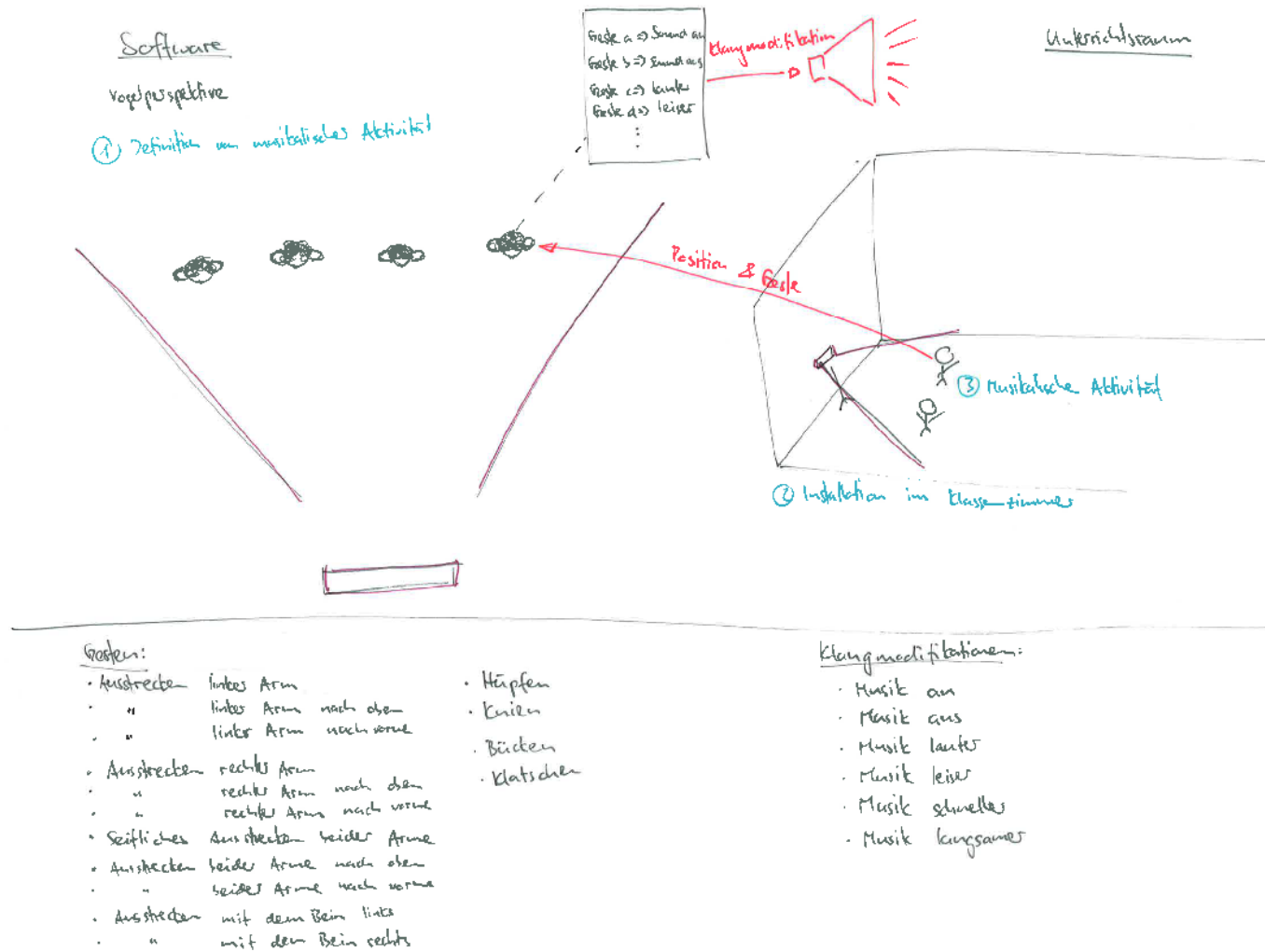
Rahmenbedingungen / Ideen

In einer ersten Phase wird das System mit einem einzelnen Sensor umgesetzt. Aus diesem Grund können maximal 6 Personen zur gleichen Zeit erkannt werden. Der Erfassungsbereich des Sensors ist auf eine Kreisfläche mit 70 Grad Winkel und einem Radius von 4.5 Metern beschränkt. Aus unserer Sicht wäre es sinnvoll ein Set mit ca. 10 Gesten zu definieren und diese dann zu implementieren. Zur räumlichen Bestimmung wo das welche Geste welchen Klang abspielt wird eine bestehende Software des Instituts für 4D-Technologien angepasst und erweitert. Diese Software ermöglicht es Körper in einem 3D Raum zu positionieren und mit Tönen zu versehen. Durch die definierten Gesten soll es möglich sein verschiedene Aktionen auszulösen (z.B.: Klang abspielen, abspielen beenden, lauter, leiser, usw.). Unsere Software soll das Zusammenspiel dieser Komponenten ermöglichen. Konkret soll eine Lehrperson im Voraus die Personen virtuell innerhalb des Raums positionieren und mit den dafür vorgesehenen Tönen verknüpfen können. Danach werden sich die Personen in den dafür vorgesehenen Bereich aufstellen und der Sensor wird eingeschaltet. Unsere Software soll die Position und die Geste welche ausgeführt wird erkennen und anhand der zuvor definierten Einstellungen einen Klang wiedergeben.

Leitfragen

1. *Wie könnte eine konkrete musikalische Aktivität aussehen (Gruppengrößen 2-6, Aktivitäten, körperliche Bewegung, Raumnutzung)?*
2. *Was für Klangmodifikationen sind aus Ihrer Sicht sinnvoll (Lauter, Leiser, Schneller, Langsamer, Ton ein, Ton aus)?*
3. *Sind die Gesten die wir definiert haben aus Ihrer Sicht sinnvoll oder würden Sie andere bevorzugen?*
4. *Würden Sie es bevorzugen, wenn die Lehrpersonen Einstellungen selber tätigen können? (Räumliches Setup, Gesten-Klang-Verbindungen)*
5. *Ist eine Live-Recording Funktion erwünscht oder soll vor allem auf vordefinierte Klänge zurückgegriffen werden?*
6. *Ist es sinnvoll, dass die Installation fix in einem Zimmer bereit steht oder sollte der ganze Aufbau mobil einsetzbar sein?*

11.3 Anhang 3 – Skizze A3 Erklärung für Expertengespräch



11.4 Anhang 4 – Auswertung Expertengespräche

Leitfragen

7. *Wie könnte eine konkrete musikalische Aktivität aussehen (Gruppengrößen 2-6, Aktivitäten, körperliche Bewegung, Raumnutzung)?*

M.C.:

- Verschiedene Raumgrößen (Aula, Schulzimmer)
- Aktivitäten in kleinen Gruppen sehr interessant. Evt. unabhängig von Lehrperson
- Es kann Gruppen- und Einzelaktivitäten geben
- Z.B. 1-2 Personen wechseln die Positionen und arbeiten so miteinander (mehrere Tonspuren – Komposition)

G.N.:

- Normale Klassengröße ist zwischen 10 und 25 Personen
- Gruppenunterteilung möglich
- Je nach räumlicher Möglichkeit wäre ein Projekt mit ganzer Klasse erwünscht
- Möglichkeit mit Kopfhörern in Betracht ziehen um andere nicht allzu sehr zu stören
- Orchestrieren, dirigieren (Gibt es bereits → Museum Berlin)
- Experimentiermodus / Improvisationsmodus

T.C.:

- Raum wird grundsätzlich aktuell zu wenig genutzt
- Nutzen zu Komposition ist vorhanden
- Gruppengröße sollte nicht wirklich relevant. In kleineren Gruppen aber sicher sinnvoll

B.L.:

- Verzögerung im gesamten System sicher wichtig
- Das Zusammenspiel hängt stark vom Timing ab
- Die spielerische Art wäre eher im Theaterbereich oder Tanz von Nutzen da es sich nicht sehr genau sein muss
- Personen die im Rollstuhl sind oder im Bett liegen
 - Personen die nur den Kopf bewegen können
 - Personen mit nur einem Arm
 - Personen die nicht die Kraft haben laute Töne mit Instrumenten zu erzeugen könnten durch unser Tool die Möglichkeit bekommen

8. *Was für Klangmodifikationen sind aus Ihrer Sicht sinnvoll (Lauter, Leiser, Schneller, Langsamer, Ton ein, Ton aus)?*

M.C.:

- Verschiedene Muster (Refrain – Standardrythmus)
- Tonleiter
- Virtuelle Instrumente (Schlagzeug)
- Verschiedene Akkorde
- Geräusche anstatt Sounds (Vogel, Strasse, usw..)

G.N.:

- Kurz, abgehackt, weich, stärker betont, weniger betont, Tonhöhe, Klangfarbe, Akzentuiert

T.C.:

- Versch. Klangfarben wären interessant (dunkel/hell)
- Klanghöhen
- Halleffekt, Frequenz beeinflussen, Tremolo Effekt

B.L.:

- MIDI Gerätesteuerung ist gewünscht
- Mit MIDI kann jeder Ton gesampelt werden
- Für wirkliche musikalische Nutzung nur in Zusammenhang mit MIDI

9. *Sind die Gesten die wir definiert haben aus Ihrer Sicht sinnvoll oder würden Sie andere bevorzugen?*

M.C.:

- Position als Geste (weiter vorne Lauter / hinten leiser)

G.N.:

- Eventuell zu mechanisch / statisch
- Welle, Stampfen, Tanzen

T.C.:

- Fließende Gestenerkennung wäre natürlich auch interessant

10. Würden Sie es bevorzugen, wenn die Lehrpersonen Einstellungen selber tätigen können?
(Räumliches Setup, Gesten-Klang-Verbindungen)

M.C.:

- Sollte möglich sein aber trotzdem soll es vordefinierte Presets geben mit versch. Anzahl Akteuren
- Es muss einfach bedienbar sein

G.N.:

- Presets wären gut aber es sollte auch möglich sein eigene Einstellungen zu definieren

T.C.:

- Für die Allgemeinheit sind vordefinierte Sets sinnvoller. Es gibt aber auch technisch interessiertere / versiertere Lehrpersonen die einen Edit-Modus sicher begrüßen würden

B.L.:

- Einige Grundeinstellungen sollten schon vorhanden sein
- Bei konkreter Nutzung ist eine Schulung sicher nötig

11. Ist eine Live-Recording Funktion erwünscht oder soll vor allem auf vordefinierte Klänge zurückgegriffen werden?

M.C.:

- Looping & Sampling wäre sehr interessant

G.N.:

- Recording / Live Modus tönt sehr interessant

T.C.:

- Looping & Recording interessant

12. Ist es sinnvoll, dass die Installation fix in einem Zimmer bereit steht oder sollte der ganze Aufbau mobil einsetzbar sein?

M.C.:

- In einem Schritt mobil

G.N.:

- Eher fix Installation
- Aufwand sollte so klein wie möglich sein

T.C.:

- Fix wäre ideal (Markierungen). Mobil erhöht die Komplexität und stellt somit eine Hemmschwelle dar

B.L.:

- Eher eine fixe Installation wäre denkbar

Weitere Anmerkungen:

M.C.:

- Gabriel Imthurn könnte Kontakt zu einer Schulklasse herstellen
- Thomi Christ behilflich bei einer Standardmusik Bibliothek

T.C.:

- Grundsätzlich soll es nicht das Ziel sein unbedingt auf bereits bestehende musikalische Aktivität die bekannt sind abzielen. Es ist vielmehr interessant was experimentell neu entdeckt werden kann. Experimentelle/Kreative Klänge.
- Die ganze Aktivität kann auch ein sozialer Aspekt haben. Einander nicht in die Quere kommen zum Beispiel.
- Durchaus auch für Oberstufe geeignet

B.L.:

- Exotisch für den täglichen Schulbetrieb (Eher im Zuge eines Projektes)

Legende:

M.C.: Markus Clovjecsek

G.N.: Gabriele Noppeney

T.C.: Thomas Christ

B.L.: Boris Lanz

11.5 Anhang 5 – Testmatrix Gesten

	Rechtes Knie anziehen	Linkes Knie anziehen	Rechter Arm gegen oben heben	Linker Arm gegen oben heben	Rechter Arm ausgestreckt vorne heben	Linker Arm ausgestreckt vorne heben	Rechter Arm ausgestreckt seitlich heben	Linker Arm ausgestreckt seitlich heben	Kopf nach vorne beugen	Oberkörper nach vorne beugen	Klatschen über Kopf	Klatschen vorne	Bücken	Kniebeuge	Hüpfen	Kick mit dem rechten Bein	Kick mit dem linken Bein	Rechtes Bein ausstrecken	Linkes Bein ausstrecken	Beide Arme nach vorne ausstrecken	Beide Arme seitlich ausstrecken	Rechter Arm nach vorne ausstrecken	Rechter Arm nach oben ausstrecken	Rechter Arm nach rechts ausstrecken	Linker Arm nach vorne ausstrecken	Linker Arm nach oben ausstrecken	Linker Arm nach links ausstrecken
Linker Arm nach links ausstrecken																											
Linker Arm nach oben ausstrecken																											
Linker Arm nach vorne ausstrecken																											
Rechter Arm nach rechts ausstrecken																											
Rechter Arm nach oben ausstrecken																											
Rechter Arm nach vorne ausstrecken																											
Beide Arme seitlich ausstrecken																											
Beide Arme nach vorne ausstrecken																											
Linkes Bein ausstrecken																											
Rechtes Bein ausstrecken																											
Kick mit dem linken Bein																											
Kick mit dem rechten Bein																											
Hüpfen																											
Kniebeuge																											
Bücken																											
Klatschen vorne																											
Klatschen über Kopf																											
Oberkörper nach vorne beugen																											
Kopf nach vorne beugen																											
Linker Arm ausgestreckt seitlich heben																											
Rechter Arm ausgestreckt seitlich heben																											
Linker Arm ausgestreckt vorne heben																											
Rechter Arm ausgestreckt vorne heben																											
Linker Arm gegen oben heben																											
Rechter Arm gegen oben heben																											
Linkes Knie anziehen																											
Rechtes Knie anziehen																											

Gleich Geste (zuverlässig)

Geste wird nicht ausgelöst

Geste wird ausgelöst --> können nicht gleichzeitig genutzt werden

11.6 Anhang 6 – Gestengruppierung

Geste	Exclusion ID
Linker Arm nach links ausstrecken	1
Linker Arm nach oben ausstrecken	2
Linker Arm nach vorne ausstrecken	3
Rechter Arm nach rechts ausstrecken	4
Rechter Arm nach oben ausstrecken	5
Rechter Arm nach vorne ausstrecken	6
Beide Arme seitlich ausstrecken	7/8
Beide Arme nach vorne ausstrecken	9/10
Linkes Bein ausstrecken	11
Rechtes Bein ausstrecken	12
Kick mit dem linken Bein	13
Kick mit dem rechten Bein	14
Hüpfen	15
Kniebeuge	16
Bücken	17
Klatschen vorne	18
Klatschen über Kopf	19/20
Oberkörper nach vorne beugen	17
Kopf nach vorne beugen	21
Linker Arm ausgestreckt seitlich heben	1/7
Rechter Arm ausgestreckt seitlich heben	4/8
Linker Arm ausgestreckt vorne heben	3/9
Rechter Arm ausgestreckt vorne heben	6/10
Linker Arm gegen oben heben	2/19
Rechter Arm gegen oben heben	5/20
Linkes Knie anziehen	13
Rechtes Knie anziehen	14

11.7 Anhang 7 – Codeabgrenzung Ludosonica

Files	Abgrenzung
SoundCollection.Kinect.js	Gesamtes Script ausser Methoden add, remove, play, stop, _findEmptySpotInSoundList
ui.js	<ul style="list-style-type: none"> - VisualFeedback (Zeile 178 - 284) - SoundProperties (Zeile 288 - 344) - KinectEventList (Zeile 1267 - 1322 / 1576 - 1620 / 1631 - 1668)
Editor.Kinect.js	<ul style="list-style-type: none"> - setSoundProperties / resetSoundProperties (Zeile 95 - 113) - startPlay (Zeile 478 - 488 / 532 - 565)
Play.Kinect.js	Ganzes Script ausser Methode preloadSounds
PlayfulExporter.js	Methode parse (Zeile 19 - 21)
Sidebars.File.js	Methode onFileInputChange (Zeile 34 - 50)
Sidebars.SoundProperties.js	Ganzes Script
Loader.js	Methode handleJSON (Zeile 405 - 407)
Tools.Kinect.Modes.js	Ganzes Script
Viewport.Kinect.js	<ul style="list-style-type: none"> - Grauer Kinect Bereich Editor Modus (Zeile 32 - 44) - Kinect Bereich Play Modus (Zeile 116 - 167)
index.html	Ganze Seite
kinect.html	Checkbox jQuery (Zeile 923 – 956)

11.8 Anhang 8 – Vorgehen Test mit Schulklasse

Um den ersten Prototyp unserer Software zu testen wird ein Test in einer realen Umgebung mit einer Schulklasse durchgeführt. Es sollen verschiedene Szenarien und Spielabläufe ausgetestet werden. Um herauszufinden wie das Testsetting die Schüler anspricht wird zum Schluss ein Feedback der Schüler aktiv eingefordert.

Szenarien

Aus verschiedenen Inputs durch die Projektbetreuer und Expertengespräche haben sich folgende Szenarien herauskristallisiert:

Test mit Geräuschen (Vogelgezwitscher, Strassenlärm, usw.)

Test mit wirklich musikalischen Klängen (Hihat, Drum, Kick, usw.)

Test ohne Visualisierung am Beamer

Test mit Visualisierung am Beamer

Die verschiedenen Tests werden immer wie folgt getestet: Zuerst läuft die erste Person an seinen Platz, dann die zweite, dritte usw. bis und mit 6 Personen (maximale Anzahl).

Test 1 - Instrumental

Aufbau

In diesem Test werden auf 2, 4 und dann 6 Körpern im Raum je einen Klang eines Instruments gesetzt. Welcher durch eine definierte Geste einmalig ausgelöst werden kann.

Ziel

Es soll den Probanden möglich sein, einen Rhythmus zusammen zu erzeugen der aufeinander abgestimmt ist.

Schlussfolgerungen

Die Response Zeit ist ein Faktor. Es ist für die Schüler schwierig damit umzugehen wenn das Klatschen einmal später kommt und einmal zum richtigen Zeitpunkt.

Test 2 - Instrumental mit Loop Gruppe

Aufbau

In diesem Test werden auf 2, 4 und dann 6 Körpern im Raum je einen Klang eines Instruments gesetzt. Welcher durch eine definierte Geste nicht einmalig sondern als wiederholenden Tonablauf ausgelöst werden kann.

Ziel

Es soll den Probanden möglich sein, einen Rhythmus zusammen zu erzeugen der aufeinander abgestimmt ist.

Schlussfolgerungen

Man kann sehr interessante Dinge machen mit dem Looping.

Test 3 - Instrumental mit Loop Einzelperson

Aufbau

In diesem Test werden auf allen 6 Körpern im Raum je einen Klang eines Instruments gesetzt. Welcher durch eine definierte Geste nicht einmalig sondern als wiederholenden Tonablauf ausgelöst werden kann.

Ziel

Es soll einem einzigen Probanden möglich sein, einen Rhythmus zu erzeugen der aufeinander abgestimmt ist.

Schlussfolgerungen

Auch interessant.

Test 4 - Musik

Aufbau

In diesem Test wird auf einem flächendeckenden Körper mehrere Gesten definiert um ein laufendes Musikstück zu starten/stoppen oder anders zu manipulieren. Es wird mit 3 Personen getestet.

Ziel

Es soll den Probanden möglich sein, ein Musikstück zu starten/stoppen und zu manipulieren. Ohne einander gross zu behindern.

Schlussfolgerungen

Sehr lustig und kein Problem mehr mit Response Time. Fließende Gesten sind sehr attraktiv.

Test 5 - Umgebungsmusik Gruppe

Aufbau

In diesem Test werden auf allen 6 Körpern im Raum je eine Umgebungsmusik definiert. Welche durch eine definierte Geste einmalig ausgelöst werden kann.

Ziel

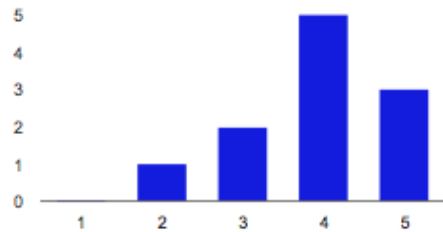
Es soll eine Art Klangspeil von einer realen Umgebung nachgespielt werden.

Schlussfolgerungen

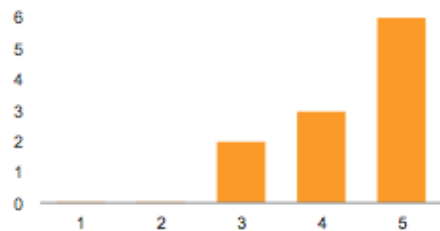
Sehr lustige Effekte ergeben sich von alleine. Problem eventuell wenn der Sound leise anfängt das man ihn nicht hört im "Chaos" und dann nochmals startet.

11.9 Anhang 9 – Auswertung Test mit Schulklasse

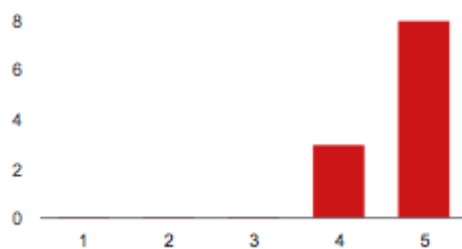
Hast du die allgemeine Bedienung des Systems als einfach empfunden?



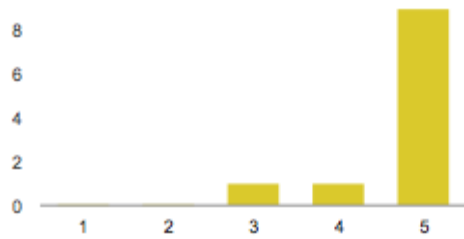
Der erste Test mit den einmalig ausgelösten rhythmischen Klängen. Wie hast du diesen empfunden?



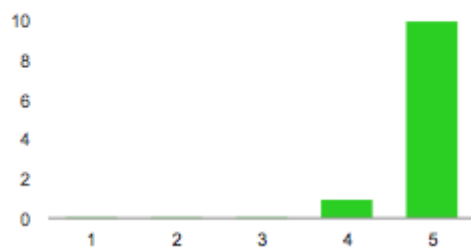
Der zweite Test mit den wiederholenden rhythmischen Klängen. Wie hast du diesen empfunden?



Der dritte Test mit den wiederholenden rhythmischen Klängen die du a Einzelperson ausgelöst hast. Wie hast du diesen empfunden?



Der vierte Test mit dem Musikstück das ihr zusammen manipuliert habt. Wie hast du diesen empfunden?



Der fünfte Test mit der Umgebungsmusik die ihr zusammen gespielt habt. Wie hast du diesen empfunden?

