FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Physics Based Music Visualization

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Medieninformatik und Visual Computing

eingereicht von

Andreas Schmid
Matrikelnummer 0725136

an der Fakultät für Informatik
der Technischen Universität Wien

Betreuung: Dr. Stefan Ohrhallinger
Mitwirkung: Alex Hauer
Jürgen Giefing

Wien, 20.12.2013

(Unterschrift Verfasser) (Unterschrift Betreuer)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Physics Based Music Visualization

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Media Information and Visual Computing

by

Andreas Schmid
Registration Number 0725136

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Dr. Stefan Ohrhallinger
Assistance: Alex Hauer
Jürgen Giefing

Vienna, 20.12.2013

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Andreas Schmid
Wiedner Gürtel 8/6

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

(Ort, Datum) (Unterschrift Verfasser)

Danksagung

Ich möchte mich an dieser Stelle bei all denen bedanken, die mich bei der Anfertigung meiner Bakkalaureatsarbeit so kräftig unterstützt haben.

Acknowledgements

I would like to take this opportunity to thank all who have supported me in preparing my bachelorthesis.

Kurzfassung

Das Ziel dieser Arbeit war es, einige Ansätze zu liefern, wie man ein Grundgerüst einer Visualisierungssoftware erstellt, welche auf Basis von menschlichen Emotionen Musik auf grafische Primitive projiziert. Dieser Prototyp eines visuellen Media Players stellt dem Benutzer ein Werkzeug zur Verfügung, um aus Musik bestimmte einzelne Werte auszulesen und diese Werte in einer Mappingdatei in bestimmte Formen, Farben und Transformationen umzuwandeln.
Dies wurde durch sogenannte Commands und einer veränderbaren Ausgabedarstellung bewerkstelligt. Mit diesen Werkzeugen hat der Benutzer die Möglichkeit, die Visualisierung komplett zu steuern. Die Commands beschreiben, was die Visualisierungssoftware beherrscht. Zum Beispiel Erstellung primitiver Formen, Bewegung von Objekten oder Farbveränderung der Formen. In der Ausgabedarstellung wird beschrieben, was das Programm anzeigen soll. Unser Prototyp beherrscht zwei Ausgabedarstellungen, die reine Textausgabe und die grafische Ausgabe. In der ersten von zwei Benutzerstudien haben wir die Probanden gebeten einzelne kurze Musikstücke und Animationen anzusehen und nach bestimmten Kriterien zu bewerten. Dies wurde dann in eine Mappingdatei übertragen und für die zweite Benutzerstudie als Basis verwendet. Im zweiten Durchgang mussten die Testpersonen fünf Media Player inklusive unserem Prototypen mit zwei verschiedenen Mappings, nach deren Visualisierung und ob Musik zur Visualisierung passt, bewerten.
Das Ergebnis war anders als erwartet, die Benutzer bewerteten die aktuellen Media Player im Bereich wie gut die Musik zur Visualisierung passt besser als unseren Prototypen. Es wird versucht diese Ergebnisse zu analysieren und in weiteren Verfahren so zu verbessern damit wir unsere Ansätze erfolgreich umsetzen können.
In dieser Bachelorarbeit befindet sich auch eine kleine Analyse der verwendeten Bibliotheken inklusive deren Vor- und Nachteile. Weiters kann man den Prototypen als Ausgangsbasis für weitere Versuche verwenden.

Abstract

The aim of our bachelor thesis was to develop a concept on the basic structure of software that visualizes human emotions when listening to music into graphic primitives. Our prototype of this visual media player allows the user to read data from a music file and map this data into a file that converts it to special forms, colors and transformations.
This goal was accomplished with special commands and a modifiable theme. With these tools, the user can completely control the visualization. The commands describe the options of what can be accomplished with the software. In our case, they create primitive forms, move them, and change their color. The theme shows what the program is able to display. Two examples of this prototype are the textual theme and the graphical theme.
In the first user study, the users were asked to listen to short music files or watch some short animations and to evaluate those. This data was then transferred to a mapping file and taken as basis for a second user study. In the second user study, the users had to evaluate five media players, including ours, using two different mappings - one was their visualization and the other an evaluation on if music and visualization match. The results were not as expected - the users evaluated the existing media players better than our prototype in visualization-music mapping. We are analyzing the results and working on redesigning our algorithms in order to have a more successful prototype.
In this bachelor thesis, there is also an analysis of the libraries used. The prototype can be used as a basis for future work in this field.

Inhaltsverzeichnis

KAPITEL 1

Einführung

1.1 Motivation

Meine Motivation liegt darin, Musikstücke verbunden mit dem mir übergebenen Mapping so darzustellen, dass die Visualisierung bessere Ergebnisse liefert, als die State-of-the-Art Player. Dies wird erreicht indem die Visualisierungssoftware nur Formen, Bewegungen und Farben dem Mapping anbietet die der Mensch am besten unterscheiden kann und wahrnimmt. Hierbei stützte ich mich auf bestimmte Thesen von Stephen E Palmer. Mit dem Einbinden der menschlichen Wahrnehmung in die Visualisierung soll das Ergebnis eine ästhetisch schönere Visualisierung darstellen.

1.2 Problembeschreibung

Wenn man die Visualisierung der State-of-the-art Player begutachtet, stellt man fest, dass diese die Visualisierung aufwendig mit vielen Effekten und ähnlichem darstellen, aber die Musik dazu nur nebenbei abgespielt wird. Das heißt, dass die Bewegungen, die dargestellte Visualisierung und die Farbe nicht mit der Musik zusammenpasst. Ein Beispiel ist, dass es nicht darauf ankommt ob ein schnelles oder langsames Musikstück abgespielt wird, die Darstellung bleibt zu großen Teilen gleich. Es fehlt ein in sich abgeschlossenes Gesamtpaket mit einer passenden auf die Musikreagierende Visualisierung.

1.3 Ziel der Arbeit

Unser Ziel ist somit herauszufindend, ob es eine Möglichkeit gibt, die menschliche Psychologie in den Prozess der Visualisierungsgenerierung einzubinden um eine Visualisierung darzustellen die nicht mit Effekten um sich wirft, sondern um den Zuseher das Gefühl zu geben, die Visualisierung passt zur Musik. Zum Beispiel wird eine laute schnelle Musik eher mit mehrkantigen Formen dargestellt als mit einem Kreis oder die Formen bewegen sich schneller durch das Bild

anstatt langsamer und weicher wie bei einem Musikstück das eine langsame Tempo besitzt. Dies soll erreicht werden indem der Visualisierung nur bestimmte Formen, Farben und Transformationen zur Verfügung stehen. Mit diesen Einschränkungen soll ein Ergebnis erreicht werden, dass dem Benutzer das Gefühl einer in sich geschlossene Darstellung mit der Musik vermittelt.

1.4 Methodischer Ansatz

Die erste kleine Aufgabe in diesem Projekt bestand darin eine passende Bibliothek zu finden die unseren Anforderungen gerecht wurde. Kurz zusammengefasst musste sie eine breite Palette an Systemen und OpenGL in Java unterstützen.
Mein Hauptaufgabe in dieser Arbeit bestand darin ein grundlegendes Framework zu erstellen, welcher folgende Punkte beinhalten musste:

- Darstellung von gut für den Menschen unterscheidbaren Formen
- Darstellung von Farbe und Ausblendung
- Darstellung von verschiedenen Transformationen
- Schnittstelle zwischen dem Einlesen des Musikstückes, dem Mapping und der Visualisierung
- Kommandos, welche die Mappingdatei aufrufen kann um der Visualisierung zu übermitteln was als nächstes angezeigt werden soll.
- Definition von relevanten Dimensionen in der Visualisierung und Diskretisierung ihrer Skalen

KAPITEL 2

Stand der Technik

2.1 Wahrnehmung

In diesem Kapitel möchte ich auf die verwendete Literatur eingehen und die für meine Arbeit wichtigsten Kapitel hervorheben und kurz beschreiben. Die wichtigsten Themen für den Part der Visualisierung waren:

- **Visuelle Wahrnehmung:** ganz allgemein gehalten
- **Farben:** welche Farben welche Bedeutung haben
- **Transformationen:** welche Transformationen für ein grafisches Framework benötigt werden

2.2 Visuelle Wahrnehmung des Menschen

Die Wahrnehmung ist ein kontinuierlicher Prozess, der es uns erlaubt mithilfe unserer Sinnesorgane wie Sehen, Hören, Riechen, Schmecken und Fühlen, Informationen und Reize aufzunehmen und auszuwerten. Dieser Prozess erfolgt durch unterbewusstes und/oder bewusstes Wahrnehmen der Umwelt. Menschen orientieren sich mittels der Wahrnehmung in Ihrer Umgebung, erkennen beispielsweise Gefahren oder bewerten die Stimmung Ihrer Mitmenschen. Dies macht die Wahrnehmung für uns Menschen überlebenswichtig. [5]

2.3 Farbwahrnehmung

Die Farbwahrnehmung ist ein Teilbereich des Sehens und wird durch das Umfeld beeinflusst. Das Auge verfügt über verschiedene Lichtsinneszellen, die von Licht jeweils unterschiedlicher Wellenlänge besonders stark wahrgenommen werden. Aus diesem Empfinden bildet das Gehirn eine Farbempfindung. [5] [8]

2.3.1 Simultankontrast

Unter Simultankontrast versteht man den Effekt, dass benachbarte Farben in ihrer Wirkung sich wechselseitig beeinflussen. Die größere Fläche wirkt auf die kleinere Fläche und die visuelle Farbunterschiede werden durch Farbton, Sättigung und Helligkeit bewertet. [5] [8]

2.3.2 Warm-kalt-Kontrast

Unter Warm-kalt-Kontrast versteht man den Effekt, dass wir durch die Assoziation von Feuer und Wärme die Farbtöne Gelb bis Rot als warm empfinden. Wasser und Schnee werden mit blauen Farbtönen in Verbindung gebracht und gehören somit zu den kalten Farben. Die Kalten und Warmen Farben stehen sich im Farbkreis gegenüber und sind somit die Komplementärfarben. [5] [8]

2.3.3 Farbassoziationen

Violett: Extravaganz, Feminismus, Macht, Feierlichkeit, Magie, Modernität, Nostalgie, Aussergewöhnliches
Blau: Technik, Natur, Wasser, Gelassenheit, Kühle, Ruhe, Seriosität
Cyan: Sachlichkeit, Kühle, Frische, Sportlichkeit, Winter, Jugendlichkeit, Distanz
Grün: Hoffnung, Natur, Gift, Frühling, Ruhe, Gesundheit, Erholung
Gelb: Sonne, Helligkeit, Modernität, Gift, Neid, Optimismus, Sauberkeit
Orange: Energie, Wärme, Unruhe, Innovation, Dynamik, Spass, Vergnügen, Künstlichkeit
Rot: Liebe, Energie, Blut, Krieg, Leidenschaft, Gefahr, Wärme, Feuer
Magenta: Jugendlichkeit, Romantik, Dynamik, Wärme, Weiblichkeit, Kommunikation
Schwarz: Geheimnis, Tradition, Macht, Sachlichkeit, Kraft, Dunkelheit
Grau: Sachlichkeit, Wahrheit, Seriosität, Neutralität, Technik
Weiss: Sauberkeit, Sachlichkeit, Gespenst, Schnee, Helligkeit, Wahrheit, Seriosität

[5]

2.4 Gestaltpsychologie

Die Gestaltpsychologie ist die Wahrnehmungslehre in der Psychologie, deren Grundannahme lautet, dass die menschliche Wahrnehmung das Ganze nicht objektiv betrachtet sondern bereits strukturiert bevor diese zum Gehirn zur Begutachtung weitergeleitet wird. Die Gestaltpsychologie wird in mehrere Gesetze unterteilt, welche die Ergebnisse der Wahrnehmung der verschiedenen Formen und ihrer Beziehung zueinander beschreiben. [5]

2.4.1 Gesetz des gemeinsamen Schicksals

Dieses Gesetz befasst sich mit der ähnlichkeit in einer dynamischen Darstellung. So werden beispielsweise Figuren, die sich in die Selbe Richtung bewegen als Gruppe angesehen. Im Allgemeinen erkennen wir Figuren oder Formen, die ähnliche Bewegungseigenschaften aufweisen, als Einheit an. [5]

2.4.2 Gesetz von der einfachen Gestalt

Das Gesetz von der einfachen Gestalt ist das Grundgesetz der menschlichen Wahrnehmung. Die Wahrnehmung wird grundlegend auf Bewegung und auf einfache geometrische Formen, wie Dreiecke, Rechtecke und Kreise zurückgeführt. Die Wahrnehmung von einfachen geometrischen Gestalten ist in uns durch die Evolution verankert. [5]

2.4.3 Gesetz der Gleichheit

Das Gesetz der Gleichheit besagt, dass Elemente die gemeinsame Unterscheidungsmerkmale zur Umgebung aufweisen, vom Betrachter als zusammengehörig wahrgenommen werden. Formen und Farben verstärken dieses Empfinden. [5]

2.4.4 Gesetz der Erfahrung

Das Gesetz der Erfahrung besagt, dass wir bekannte Formen, Symbole oder Körper auch bei starker Transformation noch erkennen. [5]

2.4.5 Gesetz der Figur-Grund-Trennung

Das Gesetz der Figur-Grund-Trennung besagt, dass wir nur Wahrnehmen können wenn das Wahrnehmungsfeld in unterschiedliche Bereiche gegliedert wird. Um ein Objekt wahrnehmen zu können muss es sich von seinem Umfeld abheben. Dieser Effekt entsteht durch Konturen, Kontraste, Texturen, Bewegungen und Farben. [5]

2.5 Transformationen

Um Objekte in einem Koordinatensystem zu transformieren, also verschieben, skalieren und rotieren, muss ein geeignetes mathematisches Konzept zugrunde gelegt werden, welches eine geschlossene Behandlung aller Operationen zulässt.
Grundsätzlich werden zwei Arten von Transformationen unterschieden:

- Nicht formverändernde Transformationen
- Formverändernde Transformationen

2.5.1 Skalierung

Wenn ein Objekt skaliert wird, wird die Höhe oder Breite des Objektes gleichermaßen verändert.

2.5.2 Rotation

Wenn ein Objekt rotiert wird, spricht man von einer Drehung des Objektes um den Ursprung.

2.5.3 Translation

Wenn eine Translation stattfindet, wird ein Objekt verschoben. Eine Translation kann aber nicht wie die Skalierung oder Rotation durch eine Multiplikation mit einer 2x2 Matrix erzielt werden. Durch die Darstellung der Transformation als homogene 3x3 Matrix können alle affinen Transformationen dargestellt werden. [4]

2.6 Analyse und Vergleich

In diesem Abschnitt werde ich die folgenden Media Player mit Visualisierungen analysieren und vergleichen.

- Windows Media Player Version 12
- WinAmp Version 5.666
- iTunes Version 11.1.3

2.6.1 Windows Media Player

Die erste Version von Windows Media Player erschien im Jahr 1991. Dieser Player spielte Animationsdateien mit der Endung .mmm ab und unterstützte durch Erweiterungen weitere Formate. Da der Windows Media Player eine Komponente von Windows war, hatte er dieselbe Versionsnummer. [7]

- **Formen:** Formen werden nicht wirklich verwendet. Darstellung ist mehr Abstrakt. Wenn Formen dargestellt werden bestehen sie meist aus Linien.
- **Farbdarstellung:** Farben sind eher matt. Farben unterscheiden sich sehr stark. Farben passen aber nicht wirklich zur Musik
- **Translation:** Linien werden bewegt, rotiert und skaliert.

2.6.2 WinAmp

1997 wurde Winamp von Justin Frankel, Dmitry Boldyrev und Gianluca Rubinacci als Shareware-Version veröffentlicht. Nach 16 Jahren, Ende November 2013 ließ AOL verkünden, dass man Winamp ab Dezember 2013 nicht mehr herunterladen werden könne. [2]

- **Formen:** Die Darstellung ist mehr abstrakt. Keine bis wenige Formen (Linien) werden verwendet.

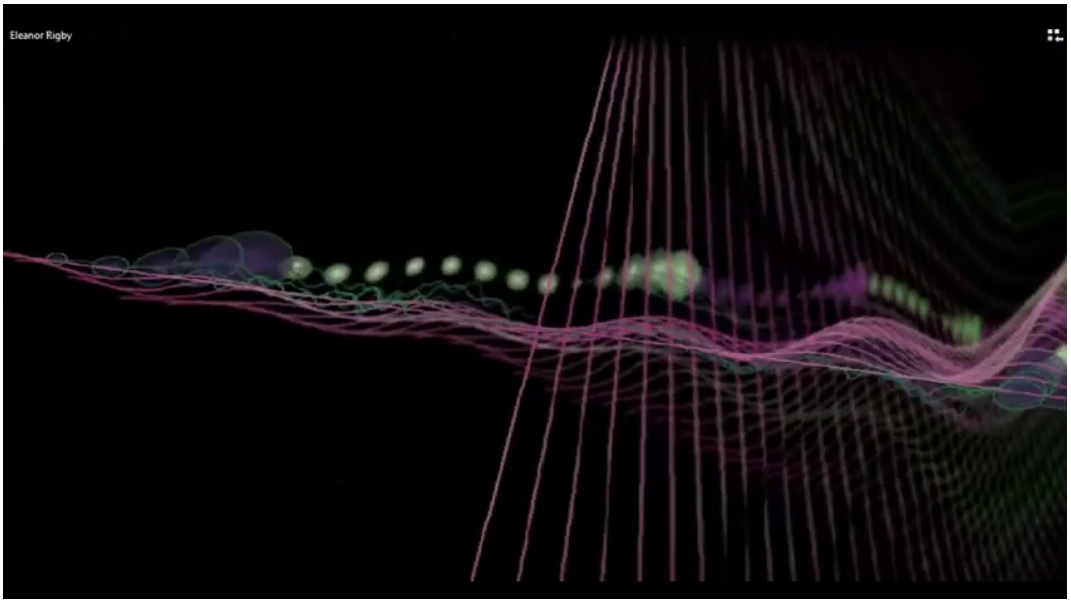

Abbildung 2.1: Visualisierung des Windows Media Players

Abbildung 2.2: Visualiserung des WinAmp Players

- **Farbdarstellung:** Es wird das komplette Farbspektrum verwendet, aber ohne die menschlichen Emotionen einzubinden. Zufällige Farben.
- **Translation:** Objekte/Effekte werden rotiert, skaliert und bewegt.

2.6.3 iTunes

Die erste Version von iTunes basierte noch aus großen Teilen von SoundJam MP, welches Apple im Jahr 2000 gekauft hatte. Die Entwickler von SoundJam MP arbeiteten fortan für Apple und entwickelten den Media Player weiter. Apple veröffentlichte dieses Produkt Im Jahr 2001 unter einem neuen Namen, nämlich iTunes. [6]

Abbildung 2.3: Visualiserung des iTunes Players

- **Formen:** Es werden aufwendige Formen, auch 3D, verwendet. Viele Effekte.
- **Farbdarstellung:** Es wird das komplette Farbspektrum verwendet. Menschliche Emotionen werden nicht berücksichtigt. Dasselbe Lied, hat unterschiedliche Farben nach mehrmaligen Abspielen.
- **Translation:** Objekte werden bewegt, rotiert und skaliert.

KAPITEL 3

Methodik

3.1 Konzept

Insgesamt wurden zwei Benutzerstudien durchgeführt, welche gemeinsam mit Alexander Hauer und Jürgen Giefing durchgeführt und anschließend ausgewertet wurden. Nachfolgend gibt es eine Auswertung der Benutzerstudien mit dem Blick auf den Visualisierungspart. Die Auswertung der anderen Themen wie das Auslesen von Musikteilen und dem Erstellen des Mappings, befinden sich in den Bachelorarbeiten von Jürgen Giefing und Alex Hauer. [1] [3]

3.1.1 Erste Benutzerstudie

In der ersten Benutzerstudie mussten die Probanden uns die Emotion nennen die sie in bestimmten Musikstücken empfanden. Um Emotionen effektiv bewerten zu können, gibt es laut Literatur nur drei Bewertungsachsen: [9]

- Aktivität
- Kraft
- Bewertung (ob sich etwas gut oder schlecht anfühlt)

Genauere Informationen zu diesem Thema finden sie in der Bachelorarbeit von Jürgen Giefing. [1] Die Lieder waren Midi Versionen und hatten somit keine Sprache. Folgende Musikstücke wurden den Personen vorgespielt:

- AC/DC - T.N.T.
- Aerosmith - Cryin
- Alcazar - Crying At The Discotheque
- Alice Cooper - Schools out

- Annie Lenox - Rain
- Aretha Franklin - Respect
- Arrested Development - Everyday People
- Average White Band - Pick Up The Pieces
- The Beatles - Eleanor Rigby
- Bee Gees - Stayin alive

3.1.2 Ergebnisse der Benutzerstudien

Nun folgen die Ergebnisse der ersten Benutzerstudie mit dem Blick auf die Visualisierung.

Formen

Ich habe mich für grundlegende Formen entschieden da diese der Mensch laut dem Gesetz der einfachen Gestalt schon von klein auf kennt und in uns verankert sind. Somit haben wir folgende klar unterscheidbaren Formen verwendet:

- Punkt
- Linie
- Kreis
- Dreieck
- Viereck

Unser Framework beinhaltet die Methode um Primitive gefüllt oder nur durch Linien darzustellen. Die Benutzerstudie hat aber ergeben dass gefüllte oder nur durch Linien dargestellte Formen keinen Unterschied ausmachen. Weitere Ergebnisse der Studie bezüglich Formen:

- **Aktivität:**
  - Dreiecke werden benutzt
  - Kreise und Rechtecke werden nicht benutzt
- **Kraft:**
  - Punkte werden nicht benutzt
- **Bewertung:**
  - Punkte werden nicht benutzt

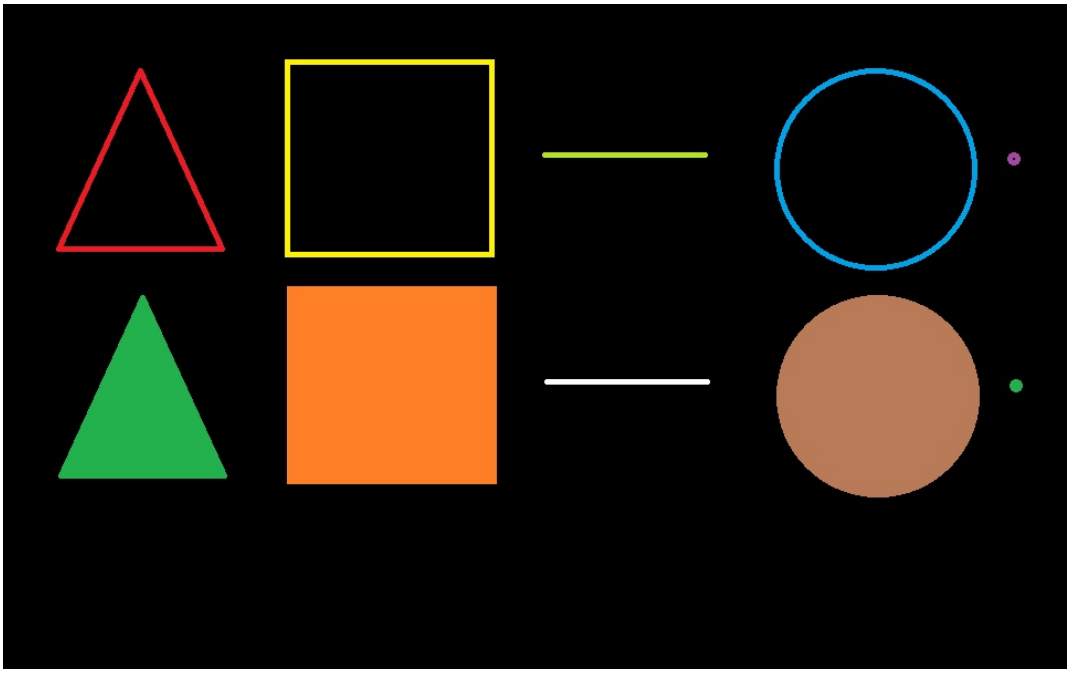

Abbildung 3.1: Formen die der Prototyp darstellen kann

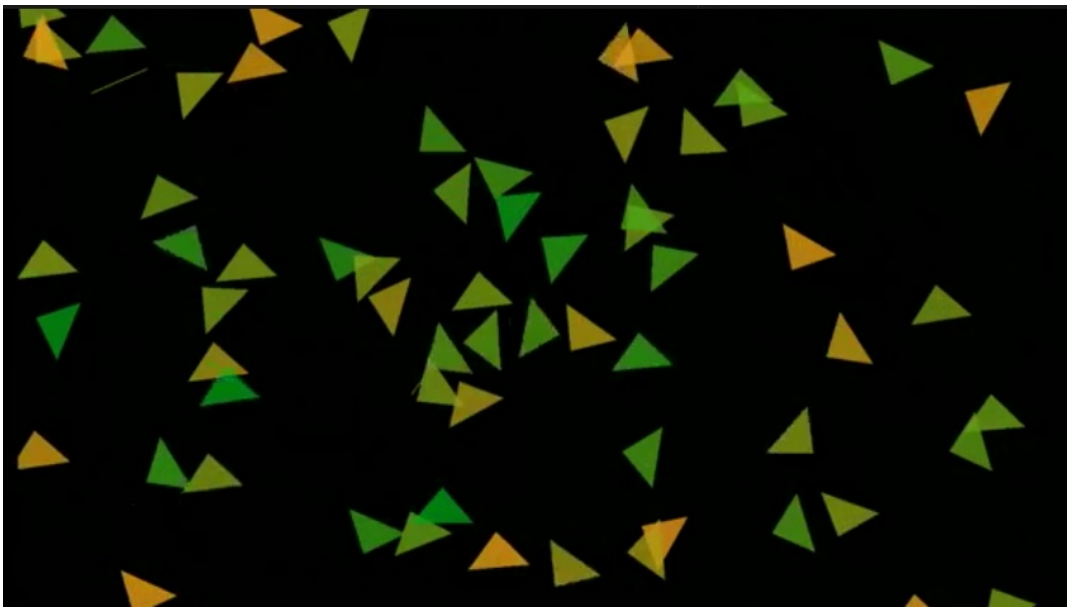

Abbildung 3.2: Prototyp in Aktion - Lied:annie lenox - rain

Farben

Bei unseren Prototypen haben wir beschlossen uns nicht nur mit den Grundfarben (Rot, Grün, Blau) zufrieden zu geben. Wir wollten das ganze Farbspektrum und teilten somit die Farben in

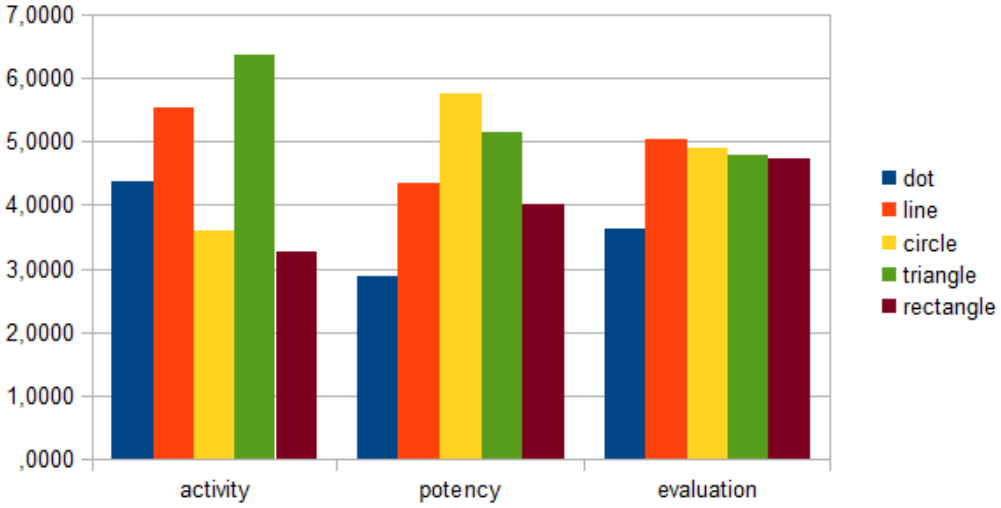

Abbildung 3.3: Ergebnisse basierend auf Formen

ihren Rot, Grün und Blau Anteil. Weiters beherrscht unser Framework noch dem Alphawert, der dafür zuständig ist, die erstellte Form dünkler oder hellen darzustellen. In der Literatur fanden wir auch noch Farbassoziationen die dabei helfen konnten wie ein Mensch auf eine gewisse Farbe reagiert.

Die Ergebnisse der Benutzerstudie bezüglich Farben:

- **Aktivität:**
  - Die Objekte/Formen haben einen hohen Grünanteil
- **Kraft:**
  - Die dargestellten Formen sind dünkler
  - Formen haben einen geringeren Blauanteil, dafür aber mehr Grün
- **Bewertung:**
  - Bezüglich Bewertung gibt es keine Aussage

Transformationen

In unserem Grundgerüst verwenden wir die typischen Transformationen die in jedem grafischen Framework angeboten werden. Mit diesen kann man jeden Punkt im Raum erreichen und verändern.

Das Ergebnis der Benutzerstudie zum Thema Transformationen:

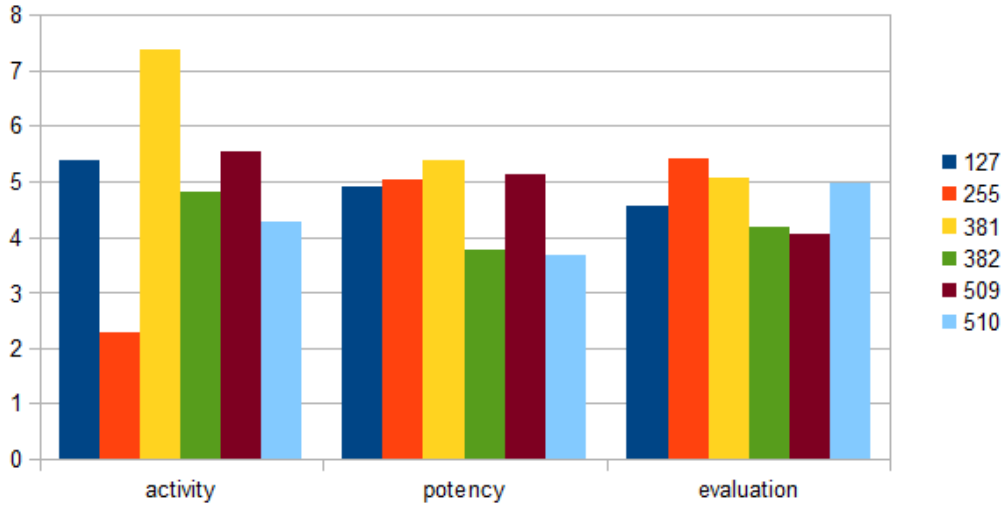

Abbildung 3.4: Ergebnisse basierend auf Helligkeit.

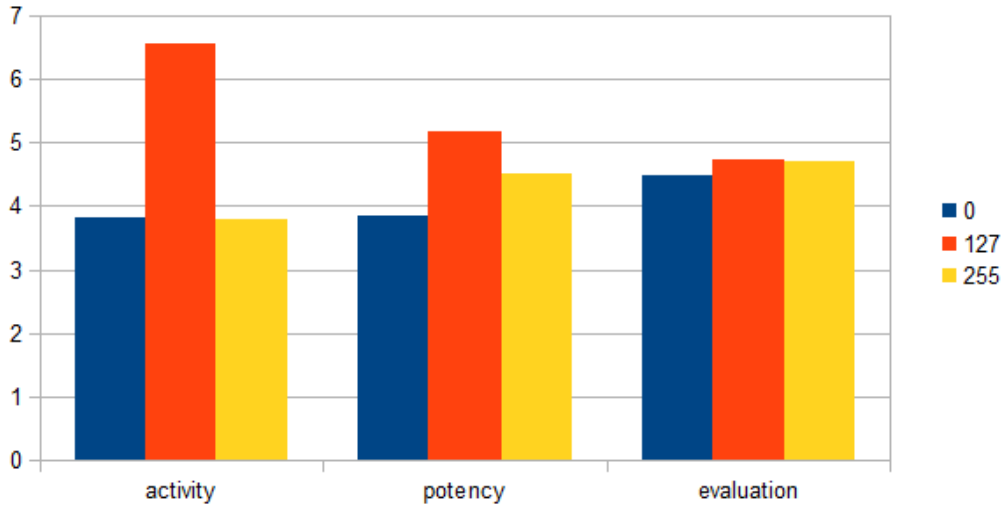

Abbildung 3.5: Eregebnisse basierend auf Rotanteil.

- **Aktivität:**
  - Objekte werden schnell bewegt und rotiert.

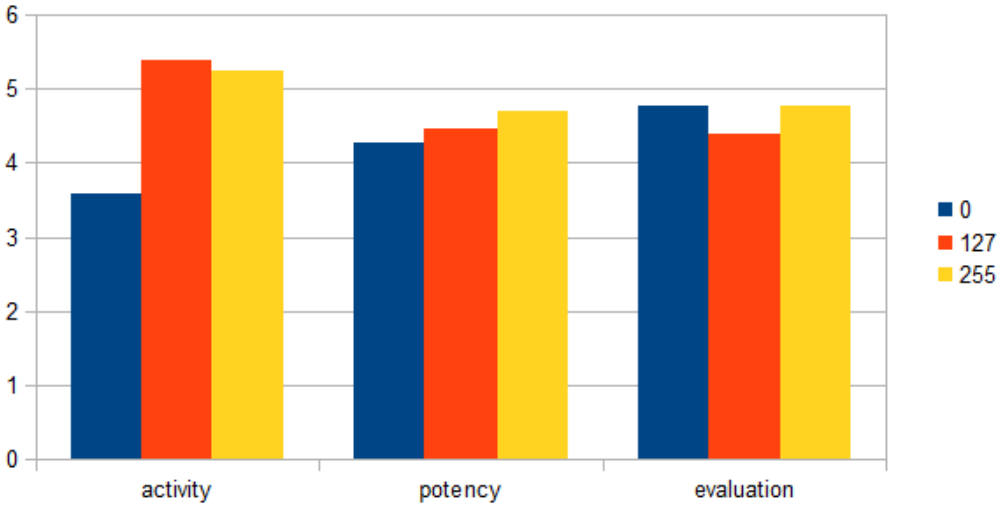

Abbildung 3.6: Ergebnisse basierend auf Grünanteil.

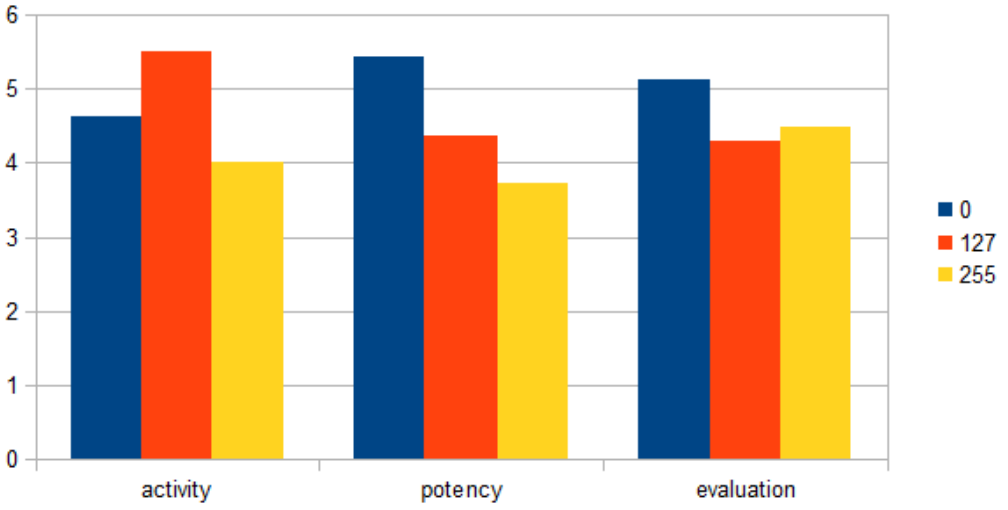

Abbildung 3.7: Ergebnisse basierend auf Blauanteil.

- **Kraft:**
  - Dargestellte Objekte sind größer
  - Formen bewegen sich nicht oder nur mit sehr hoher Geschwindigkeit

- **Bewertung:**
  - Objekte/Formen rotieren mit hoher Geschwindigkeit
  - Formen bewegen sich sehr langsam oder gar nicht

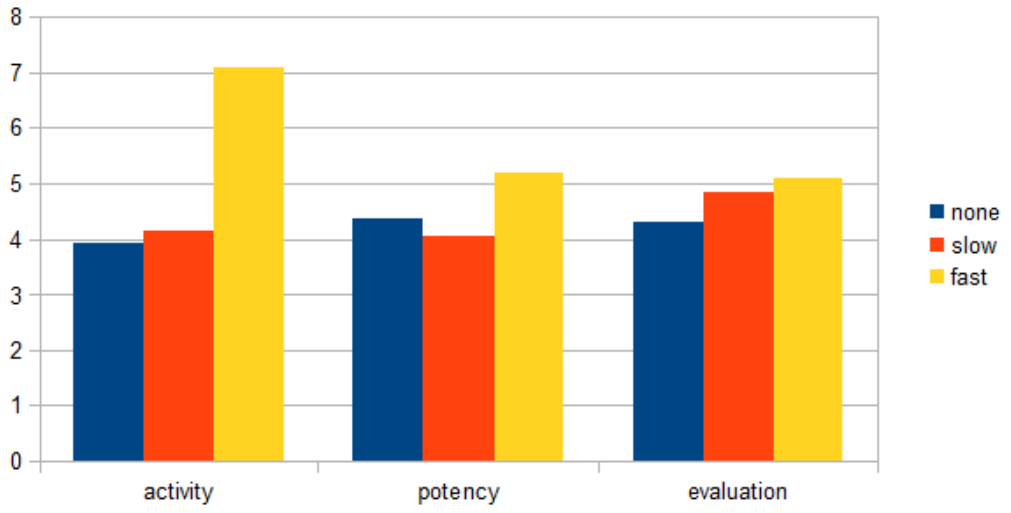

Abbildung 3.8: User study results based on the speed of rotation.

3.1.3 Zweite Benutzerstudie

In der zweiten Benutzerstudie spielten wir den Benutzern die oben genannten Lieder auf fünf Media Playern inklusive unserem Prototypen ab. Da unser Prototyp nur primitive Formen darstellt und wir verhindern wollten, dass die Benutzer nur die Effekte bewerten, wurden Sie gebeten einmal die Visualisierung zu bewerten und einmal die Übereinstimmung von Musik und Visualisierung. Wie im Kapitel 2 erwähnt wurden folgende Media Player verwendet:

- WinAmp
- Windows Media Player
- iTunes
- Prototyp mit Mapping aus der ersten Benutzerbefragung
- Prototyp mit dem Antimapping

Das Ergebnis dieser Benutzerstudie befindet sich in Kapitel 5 unter dem Punkt Erkenntnisse.

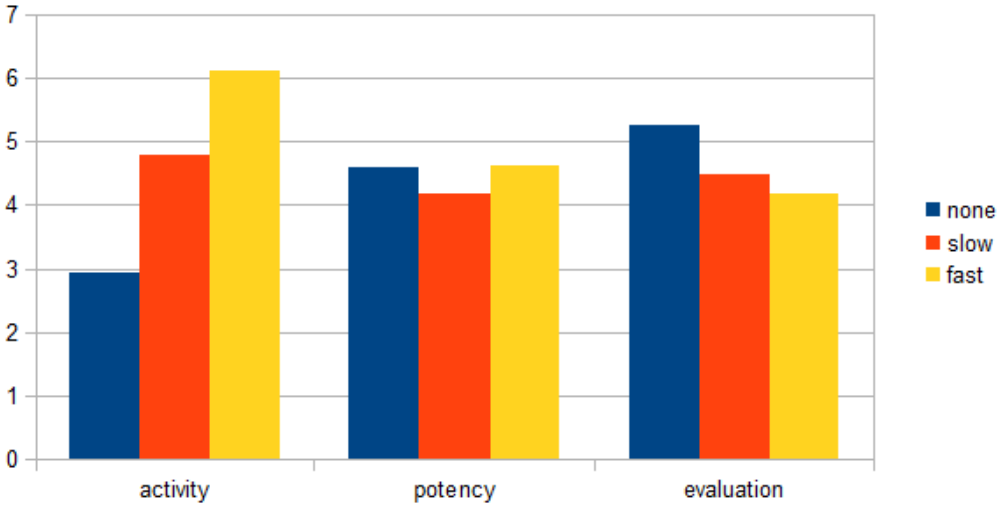

Abbildung 3.9: User study results based on the speed of translation.

3.2 Architektur

Nachfolgend finden Sie ein Klassendiagramm unseres Prototypen.

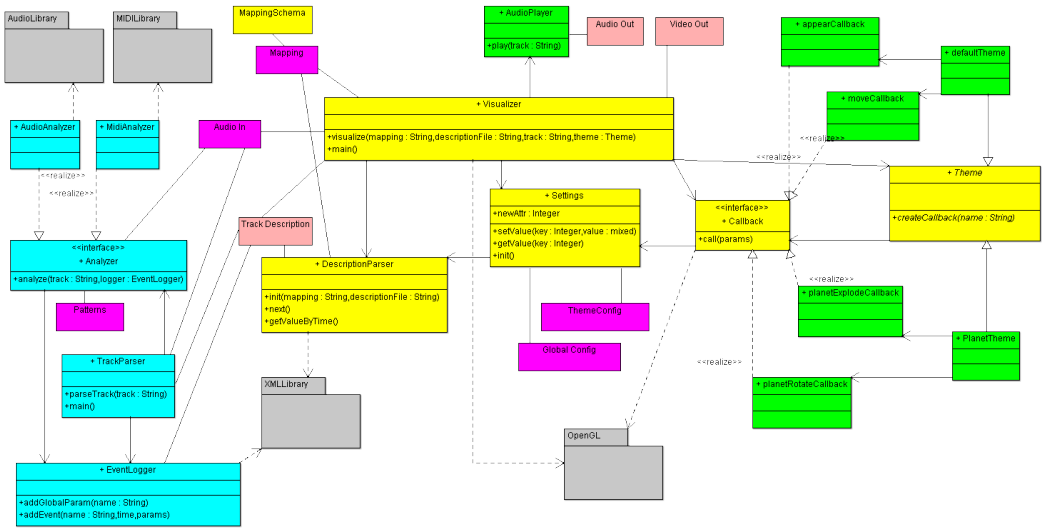

Abbildung 3.10: Klassendiagramm

3.2.1 Definition config.xml

In der Konfigurationsdatei werden folgende Werte gespeichert:

- **theme:** Beschreibt die gewünschte Ausgabedarstellung.
- **debugMode:** Beschreibt ob der Debugmodus verwendet werden soll. Der Debugmodus liefert, wenn aktiviert, gewünschte Debugmeldungen an die Konsole weiter.
- **msPerFrame:** Beschreibt wie viele Millisekunden pro Frame vergehen sollen. Diesen Wert benutzen wir um die Synchronisation zwischen Musik und Visualisierung aufrecht zu erhalten.

3.2.2 Definition Ausgabedarstellung

In unserem Projekt besteht die Möglichkeit eigene Ausgabedarstellungen zu entwickeln. Derzeit sind zwei Darstellungsmethoden definiert und einsatzbereit. Die erste stellt eine einfache Textuelle Ausgabe dar. Das heiśt wenn eine Form auf dem Bildschirm dargestellt werden sollte, erscheint in der Konsole der entsprechende Befehl. Die zweite Ausgabedarstellung bietet dem Benutzer verschiedene Formen, Farben und Transformationen an. Der Benutzer könnte somit in seiner Mappindatei beschreiben welche Noten, die Lautstärke und weitere Parameter auf welche Formen, Farben und Transformationen gemappt werden.

3.2.3 Definition Mapping.xml

In der Mappingdatei wird definiert wie sich die Visualisierung zur Musik verhalten soll. Das derzeitige Mapping basiert auf psychologischen Erkenntnissen wie Musik mit bestimmten Formen, Farben und Transformationen zusammenhängen.
Zusätzlich hat der Benutzer die Möglichkeit seine eigene Mappingdatei zu erstellen. In dieser hat er die Möglichkeit selbst zu definieren was in dem Musikstück wie Visualisiert werden soll. Zum Beispiel passt es dem Benutzer nicht, dass Dreiecke bei bestimmten Noten erzeugt werden und ändert es mit wenigen Handgriffen zu Linien um.
Folgende Werte werden in der Mappingdatei definiert:

- **event:** Beschreibt bei welchem Event welcher Command ausgeführt werden soll. Dieser besteht aus beliebig vielen <command> Tupeln.
- **Command name:** Beschreibt welcher Command mit welchen Parametern in der Visualisierung aufgerufen werden soll. Besteht aus beliebig vielen <param name> und <param value> Tupeln.
- **param name und value:** Der Name beschreibt welcher Parameter gemeint ist und Value übergibt den gewünschten Wert der Visualisierung.

Eine Mappingdatei besteht aus beliebig vielen <event> Tupeln.

3.2.4 Definition TrackDescription

Die TrackDescription beinhaltet alle ausgelesenen relevanten Information zum Musikstück und besteht aus beliebig vielen Tupeln der Form <tdGlobal> oder <tdEvent> welches wiederrum aus beliebig vielen Tupel der Form <tdEventParam> besteht. Folgende Werte werden gespeichert:

- **tdEvent**
  - **name:** Beschreibt ob der Typ eine Note oder eine Pause ist.
  - **time:** Zeitpunkt an dem die Note oder Pause stattfindet.
- **tdEventParam**
  - **paramName:** Kann einen der unten angegebenen Typen annehmen.
    - **key:** Beschreibt die Art der Note.
    - **volume:** Gibt die Lautstärke wieder.
    - **duration:** Die Länge der Note.
    - **frequency:** Die Frequenz.
    - **pan:** Gibt an, an welchem Lautsprecher man die Note hört.
    - **pitch:** Numerischer Wert der Note.
    - **instrument:** Ein Zahlenwert der die Instrumente identifiziert.
    - **channel:** Beschreibt über welchen Kanal das Signal ankommt.
  - **paramValue:** Speichert den Wert des Parameters.

3.2.5 Defintion Command

Ein Command beschreibt in unserem Projekt was zur Laufzeit zu einem bestimmten Zeitpunkt unternommen werden soll. Damit man es sich besser vorstellen kann was genau ein Command beschreibt, hier ein Beispiel: CreateCommand ist eine Methode die es dem Mapping erlaubt verschiedene Formen mit vordefinierten Parametern (Position, Gröše,. . .) zu erstellen. Die Commands werden dann zur Laufzeit mit einem Zeitwert in Listen gespeichert und in der Visualisierungsschleife pro Frame abgearbeitet.

3.2.6 Programmiersprache

Als Auswahl an Programmiersprachen standen uns C++ und Java zur Verfügung. Da meine Kollegen noch nicht mit C++ programmiert hatten und es auch Bibliotheken gibt die OpenGL in Java ermöglichen, entschieden wir uns für Java.

3.2.7 Verwendete Bibliotheken

Im folgenden Abschnitt möchte ich auf Bibliotheken eingehen, die wir getestet haben beziehungsweise dann verwendet haben.

Motivation zur Verwendung einer Bibliothek

Die Motivation eine Bibliothek zu benutzen besteht darin, das Rad nicht neu zu erfinden und sich darauf zu konzentrieren was man eigentlich entwickeln möchte. Weiters hat uns die Bibliothek viel Arbeit im grafischen Bereich abgenommen. Folgenden Anspruch hatten wir an die Bibliothek.

- Grafische Ausgabe mit OpenGL 3.3+
- gut dokumentiert, viele Tutorials
- Unterstütztung vieler Systeme
- Arbeitserleichterung (Helperklassen, Primitive,. . .)

JOGL - Java Binding for the OpenGL api

Jogl ist eine Bibliothek die es erlaubt OpenGL in Java zu benutzen. Da für dieses Framework nur wenige aktuelle Tutorials (OpenGL 3.3+) vorhanden sind und nach ausführlichem Testen die Erstellung von Primitiven schwierig war, haben wir uns gegen diese Bibliothek entschieden.

Lwjgl - Lightweight Java Game Library

Dieses Framework erlaubt es auch OpenGL in Java zu benutzen, aber sie stellt auch nur ein paar nützliche Helperklasse zur Verfügung. Für diese Bibliothek gibt es aber auch nur wenige aktuelle Tutorials (OpenGL 3.3+) und ein weiteres Problem bestand darin, dass die grafische Ausgabe nicht auf den Notebooks meiner Kollegen funktioniert hat. Somit haben wir uns auch gegen dieses Framework entschieden.

Libgdx

Libgdx übernimmt mittels Helperklassen (primitive, Mathematik, Vektoren,) einen großen Teil der Arbeit und man findet sehr viele gute Tutorials und Bespiele die einem helfen sich in dieses Framework zu Recht zu finden. Ein weiterer Vorteil ist, dass dieses Framework auf weit mehr Rechnersystemen Unterstützung findet als JOGL und Lwjgl und es besteht auch die Möglichkeit die erstellten Applikation auf Linux, Mac OS, Android usw. portieren.

Entscheidung

Am Ende haben wir uns für die Bibliothek Libgdx entschieden, da sie die am einfachsten zu benutzende Bibliothek war, viele Tutorials angeboten werden und auf vielen Systemen Unterstützung findet.

KAPITEL 4

Implementierung

4.1 Visualisierungsreihenfolge

In diesem Kapitel möchte ich Visualisierungsreihenfolge erläutern.

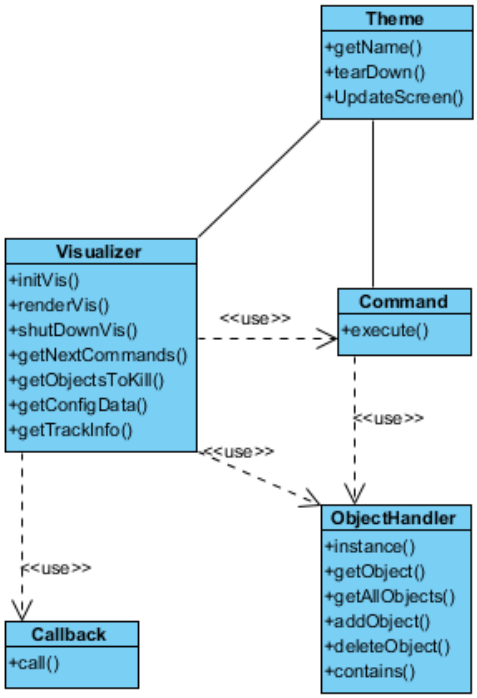

Abbildung 4.1: Ausschnitt aus dem Klassendiagramm

4.1.1 Bei der Erstellung des Visualizers werden folgende Schritte durchgeführt

Die config.xml-Datei wird eingelesen und abgearbeitet. Das gewählte Ausgabedarstellung wird aus der config.xml-Datei gelesen, die dazu passende Klasse geladen und initialisiert. In der Initialisierungsmethode der Mapping-Datei werden die TrackDescription und die Mapping.xml-Datei eingelesen. Es werden aus der TrackDescription-Datei die einzelnen Informationen ausgelesen und mit dem dazu gemappten Befehl ein Command erstellt.

4.1.2 Initialisierung des Visualizers

Bei der Initialisierung des Visualizers werden folgende, vom Programm benötigte Variablen gesetzt. Die Variablen liefern die TrackDescription-Datei und die config.xml-Datei. Gebraucht werden folgende Werte: Die Tracklänge in Millisekunden. (von TrackDescription) Das Tempo des gewählten Musikstückes in Millisekunden. (von TrackDescription) Die Millisekunden pro Frameeinheit. (von config.xml)

4.1.3 Nach der Initialisierung des Visualizers startet das Rendern der einzelnen Frames

In der Funktion renderVis() läuft eine Schleife solange ein Framezähler kleiner als die Länge des Musikstückes ist. In jedem Schritt holt sich die Schleife mit der Methode getNextCommands() die nächsten Commands die vom letzten Schritt bis zum jetzigen Schritt erzeugt worden sind. Als nächstes werden diese Commands nach deren zeitlichen Eintrag ausgeführt.

4.1.4 Handhabung der Objekte (Primitive)

Ein ObjectHandler wird am Anfang des Programms erstellt, welcher sich um die Objekte im Programm kümmert. Folgende Befehle stellt er zur Verfügung:

- **addObject:** Ein Objekt der Liste hinzufügen.
- **getObject:** Liefert das gewünschte Objekt.
- **deleteObjekt:** Löscht das Objekt wird der übergebenen ID.

4.1.5 Ausführen eines Kommandos

Die Funktion execute() von einem Command holt sich zuerst alle benötigten Parameter von der Mapping.xml-Datei. Beispiele für solche Parameter wären: die Grösse in X-Richtung oder der Winkel. Im nächsten Schritt wird dann entweder

- ein Primitive erstellt, (CreateCommand)
- ein Primitive aus der Liste gesucht und gelöscht (DeleteCommand) oder
- ein Primitive aus der Liste gesucht und der gewünschte Befehl ausgeführt.(move, scale, rotate,...)

4.1.6 Darstellen des Bildes

Nachdem die Commands für diesen Schritt abgearbeitet worden sind, wird das ganze Bild dargestellt. Dazu ruft die Methode renderVis() die Funktion updateScreen() in der erstellten Ausgabedarstellung auf, welche wiederum die Renderfunktion renderCycle() der jeweiligen Primitive aufruft.

4.1.7 Beenden des Visualizers:

Wenn das Musikstück zu Ende ist und alle Commands abgearbeitet worden sind wird die Methode tearDown() im Ausgabedarstellung aufgerufen. Diese bewirkt, dass alle Objekte aus dem ObjectHandler geholt, gelöscht und somit der Speicherbereich wieder freigegeben werden.

4.1.8 Commands die dem Visualizer zur Verfügung stehen

CreateCommand()

Dieser Command erstellt mit den unten angeführten Parametern ein neues Primitiv.

- **setup(sizeX, sizeY, posX, posY, r, g, b, a, angle, shape, renderType)**
- **sizeX, sizeY:** Die Skalierung in X,Y-Richtung posX, posY: Die X,Y-Postion des Primitives.
- **r, g, b, a:** Die Farbe im RGBA-Raum.
- **angle:** Der Winkel des Primitives in dem es starten soll.
- **shape:** Folgende Primitive können erstellt werden: Kreis, Punkt, Linie, Rechteck, Dreieck.
- **renderType:** Mit rendertype kann man einstellen ob das Primitive ausgefüllt gezeichnet werden soll oder nur die Linien sichtbar sein sollen.

DeleteCommand

Dieser Command löscht das Primitive aus dem ObjectHandler mit der gelieferten ID.

- **destroy():** Methode die das Objekt im Speicher wieder frei gibt.

MoveCommand

MoveCommand bewegt das Primitiv an die mit Parametern übergebene Position.

- **move(posEndX, posEndY, duration)**
- **posEndX, posEndY:** Die Position an der das Primitiv verschoben werden soll.
- **duration:** Die Zeit die das Objekt hat an die endgültige Position verschoben zu werden.

RotateCommand

RotateCommand rotiert das Objekt gegen den Uhrzeiger um den angegebenen Winkel.

- **rotate(angle,duration) angle:** Aktueller Winkel plus angle ist der Endwinkel in Grad.
- **duration:** Die Zeit die das Objekt hat sich an den endgültigen Winkel zu drehen.

ScaleCommand, ScaleXCommand, ScaleYCommand

Die ScaleCommands verzerren das Obkjekt an der gewünschten Achse.

- **scale(endSize(/X/Y), duration)**
- **endSize /X/Y:** Endgrösse der Achse/Achsen.
- **duration:** Die Zeit die das Primitiv hat an die endSize-Grösse zu skalieren.

SetColorCommand

Dieser Command setzt die übergebene Farbe im nächsten Frame.

- **setColor(r, g, b, a)**
- **r,g,b,a:** Die Werte im RGBA-Farbraum.

InterpolateColorCommand

Dieser Command interpoliert die Farbe von der aktuellen Farbe zu der gewünschten Farbe.

- **interpolateColor(endR, endG, endB, endA, duration)**
- **endR,-G,-B,-A:** Die Farbe die das Objekt nach der Interpolation haben soll.
- **duration:** Die Zeit die das interpolieren der Farbe dauern darf.

KAPITEL 5

Zusammenfassung und zukünftige Arbeit

5.1 Zusammenfassung

Die kompletten Ergebnisse der zweiten Benutzerstudie befinden sich in der Bachelorarbeit von Jürgen Giefing im sechsten Kapitel. [1]

5.1.1 Erkenntnisse

In der Grafik 5.1 kann man erkennen, dass unser Mapping das schlechteste Ergebnis liefert. Zusammenfassend kann gesagt werden, dass das gelieferte Ergebnis nicht unseren Vorstellung entspricht. Jetzt stellt sich natürlich die Frage woran das liegen könnte.
Die naheliegendste Theorie wäre, dass die These welche behauptet hat, dass man Musik auf Farben, Formen und Transformationen projizieren falsch ist. Wir gehen aber von folgenden Problemen beziehungsweise falsche Annahmen aus:

- Die Visualisierungen der anderen Media Player beinhalteten zu viele Effekte und behinderten die Testpersonen daran objektiv an die Bewertung heran zu gehen.

- Ein weiteres Problem könnte sein, dass wir nur einzelne Parameter des Musikstückes auf Formen, Farben und Transformationen gemappt haben. Die Lösung dafür wäre, dass man sich auch die Kombination von Musikparametern anschaut und somit das Mapping komplexer gestaltet.

- Die derzeitige für diesen Prototypen erstellte Ausgabedarstellung könnte auch ein Problem gewesen sein. Man könnte versuchen, aufwendigere und komplexere Commands einzufügen, damit ein komplexeres Mapping mehr Visualisierungsmöglichkeiten besitzt.

Zum Abschluss möchten wir noch erwähnen, dass wir weiterhin davon ausgehen, dass man Musik auf Visualisierung projizieren kann und dass die Musik mit der Visualisierung und dem menschlichen Empfinden zusammenhängt.

5.1.2 Tabellarisches Ergebnis der Benutzerstudie

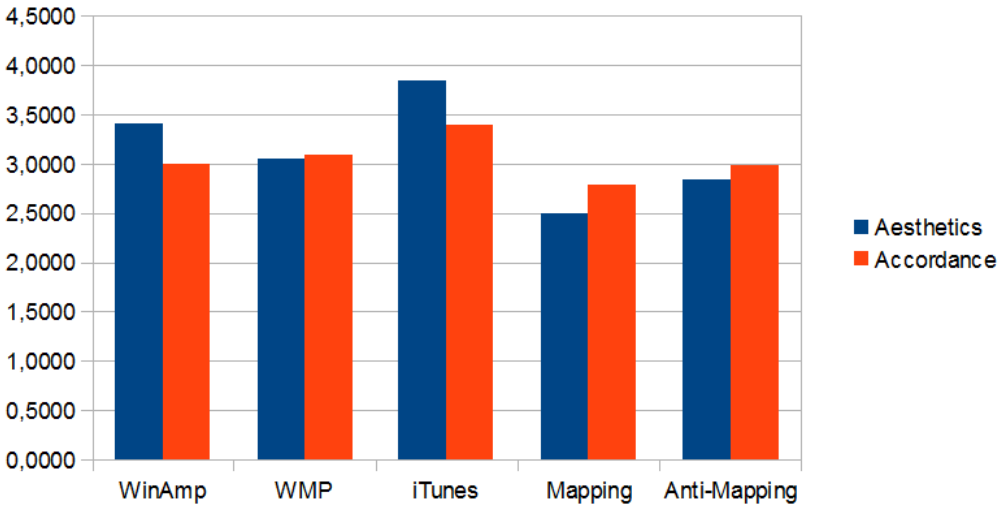

Abbildung 5.1: Das Ergebnis der zweiten Benutzerstudie als Diagramm inklusive einer Bewertung von 1 (schlecht) bis 5 (gut)

5.2 Probleme

Eines unser größten Probleme in dieser Arbeit bestand darin eine passende Bibliothek in Java und OpenGL zu finden. Die ersten beiden gefundenen Bibliotheken hatten entweder keine ausreichende Dokumentation für OpenGL 3.3+ oder sie haben schlichtweg auf einigen Systemen zu Problemen geführt.

5.3 Zukünftige Arbeit

Hier möchte ich nur einige Beispiele nennen, die Ideen für zukünftige Arbeiten an der Visualisierung behilflich sein könnten.

5.3.1 Texturen

Texturen könnte man benutzen um die Visualisierung zu verbessern oder die Darstellung realistischer zu gestalten.

5.3.2 Automatisierte Generierung von Primitiven durch Musik

Eine sinnvolle Möglichkeit das Programm zu erweitern wäre, keine vorgefertigten Primitive (Kreis, Dreieck,...) zu verwenden sondern das Programm inklusive dem Mapping entscheiden zu lassen welche Primitive erstellt werden soll. Ein friedlicher langsamer Musikstil bringt Primitive mit einer geringen Anzahl von Punkten hervor wo hingegen ein lauter Stil welche mit vielen Eckpunkten erstellt.

5.3.3 Fraktale

Eine weitere Möglichkeit die Arbeit fortzusetzen wäre Fraktale einzubauen. Die Idee dahinter wäre, das Fraktal zum Beispiel eine Blume bei langsamer Musik langsam wachsen zu lassen oder bei einem aggressiven Musikstil die Blüten verwelken zu lassen.

KAPITEL 6

Appendix

6.1 Download des Prototypen

Der Prototyp und der dazugehörige Sourcecode kann unter `https://bitbucket.org/juergen.giefing/physics-based-music-visualization` heruntergeladen werden.

6.2 Beispiel mapping.xml

```
<mapping>
 <pattern name="_global">
  <event name="note">
   <command name="+track.mode|*track.instruments|*0.05|+2:CreationCommand">
    <param name="sizeX" paramValue="VALUE" />
    <param name="sizeY" paramValue="VALUE">
    <param name="r" paramValue="*random" />
    <param name="g" paramValue="VALUE" />
    <param name="b" paramValue="VALUE" />
    <param name="a" paramValue="*0|+track.mode|-track.tempo|*0.01|+2" />
    <param name="x0" paramValue="+1|*random|-1" />
    <param name="y0" paramValue="+1|*random|-1" />
    <param name="angle" paramValue="+179|*random" />
    <param name="shape" paramValue="CbShape" />
    <param name="renderType" paramValue="+1" />
    <param name="id" paramValue="*0" />
   </command>
   <command name="MoveCommand">
    <param name="posEndX" paramValue="+2|*random|-1.5" />
    <param name="posEndY" paramValue="+2|*random|-1.5" />
```

```
    <param name="duration" paramValue="+1" />
    <param name="id" paramValue="lastId" />
   </command>
   <command name="RotateCommand">
    <param name="angle" paramValue="*0|+track.mode|*360" />
    <param name="duration"VALUE" />
    <param name="id" paramValue="lastId" />
   </command>
  </event>
</pattern>
</mapping>
```

6.3 Beispiel config.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<config>
    <configEntry configKey="theme" configValue="main" />
    <configEntry configKey="debugMode" configValue="0" />
    <configEntry configKey="msPerFrame" configValue="40" />
</config>
```

6.4 Benutzerstudien

Die Benutzerstudien wurden mit Alexander Hauer und Jürgen Giefing zusammen ausgearbeitet und befinden sich in ähnlicher Form in deren Bachelorarbeiten. Die Tabellen stammen aus der Bachelorarbeit von Jürgen Giefing und werden daher in Englisch angezeigt. [1] [3]

6.4.1 Testpersonen

Die folgenden zwei Tabellen zeigen wie die Testpersonen ihr Verständnis gegenüber Musik (erste Tabelle) und ihr Verständnis gegenüber Grafik beziehungsweise Visualisierungen (zweite Tabelle) bewertet haben. Die Bewertung reichte von 1 (kein Verständnis) bis 5 (volles Verständnis).

affinity	count
1	0
2	1
3	4
4	8
5	6

Tabelle 6.1: Self assessment of the test persons concerning their affinity to music

affinity	count
1	0
2	4
3	8
4	4
5	3

Tabelle 6.2: Self assessment of the test persons concerning their affinity to visual elements

sex	count
female	9
male	10

Tabelle 6.3: Composition of the sample of test persons based on sex

age	count
20-29	5
30-39	11
40-49	1
50-59	2

Tabelle 6.4: Composition of the sample of test persons based on age

Title	Artist	Mode	Tempo	Instruments
Cryin	Aerosmith	major	102	9
Crying at the discotheque	Alcazar	minor	136	7
Eleanor Rigby	The Beatles	minor	130	5
Everyday People	Arrested Development	major	91	5
Pick up the pieces	Average White Band	major	107	6
Rain	Annie Lennox	minor	126	6
Respect	Aretha Franklin	major	116	7
School's out	Alice Cooper	major	132	5
Stayin Alive	Bee Gees	minor	104	8
TNT	AC DC	major	130	6

Tabelle 6.5: Songs used in our user studies

	red	green	blue	translation	rotation	shape	size	number
1	0	127	255	slow	none	dot	undefined	>5
2	255	0	127	fast	none	dot	undefined	2-5
3	255	0	255	none	slow	line	small	>5
4	127	255	0	slow	fast	line	big	2-5
5	127	127	255	fast	none	circle	small	>5
6	255	0	0	none	none	circle	big	2-5
7	0	127	0	slow	slow	triangle	big	1
8	127	127	127	fast	fast	triangle	small	>5
9	0	0	255	none	slow	rectangle	small	2-5
10	0	255	127	slow	none	rectangle	big	1

Tabelle 6.6: Animations used in our user studies

axis	minimum	maximum
activity	1 (not active)	10 (very active)
potency	1 (not potent)	10 (very potent)
evaluation	1 (very negative)	10 (very positive)

Tabelle 6.7: Scales used in our user studies

shape	axis	min	avg	max
dot	activity	1	4.3684	9
dot	potency	1	2.8947	7
dot	evaluation	1	3.6316	7
line	activity	1	5.5263	9
line	potency	1	4.3421	9
line	evaluation	1	5.0263	10
circle	activity	1	3.6053	10
circle	potency	1	5.7632	10
circle	evaluation	1	4.8947	10
triangle	activity	3	6.3684	9
triangle	potency	2	5.1316	9
triangle	evaluation	1	4.7895	9
rectangle	activity	1	3.2895	8
rectangle	potency	1	4.0000	9
rectangle	evaluation	1	4.7368	8

Tabelle 6.8: Ratings based on the shape used in the animation

brightness	axis	min	avg	max
127	activity	3	5,3684	9
127	potency	2	4,8947	8
127	evaluation	2	4,5263	8
255	activity	1	2,2895	8
255	potency	1	5,0263	10
255	evaluation	1	5,3947	10
381	activity	5	7,3684	9
381	potency	2	5,3684	9
381	evaluation	1	5,0526	9
382	activity	1	4,8026	9
382	potency	1	3,7895	9
382	evaluation	1	4,1974	9
509	activity	1	5,5263	10
509	potency	1	5,1053	9
509	evaluation	1	4,0526	7
510	activity	1	4,2632	7
510	potency	1	3,6842	8
510	evaluation	1	4,9474	10

Tabelle 6.9: Ratings based on the brightness of the color used in the animation

blue	axis	min	avg	max
0	activity	1	4,6140	9
0	potency	2	5,4386	10
0	evaluation	1	5,1228	10
127	activity	2	5,4912	9
127	potency	1	4,3684	9
127	evaluation	1	4,2807	9
255	activity	1	4,0000	10
255	potency	1	3,7105	9
255	evaluation	1	4,4868	10

Tabelle 6.10: Ratings based on the proportion of blue used in the animation

green	axis	min	avg	max
0	activity	1	3,5658	9
0	potency	1	4,2763	10
0	evaluation	1	4,7763	10
127	activity	1	5,3947	10
127	potency	1	4,4474	9
127	evaluation	1	4,3816	9
255	activity	2	5,2368	9
255	potency	1	4,6842	9
255	evaluation	1	4,7632	9

Tabelle 6.11: Ratings based on the proportion of green used in the animation

red	axis	min	avg	max
0	activity	1	3,8158	9
0	potency	1	3,8289	9
0	evaluation	1	4,4737	8
127	activity	1	6,5614	10
127	potency	1	5,1579	9
127	evaluation	1	4,7368	9
255	activity	1	3,7895	9
255	potency	1	4,4912	10
255	evaluation	1	4,6842	10

Tabelle 6.12: Ratings based on the proportion of red used in the animation

number	axis	min	avg	max
1	activity	2	4,5263	9
1	potency	1	4,6316	9
1	evaluation	1	4,4737	8
2 - 5	activity	1	4,1974	9
2 - 5	potency	1	4,6053	10
2 - 5	evaluation	1	4,8158	10
> 5	activity	1	5,1184	10
> 5	potency	1	4,1447	9
> 5	evaluation	1	4,4868	10

Tabelle 6.13: Ratings based on the number of objects used in the animation

size	axis	min	avg	max
undefined	activity	1	4,3684	9
undefined	potency	1	2,8947	7
undefined	evaluation	1	3,6316	7
small	activity	1	5,0132	10
small	potency	1	4,4474	9
small	evaluation	1	4,7763	10
big	activity	1	4,3816	9
big	potency	1	5,1711	10
big	evaluation	1	4,9474	10

Tabelle 6.14: Ratings based on the size of objects used in the animation

speed	axis	min	avg	max
none	activity	1	3,9263	10
none	potency	1	4,3368	10
none	evaluation	1	4,2947	10
slow	activity	1	4,1754	9
slow	potency	1	4,0702	8
slow	evaluation	1	4,8421	10
fast	activity	4	7,0789	9
fast	potency	2	5,1842	9
fast	evaluation	1	5,0789	9

Tabelle 6.15: Ratings based on the rotation of objects used in the animation

speed	axis	min	avg	max
none	activity	1	2,9474	8
none	potency	1	4,5789	10
none	evaluation	1	5,2456	10
slow	activity	1	4,7895	9
slow	potency	1	4,1711	9
slow	evaluation	1	4,4868	9
fast	activity	1	6,1053	10
fast	potency	1	4,6140	9
fast	evaluation	1	4,1579	9

Tabelle 6.16: Ratings based on the translation of objects used in the animation

tempo	axis	min	avg	max
activity	2	4.8772	10	
potency	1	5.1053	10	
evaluation	2	5.5789	10	
activity	2	5.8421	10	
potency	1	5.5263	10	
evaluation	2	5.6667	10	
activity	3	6.2895	10	
potency	3	5.2105	9	
evaluation	3	6.2368	9	
activity	4	6.5263	9	
potency	2	4.8947	8	
evaluation	3	6.7368	9	
activity	2	4.8947	10	
potency	2	4.4737	10	
evaluation	2	6.3684	10	

Tabelle 6.17: Ratings based on the number of instruments used

tempo	axis	min	avg	max
1	1	2	5.3596	10
1	2	1	5.0526	10
1	3	2	5.8684	10
2	1	2	6.0000	10
2	2	2	5.3421	9
2	3	3	6.0263	10

Tabelle 6.18: Ratings based on the mode

tempo	axis	min	avg	max
91	activity	2	4.0000	10
91	potency	1	4.2632	10
91	evaluation	2	5.2105	10
102	activity	2	4.8947	10
102	potency	2	4.4737	10
102	evaluation	2	6.3684	10
104	activity	4	6.5263	9
104	potency	2	4.8947	8
104	evaluation	3	6.7368	9
107	activity	5	6.8947	9
107	potency	1	5.0526	9
107	evaluation	2	6.1579	9
116	activity	3	5.7895	9
116	potency	3	5.1053	9
116	evaluation	4	6.1053	9
126	activity	3	5.7368	8
126	potency	2	5.4211	8
126	evaluation	3	5.2105	9
130	activity	2	4.9211	10
130	potency	1	5.9211	10
130	evaluation	2	5.7105	10
132	activity	3	5.6842	8
132	potency	2	5.3158	8
132	evaluation	3	5.7368	8
136	activity	3	6.7895	10
136	potency	3	5.3158	9
136	evaluation	3	6.3684	9

Tabelle 6.19: Ratings based on the tempo

Literaturverzeichnis

[1] Juergen Giefing. Physics based music visualization. 2013, Österreich, Tu Wien.

[2] Gregor Lawatscheck. http://de.wikipedia.org/wiki/winamp. Last Accessed: 2013-12-19.

[3] Alexander Hauer. Physics based music visualization. 2013, Österreich, Tu Wien.

[4] Ines Weimer. http://computergrafik.informatiker-wissen.de/transformationen.html. Last Accessed: 2013-12-19.

[5] P. Schlaich J. Bhringer, P. Bhler. *Kompendium der Mediengestaltung*. springer, 5. edition, 2011 (in German).

[6] Kincaid, Bill. http://en.wikipedia.org/wiki/itunes. Last Accessed: 2013-12-19.

[7] Marc Liron. http://www.updatexp.com/windowsxpmediaplayer.html. Last Accessed: 2013-12-19.

[8] Claus Chr. Liebmann Norbert Welsch. *Farben - Natur, technik, Kunst*. Spektrum, 3. edition, 2012 (in German).

[9] Stephen E Palmer, Karen B Schloss, Zoe Xu, and Lilia R Prado-León. Music–color associations are mediated by emotion. *Proceedings of the National Academy of Sciences*, 110(22):8836–8841, 2013.