RECONFIGURABLE UNIT VISUALIZATION FOR EXPLORING MUSIC THEORY

A FLEXIBLE TOOL FOR READING, ANALYZING, AND RECOMPOSING MUSIC

CHRISTIAN NORDSTRØM RASMUSSEN, 201304213

MASTER'S THESIS

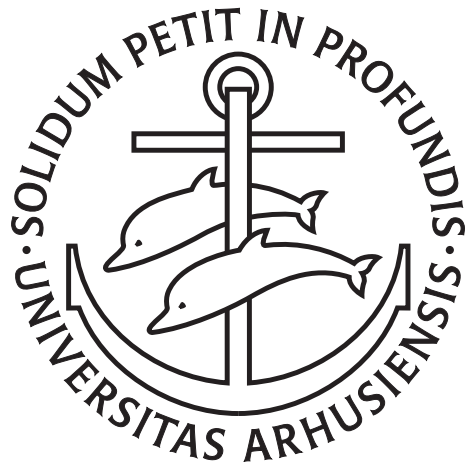
June 2021

Advisor: Hans-Jörg Schulz

AARHUS
UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

RECONFIGURABLE UNIT VISUALIZATION FOR EXPLORING MUSIC THEORY

CHRISTIAN NORDSTRØM RASMUSSEN



A flexible tool for reading, analyzing, and recomposing music

Master's Thesis
Department of Computer Science
Science & Technology
Aarhus University

June 2021

Music is like cooking for me: you mix the ingredients together in one big pan and see how they end up. Through experimenting, you find what you really like and stick with it.

— Jacob Collier

ABSTRACT

This thesis examines how to make the complex body of music theory more accessible through reconfigurable unit visualization. The thesis first conducts a survey of 32 works within music theory visualizations and categorizes them by their supported interaction intents and topic within music theory. From this categorization, several underexplored research topics are identified, mainly the interaction intent of reconfiguration. It is also identified that none of the previous works have used the technique of unit visualization in their solutions. Based on these findings, a conceptual framework for a reconfigurable unit visualization is proposed. This framework is based on a use case grounded in Bloom's Revised Taxonomy for Music Education and on the concept of fundamental smoothing operations within exploratory data analysis: specifically conversions, constraints, chunks, computation, and comparison. The framework is partly implemented in a prototype visualization based on D3.js. The visualization is evaluated in a walkthrough demonstration and semi-structured interview with a conservatory-educated musician. Then, the utility of the visualization is illustrated with specific findings in the musical data, supported with audio examples created in FL Studio. Through these activities, the thesis demonstrates a range of valuable utilities of reconfigurable unit visualizations for accessible music theory. It also highlights the potential for further research on this topic. Further work should focus on developing even more reconfigurations to reveal even deeper insights into music theory and create even more support for the workflows of musicians.

CONTENTS

LIST OF FIGURES

ACRONYMS

MTV Music Theory Visualization

MT Music Theory

MN Music Notation

DAW Digital Audio Workstation

ESDA Exploratory Sequential Data Analysis

1 INTRODUCTION

Music Theory (MT) is a complex subject that serves to provide an understanding of musical principles through analysis, and the best musicians have internalized these principles to a degree where it becomes intuition. Through time, many notation styles, models, and analytic techniques have been proposed to aid this goal. This poses an important question: How is such a complex body of theory synthesized into an accessible and versatile learning tool? This thesis will address this question by proposing a reconfigurable unit visualization, in which the users can seamlessly connect related concepts through exploration and animated interactions. To do this, the thesis will first survey the current field of Music Theory Visualization (MTV) to identify underexplored interaction concepts. A conceptual framework for a reconfigurable unit visualization will be presented, with a proof-of-concept implementation to demonstrate its usefulness. Lastly, the prototype will be evaluated with a conservatory educated musician, which will provide a well-informed basis for discussing its validity.

A common approach to acquiring musical knowledge is through analysis and re-interpretation of musical scores - i.e., sheets containing musical instructions. To this end, traditional Music Notation (MN) [70] is the most widely accepted representation in western musical culture. A strength of MN is that it affords the reader to learn the basic building blocks of music. It can, however, appear arcane at first glance, and the initial mental investment of memorizing the symbolic representations of rhythm, melody, and harmony will discourage many novice musicians to the degree where they give up [22]. At least two other factors add to the challenges of teaching MN: first, other notation styles, such as piano roll, becifrings [79], or tablature [80], can seem more intuitive to beginners who want to learn an instrument straight away. However, these 'easier' notations can also be discouraging later on, as they will increase the gap between the practical and theoretical skills of the student. Second, information, such as harmonic relations, are not explicitly represented in MN, essentially making harmonic analysis [65] and many other tasks their own separate skills. Much research has gone into creating notations that balance the learning curve with the complexity of information. However, the problem may be less about providing all the data up-front and more about providing the right data at the right time.

How data visualization approach this problem can be understood through the definition by McCormick et al.: *"[data visualization] transforms the symbolic into the geometric, [...] Visualization offers a method for seeing the unseen. It enriches the process of scientific discovery and fosters profound and unexpected insights."* [6]. This definition entails that a good visualization of MT should convey the data relations of interest effectively through geometric representations, thus easing the mental load of learning, analyzing, and re-interpreting music for the user. Data visualization typically supports the process of scientific discovery by allowing exploration through a pattern of interaction that adheres to Shneiderman's mantra [44]: *"overview first, zoom and filter, then details on demand."* This interactivity has several advantages to current, static notation styles. Especially, providing details-on-demand is necessary to connect the multifaceted principles of MT, as this empirically can not be achieved by any single representation.

This thesis will use the approach of unit visualization [34] to create a MTV that binds together theoretic principles by allowing reconfiguration of musical information. The aim is to create a visualization of modular components that can be reconfigured to support a wide range of tasks, such as reading, analyzing, reharmonizing and understanding theory.

This section has introduced the research problem of the thesis and an overview of the topics to be explored. The following section will provide an abstraction of musical data.

1.1 AN ABSTRACTION OF MUSICAL DATA

This section will characterize musical data using the terminology of data context and data content used by Schulz et al. in [43]. The characterization aims to encapsulate all basic musical terms and concepts used in this thesis in order to bridge the knowledge for readers with backgrounds in both data visualization and music.

1.1.1 *Data context*

Tempo is the word used for the pace of music and is usually measured in beat units per minute, where the beat unit is often partitioned in quarter notes [76]. Tempo is an independent variable, usually global in extent, but it is not unheard of that a piece of music can vary in tempo when transitioning between parts.

Beats are discrete accentuated pulses that make up the basic time units of a song (figure: 1)[59]. They can be seen as chunks of time, local in extent. Few pieces are best described as having no beat/pulse

if they are perceived as a continuous flow of sound, but these are outliers, as pulse is central to musicality.

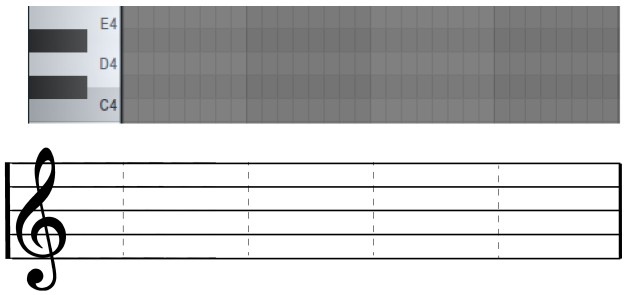


Figure 1: Four beats shown in respectively piano roll and music notation (with added dotted lines). Piano roll notation usually has four beats in a measure. In music notation, the number of beats in a measure depends on the time signature.

Bars or measures are units that contain a set amount of beats defined by the time signature [58]. Thus, they are discrete and local sections in a piece, useful for providing temporal structure to a piece.

Time signature defines the number of beats in each measure, which will also affect the meter of a piece (figure: 1) [77]. In a 3/4 time signature, the numerator shows the number of beats in a measure, while the denominator shows the beat units - in this example, a measure contains three quarter notes. Time signature can be global and local. The value of a time signature is a fraction with discrete values as its numerator and denominator.

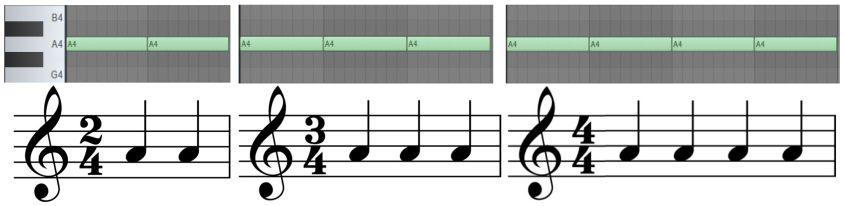


Figure 2: Three measures containing respectively two, three and four quarter notes

The **meter** is the underlying rhythm of a piece [69]. This rhythm is made up of a beat pattern in which specific beats are emphasized more than others. In 2/4 time, the first beat is strong, and the second beat is weak. In 3/4 time, the first beat is strong, while the second and third beat is weak. 4/4 time has a strong first beat, weak second, medium third, and weak fourth beat (figure: 3). The meter is, as the time signature, most often global in extent.

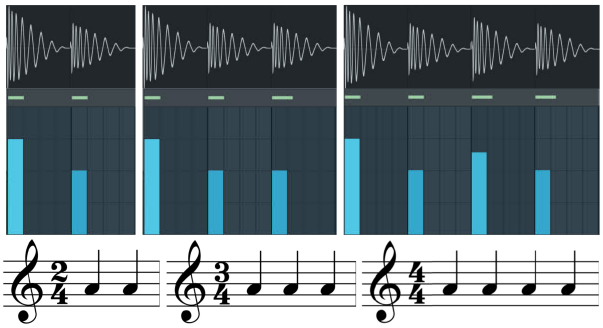


Figure 3: Three varying meters shown as signal (top), velocity (middle), and music notation (bottom). The time signature determines the force of each note - e.g., 2/2 time has a strong note, then a weak.

1.1.2 *Data content*

Pitch is the vibration frequency of a note (figure: 4)[73]. It is a continuous scalar representing a frequency within the range of the audible spectrum (20 - 20.000 Hz). The amplitude of the oscillations (the dynamic) determines the loudness and is as continuous scalar values (e.g., 20 dB).

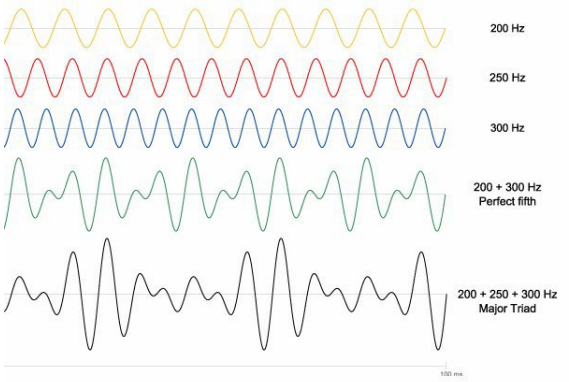


Figure 4: A frequency of 200Hz is approximately equal to a G3 pitch, while 250Hz ≈ B3 and 300Hz ≈ D3. Together, they create a more complex pattern, that will be perceived as a triadic chord - In this case G major. Image source: [56].

Each 'valid' pitch in western music is assigned a **pitch class** within 7 ordinal categories ranging: [A, B, C, D, E, F, G][72]. Pitch classes repeat in octaves, meaning pitch intervals corresponding to a doubling in pitch, as this is perceived to produce the same tone quality. E.g., the pitch of is 440 Hz is A in the fourth octave (A4), and the pitch of 880 Hz is A in the fifth octave (A5). Notes can sharpened (raised) and flattened (lowered) by a half-step by adding the suffix '#' or 'b' to the pitch class respectively. These suffixes, called **accidentals**, are added, as western music has twelve and not seven tones. By convention, there are half-steps between all pitch classes except between B

- C and E - F. So an A#/Bb is a unique pitch, while E# shares pitch with F and an Fb shares pitch with E (figure: 5).

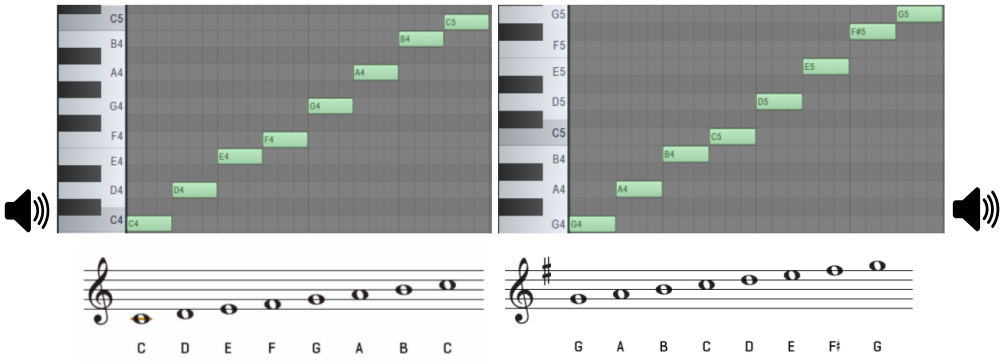


Figure 5: A C major (left) and a G major (right). The black key between F and G can be named F# or Gb, depending on the key. Pitch classes avoid redundant naming in any key, so in G major, F# is used instead of Gb, as it would otherwise contain two G-notes.

The **duration** of a pitch is relative to the beats. In MN, pitch duration is represented using ordinal categories usually ranging from whole-notes to sixty-fourth notes (figure: 6)[64]. Setting aside this categorization, duration may be described with simply a continuous scalar value.

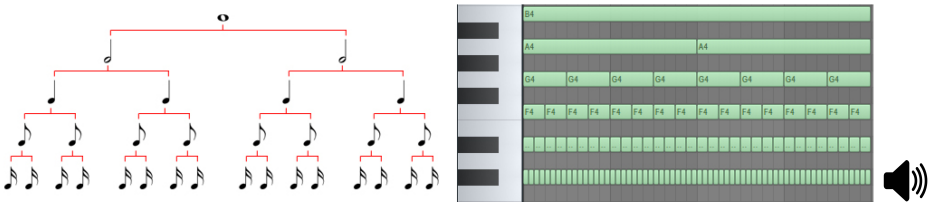


Figure 6: On the left are the MN symbols for whole-, half-, quarter-, eighth- and sixteenth-notes. Each step down the hierarchy halves the duration. The piano roll shows note duration categories down to 32nd notes.

A **note** can either refer to a pitch class or, more often, to pitch over duration [71]. The last definition can be described with a tuple containing a pitch class and a discrete time value as such: ('A#', 2.05).

Intervals in MT refer to the distance in pitch between two notes [67]. This distance could be described as a scalar value, but ordinal categories are more appropriate here. This ordinal categorization ranges: [perfect unison (P1), minor second (m2), major second (M2), minor third (m3), major third (M3), perfect fourth (P4), tritone (tt), perfect fifth (P5), minor sixth (m6), major sixth (M6), minor seventh (m7), major seventh (M7), perfect octave (P8)] (figure: 7). One advantage is that these categorizations offer a vocabulary for talking about the quality of that interval. The more major intervals a sound contains,

the brighter and more uplifting it will sound. The more minor intervals, the darker and more somber it will sound.

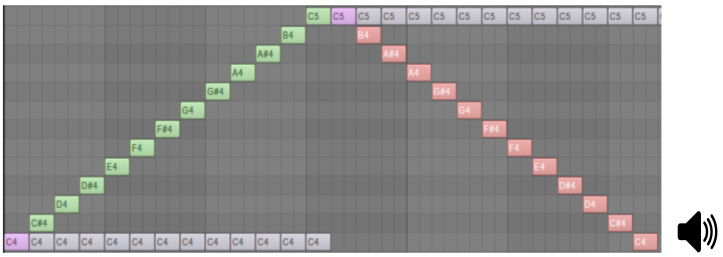


Figure 7: Greys are root notes; purples are unison intervals; greens are ascending intervals, and reds are descending intervals. The ascending and descending intervals are shown in chromatic order: [P1, m2, M2, m3, M3, P4, tt, m6, M6, m7, M7, P8].

Scales are, in concept, interval patterns from which a set of notes can be constructed, starting from a specific root note [74]. Applying the major scale (P1, M2, M3, P4, P5, M6, M7) to the root note C will result in the following set of notes: [C, D, E, F, G, A, B]. This same scale applied to the root note D will result in a set of notes with accidentals: [D, E, F#, G, A, B, C#]. A related concept is that of modes (figure: 8). The seven modes can be built from each of the major scale steps and its six consecutive steps. The Ionian mode of C starts on the first step, C, and is therefore identical to C major. The Dorian mode built from C major starts from the second step, D, and thus contains the set: [D, E, F, G, A, B, C]. The critical point here is that the Dorian mode revolves around D, so while its notes are equal to those of C major, the sound produced by this mode is not identical. Scales and modes can be viewed as vectors of pitch classes or as tuples with a name and a vector, e.g.: ('E Phrygian', [E, F, G, A, B, C, D]).

A **key** is the term for the tonality of a piece of music, and is defined with a scale with root in a specific pitch class (figure: 9)[68]. Most pieces are either notated in the major key or the natural minor key, and notes deviating from that key are marked with an **accidental**. In MN, a key is notated using the key signature, which includes a clef for reference and a number of accidentals in the range of 0-6, symbolizing which notes are raised or lowered.

Chords are the harmonic components of music and consist of three or more concurrent notes (figure: 10) [61]. Abstractly, chords are interval patterns that are used to construct a set of notes from a root note. Chords can be described as a tuple with a name and vector: ('C major', [C, E, G]). Chords can have various **qualities** based on the intervals it contains, e.g., major, minor, augmented, etc. Chord notes have a fixed order of chord degrees, e.g. [root, major third, fifth] but

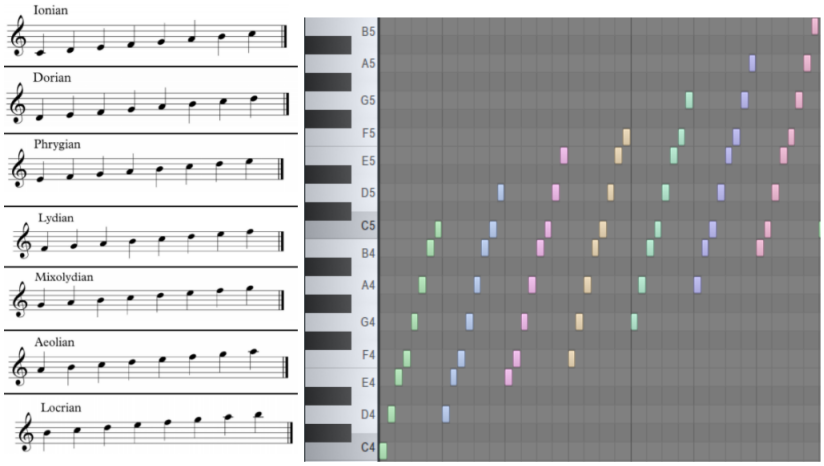


Figure 8: Illustrating all seven modes that can be built from C(4) Major. Each mode is shown with unique color. Ionian starts in C, Dorian in D, Phrygian in E, Lydian in F, Mixolydian in G, Aeolian in A, and Locrian in B. All modes contain only notes of C major

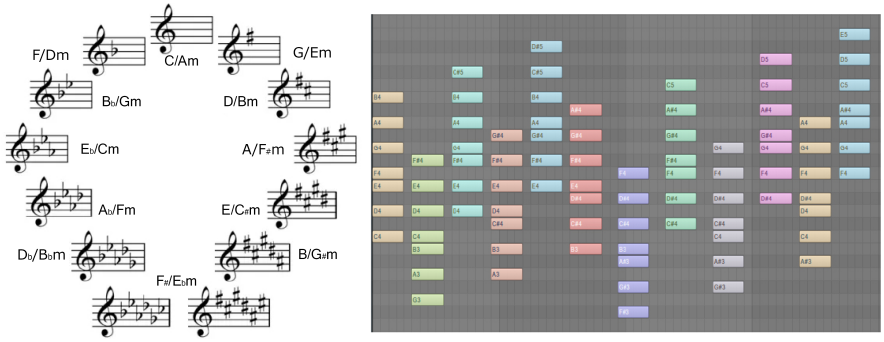


Figure 9: In MN, key is notated its corresponding accidentals at the beginning of the score. The circle of fifths (left) shows the relationship between key and amount of accidentals. All notes in each key are displayed vertically in the piano roll - moving from C major/A minor and clockwise.

can be inverted in N-1 degrees, where N is the number of chord notes.

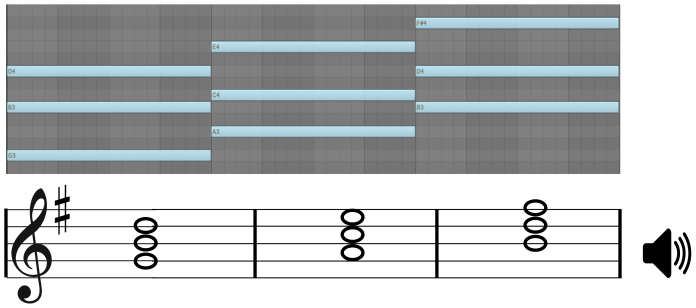


Figure 10: Basic triadic chords, here G major, A minor, and B minor, appear as stacked in MN. The intervals are, however not the same, as can be seen in the piano roll.

Scale and chord degrees are the ordinal steps of a note within a scale or chord [62]. They are typically represented using discrete numbers with a hat operator above: [$\hat{1}$, $\hat{2}$, $\hat{3}$, $\hat{4}$, $\hat{5}$, $\hat{6}$, $\hat{7}$] and alternatively: [1st, 2nd, 3rd, 4th, 5th, 6th, 7th]. Degrees are also referred to by their ascending interval distance from the root note, e.g. [P1, M2, M3, P4, P5, M6, M7]. Note, that descending intervals are not used for this, as they would produce a different result (figure: 11).

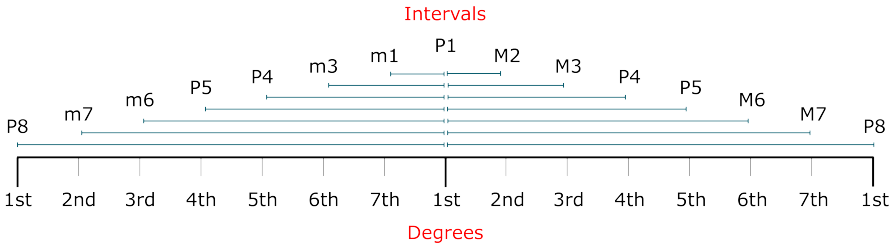


Figure 11: Ascending and descending intervals do not arrive at the same degree.

Chord progressions are groups of chords that make up larger harmonic structures. A similar concept is that of a **cadence**, which is defined as: "a melodic or harmonic configuration that creates a sense of resolution" [60]. Chord progressions and cadences are easiest represented as vectors of chord degrees in roman numerals, e.g. [I, V, vi, IV]. Providing additional information, such as the name of the progression or cadence, can be informative as well. In these cases, a tuple will be sufficient: ("Authentic Cadence", [V, I]).

This section has been an abstraction of the musical data of interest to this thesis. The next section will survey current scientific contributions to the field of MTV. This thesis understands MTV as a type of music visualization that conveys practical information of MT concepts. Practical information here means information on how to reinterpret, understand or analyze a musical piece, collection of pieces, or a theoretic framework or interrelated frameworks. Low-level representations of musical data are thus excluded, as they provide no information of practical application to the music student.

2

RELATED WORK

The previous section has characterized musical data that are the core building blocks of MT. This section will survey the current contributions made to the field of MTVs.

A survey from 2020 by Khulusi et al. [21] reviews 129 works that connect musicology and visualization and categorize the data into four groups: musicians, instruments, musical collections, and musical work - including musical scores and musical sound. The survey conducted in this thesis will have a narrower and more in-depth focus on MTVs. Compared to the paper of Khulusi et al., this will exclude topics such as musicians and instruments. On the other hand, contributions that focus mainly on theoretical principles will be included. Chan et al. survey contributions within Musical Structure Visualizations in [4]. These works can be seen as a subcategory to the MTV. Compared to the survey of Chan et al., the survey conducted in this thesis will be wider, with more emphasis on the interactive aspects of visualization.

The related work will be grouped into categories by the interaction they support. There are many frameworks for describing interaction within data visualization, but this paper will use the interaction intent framework proposed by Yi et al. [50], as this framework supports many mid-level tasks from which the related work can be categorized. Static representations will be categorized as 'non-interactive' (figure: 12).

In addition to the interaction intents, the related work has been categorized into five categories as to which aspect of MT they focus on: harmony, rhythm, melody, expression, and structure. Harmony refers to tonal relationships, intervals, chords, and harmonic progressions. Rhythm refers to note time, meter, note duration, beats, and dynamics. Melody refers to identifiable motifs and phrases created by tonal arranged over time. Expressiveness is about how musical pieces are re-interpreted by individual musicians to create individual expression. Structure is about providing an overview of music pieces to promote distant-reading analysis. While most visualizations will fall into more than one category, most will have a clear main focus, which they will be grouped by.

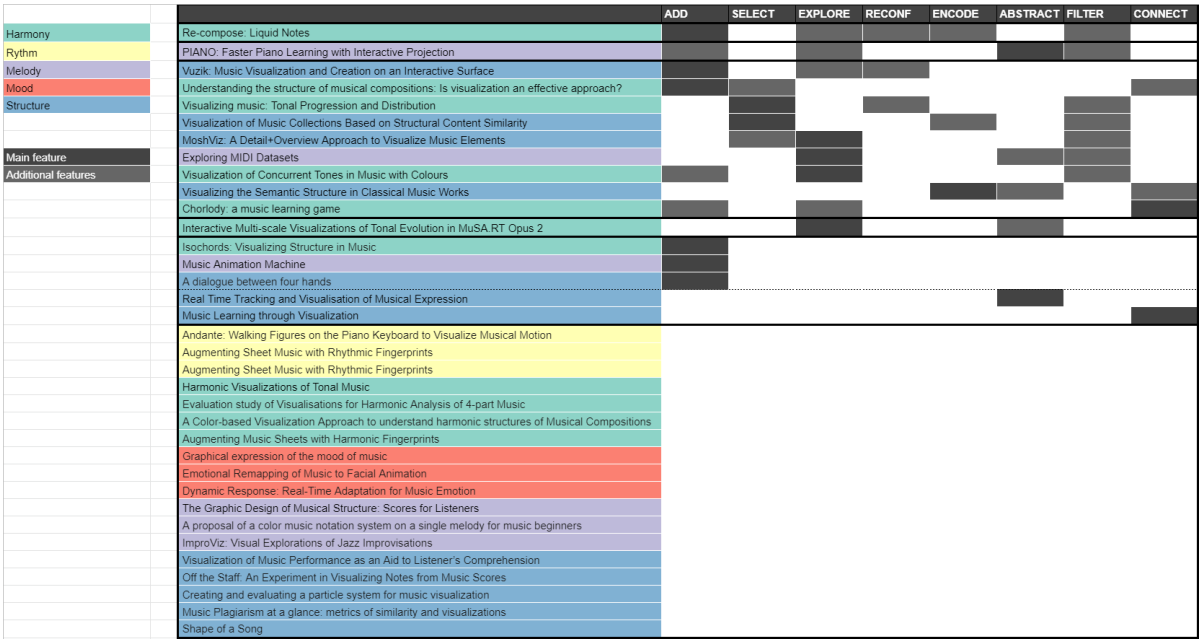


Figure 12: Overview of related work classified by interaction intent

2.1 NON-INTERACTIVE VISUALIZATION

The majority of contributions focus mainly on proposing alternative representations that can augment or replace musical notation. Therefore, they focus on static representations, and these representations can only be altered through means of scripting or rewriting source code.

Fürst et al. propose rhythmic fingerprints to augment musical notation with glyphs to support easier rhythmic analysis [15]. The glyphs are displayed above each measure and consist of a polar chart, onto which the hierarchical tree structure of note duration is mapped, from whole notes in the center and to 32nd-notes in the outer ring. Two complementary color scales are used to show respectively notes and rests, and the luminance of the colors is increased outwards (figure: 13 a). The evaluation suggests an improvement in rhythmic identification and support of both close- and distant reading of scores. Xiao et al. propose *Andante* [49], a visualization that projects animated human-like figures onto a piano to convey instructions on rhythm and dynamics (figure: 13 b). This concept aims to supplement students in learning rhythm by mapping the rhythmic features to a familiar concept of cartoonish walking styles and postures.

Craig Stuart Sapp applies three techniques for displaying the output of the Krumhansl-Schmuckler Key-Finding algorithm, which increases in specificity over time, as it partitions a piece into smaller and smaller sections (figure: 14 a)[42]. The visualization is a map showing

(a) Rhytmic fingerprints above two measures

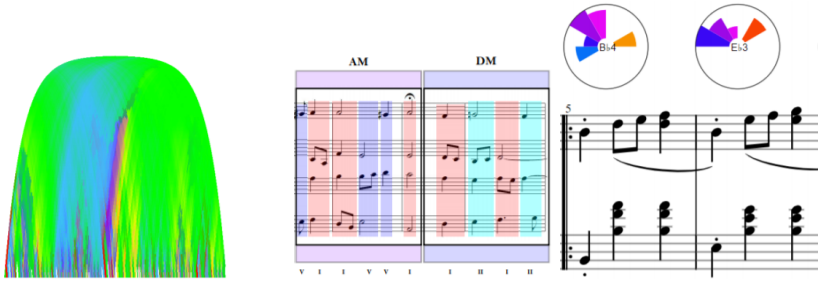
(b) Sneaking human-like figures symbolizing light playing

Figure 13: Non-interactive visualizations of rhythm

the estimated key with a color class moving from the whole piece (top) to the smallest partitioning (bottom) in a tree-like layered hierarchy. A square, triangular and logarithmic-vertical-scale view is presented, each showing time on the x-axis and layers on the y-axis. Each view reveals different trade-offs in respect to errors. To assist the tree visualizations, a line-chart is provided, showing calculation uncertainty over time. The proposed visualizations serve as an effective proposal for deriving meaningful results from the KS key-finding algorithm.

Another approach to aid harmonic analysis is through augmenting scores as presented by Melandrino et al. [28] and evaluated by De Prisco et al. in [37]. This visualization overlays the music notation with seven different categorical colors, corresponding to the harmonic function of the current harmony. On the edged of the score, another color scheme is used to display the tonality (key) within those measures (figure: 14 b). The evaluation shows that students perform better in harmonic analysis tasks using this visualization. However, it is not yet studied whether this translates to better long-term learning. Miller et al. propose harmonic fingerprint glyphs to ease the harmonic analysis task of scores [31]. The glyphs are polar charts with 12 sections that map the tones of the circle of fifths. Each section is assigned a colored area, consistent with the instances of its occurrences within a measure (figure: 14 c). The fingerprints were effective in helping test subjects in recognizing recurring harmonic patterns and identifying harmonic constructs.

[19] by Hiraga and Matsuda presents a heat map visualization of how musical performers will deviate from the musical instructions in tempo and note duration for expressive effect. This is done by comparing the original score with a MIDI file of a performance. This deviating can be seen as the musician's unique way of re-interpreting the piece to evoke certain feelings in the audience. Two visualization



(a) Visualization of KS-algorithm

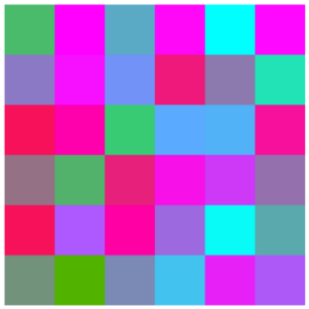
(b) Scores augmented with color

(c) Harmonic fingerprints above score

Figure 14: Non-interactive visualizations of harmony

examples are given where the first has many different colors, meaning that it is played in more free form than anticipated (figure: 15 a). Another visualization is dominated by red colors, meaning that this part is more hastily performed by pushing the tempo and dragging out notes. In extension to their previous research, Hiraga and Matsuda present another visualization for emotion in musical performance based on tempo change, articulation, and change in dynamics. This visualization uses a variation of a proportional symbol chart, where the x-axis shows time, vertical lines show the onset of notes. The notes are symbolized with boxes, where the height is their relative dynamics, and the width is relative duration. If the interval between two notes is small, then the tempo accelerates. Again, this visualization can be used to overview how performances deviate from the original piece. Dipaola and Arya derive emotions from extracting MIDI data of rhythm, sound level, timbre, articulation, melody, tonality, and duration [11]. The mapping process draws inspiration from two-dimensional emotional models, where an energy axis and stress axis determines whether the emotion is in the domain of contentment (low stress, low energy), depressions (high stress, low energy), exuberance (low stress, high energy) or anxiousness/franticness (high stress, high energy). To visualize the derived emotions, they use the iFace software, which can generate non-photorealistic 3D faces that displaying the given emotions (figure: 15 b).

To make music notation more accessible to beginners, Kuo and Chuang propose a color and shape-based notation for decoding melodies [22]. This notation uses seven color classes from the rainbow color map to map the pitch classes without accidentals. Round shapes symbolize default pitch classes; triangles show symbolize sharpened pitches, and rectangles show pauses in melody. Dynamics are encoded to size, and the duration of notes are represented with different border strokes in the rectangular containers for the notes. This makes for an easily understandable notation style that can be used

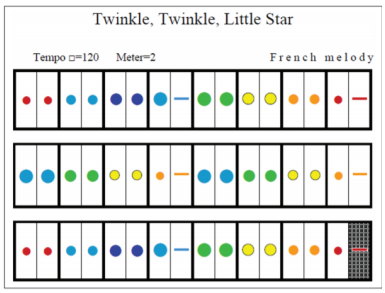


(a) Heatmap of free-form expressiveness

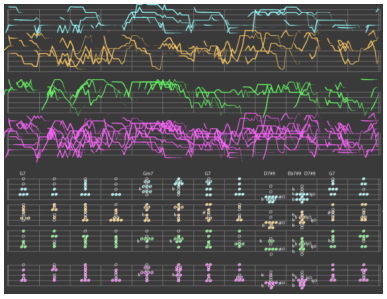
(b) Expressive 3D faces

Figure 15: Non-interactive visualizations of expression

for teaching children simple melodies while they learn about the basic building blocks of music (figure: 16 a). *ImproViz* by Snydal and Hearst is a concept for visualizing signature improvisational patterns of jazz musicians over commonly re-interpreted musical pieces [45]. The concept consists of two parts: a melodic landscape and a harmonic palette. The melodic landscape is a line chart that connects all the notes played by the artist. The edges are stylized so that they fade out, similar to how a musical phrase eases in out dynamically (figure: 16 b). The harmonic palette shows the notes that the artist chooses to play and leaves out of the original score. This visualization can inform jazz students on how to appropriate the patterns of their jazz icons.



(a) Color and shape based notation

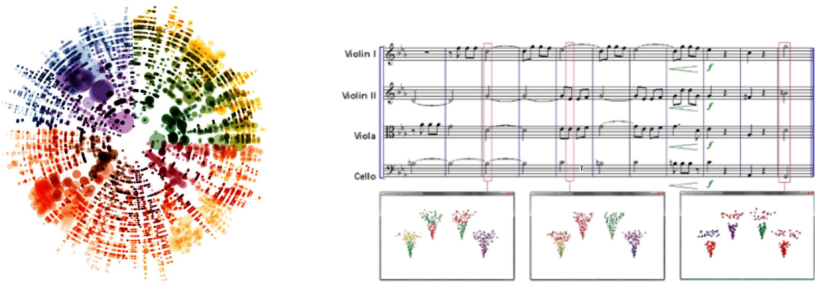


(b) ImproViz mock-up of All Blues

Figure 16: Non-interactive visualizations of melody

In [40] Nicholas Rougeux describes his design process and design of the visualization *Off the Staff*. This visualization uses MIDI files of music and transforms the note pattern into dots in a circular animation, that in turn, displays the entire structure of a piece. In *Off the Staff*, the pitch is presented with distance from the center. The distinct color classes are consistent with the different instruments playing. Lastly, the duration of notes is mapped to the dot size (figure: 17 a). The aim of this visualization is to represent musical notation

in a single image for music fans can enjoy. It is, however, also useful for overviewing how the evolving structure of a piece. In [13], Fonteles et al., create animations of particle emitting fountains to ease the learning curve of understanding musical structure. This visualization augments musical notation by adding the view of fountains. Their prototype, made in Java for the OpenGL 3D API, takes in MIDI data and assigns one fountain for each voicing or instrument in the score. The color of particles is determined by the notes, using a categorical rainbow color map, and the size of the particles is determined by the dynamics of the notes (figure: 17 b). The evaluation study carried out on the visualization shows a strong perceptual connection between the animation and the sound. Compared to musical notation, the perception of rhythm and melody is also improved.

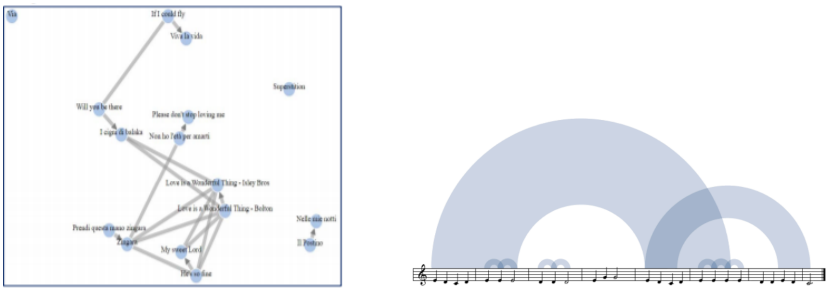


(a) Off the Staff (b) Particle fountains below score

Figure 17: Non-interactive visualizations of structure

In [10] De Prisco et al. evaluates four variants of graph visualizations that convey plagiarism between pieces by calculating melodic vectors. In Viz1, equal melodic parts are marked with a red color; in Viz2 melodic distance is mapped to the thickness of the edges; in Viz3, melodic distance is mapped to the distance between graph nodes; in Viz4 melodic distance is mapped to both of these visual channels. Viz3 performs best out of the four graphs in terms of how precisely the users identify similarity in songs (figure: 18 a). *Shape of a Song* is an arch diagram visualization created by Martin Wattenberg [48]. It can be applied to musical scores to visualize repetitions in musical phrases and motifs, thus providing a structured overview of the entire song (figure: 18 b). *Shape of a Song* can be visualized using a multi-view, where the musical score is represented in the bottom view for reference, or in a more abstract single view where only the arch diagram is displayed. This visualization offers an effective way of creating a structured overview of musical pieces that can also support analysis tasks such as identifying trends and comparing pieces or genres by structure.

A dialogue between Four Hands is an artistic visualization by the designer Georgia Lupi that aims to challenge the traditional music notation and the amount of information it can convey [26]. This visual-



(a) Viz3 for detecting plagiarism (b) Shape of a song

Figure 18: Non-interactive visualizations of structure

ization splits music up in bundles of lines, where the part above the junction shows notes played on the left hand of a guitar, while the part below shows the strikes of the right hand of a guitar. Measures are shown with thicker lines, and colors are used differently from the top and bottom views: colors in the top show the articulation of the notes, while colors in the bottom show which finger is used to strike the string. This visualization shows well how far from traditional musical notation a visualization can be while still maintaining the essence of the data. It also shows how animation and stylization can be used to make a visualization that is also aesthetically pleasing (figure: 19).

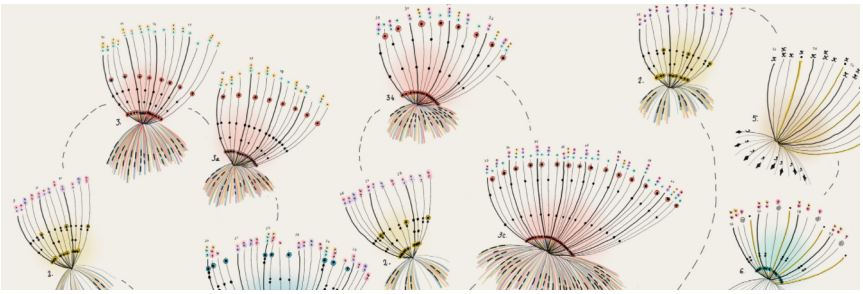


Figure 19: Non-interactive visualizations of structure: Dialogue between four Hands

2.2 SELECT

Soriano et al. present a novel technique for viewing one's music collection based on similarity from a MT viewpoint. This is achieved by representing tonal sequences as vectors and comparing them comparing using a Dynamic Time Warping distance function [46]. From this vector, a single stacked bar-glyph is constructed for each song, while a coordinated view lays all songs out as nodes in a graph. The node positions are calculated using a Self-Organizing-Map neural network, which also results in a somewhat structured clustering consistent with the genres of the songs (figure: 20). The main interaction intent

supported in this visualization is that nodes can be selected to view the consistent songs in a media-player view and in the stacked-bar glyph view. Moreover, more or less detail can be provided by zooming in and out, supporting the abstraction/elaboration intent, while the interface can be reconfigured to show/hide legends, change the color scheme of the nodes.

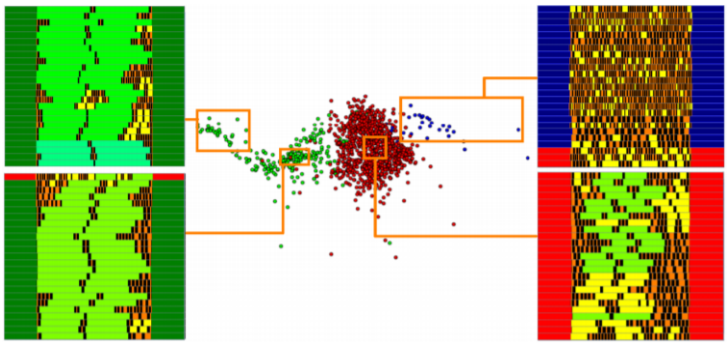


Figure 20: Graph showing structural similarity

2.3 EXPLORE

Ciuha et al. presents a piano roll in 3D-space to visualize concurrent tones with colors to provide insight into the harmonic structure of a score [8]. The coloring is achieved by adding vectors that point from the center of a circular rainbow color map with low saturation in the center and high saturation in the periphery. The vectors point towards either of 12 nodes (representing notes) that are arranged around the map in a circle of thirds, where closely related notes are positioned adjacent to each other. Thus, consonant notes will be a mix of closely related hues, and dissonant notes will be a color with low saturation after the vector addition. The 3D space can be rotated, panned, and zoomed to explore different views of the piano roll (figure: 21). The exploration is also supported by a timeline of the musical piece, which can be adjusted to the point in time that the user wants to see. Besides the explorative interactions provided, real-time MIDI data can be inputted, and the user can thus input data directly from an instrument to see how this data connects to the other data points.

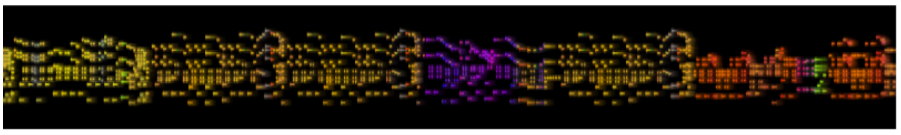


Figure 21: Concurrent tones visualized with color

MuSA.RT Opus 2, proposed by Chew and François, is an interactive visualization that maps real-time musical input to a helix-shaped

pitch space [7] - distributing notes equidistantly following a circle of fifths pattern. By connecting the concurrent input of notes from their position on the helix, harmonic structures become apparent through different geometric patterns (figure: 22). *MuSA.RT Opus 2* is envisioned to serve both as a tool for pattern recognition for expert musicians, learning patterns for novice musicians, and for an entertaining visualization for general audiences - however, the implementation is currently limited to showing major and minor chords. The system offers two main ways of viewing the visualization: the first allows users to explore the visualizations by actions of moving the camera up and down, rotating the view around the helix, and zooming in and out. The second mode, called auto-pilot, is calculated by the system itself to smoothly transition through the shortest path to view the new patterns. As the system allows for user input, another utility would of the system is that the user themselves can input signal from a MIDI instrument to see how they connect geometrically. The software is available for commercial use on Apple products through App Store [14].

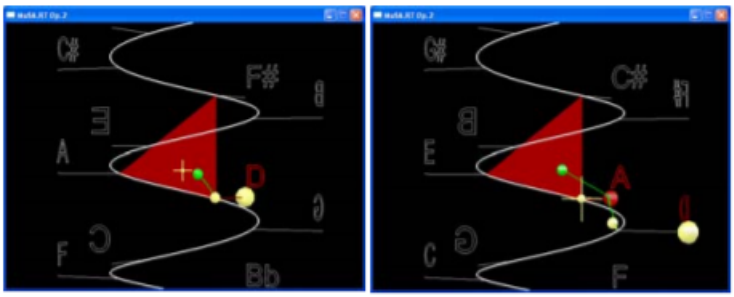


Figure 22: Helix shaped pitch space of MuSA.RT Opus 2

In [32] the system *comp-i* is proposed by Miyazaki et al. This visualization represents MIDI data in a 3D-virtual space with a temporal axis, a pitch-space axis, and an axis dedicated to each channel in the MIDI input. In comp-i, notes are represented as cylinders where the diameter is mapped to the dynamics, and the saturation is mapped to the duration of the notes. Each MIDI channel is also assigned a categorical color hue so that they are easier to distinguish from one another. When the MIDI data is played, a plane layer follows the time axis to represent the current point in time (figure: 23). The main interaction intent that this visualization supports is the one of exploration, allowing users to pan and change the camera viewpoint. One can also zoom in and out (abstraction), and the data can be filtered by manipulating the visibility of the objects in the 3D environment. Comp-i allows a unique way of exploring how melodies of a given piece develop.

MoshViz, is a visualization by Cantareira et al. that provides detail + overview of a musical piece through coordinated view along with a

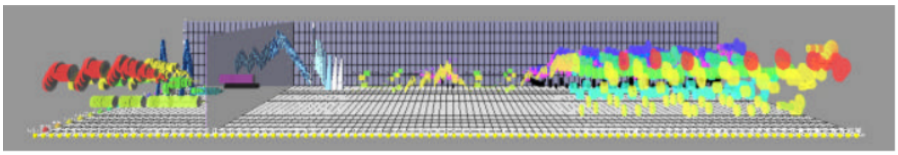


Figure 23: Comp-i: Overview of a MIDI dataset

unique presentation of the stability and harmony [2]. In the detailed view, each note is represented with a circle, where the size is proportional to the dynamics of the note, and the color mapping from green to red is proportional to respectively the dissonance and consonance of that note in relation to its chord. The circles are combined with a line for representing the duration via the length-channel, making the view somewhat similar to a connected dot-plot. In the overview representation, each rectangle corresponds to a measure. The height of the rectangle is mapped to the interval-span of notes within that measure, the transparency maps the density of notes, and color maps the instability/stability of the harmonic construct. The mean-note played within a measure is shown with a black line in the rectangle, making this representation visually similar to a box-and-whiskers plot. Below this representation are several other representations, three of them showing respectively complexity, repetition, and interval variation in green, purple, and blue-hue heat maps (figure: 24). *MoshViz* allows exploration through adjusting the timeline to the point of interest. Users can zoom on interesting data in the detail view. Filtering of different instrumental channels is also allowed. While *MoshViz* is extremely visually dense, the information that can be derived from this visualization is also proportionally vast.

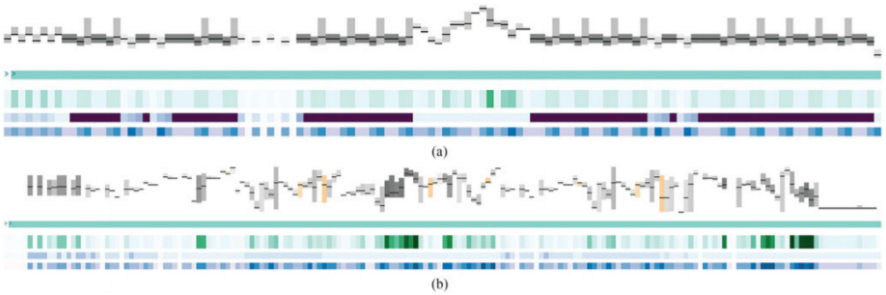


Figure 24: Overview of the MoshViz visualization

2.4 ENCODE

To visualize the structure of classical pieces, Chan et al. propose three prototypes for a highly customizable visualization [3]. The first prototype is named Layer Braid, inspired by a plaiting metaphor. This visualization shows how the different instruments in a composition

take on different roles in a musical piece over time, meaning that, e.g., violins can go from a dominating position in a piece to providing contrast to other instruments later. This visualization is somewhat similar to parallel coordinate systems, where the first axis would show the instrument category, and all other axes would be what task they fulfill at a certain time point. The two other prototypes show additional information by adding a Theme Glyph, which is inspired by the classical notation of a note. The glyph is altered in different ways to present different features of themes, so if a theme is transposed, diminished, augmented, inverted, shortened, etc. This is reflected in the appearance of the glyph. Different themes are at the same time shown with different amounts of flags on the glyphs. These two prototypes take different approaches to organizing the glyphs and layer braids (figure: 25). The interaction supported in these visualizations is encoding through customizing the color scheme of the representations. The user can also connect similar segments by linking and brushing, and a fish-eye lens is provided to zoom in for a more detailed view.

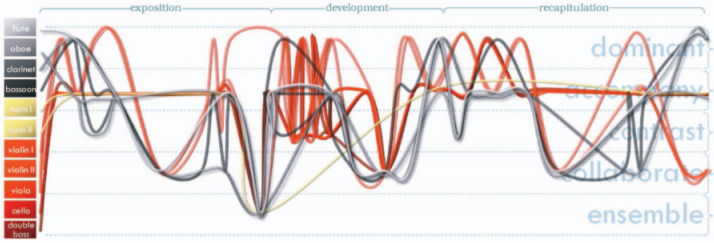


Figure 25: Layer braid representation of instrument tasks

2.5 ABSTRACT

Rogers et al. [39] propose the interactive *P.I.A.N.O.* system, which aims to support faster piano learning for students with no prior sight-reading experience. The piano-roll visualization is projected onto a real piano and animated, such that the notes approach the player at a speed equal to the tempo of the song. Additional channels are added to the marks to convey information to the player. Rounded corners mean lighter accent, sharper corners mean heavier accent, trill and grace notes are shown with small glyphs, and finger placement is mapped to color (figure: 26). *P.I.A.N.O.* Offers three modes of playing back the visualization: Listen mode, Practice mode, and Play mode. The first mode merely plays the visualization to provide an overview. The practice mode supports correct plating by waiting until the right note is pressed to progress temporally. Here right and wrong notes are highlighted with red and green, respectively. The player can jump in time using a fast forward or backward button. The play mode supports real-time feedback while playing from one end to another. Here, the tempo can also be adjusted. The main interaction of this visual-

ization is that the player can, in a sense, abstract which granularity is of information is desired through the different playback modes. The player can also explore through panning to other places in time and connect new data by inputting data and receiving feedback through color.

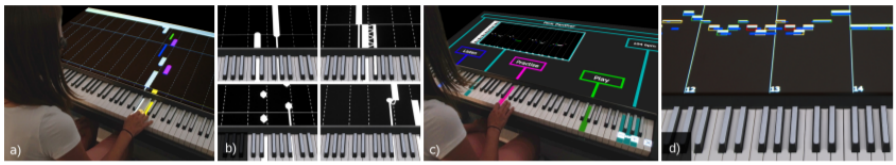


Figure 26: The P.I.A.N.O. system supporting different tasks

In [12] Dixon et al. provide a visualization for representing automatic analysis of expressive music performance, meaning the re-interpretation of a performer of an existing piece. While other papers have focused more broadly on expressive dimensions, Dixon et al. restrict their focus to tempo and dynamics. From calculating dynamics and tempo from audio files, this data is turned into a representation which the authors coin, the Performance Worm. This representation is a coordinate system with the x-axis dedicated to tempo and the y-axis dedicated to loudness, with each time point in the musical performance plotted as a circular marker. The most recent circles are drawn on top of the other circles with a darker fill color, which creates a worm-like line (figure: 27). This visualization allows comparison of different performances on the basis of the artist's individual re-interpretations. The main supported interaction is zooming in and out through adjusting the axis.

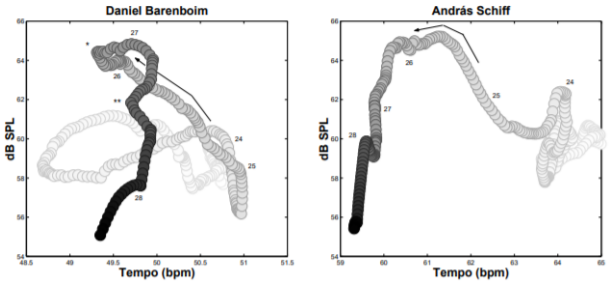


Figure 27: The *Performance Worm* can be used to compare expressive parameters between performances

2.6 FILTER

Mardirossian and Chew present a visualization for showing tonal progressions and distributions in a pitch space proposed by Lerdahl [30]. The pitch space consists of a dotted grid of pitch classes arranged in a pattern that can be repeated infinitely. Whenever a note occurs in the MIDI input, the dot of the pitch class grows in radius (figure: 28). The

visualization can be either a static presentation of the note distribution or an animation of how the piece evolves over time. This makes for interesting discoveries of how music develops harmonically and is especially effective for detecting pattern variations between different genres. The main interactive component to this visualization is that the visualizations allow you to filter what song you want to see the data for and whether you want the data to be presented as an animation or as a static view. The user is, however, also able to zoom in and out on the view, which falls under the abstraction/elaboration intent.

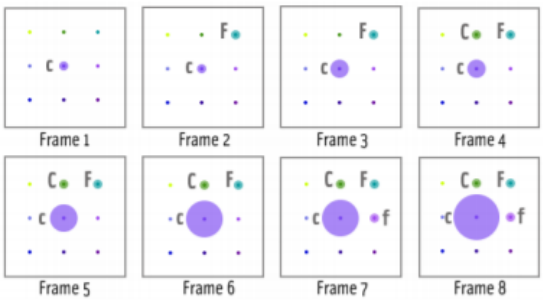


Figure 28: The circles grow over time as the piece progresses

2.7 CONNECT

[25] presents *Chorlody*, which is an interactive music learning game. The core of the game is to press approaching notes at the right point in time, like, e.g., the commercial game Guitar Hero, but unlike Guitar Hero, the game focuses on chord shapes rather than individual notes. At the center of the interface, the game has a circle of notes, which are chromatically arranged. Each of these notes has a hue assigned to them, similar to the rainbow color maps. The chords are represented as geometric shapes, with each edge pointing towards one note in the circle. These shapes are named and colored, corresponding to the contained notes. An additional view provides a histogram column, where each tick in the histogram corresponds to the color of the played chord (figure: 29 a). The main interaction of the game is connecting the data notes via pressing the correct chord notes when a shape aligns with the circle. Notes can be inputted in real-time to see how they connect to the existing data.

[18] Hiragi et al. propose a system with two views to aid musicians in exchanging performance information. The first view provides overview + detail of the musical score. Measures that are not in focus are compressed horizontally. Concurrent notes are shown with lines, where brightness/darkness maps the number of notes, and height of the line maps the dynamics. Measures are expanded when in focus

and resemble regular musical notation. The other view shows expressive cues through Chernoff's face-glyphs. The expressive ques shown with these glyphs are tempo, articulation, and sound. The eyeballs of the glyph show tempo by looking backward or forwards (dragging/-pushing). The contour of face and mouth shows legato or staccato, the directions of the nose show a strong or soft tone (figure: 29 b). The main interaction of this contribution is choosing a measure and see how that connects with the performance expression. Abstraction is of course also a central feature as you can focus on some measures while others just provide context.

(a) Chorlody: interactive music learning game

(b) Performance information exchange system

Figure 29: Applications supporting the connect intent

2.8 CONNECT NEW DATA

Liquid Notes by the company Re-Compose is a software plugin for a Digital Audio Workstation (DAW) [63] such as FL Studio, Logic Pro, or Ableton Live [38]. Liquid notes provide a unique and highly interactive approach to writing chord progressions based on MT. The plugin provides two coordinated views, where the top view is a classic piano roll representation, and the other view is a control panel with drop-down lists and knobs and sliders to customize the chords. The sliders provide three steps where the user can choose a I, IV, V chord consistent with the chord functions tonic, subdominant, and dominant. One knob provides control of the amount of tension that the user wants to include in the chord. The higher the tension, the more dissonant notes will be added to a given chord. Through the drop-down, the user can choose any given chord fulfilling the criteria of the chosen function. The main interaction is that of connecting new data. The user can also explore the piano roll through panning and scrolling. Notes corresponding to a specific control panel can be encoded with different colors for easier organization, and the user can filter the possible tones by adjusting the criteria set-up in the panel (figure: 30).

In [36] De Prisco et al. propose a tool for effective learning of musical composition called VisualMelody. Their system takes offset in a specific branch of classical music called chorale, which strictly adheres to the harmonic guidelines for writing three- or four-voice compositions. The visualization provides two coordinated views, where the lower view is the MN, and the upper view is either a line chart representing the melodic trajectories with colored lines or a scatter plot representing the harmonic intervals with colored elliptical shapes. In both representations, color is used for mapping the relation between views following the rule of self-evidence. Combinations that violate guidelines are highlighted with red line- or border strokes (figure: 30). Evaluations carried out in the paper show improvement of performance in creating good chorale compositions. The interaction intent covered in this system is inputting notes through the music editor and see how they connect harmonically to the other notes in the composition.

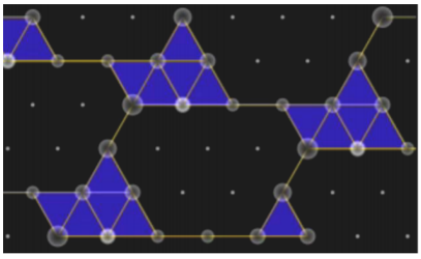
(a) The LiquidNotes interface

(b) VisualMelody showing trajectories and intervals

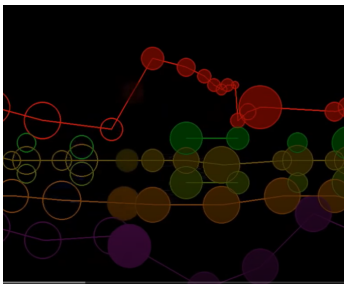
Figure 30: Applications supporting connecting new data

IsoChords is visualization proposed by Bergstrom et al [1]. *Isochord* visualizes musical structure using animated nodes within a Tonnetzgrid [75]. The grid consists of upward and downward pointing triangles in alternation, with a node at each vertice representing a pitch class. The arrangements of notes can be seen such that from any note, a step to the right will add a P5 interval, a step down the left-to-right descending diagonal will add an m3 interval, and any step up the left-to-right the ascending diagonal will add an M3 interval (figure: 31 a). *IsoChords* is animated in real-time as the music plays, and the geometric shapes of the grid light up in blue if all three surrounding notes are played simultaneously. The interesting outcome from this is that harmonic relationships can be perceived with ease as the user learns to recognize the shapes. As an example, all major and minor triad chords are represented with respectively upwards and downwards triangles. *IsoChords* allows the user to connect inputted MIDI data in real-time through e.g., a MIDI instrument.

The Music Animation Machine is a visualization software made by the artist Stephen Malinowski's [29]. It takes MIDI input and transforms it into geometric representations with duration on the x-axis and pitch on the y-axis. The x-axis is animated one-to-one with the time, such that the current point in time will always appear in the middle of the screen. The representations can vary from a regular piano roll to more experimental visuals, where the melodic trajectory is represented with connected lines and the duration of notes is represented with a radius of each of the notes connecting the trajectory. In the latter representation, the radius will shrink with time as the note is closer to its ending (figure: 31 b). Other visual variables can also be mapped to other channels, such as color to show harmonic information. *The Music Animation Machine* aims to display music in a more accessible way so that the users can understand it from seeing and hearing a piece simultaneously. The main interactive component of this visualization is that MIDI data can be inputted in real-time to see how it connects to other inputted data.



(a) The IsoChord visualization



(b) Music Animation Machine. Image source: [81]

Figure 31: Applications supporting connecting new data

Pon et al. present *Vuzik*, which is a visualization that supports music creation on an interactive surface [35]. This visualization aims to be accessible to children who are new to creating music. Users can input notes via drawing on the surface with their hands or using a tangible interface consisting of a physical paintbrush and color palette with embedded electronics. When played, the y-axis of the painting canvas corresponds to pitch, while the x-axis corresponds to time - except that lines will be played back from the direction in which they are drawn. Additionally, the color that the users choose will determine what timbre the notes are played with (figure: 32). *Vuzik* supports inputting data as the main feature, while exploration is somewhat supported by the implementations of playback and play forward functions.

Figure 32: The Vuzik interface with the TUI paintbrush and color palette

2.9 OUTLINE OF CURRENT RESEARCH

This section has presented the current state of research in MTV. The related work presented here should not be viewed as a complete collection, but it should offer a thorough and detailed overview of current research in MTV.

An obvious finding from surveying the work is that nearly half of the papers focus on augmenting or replacing sheet music with other static representations. These are great supplementation to MN, and nearly all the papers document promising results in aiding the musical tasks not implicitly supported in MN and other modern notation styles.

Another apparent finding is that there is substantial room for supporting new interaction intents within all five music categories. An example is that no interaction intents to support rhythm have been researched. By comparing which interaction intents have been explored in which categories, it is clear that much research still needs to be done. Lastly, looking across all music categories, reconfiguration and selection stand out as being vastly underexplored.

The last point worth highlighting is that none of the current contributions tries to tackle the problems of designing an effective MTV through unit visualization. This is conspicuous, as this would be a powerful technique to create connections between the many multifaceted representations and models of musical data and MT. In the next section, a conceptual framework will be presented based on these learnings.

3 CONCEPT

As identified in RELATED WORK, the interaction intent of reconfiguration is largely underexplored within the field of MTV. Reconfiguration, however, holds much potential in resolving the problems with the current music notation outlined in INTRODUCTION. To demonstrate this potential, this thesis will propose a concept for a modular music visualization that supports reconfiguration and exploration at its core.

3.1 UNIT VISUALIZATION

One of the main challenges in designing a visualization that supports reconfiguration is to encode the data into modular elements easily combined in new patterns to reveal new insight. To address this challenge, the conceptual framework will take offset in the technique of unit visualization. As defined by Park et al. [34], unit visualization encodes each data element into a unique visual mark that maintains its identity and relation to that data piece throughout.

Park et al. identify the strengths of unit visualization as following. First, the direct one-to-one mapping of data is intuitive to understand, as it requires no degree of mental abstraction from the user. Second, using animated transitions, data elements can easily be tracked throughout sequences of interaction. Third, unit visualization is accessible to novices, as it is in line with how they empirically approach constructing a visualization with tangible tokens [20]. Last, users can get details-on-demand through interaction, down to the smallest resolution of data. These strengths go hand-in-hand with the recognized goal of this thesis: providing a modular, reconfigurable visualization that bridges musical concepts in a seamless and accessible way.

The weaknesses of unit visualization are that there is a limit to the computational scalability that varies from device to device, how many elements can be displayed in one screen while still being distinguishable, and how many details a human vision can perceive and track. These are all points that the visualization will have to address. For this reason, it is essential that the visualization is designed cleverly around the workflows of a musician so that no more than the necessary amount of details is available at all times, which will be attempted in the Use-case section.

3.2 THE ELEMENTS OF THE VISUALIZATION

The unit visualization will take offset in the piano roll notation, which plots rectangular markers representing notes, into a coordinate system. The y-axis represents the ordinal pitch classes from lowest to highest octave. The x-axis represents time, partitioned into the beat unit of choice. The reason for taking inspiration in this notation is that it is expressive, as it conveys the nature of notes as discrete elements that make up the larger piece. It is also in line with Mackinlay's ranking of perceptual tasks presented in [27]: The position channel is used to convey the ordinal pitch class and the time onset of notes. Duration is encoded with the length channel, ranking second highest to position for quantitative data. Third, the piano roll notation is widely used in DAWs [63], and commercial music games, such as Guitar Hero [66], and the user will therefore likely have a conceptual model of how to approach it. [75]. Lastly, the rectangular marker style is an obvious fit for a flexible unit visualization due to its geometric simplicity.

The elements adopted from the piano roll are the pitch axis, the temporal axis, and the rectangular marker style for notes (figure: 33). In addition, it includes three types of harmonic relationships, all represented with semi-transparent lines. The first, chord intervals, are represented with lines connecting notes vertically. The second, intervals between subsequent notes, is represented with lines that connect notes horizontally. The last, the possible trajectories from a given chord, is represented with lines pointing from the given root note towards the possible directions allowed in the rules of tonal harmony (figure: 34). The first two mentioned harmonic relationships are color-coded to convey the quality of the note intervals: consonance, imperfect consonance, dissonance, and major/minor qualities. The color scheme for these elements is based on four easily distinguishable color categories: Grey maps consonance, cyan maps major + consonance, yellow maps minor + consonance, and red maps all dissonances. The colors were chosen to give more emphasis to major and minor consonances and even more to dissonances. For tonal trajectories, two types of color maps can make sense; either using three distinct colors to map harmonic function of either tonic, subdominant or dominant or use an ordinal color scale to map the seven different degrees which a note can go to next.

Above the notes, chords are shown with rectangular labels containing the chord names. This information is often included in various notation styles and is helpful for providing an overview or when playing chords only. The width of the labels maps the temporal span of the included notes each chord. The harmonic function of the given

chord is encoded to the background color of the label. This color scheme would be similar to that described for the harmonic trajectories.

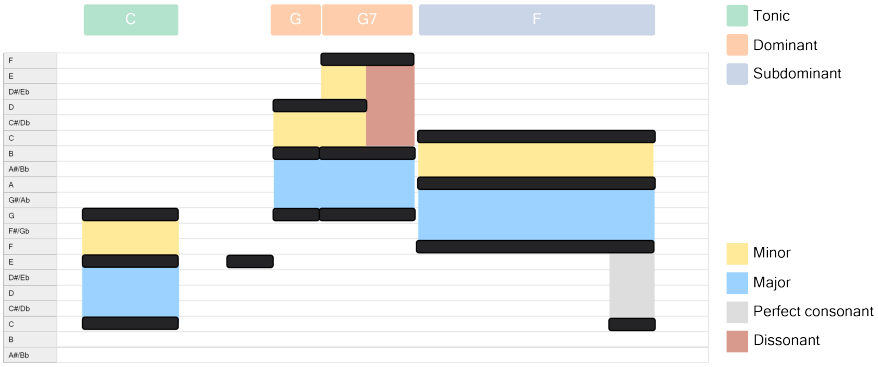


Figure 33: The default view of notes, harmonic relations and chord labels. An additonal control panel for the user was positioned above.

Figure 34: Harmonic relationships between subsequent notes (left), and harmonic trajectories (right). The former relationship uses the color scheme associated with intervals, while the latter uses the color schem associated with function, as these are the relationships they convey.

Before going into detail with the interactions, a use case will be described to diagnose at which steps, which tasks should be supported.

3.3 USE-CASE

This anticipated use-case will be grounded in Bloom's Revised Taxonomy for Music Education [16] for a realistic learning scenario.

3.3.1 *Remembering*

Before engaging with the visualization, the musician will need to recall the fundamentals of music, such as where different pitch classes are located on their instrument. Assuming the player uses a piano, the degree of abstraction required of the musician is limited to identifying the correct octave on the piano.

3.3.2 *Understanding*

When a musician first sees a new piece, they will need to get an overview. The visualization will accommodate this by seeing the default view zoomed out on the x-axis to fit most information into the view. The user will quickly be able to overview the length of the piece, the overall complexity, and the different sections that make up the piece. The harmonic relations will also be visible at first to view the harmonic complexity of the piece.

In the control panel, independent variables of tempo and time signature will be displayed at all times to provide temporal and rhythmic context to the piece.

3.3.3 *Applying*

The musician will want to start practicing the piece, measure by measure, or chord by chord. This is accommodated by zooming in on the details. A play-along feature should be provided, allowing the user to play along in an optional tempo since this can be helpful while acquiring muscle memory.

3.3.4 *Analysing*

When a musician has mastered playing a piece, they will want to analyze how the piece is composed. This can be achieved via conversion operations that, e.g., reorders the y-axis to show how the tones relate to theoretical concepts. Also using the conversion operations, the user can toggle on and off harmonic relationships and the harmonic trajectories for selected notes. Analysis may also benefit from computation operations such as aggregation of notes or intervals into a bar-chart. The specific smoothing operations will be elaborated on in the Smoothing operations section. At last, the user might want to compare the piece to other similar pieces. This framework propose allowing for comparison operations through either overlaying the original views or providing an extra set of coordinated views with data from the other piece.

3.3.5 *Evaluating*

For evaluation, users may want to identify dissonances and locate notes that are out-of-key (outliers). Inspecting if any progressions are breaking the rules of tonal harmony by viewing harmonic trajectories may also be helpful, and lastly, identifying which cadences and chord progressions were used will make sense. The 'norms' for what is al-

lowed will vary much for different genres, but activities will at least lay the ground for a sound evaluation.

3.3.6 *Creating*

Creation involves the knowledge of musical theory applied to, e.g., reharmonization, improvisation, and composing [16]. However, this thesis hypothesizes that reharmonization (e.g., making changes to an existing piece) is also an effective way of internalizing knowledge. Reharmonization/composing is supported through constraint operations such as sorting notes in new patterns and comparison/conversion operations such as viewing and changing scales, modes or chords.

A use case has now been presented to sketch out an anticipated workflow and justify the appropriateness of this kind of visualization. The next section will present the specific concepts for smoothing operations that are part of this framework.

3.4 SMOOTHING OPERATIONS

The proposed unit visualization gains its strength from its flexibility, which is accessed through exploratory interaction. The interactions are grounded in the concept of smoothing operations (C8) presented in the Exploratory Sequential Data Analysis (ESDA) approach [41]. Smoothing operations can be split into two categories as identified in [51]: data exploration operations and annotation operations. This thesis is will only concern itself with operations of the former category:

- **Conversions:** Operations that transform data to reveal new patterns. Through the lens of unit visualization, this will mean changing the context in which visual variables appear since the visual variable itself is atomic and immutable.
- **Constraints:** Operations that show data conditionally through e.g., filtering, sorting, or grouping
- **Chunks:** Chunking allows the user to carry out the same operation on a group of data through selection techniques like bounding box selection.
- **Computations:** Computation operations provide a summarized view in a representation that is easily digestible.
- **Comparisons:** Comparison operations allow the user to explore relationships between data objects or chunks of objects.

3.4.1 *Conversions*

A core concept for conversions operations is to reorganize the pitch-axis on the visualization. The y-axis can be reorganized into a line-of-fifths, similar to circle-of-fifths, which has been heavily utilized in related works such as IsoChords [1] and [8]. The benefit of displaying the y-axis in a pattern of fifths is that it provides a measure for consonance and dissonance in the given tonality. I.e., the tones that appear furthest away from the tonal center in the circle of fifth are the most dissonant to that key.

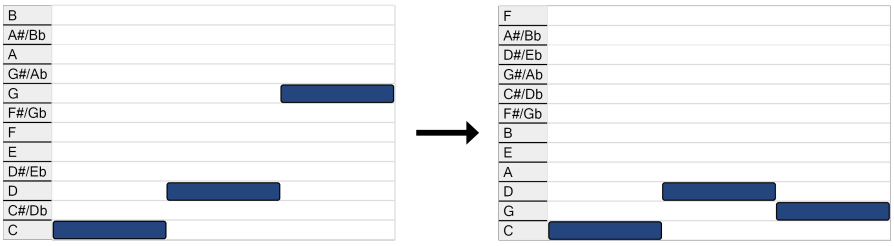


Figure 35: Conversion from a chromatic ordering of the y-axis to an ordering by fifths.

A similar operation is showing the y-axis in pitch classes separated by thirds instead. Ordering the notes like this will make it easy to identify common chords, as the notes will, in most cases, stack on top of each other in this view. This might give the user an insight into the musical piece, and in any case, it can serve as a tool for visualizing how chords are built.

Another operation on the y-axis is to only show pitch classes of a given tonality. As a result of this, the pitch distance of will not be equidistant half steps, but an interval pattern corresponding to that of the tonality: for a major tonality, the steps of the y-axis would be distanced W, W, H, W, W, W, H starting from the root. This reconfiguration appropriates the layout of traditional MN, which is arguably less effective for sight-reading. The benefit is that it gets easier to identify the degrees of notes within the tonality. A similar conversion operation is simply greying out notes that are not included in the tonality. This will make it easy to spot notes that are outliers with respect to the tonality.

The semi-transparent lines representing harmonic relationships can be toggled on and off for all notes or a selected chunk of notes. This data is good to show for Analysing and Evaluating, but counter-productive to Applying.

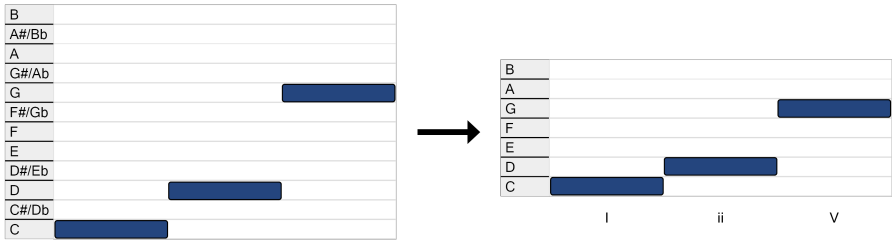


Figure 36: Conversion from chromatic ordering to showing only notes within the key

3.4.2 *Constraints*

Performing constraint operations through reconfiguration interactions is arguably limited, as both grouping and filtering operations are not reconfigurations. Sorting, on the other hand, is an interaction that reconfigures the data.

The first constraint operation is to sort selected notes or chords by an ascending pattern. This can be done by moving tones by lowering or raising them an octave. The same applies to chords. However, in some cases, it will make more sense to simply invert a chord to conform to the ascending pattern if possible. This will reduce the degree to which the chords are spread out over more octaves.

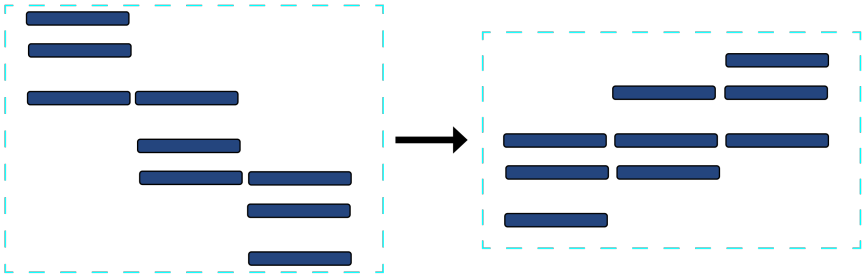


Figure 37: Conforming notes from descending to an ascending pattern

A similar operation is that of sorting a selection by a descending pattern. Likewise, tones can be sorted to conform to a pattern in which they are positioned as closely as possible or as far away as possible. Providing the user a tool for customizing sorting patterns will also allow a high degree of freedom in exploring different rearrangements. The purpose of the suggested sorting operations mainly falls within Creating. By quickly sorting the chords into other patterns, the user can play them on an instrument and see how this affects the overall sound.

Another constraining operation is to partition a voicing into more views, in case some part of the piece is more interesting to analyze or if some part should simply be left as is. For instance, bass pat-

terns tend to be very dominating in regards to perceived tonality and should sometimes not be altered too much. Another case is that different voicings like vocals, strings, percussion, and guitar should be treated differently, as they generally behave differently in compositions. An opposite operation of the partitioning is joining. This could make sense when pieces of a song need to be explored simultaneously.

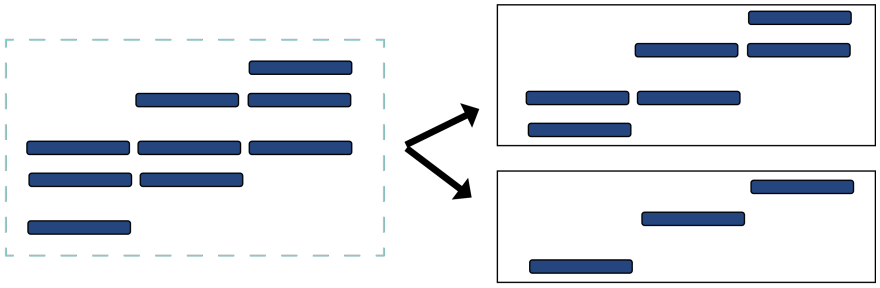


Figure 38: Partitioning a selection into two views

3.4.3 *Chunks*

The first chunking operation is basic user-defined selection. This operation can chunk any notes together from anywhere in the representation. This operation is supported by either clicking every single note and selecting more notes within a bounding box.

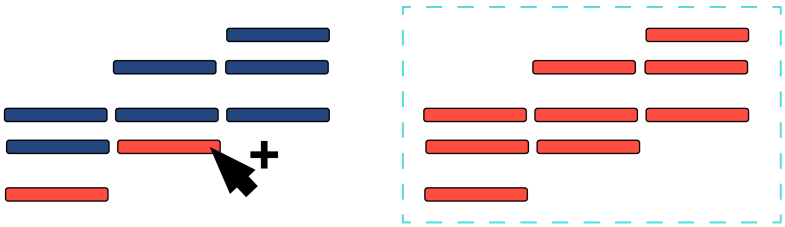


Figure 39: Chunking by pointing and clicking or by bounding box selection.

Chunking operations should also consider that music consists of repeating patterns that make sense to select as groups. This is allowed through selecting a set of matching criteria. Matching criteria add all elements matching the selection to that selection. They include matching by: pitch class, interval, beat onset, duration, rhythm, harmony.

3.4.4 *Computations*

A category of proposed computation operations is aggregation. A concept for this is to arrange all notes in a bar chart, showing the distribution of the piece. This concept is particularly powerful combined with the line-of-fifths representation presented in Conversions,

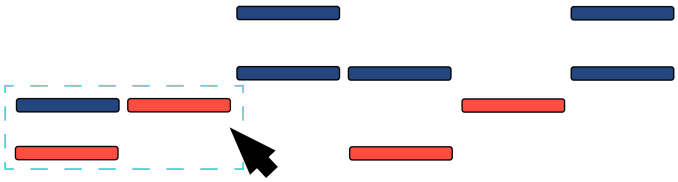


Figure 40: Notes that match a given criteria will be included in the selection.

as the user can quickly identify the key pitch class by the longest of bars, while the other viewing how the most consonant notes to that key distribute around that pitch class 41). Similar operations allow the user to aggregate all harmonic relationships into a bar chart to view which intervals are most commonly used in the piece. This quantifies the interval qualities used in a piece. Lastly, the user can aggregate all chord labels into a bar chart, showing which chords are most common.

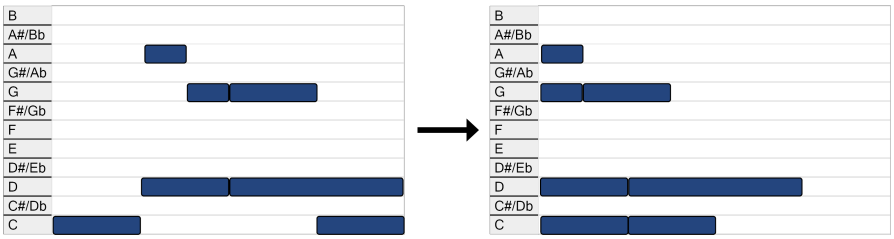


Figure 41: All notes are floated to the left to create a bar chart.

Another computation operation is to cluster together chords by harmonic similarity, chord quality, or function. This allows for easy identification of repetitions or variations to chords, as well as seeing how closely chords are related. Clustering by function shows which chord function is used most: tonic, subdominant, or dominant chords. Another application to this is, that a user can analyze and adopt certain chords typical for a song/genre/artist.

Last is a concept named the tonal harmony flowchart. This highlights a line connecting root notes of chords in a piece and color encoding them by their function. At each node of that line are semi-transparent trajectories pointing towards the other possible trajectories allowed within the theory of tonal harmony. This gives the user a summary of the choices made in the harmonic writing.

3.4.5 *Comparisons*

The comparison operations allow the user to view alternatives to the current harmonization. The first proposed comparison operations allow the users to view a transposition of all notes in their selection. Transposing, meaning to raising or lowering notes by a number of

half-steps (figure: 42). A similar operation shows where notes would be placed if the selection was written in another scale or modularity. This same concept can be applied to a selected chord, mimicking to the theory of borrowed chords [78] in which chords are borrowed from related modalities.

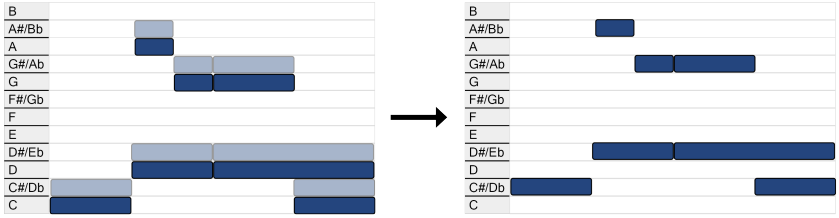


Figure 42: This figure shows all notes being raised by one half step using transposition

A more complex comparison operation draws on the concept of negative harmony [5], in which notes are mirrored in a line drawn through the circle-of-fifths between a tonic and a fifth. With this operation, all notes are essentially negated into their harmonic counterpart, which means that a major interval will become negative and vice versa. This allows the user to explore how a piece will look in its harmonically mirrored version.

An extension to these operations is the option to commit the change, which will perform a conversion operation by moving all original notes to their new positions, rewriting the piece.

Lastly, comparison operations should allow the user to compare pieces by either overlaying one piece with another or viewing them in two sets of coordinated views, as mentioned in the Analysing section.

3.5 INTERACTION DESIGN

This last section of the concept chapter will describe the interaction design of the concept.

Central to the interaction design is that all reconfiguration interactions are animated, which will allow for clear perception and tracking of notes across the different configurations. The animations also apply all other reconfigurable elements: semi-transparent lines, axes, measure lines, bar lines, etc. Next, the interaction should aim to reduce the Decision costs and Physical-motion costs as much as possible [23]. In order to do so, the concept will draw inspiration from everyday interactions used on desktop and laptop devices.

Selection interactions can be carried out in two ways, by either clicking or bounding box selection. To select more than one element by clicking, the user needs to press ctrl+click. Elements are unselected by clicking outside the selection or by ctrl+clicking a selected element again. Using matching criteria for selection is also an option. To do this, checkboxes are provided in a filtering menu at the top of the screen. Selecting all notes in a piece is done by pressing ctrl+a.

To view/hide harmonic relations, a toggle button is provided for each of the three harmonic categories. It can also be done by clicking the alternation button: hiding chord relations is alt+c, hiding interval relations of subsequent notes are alt+s, hiding harmonic trajectories is alt+t. The toggle function is performed globally if nothing is selected and locally if a group of notes is selected.

Zooming in or out is achieved by clicking and stretching an axis to the desired scale. Panning is done by placing the cursor in the side of the view that should be navigated to. The further to the side, the faster the panning.

Reordering of the y-axis should be allowed through right-clicking the axis and choosing the pattern to sort by. Additionally, a small burger icon should indicate that the axis is interactive through list selection.

Computational operations are done via clicking the corresponding buttons in the control interface. The default sorting operations are done in the same manner but must be applied to a selection. Last, the user will have the option to draw a custom trajectory that notes or chords should conform to. This happens through selecting a group of notes and clicking the custom-sort icon in the control interface. The cursor will then transformed into a pen tool, so the user can draw the desired trajectory within the bounding box.

Transposing, inversion, or negation of harmony, along with changing of modes, is done via clicking a representative icon in the control interface - or by pressing T (transpose), I (invert), N (negate) or M (modulate). Upon this action, the user can then view the different comparisons by selecting a group of notes and scrolling with the mouse wheel. A user can e.g., press T, select a group of notes, and scroll the mouse-wheel up one tick to transpose the notes up one halftone. The user can also press I, select a group of chord notes, and scroll down to invert the chord. This will cause a 'walking spring' effect, meaning that every inversion will move the highest note to the bottom, until the the chord is back to its normal inversion one octave lower. When changing modes, the nearest mode will be one scroll up,

while the furthest away of the seven modes will be one scroll down. If the user has clicked the N key, the note or chord selection will be negated with every scroll.

4 IMPLEMENTATION

This chapter will describe the proof-of-concept implementation of the unit visualization. The description will touch on which parts of the concept were implemented and which technologies, design patterns, and algorithms were used to implement them.

4.1 TECHNOLOGIES

The visualization was built in web-based markup- and programming languages: HTML, CSS, and JavaScript - including the D3 library. The D3.js allows for highly customizable visualizations through data-driven transformations of the DOM elements. The data that can be bound to DOM elements can be anything from CSV or JSON to a plain array of data types or objects [9]. Like other query-based libraries, such as JQuery, selections can be targeted through any valid CSS-selectors such as classes and IDs, which also makes it easy to work with animations and changing styling. For these reasons, and for the overall flexibility and speed of D3.js, it was a good fit for creating a highly reconfigurable and animated visualization.

4.2 DATA-FORMAT

The data used for the visualization was originally retrieved in musicXML-format from the MuseScore, an online service for sharing sheet music [33]. The musicXML was converted to JSON via an online XML-to-JSON converter for two practical reasons: first, JSON is easy to work with in D3.js, which was the library of choice for building the prototype. Second, it was, anecdotally, easier to overview the hierarchical structure of the data, which would ease the workflow of building an application around it. Once conterved, the data was arranged in a hierarchy like this:

```
{
 "@id": "P1",
 "measure": [...
     "attributes": {
          "divisions": "4",
          "key": {
             "fifths": "1"
          },
          "time": {
```

```
            "beats": "4",
            "beat-type": "4"
         },
         "clef": {
            "sign": "G",
            "line": "2"
         }
      },
     "note": [
      {
          "rest": [],
          "duration": "4",
          "voice": "1",
          "type": "quarter"
      }...
```


The data was profiled before use. One assumption was that the data that was accessible on MuseScore followed the conventions of the musicXML standard, as the website itself was able to parse and represent the data correctly as sheet-music. This assumption was made in the lack of a better tool validating musicXML. Another key consideration was if the data was correctly transcribed. This was checked by first listening to the transcript in the MuseScore audio player, and then skimming the transcript for errors. Potential errors could be the following: Using too many accidentals. This suggests that the song is transcribed in the wrong key; Using too many 8th, 16th, and 32th notes. This suggests that the tempo is not correctly notated; Using a wrong time signature and notating chord names inconsistently with the actual notes played.

The best musicXML files for the visualization were correctly transcribed, intermediate-level pieces containing melody, harmonic, and information chord names. The visualization prototype took offset in a musicXML transcript of *New Light* by John Mayer.

4.3 TECHNICAL IMPLEMENTATION

Following features were implemented, to demonstrate the most diverse palette of the interactions described in CONCEPT:

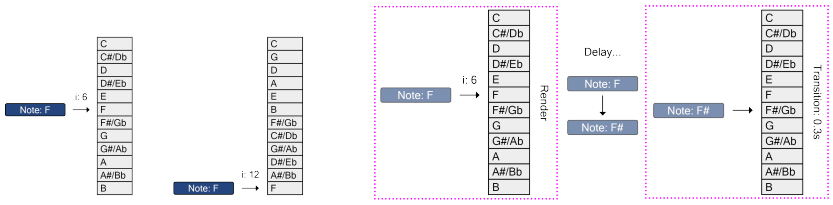
- Conversions: reordering the y-axis into chronological order and line-of-fifths
- Comparisons: view transposition, other modality, and negative harmony
- Computations: aggregating notes and intervals into histograms

4.3.1 *Positioning notes on the y-axis*

Note-objects in the musicXML standard contain an attribute called '@default-y' which is used to position notes in correspondence to the staffs of a sheet. However, since MN does not show pitch class-distance equidistantly like the visualization was supposed to, these values were overwritten on load.

This new '@default-y'-value was calculated by matching the pitch and octave attributes of a note object to an ordered array of pitch classes to get its index. The index was then multiplied by a variable y_axis_scale, which could be adjusted to the scale which the user found suitable. The reason for this design choice was straightforward – if the notes needed to be sorted differently, their index could be looked up in another array. Furthermore, if notes were assigned a new pitch or octave, they could simply be sorted due to the desired order again. So when, e.g., transposing a note, its current index was looked up in an array of pitches and stored in an index variable. The given number of half steps was then added or subtracted from the index, and the associated pitch of the new index was lookup up. Once the new pitch had been assigned, the index could be used to render the note object at its new position.

Transposing, thus happened in two steps: assign a new pitch to the note object, then lookup the index and move the note to the new position. Assigning the new pitch happened as such: the current index was looked up in an alphabetically sorted array of pitches, then stored in a variable, then the number of half steps to transpose the note were added or subtracted, and then the new pitch was looked up in the same array using the new index. A generic function for calculating the y-position was then called.



(a) Sorting: The note has index 6 in the default array, and index 12 in array ordered by fifths.

(b) Transposing: note is first rendered, then assigned new pitch and moved to new position

Figure 43: Matching note objects to array of the desired order for y-positioning

The same two steps were required for modulating notes objects. Here assigning the new pitch happened as such: An array with all

notes within the current mode was constructed using the current key, the interval pattern of the current mode, and the alphabetically sorted array of pitch classes. Then an array for the new mode was constructed in a similar fashion. The index for each note was then looked up in the current mode-array and used to locate the corresponding note in the new mode-array. Once the new note was assigned, the same generic positioning function was called.

```
for (j = 0; j < 7; j++) {
        if (d.pitch.step == previous_mode_notes[j]) {
            d.pitch.step = new_mode_notes[j];
        }
    }
    return update_y_pos(d);
}
```

For negation, two arrays were constructed, each representing one half of the circle of fifths (figure: 44). Then the same logic of matching the two arrays was used.

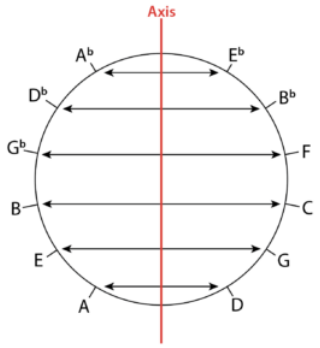


Figure 44: In negative harmony, notes are mirrored in an axis drawn between the key and the fifth of the key. Image source: [47]

```
for(i = 0; i < 6 ; i++){
    right_half.push(circle_of_fifths_array[clockwise_index]);
    clockwise_index = (clockwise_index+1)%12;
}
for(i = 0; i < 6 ; i++){
    left_half.push(circle_of_fifths_array[counter_clockwise_index]);
    counter_clockwise_index = (counter_clockwise_index > 0) ?
    counter_clockwise_index-1:(counter_clockwise_index-1)+12;
}
```

Once the arrays had been constructed, the same logic of matching the two arrays was used.

```
for (j = 0; j < 6; j++) {
```

```
    if (d.pitch.step == right_half[j]) {
        d.pitch.step = left_half[j];
    } else if (d.pitch.step == left_half[j])
        d.pitch.step = right_half[j];
    }
return update_y_pos(d);
```

4.3.2 *Animated transitions*

Transitions were very straightforward, as D3.js allowed for procedural chaining of functions in which a transition could simply be integrated.

```
sheet.selectAll(".note")
    .attr("y", d => { //old position })
    .transition()
    .duration(300)
    .attr("y", d => { //new position})
```

The challenge was to transitions reversible, as overwriting dataset attributes would obviously complicate this. The solution was inspired by the reference model-design pattern, described in [17], in which the DataSource, the DataSet, and the Visualization are treated as separate (figure: 45). Operations that merely affected the visualization, such as sorting and aggregation, were only allowed to modify attributes DOM-elements. Operations that changed the actual data, such as transposition, modulation, and negation, were allowed to modify the dataset that DOM elements referred to.

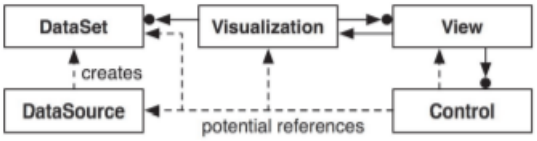


Figure 45: The reference model is an extension to the MVC pattern, but the view is partitioned into a DataSource, a Dataset and the Visualization. Image source: [17]

4.3.3 *Rendering harmonic relationships*

The semi-transparent lines were constructed in a similar way to the notes, except there was a check for whether a previous note, with an equal x-value existed, as there are, e.g., only one interval between two notes and two intervals between three notes, etc. The height was calculated as the difference between the '@default-y' attribute of the current note and the '@default-y' of the previous note as musicXML. Notes within a chord needed always to be sorted by descending

'@default-y' for the harmonic relations not to overlap. This order was also maintained, to avoid problem of descending intervals illustrated in (figure: 11) of the Data content section.

The color encoding of the semi-transparent interval lines was achieved using the production rule, also described in [17], in which a series of if-else rules are used to assign an encoding. The intervals between two given notes were first calculated by subtracting their index numbers. Then the color scheme was assigned based on the interval. Three color schemes could be toggled between: one scheme rendering consonants (grey), majors (blue), minors (yellow) and dissonance (red), one scheme rendering either consonance or dissonance, and one scheme rendering only majors or minors.

5 EVALUATION

The first part of the evaluation was a hybrid of a semi-structured interview and walkthrough demonstration as described in [24]. The benefit of keeping the interview style semi-structured was that it allowed the flexibility of going off script, when the test person would pull the topic in an interesting and unanticipated direction due to his competence within the field. The benefit of doing a walkthrough demonstration was that the focus was on the utility of the visualization and not the usability, which had not been exploring in-depth with the prototype.

5.0.1 *Evaluational interview with Jacob*

Jakob is 25 years and has about 18 years of musical experience. He is currently in the third year of his conservatory degree in classical piano. Besides his education, he is active as a freelance organist and plays in a psychedelic rock/post-punk band. The piano has been his main interest throughout the years, but he also has experience with playing organ and guitar. For musical production, he has experience with the DAWs Ableton and Logic. For transcribing music notation digitally, Sibelius and PriMus. He mainly reads music using MN and chord notation.

Presented to the default presentation of the visualization, Jakob was asked if he was experienced with reading piano roll notation: *"No, I have no experience with reading piano-roll, but I have seen it being used in YouTube videos."* Even though he had no experience with the notation, he could easily see the utility of zooming and panning functionality: *"it is good for getting an overview of the piece."*

Asked about the semi-transparent lines, showing harmonic relationships, Jakob remarked that: *"if you are not that proficient in chords, it can give a good insight into the intervals that the chords consist of,"* and later elaborated: *"especially with the dissonant intervals, it makes a lot of sense."*

Jakob was initially was a bit confused by the reconfiguration of the y-axis into a line-of-fifths: *"I think you really need to be into the circle-of-fifths to make sense of this."* He was then shown the aggregation technique that arranged all nodes in a bar chart. This made more sense to him: *"Oh okay, that is actually smart, if you want to know which tone*

you are in." I pointed out that this could be helpful both for a simple musical piece and for a more complex one, and he remarked: *"Yes, I was actually thinking the same thing because if it, for example, starts in one tonality here, then, via this [the reconfiguration], you can read off that it is moving towards something else [another key]."*

Jakob was shown the concept of aggregating chords by harmonic similarity on a sketch as this had not been implemented: *"In regards to this piece specifically, I don't think it would make that much sense, but if maybe had something... maybe a jazz piece for example - where there can be a lot of different augmentations of the same chord in some way - then it would make a lot of sense. Because then you can say, now I have to learn this part... and then C can occur in all these different configurations. Then I can just quickly familiarise myself with... I have this chord, and I have this augmentation, and it can have an 'add 9', maybe also an 'add #11' or something... in that way it would make sense, if you maybe had to learn all chords before starting to play the piece."*

The next functionality discussed was modulation: *"I think this could be very fun to use - if you, for example, want to make a piece sound more old-fashioned, like folk music."* He thought the semi-transparent lines could serve as a good learning tool: *"It would make sense if you had to teach someone the modes, then you could present them a piece and say, now we rewrite it into Dorian mode, and in that case, it is these specific intervals that are changed, because it has to fit into this mode. This could be very nice for a learning situation."*

Jakob related the negation functionality to an activity within his education: *"I have tried this [writing negating harmony] somewhat before in the context of songwriting. In these situations, you often impose different obstructions on yourself to see if something new will come out of this. In this context, it could be fun to just flip it all the way around, and maybe you could pick out some parts that sound good and use them in the songwriting. It could be used as a tool to invest some new sounds, and it could probably also be fun to just play the whole piece in negation."* Jakob was, however, not able to perceive the axis around which the notes were flipped.

Asked about the utility of the visualization in general, Jakob answered: *"If I were to teach someone how to analyze a piece, it would be really nice to present it like this. I don't know if it is easy to play along with, but that might be because I simply haven't practiced this notation. I could also imagine that it would be really nice to learn if you know that you have to work a lot with music software like Ableton because if you have to, for example, make melodies in a MIDI track, then it looks exactly like this. Then you have a much better overview of when you have to transfer it"*.

When asked what level of musicians he saw it fit for: *"I think, for classical musicians, like I am, and on a somewhat professional level, then it would be a bit frustrating to suddenly have to read music in a new way, but I think for many novices, it would make sense to learn it this way, and also on an intermediate level it would be a good supplement, to get a different kind of overview."*

Lastly, I asked him if he saw some other opportunities to visualize: *"In classical music.. then it could be nice. In, for example, music from Wiener Classicism and Baroque and Romanticism - here it might get a little spicier in regards to harmony - but here it could be nice to view every time there is a tonal cadence, where you go from the fourth step to the fifth step to the first step, so whenever there is some kind of cadence, this would be nice to emphasize. And it can be used when a piece changed tonality. Then you can see how the piece actually transitions to that new tonality."*

After the camera was off, Jakob added that it would also make sense to display info about the harmonic function of chords.

5.1 INSIGHTS FROM THE VISUALIZATION

This section will exemplify some of the utility of the visualization through representations of different songs.

5.2 AGGREGATED NOTES SORTED BY FIFTHS

Investigation how *New Light* by John Mayer (figure: 46) distributes over a bar chart, it is clear to see that notes are heavily distributed around the note G, at least in the lower two octaves. Another heavily emphasizes note is D, which sitting a fifth above G - this is a clear indication of the tonality being G major, which also makes sense due to most notes played appearing in this scale: G, A, B, C, D, E, F#/Gb, while the notes B#/Ab and D#/Eb, from outside the scale only appear in a total of 9 times during the piece. Interestingly, only three notes appear in the lowest octave: G, A, and B. Many guitarists with a background in blues will keep the basslines simple, as their fingerpicking style has the thumb allocated for the three lowest strings, while their index and middle finger are used for the three brighter strings [57]

Looking at the piano transcript of the song *Garden* by SZA (figure: 47), the octave span is wider, which is allowed by the piano having a wider pitch range than guitar. The piece is in A major, and A is also very dominant in the lower three octaves. Looking at the duration of notes in the bar chart, most notes are short, while A, B, E, span over a longer duration. This tells that SZA will emphasize the first, second and fourth degree of the key, while increasing the pace between these

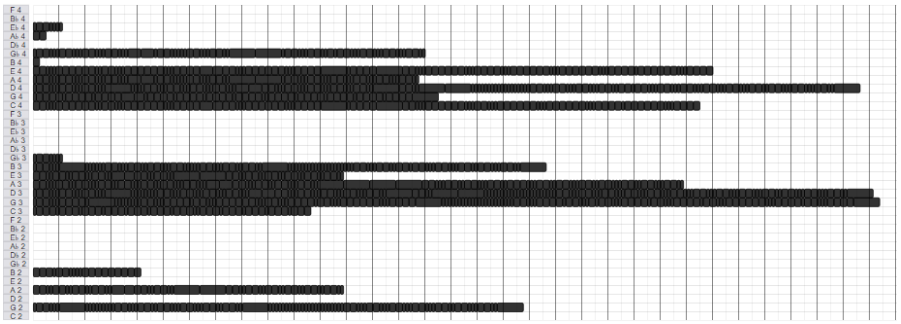


Figure 46: Acoustic guitar voicing from transcript of John Mayers *New Light*. MusicXML retrieved from [54].

between these central notes. This fits well into her modern R'n'B style, in which the phrasing tends to mix rapping and singing. The note distribution additionally tells that the song adheres strictly to the seven notes within the key.

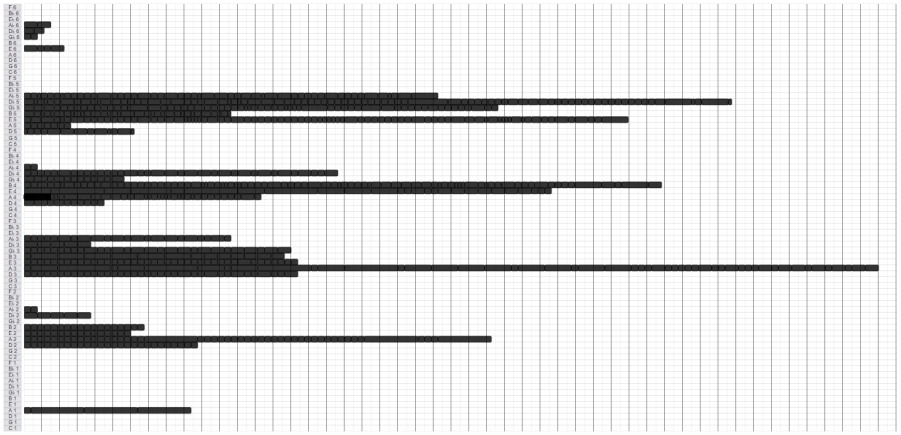


Figure 47: Piano voicing from transcript of *Garden* by SZA. MusicXML retrieved from [53].

The third bar chart shows a transcript of a section of *Claire de Lune* by Claude Debussy, transposed to C major. The note of C, however, occurs less than its fifth, G, and its third, E - this may be due to the fact that the transcript is incomplete. Harmonically, this piece is quite complex as it contains many out-of-key notes. The fact that bar chart almost looks like normal distribution curve when sorted chromatically, is also interesting.

The last piece is an atonal piece composed by the MuseScore user Hudson Holland (figure: 50). The piece is unsurprisingly a lot more flatly distributed along the pitch axis. However, some notes occur more seldom than others, such as Bb. It is hard to derive anything meaningful from this view, despite the fact that there is no clear harmonic focal point, which will make it sound very dissonant.

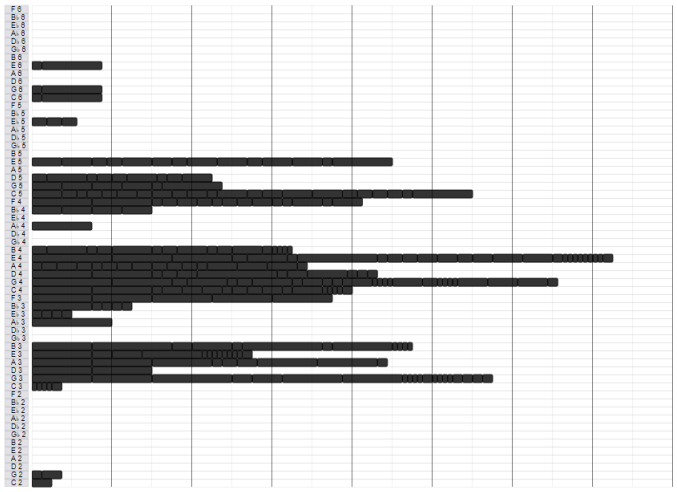


Figure 48: Piano voicing from transcript of *Claire de Lune* by Claude-Achille Debussy. MusicXML retrieved from [52].

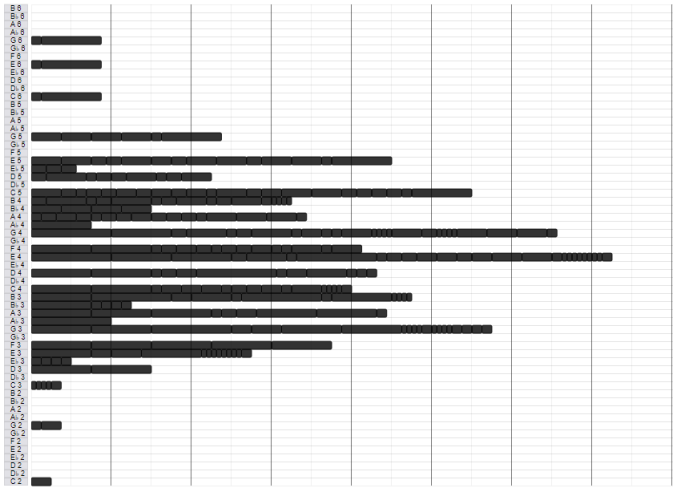


Figure 49: *Claire de Lune* with chronologically sorted notes. MusicXML retrieved from [52].

5.3 EXPLORING INTERVALS

For gaining an overview of the intervals used in *New Light*, the aggregated interval representation is useful (figure: 51). This bar chart reveals that the major third, minor third, perfect fourth, and perfect fifth intervals occur almost the same amount of times throughout the piece. Compared to the default view, it is an obvious result of how John Major constructs his chords: minor chords are usually constructed by stacking a P5, P4, m3, and an M3 interval on top of each other; major chords are almost identical except for the m3 and M3 intervals being swapped. An augmented chord, B7, is also used. This is constructed the same entirely the same as a major chord, but with an additional m3 interval stacked on top to augment the chord with

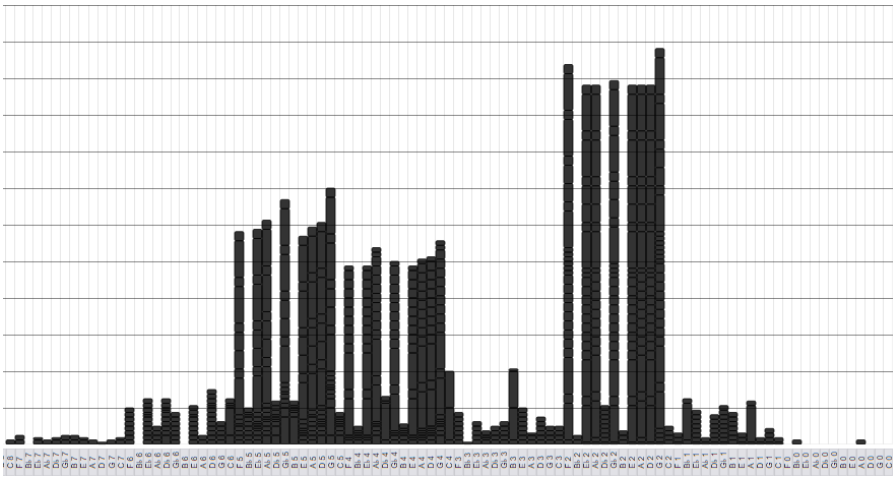


Figure 50: Piano voicing from an atonal piece, *Prelude (Atonal)* by Hudson Holland. MusicXML retrieved from [55].

a minor 7 from B (figure: 52 a).

Revisiting the aggregated interval view, another often utilized interval is the m6. Comparing this to the default representation, the m6 usually occurs as a substitute for the bass note of the C chord when it is about to descend to the A minor chord, creating an intermediate step between the two chords. It does, however, also appear in an A minor chord once, thus creating a Am7 chord (figure: 52 b).

Lastly, the song only contains a single instance of a dissonant note. This is a tritone that occurs at the end of an ascending build-up that leads to a chorus.

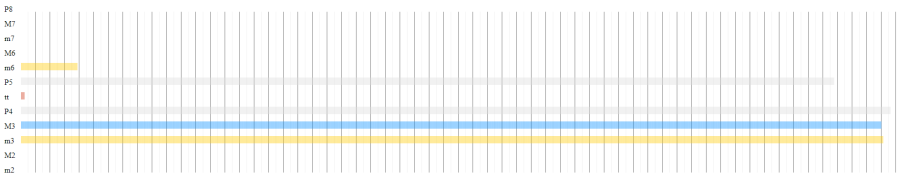
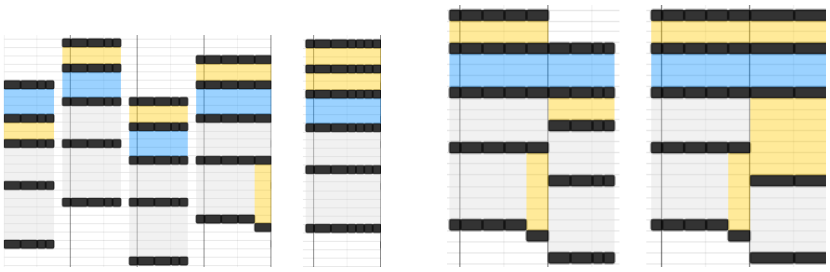


Figure 51: *New Light*. MusicXML retrieved from [54].

Claire de Lune uses intervals very differently than the previous example (figure: 53). There are very few perfect intervals - almost all intervals are either major or minor. Also, considering the short length of this transcript, the dissonant M2 interval also occurs quite a lot within the piece.

Inspecting the default view, it is apparent that all of the harmonies consist solely of two tones played simultaneously. These can not be considered chords, as a chord contains at least three notes, but they will resemble the chord-qualities of major and minor by keeping the



(a) Chords: Am, D, G, C and B7

(b) Minor sixth used as intermediate bass note and in Am7 chord

Figure 52: Chords constructed with P4, P5, m3, M3 and m6 intervals

essential sounds of these basic triadic chords - the root and the third (figure: 54)

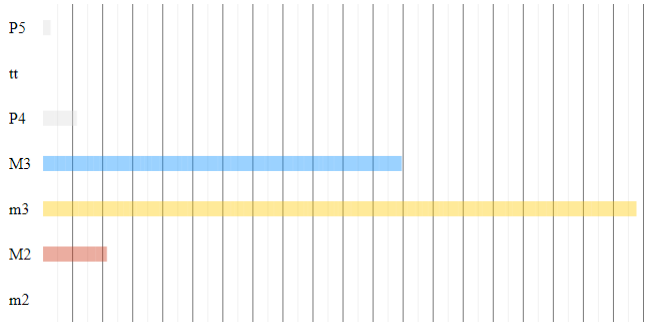


Figure 53: Barchart showing all intervals that occur in *Claire de Lune*. MusicXML retrieved from [52].

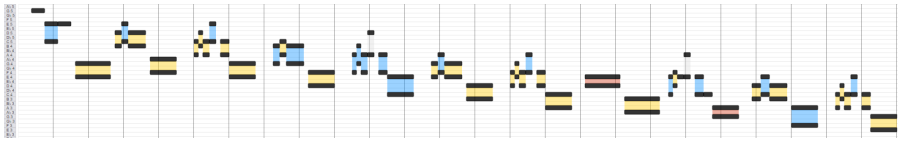


Figure 54: Default representation of *Claire de Lune*. MusicXML retrieved from [52].

5.4 EXPLORING ALTERNATIVE HARMONIES

Transposing was quite straightforward, and little new insight outside being explicitly able to perceive the movement from an old pitch to a new. (figure: 55).

The modulation and negation feature supported reharmonization easily and allowed an overview of how reharmonization affected the harmonic structure. From (figure: 55), it is easy to see that the Ionian, Dorian, Mixolydian, and negative Harmony reconfigurations result

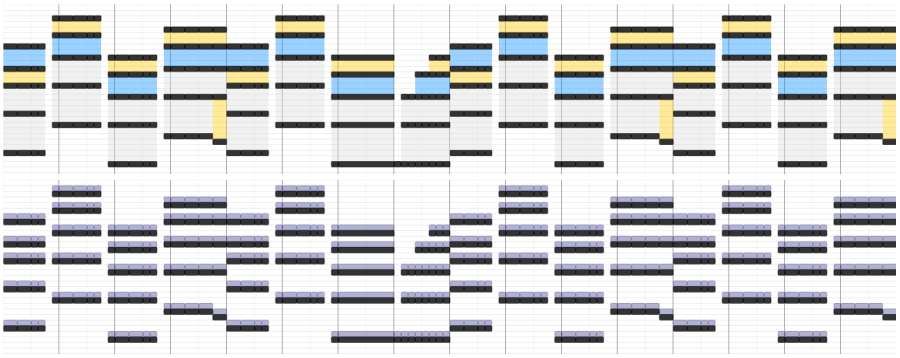


Figure 55: New Light transposed by one from G to G#/Ab. The semi-transparent notes are animated such that transition away from the previous position each time the input is incremented.

in the most consonant configurations of notes.

From (figure: 56), a problem sticks out: the negative harmony view does not show entirely the opposite intervals to the Ionian mode. This highlights a design problem, as all notes are actually correctly mirrored to their negative counterpart within the octave. Nevertheless, since the order of the notes is shuffled, the note of A is no longer compared to C, and the minor third interval between them is therefore not displayed. This will be elaborated on further in the discussion section. Audio clips with the melodic line included are provided below.

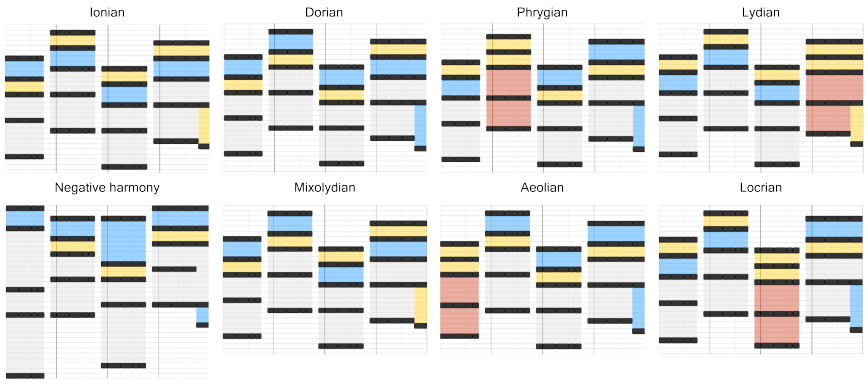


Figure 56: The Ionian, Dorian, Mixolydian modes, along with the negative harmonization of the Ionian mode, all have no dissonance. On the opposite Phrygian, Lydian, Aeolian and Locrian modes have introduced more dissonance to the piece.

New Light - Ionian | New Light - Dorian | New Light - Phrygian | New Light - Lydian

New Light - Negative | New Light - Mixolydian | New Light - Aeolian | New Light - Locrian

New Light - Negative - Octaves Adjusted

Lastly, looking at the different configurations of *Claire de Lune* (figure: 57), we can see that most configuration add little additional dissonance to what was already there. However, many of the intervals have changed from major to minor and vice versa. Audio of the first few measures is included below.

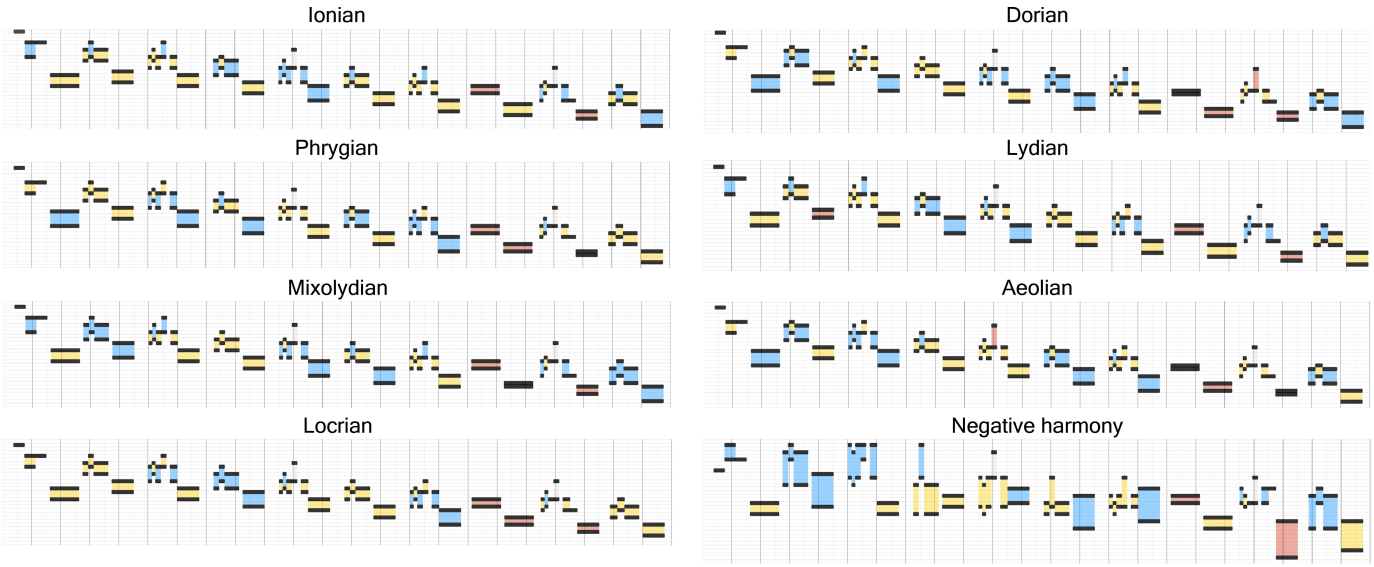


Figure 57: *Claire de Lune* harmonized in eight different ways.

Claire de Lune - Ionian

Claire de Lune - Dorian

Claire de Lune - Phrygian

Claire de Lune - Lydian

Claire de Lune - Mixolydian

Claire de Lune - Aeolian

Claire de Lune - Locrian

Claire de Lune - Negative

Claire de Lune - Negative - Octaves Adjusted

6 DISCUSSION

6.1 DISCUSSION POINTS FROM WALKTHROUGH DEMONSTRATION

From the walkthrough demonstration with Jakob, a clear shortcoming to him was the lack of compatibility with music notation. This was expected, as he has spent 18 years using the traditional notation style, but nonetheless, an obstacle if a professional musician should adopt the software. While the choice of making the default representation resembles the piano roll notation, it is worth speculating if the units can be configured in a way that more closely resemble that of MN. One compromise could be providing the possibility of an MN-like five-staff layout while still using the note markers from the piano roll notation. This would allow for easy transitions back and forth between the two styles and would not conflict with the unit visualization technique. An additional encoding would, however, need to be added in order to show the accidentals (figure: 58).

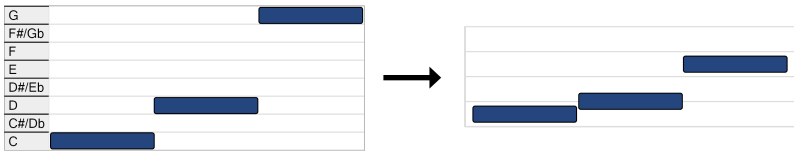


Figure 58: Adapting the unit visualization to the five-staff layout of MN.

As described in 5.0.1, Jakob did not find the ordering-by-fifths configuration particularly helpful unless it was used in combination with the bar chart configuration. Since there is no valid use case for the single operation, the two operations may be combined into one (conversion + computation). As the transition from the default representation involved two steps, the conversion, and the computation, a staged transition may be the most expressive way to animate the path of the notes.

Jakob also expressed that he would like the bar chart representation to reflect if a piece was moving from one tonality to another. As the current function aggregates all notes for the entire piece, this could be improved upon. One solution is to make the bar chart representation animated, such that the bar char stacks up the notes in real-time as the music plays - notes could also be made to disappear at a certain time after being played for a more accurate snapshot of the current time frame. This solution would have many similarities to the visualization proposed by Mardirossian and Chew in [30]. Another solution

would be to have two coordinated views - one with the default view and one with the bar chart - and connecting them through linking and brushing.

When demonstrating the configuration of harmonic negation, Jakob was not able to perceive the axis around which the notes were flipped. This could be due to the line not being emphasized. A solution to this could be to highlight the axis while the transitioning is happening. Another reason could be that the animations implemented in the visualization were limited to 2D. If the notes were actually flipped in 3D with around the x-axis, like flipping a page in a book, this might have been easier to understand perceptually. One last reason may be that the notes were flipped within their respective octave. This requires the user to keep track of several pivot axis. This solution will be discussed more in-depth in section 6.2.

Jakob suggested that the visualization could a good fit for presentational context like teaching, in which he was to illustrate new concepts to novice musicians. This is what. He was also enthusiastic about using it in the context of digital audio production for idea generation. A seamless way for integrating the visualization into such a workflow would be through providing it as a Digital Audio Workstation plug-ins, similar to that of Liquid Notes [38].

Lastly, Jakob remarked that he would wish two other features from the visualization. The first was support for the task of identifying cadences, and the second was support for the task of identifying the harmonic function of chords. Both of these features have been successfully covered by Malandrino et al. [28], and support for the latter of these tasks was also envisioned for the concept as described in 3.2. Representing the cadences could be incorporated into the current concept as part of the chord labeling if it were to be displayed in symphony with the notes. An effective and reconfigurable way of encoding this relationship would, however, need to be suggested.

6.2 DISPLAYING INTERVALS

As apparent in the negated harmony shown in (figure: 56), there is an unsolved problem in regards to which intervals should be shown. Technically, the 'negate harmony'-algorithm worked correctly in the fact that it mirrored the notes to their correct counterparts. But since each note was mirrored within their respective octave, the order of the notes was swapped, resulting in the visualization displaying different intervals than anticipated - see (figure: 59)

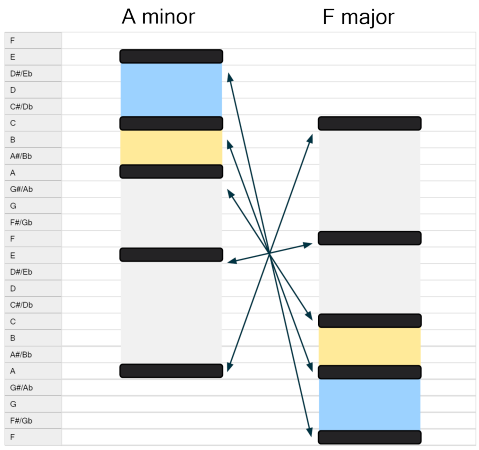


Figure 59: Here notes of all octaves are flipped around the same axis, resulting in a reverse order of the intervals.

So, what is most interesting to show? The interval between its a given note and its predecessor, the interval from the root note to all other notes in the chord, or the intervals between any given combination of notes within a chord? The answer might be all of them. Intervals between adjacent notes may support analyzing how chords are structured with the harmonic building blocks. Showing intervals from the root may help to identify the chord quality quickly. Comparing all intervals within the chord may actually give the most in-depth overview of the harmonic relationships. Most of this could be accommodated by allowing the user to select coloring options (figure: 60). For comparing all intervals within a chord, other configurations would be necessary (figure: 61).

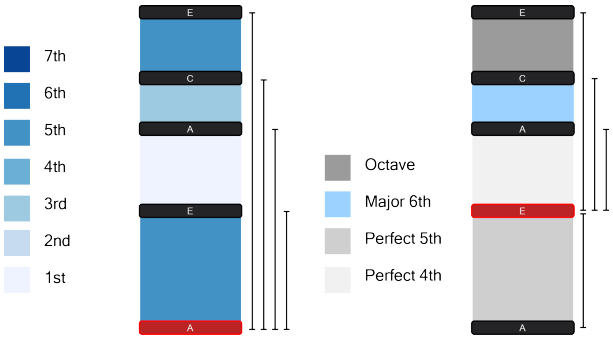


Figure 60: On the left, a continuous color scale for identifying the ordinal scale degrees of the key of A. On the right, an concept for selecting an arbitrary note within a chord, and viewing all intervals from that note.

Figure 61: The edge-centric heatmap could make for easy comparison of all intervals within a chord. This could be constructed by the units of the visualization, i.e. using notes along the axis. More color options may be necessary to show intervals larger than an octave.

6.3 TECHNICAL SOLUTION

MusicXML was chosen as the data source, since it was easy to retrieve good quality transcriptions in the format, and since the structure of the data was easy to understand. The main downside was that it is intended for sheet music, and therefore objects like notes needed to be repositioned, while other objects like rests needed to be removed entirely. The MusicXML was, as described in Data-format, converted into JSON, so that it was easier to work with. This decision was not made taking into account that it would make sense to export an altered version of the MusicXML, that could be used in other software supporting the format.

Other formats like MIDI requires parsing from its initial 8-bit format into, e.g., JSON. However, once converted, the JSON is very similar to that created from musicXML, but with the advantage that it is not made specifically for sheet music and that it has a wider range of information on e.g., velocity and pitch bends. A short example can be seen below:

```
{
  "instrument": {
    "family": "piano",
    "name": "acoustic grand piano",
    "number": 0
  },
  "name": "Basic",
  "notes": [
    {
      "duration": 2.03125,
      "durationTicks": 390,
```

```
      "midi": 62,
      "name": "D4",
      "ticks": 0,
      "time": 0,
      "velocity": 0.7480314960629921
    }...
  ],
}
```

Tonejs.github.io [82] offers a converter for MIDI format into JSON, and a library that actually allows for playing music in the browser. Last, the MIDI format is supported by a wide array of DAWs and other software, compared to musicXML. These points suggest that MIDI might have been a better data source for this kind of visualization.

6.4 FUTURE WORK

This thesis has been concerned with solving some specific musical tasks using reconfigurable unit visualization. However, a lot of potential still remains within this topic. Further research in this direction should focus on experimental reconfigurations that take remove the units entirely from the context of the piano roll view, to see if new and informative insight can be gained from this.

Adding the modality of audio may also improve vastly on the effectiveness of the visualization as the main data type is sound-based. Providing a combination of visual and sound will make it easier for the users to orient themselves in the piece, understand the concepts and use the visualization for creative tasks. Pragmatically, this would also make the software more attractive to, e.g., electronic music producers, as Jakob also highlighted in the walkthrough.

While animations have been used in a lot in this concept in this thesis, it may make sense to add even more animations for communicating the information. An example is the bar chart, which could benefit from being linked to the temporal progression of a piece so that the user can track if the music is actually transitioning into a new tonality. Further research could look into whether more views could benefit from animating the temporal progression.

7 CONCLUSION

This thesis has examined how the complex body of MT can be made more accessible through data visualization. To achieve this, a data abstraction was initially performed to bridge the understanding of readers and to understand better the underlying data better before designing the visualization on top of it.

Thereafter, the survey of current research within MTVs was conducted. The survey categorized 32 contributions by the interaction intents which they supported: non-interactive, selection, exploration, reconfiguration, encoding, abstraction, filtering, and connecting. The works were further categorized from their primary focus within music: harmony, rhythm, melody, expression, and structure. The survey identified that nearly half of the contributions focused on non-interactive representations that either aimed at supplementing or replacing the traditional music notation. Among the works that supported interaction, there was much room for exploring all categories of interaction intents, but especially reconfiguration and selection were underexplored. Among the musical categories, rhythm and melody were the least explored topics, with the reservation that there are a vast amount of subtasks to support each of these topics. Lastly, no works have tackled the problem of MTV with the technique of unit visualization.

The thesis builds upon the current research by proposing a unit visualization that allowed for exploring harmonic concepts in musical pieces through reconfiguration. To develop the conceptual framework, first, an anticipated use-case was laid out based on Bloom's Revised Taxonomy for Music Education, which proposes concrete objectives important for learning music. Then, several reconfiguration concepts were proposed based on the framework of smoothing operations, which describe ways of manipulating data that are fundamental to exploratory analysis. Specifically, the operations of interest were: conversions, constraints, chunks, computation, and comparison.

Lastly, a prototype of the unit visualization was implemented with functionality covering four of the interaction intents: conversions, constraints, computation, and comparison. A technical chapter described the implementation and its challenges. Finally, the prototype was evaluated with Jakob, who is a conservatory-educated musician, and

specific findings from exploring the visualization were discussed.

Jakob found the visualization a good tool for idea illustration and generation - especially if used together with a DAW. As a classical musician, he expressed the need for a configuration that was closer to traditional music notation. He also expressed a need for supporting the labeling of cadences and harmonic functions. The visualization revealed interesting insights into the different composition, especially the reharmonization tools (modulating and negating harmonies) seemed promising from the perspective of exploring and rewriting music.

While this thesis presents a range of concepts for a reconfigurable unit visualization, a lot is yet to be explored. Further work should focus on developing even more reconfigurations that can reveal a deeper insight into this field.

ACKNOWLEDGMENTS

Many thanks to my supervisor, Hans-Jörg Schulz, for constructive guidance and to Jakob Ferdinand Hansen for participating in the evaluation.

Also many thanks to my fellow students for checking in once in a while during lockdown. And to my family, friends, dorm mates and colleagues for moral support.

BIBLIOGRAPHY

[1] Tony Bergstrom, Karrie Karahalios, and John Hart. "Isochords: visualizing structure in music." In: Jan. 2007, pp. 297–304. DOI: 10.1145/1268517.1268565.

[2] Gabriel Dias Cantareira, Luis Gustavo Nonato, and Fernando V. Paulovich. "MoshViz: A Detail+Overview Approach to Visualize Music Elements." In: *IEEE Transactions on Multimedia* 18.11 (2016), pp. 2238–2246. DOI: 10.1109/TMM.2016.2614226.

[3] Wing-Yi Chan, Huamin Qu, and Wai-Ho Mak. "Visualizing the Semantic Structure in Classical Music Works." In: *IEEE transactions on visualization and computer graphics* 16 (Jan. 2010), pp. 161–73. DOI: 10.1109/TVCG.2009.63.

[4] Winnie Wing-Yi Chan. "A Report on Musical Structure Visualization." In: 2007.

[5] Samuel Chase. *What Is Negative Harmony?* https://hellomusictheory.com/learn/negative-harmony. 2021 (accessed May 10, 2021).

[6] Min Chen, Luciano Floridi, and Rita Borgo. "What Is Visualization Really For?" In: *The Philosophy of Information Quality*. Ed. by Luciano Floridi and Phyllis Illari. Cham: Springer International Publishing, 2014, pp. 75–93. ISBN: 978-3-319-07121-3. DOI: 10.1007/978-3-319-07121-3_5. URL: https://doi.org/10.1007/978-3-319-07121-3_5.

[7] Elaine Chew and Alexandre R. J. Francois. "Interactive Multi-Scale Visualizations of Tonal Evolution in MuSA.RT Opus 2." In: *Comput. Entertain.* 3.4 (Oct. 2005), 1–16. DOI: 10.1145/1095534.1095545. URL: https://doi.org/10.1145/1095534.1095545.

[8] Peter Ciuha, Bojan Klemenc, and Franc Solina. "Visualization of concurrent tones in music with colours." In: Oct. 2010, pp. 1677–1680. DOI: 10.1145/1873951.1874320.

[9] D3.js. *D3.js: Data-Driven Documents*. https://d3js.org/. 2021 (accessed May 7, 2021).

[10] Roberto De Prisco, Antonio Esposito, Nicola Lettieri, Delfina Malandrino, Donato Pirozzi, Gianluca Zaccagnino, and Rocco Zaccagnino. "Music Plagiarism at a Glance: Metrics of Similarity and Visualizations." In: *2017 21st International Conference Information Visualisation (IV)*. 2017, pp. 410–415. DOI: 10.1109/iV.2017.49.

[11] Steve Dipaola and Ali Arya. "Emotional remapping of music to facial animation." In: (July 2006), pp. 143–149. DOI: 10.1145/1183316.1183337.

[12] Simon Dixon, Werner Goebl, and Gerhard Widmer. "Real Time Tracking and Visualisation of Musical Expression." In: 2445 (Mar. 2002). DOI: 10.1007/3-540-45722-4_7.

[13] Joyce Horn Fonteles, Maria Andréia Formico Rodrigues, and Victor Emanuel Dias Basso. "Creating and Evaluating a Particle System for Music Visualization." In: *J. Vis. Lang. Comput.* 24.6 (Dec. 2013), 472–482. ISSN: 1045-926X. DOI: 10.1016/j.jvlc.2013.10.002. URL: https://doi.org/10.1016/j.jvlc.2013.10.002.

[14] Alexandre Francois. *MuSA$_R$T*. https://apps.apple.com/us/app/musa-rt/id506866959?mt=12. 2021 (accessed May 7, 2021).

[15] Daniel Fürst, Matthias Miller, Daniel A. Keim, Alexandra Bonnici, Hanna Schäfer, and Mennatallah El-Assady. "Augmenting Sheet Music with Rhythmic Fingerprints." In: *2020 IEEE 5th Workshop on Visualization for the Digital Humanities (VIS4DH)*. 2020, pp. 14–23. DOI: 10.1109/VIS4DH51463.2020.00007.

[16] Wendell Hanna. "The New Bloom's Taxonomy: Implications for Music Education." In: *Arts Education Policy Review* 108.4 (2007), pp. 7–16. DOI: 10.3200/AEPR.108.4.7-16. eprint: https://doi.org/10.3200/AEPR.108.4.7-16. URL: https://doi.org/10.3200/AEPR.108.4.7-16.

[17] Jeffrey Heer and Maneesh Agrawala. "Software Design Patterns for Information Visualization." In: *IEEE Transactions on Visualization and Computer Graphics* 12.5 (2006), pp. 853–860. DOI: 10.1109/TVCG.2006.178.

[18] R. Hiraga, F. Watanabe, and I. Fujishiro. "Music learning through visualization." In: *Second International Conference on Web Delivering of Music, 2002. WEDELMUSIC 2002. Proceedings.* 2002, pp. 101–108. DOI: 10.1109/WDM.2002.1176199.

[19] Rumi Hiraga and N. Matsuda. "Graphical expression of the mood of music." In: July 2004, 2035 –2038 Vol.3. ISBN: 0-7803-8603-5. DOI: 10.1109/ICME.2004.1394664.

[20] Samuel Huron, Yvonne Jansen, and Sheelagh Carpendale. "Constructing Visual Representations: Investigating the Use of Tangible Tokens." In: *IEEE Transactions on Visualization and Computer Graphics* 20.12 (2014), pp. 2102–2111. DOI: 10.1109/TVCG.2014.2346292.

[21] Richard Khulusi, Jakob Kusnick, Christofer Meinecke, Christina Gillmann, Josef Focht, and Stefan Jänicke. "A Survey on Visualizations for Musical Data." In: *Computer Graphics Forum* 39 (Mar. 2020). DOI: 10.1111/cgf.13905.

[22] Yi-Ting Kuo and Ming-Chuen Chuang. "A proposal of a color music notation system on a single melody for music beginners." In: *International Journal of Music Education* 31.4 (2013), pp. 394–412. DOI: 10.1177/0255761413489082. eprint: https://doi.org/10.1177/0255761413489082. URL: https://doi.org/10.1177/0255761413489082.

[23] Heidi Lam. "A Framework of Interaction Costs in Information Visualization." In: *IEEE Transactions on Visualization and Computer Graphics* 14.6 (2008), pp. 1149–1156. DOI: 10.1109/TVCG.2008.109.

[24] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. "Evaluation Strategies for HCI Toolkit Research." In: CHI '18. Montreal QC, Canada: Association for Computing Machinery, 2018, 1–17. ISBN: 9781450356206. DOI: 10.1145/3173574.3173610. URL: https://doi.org/10.1145/3173574.3173610.

[25] Yang Liu, Ni Yan, and Dili Hu. "Chorlody: A music learning game." In: *Conference on Human Factors in Computing Systems - Proceedings* (Apr. 2014). DOI: 10.1145/2559206.2580098.

[26] Giorgia Lupi. *A dialogue between four hands*. http://giorgialupi.com/a-dialogue-between-four-hands-my-ongoing-collaboration-with-kaki-king. Not informed (accessed May 7, 2021).

[27] Jock Mackinlay. "Automating the Design of Graphical Presentations of Relational Information." In: 5.2 (Apr. 1986), 110–141. ISSN: 0730-0301. DOI: 10.1145/22949.22950. URL: https://doi.org/10.1145/22949.22950.

[28] Delfina Malandrino, Donato Pirozzi, Gianluca Zaccagnino, and Rocco Zaccagnino. "A Color-Based Visualization Approach to Understand Harmonic Structures of Musical Compositions." In: *2015 19th International Conference on Information Visualisation*. 2015, pp. 56–61. DOI: 10.1109/iV.2015.21.

[29] Stephen Malinowski. *Music Animation Machine*. https://musanim.com/. 2021 (accessed May 7, 2021).

[30] Arpi Mardirossian and Elaine Chew. "Visualizing Music: Tonal Progressions and Distributions." In: Jan. 2007, pp. 189–194.

[31] Matthias Miller, Alexandra Bonnici, and Mennatallah El-Assady. "Augmenting Music Sheets with Harmonic Fingerprints." In: *Proceedings of the ACM Symposium on Document Engineering 2019*. DocEng '19. Berlin, Germany: Association for Computing Machinery, 2019. ISBN: 9781450368872. DOI: 10.1145/3342558.3345395. URL: https://doi.org/10.1145/3342558.3345395.

[32] Reiko Miyazaki, Issei Fujishiro, and Rumi Hiraga. "Exploring MIDI Datasets." In: *ACM SIGGRAPH 2003 Sketches amp; Applications*. SIGGRAPH '03. San Diego, California: Association for Computing Machinery, 2003, p. 1. ISBN: 9781450374668. DOI: 10.1145/965400.965453. URL: https://doi.org/10.1145/965400.965453.

[33] Musescore. *Musescore*. https://musescore.com/. 2021 (accessed May 7, 2021).

[34] Deokgun Park, Steven M. Drucker, Roland Fernandez, and Niklas Elmqvist. "Atom: A Grammar for Unit Visualizations." In: *IEEE Transactions on Visualization and Computer Graphics* 24.12 (2018), pp. 3032–3043. DOI: 10.1109/TVCG.2017.2785807.

[35] Aura Pon, Junko Ichino, David Eagle, Ehud Sharlin, and Sheelagh Carpendale. "Vuzik: Music Visualization and Creation on an Interactive Surface." In: *In Proceedings of Audio Mostly*. 2011.

[36] Roberto De Prisco, Delfina Malandrino, Donato Pirozzi, Gianluca Zaccagnino, and Rocco Zaccagnino. "Understanding the structure of musical compositions: Is visualization an effective approach?" In: *Information Visualization* 16.2 (2017), pp. 139–152. DOI: 10.1177/1473871616655468. eprint: https://doi.org/10.1177/1473871616655468. URL: https://doi.org/10.1177/1473871616655468.

[37] Roberto Prisco, Delfina Malandrino, Donato Pirozzi, Gianluca Zaccagnino, and Rocco Zaccagnino. "Evaluation Study of Visualisations for Harmonic Analysis of 4-Part Music." In: July 2018, pp. 484–489. DOI: 10.1109/iV.2018.00090.

[38] Recompose. *Recompose: Liquid Notes*. https://www.re-compose.com/liquid-notes-music-software.html. 2021 (accessed May 7, 2021).

[39] Katja Rogers, Amrei Röhlig, Matthias Weing, Jan Gugenheimer, Bastian Könings, Melina Klepsch, Florian Schaub, Enrico Rukzio, Tina Seufert, and Michael Weber. "P.I.A.N.O.: Faster Piano Learning with Interactive Projection." In: *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*. ITS '14. Dresden, Germany: Association for Computing Machinery, 2014, 149–158. ISBN: 9781450325875. DOI: 10.1145/2669485.2669514. URL: https://doi.org/10.1145/2669485.2669514.

[40] Nicholas Rougeux. "Off the Staff: An Experiment in Visualizing Notes from Music Scores." In: *Patterns* 1 (June 2020), p. 100055. DOI: 10.1016/j.patter.2020.100055.

[41] Penelope Sanderson and Carolanne Fisher. "Exploratory Sequential Data Analysis: Foundations." In: *Human-Computer Interaction* 9 (Sept. 1994), pp. 251–317. DOI: 10.1080/07370024.1994.9667208.

[42] Craig Sapp. "Harmonic Visualizations of Tonal Music." In: (Jan. 2001).

[43] Hans-Jörg Schulz, Thomas Nocke, Magnus Heitzler, and Heidrun Schumann. "A systematic view on data descriptors for the visual analysis of tabular data." In: *Information Visualization* 16.3 (2017), pp. 232–256. DOI: 10.1177/1473871616667767. eprint: https://doi.org/10.1177/1473871616667767. URL: https://doi.org/10.1177/1473871616667767.

[44] Ben Shneiderman. "The eyes have it: A task by data type taxonomy." In: *Proceedings of IEEE Symposium on Visual Languages*. Vol. 96.

[45] Jon Snydal and Marti Hearst. "ImproViz: visual explorations of jazz improvisations." In: Jan. 2005, pp. 1805–1808. DOI: 10.1145/1056808.1057027.

[46] Aurea Soriano Vargas, Fernando Paulovich, Luis Nonato, and Maria Cristina Oliveira. "Visualization of Music Collections Based on Structural Content Similarity." In: Aug. 2014. DOI: 10.1109/SIBGRAPI.2014.53.

[47] Dan Tepfer. *Negative Harmony: a primer*. https://dantepfer.com/blog/?p=368. 2020 (accessed June 11, 2021).

[48] Martin Wattenberg. *The Shape of Song*. http://www.bewitched.com/song.html. 2021 (accessed May 7, 2021).

[49] Hiroshi Ishii Xiao Xiao Basheer Tome. "Andante: Walking Figures on the Piano Keyboard to Visualize Musical Motion." In: *NIME*. 2014.

[50] Ji Soo Yi, Youn ah Kang, John Stasko, and J.A. Jacko. "Toward a Deeper Understanding of the Role of Interaction in Information Visualization." In: *IEEE Transactions on Visualization and Computer Graphics* 13.6 (2007), pp. 1224–1231. DOI: 10.1109/TVCG.2007.70515.

[51] Jian Zhao, Michael Glueck, Simon Breslav, Fanny Chevalier, and Azam Khan. "Annotation Graphs: A Graph-Based Visualization for Meta-Analysis of Data Based on User-Authored Annotations." In: *IEEE Transactions on Visualization and Computer Graphics* PrePrints (Jan. 2017), pp. 1–1. DOI: 10.1109/TVCG.2016.2598543.

[52] MuseScore entry. *MuseScore: Clair De Lune - Claude Debussy*. https://musescore.com/score/2701381. 2016 (accessed June 7, 2021).

[53] MuseScore entry. *MuseScore: Garden (Say It Like Dat) - SZA*. https://musescore.com/user/12289476/scores/5156383. 2018 (accessed June 7, 2021).

[54] MuseScore entry. *MuseScore: New Light - John Mayer*. https://musescore.com/user/29980887/scores/6051145. 2020 (accessed June 7, 2021).

[55] MuseScore entry. *MuseScore: Prelude No. 6 (Atonal) - Hudson Holland*. https://musescore.com/score/2701381. 2020 (accessed June 7, 2021).

[56] Stackexchange entry. *Stackexchange: Is 'fundamental frequency' the same thing as 'fundamental bass' (chord root?)* https://music.stackexchange.com/questions/58372/is-fundamental-frequency-the-same-thing-as-fundamental-bass-chord-root. 2017 (accessed May 1, 2021).

[57] Wikipedia entry. *Wikipedia: Fingerstyle guitar*. https://en.wikipedia.org/wiki/Fingerstyle_guitar#Technique. 2021 (accessed June 7, 2021).

[58] Wikipedia entry. *Wikipedia: Bar (music)*. https://en.wikipedia.org/wiki/Bar_(music). 2021 (accessed May 1, 2021).

[59] Wikipedia entry. *Wikipedia: Beat (music)*. https://en.wikipedia.org/wiki/Beat_(music). 2021 (accessed May 1, 2021).

[60] Wikipedia entry. *Wikipedia: Cadence*. https://en.wikipedia.org/wiki/Cadence. 2021 (accessed May 1, 2021).

[61] Wikipedia entry. *Wikipedia: Chord (music)*. https://en.wikipedia.org/wiki/Chord_(music). 2021 (accessed May 1, 2021).

[62] Wikipedia entry. *Wikipedia: Degree (music)*. https://en.wikipedia.org/wiki/Degree_(music). 2021 (accessed May 1, 2021).

[63] Wikipedia entry. *Wikipedia: Digital audio workstation*. https://en.wikipedia.org/wiki/Digital_audio_workstation. 2021 (accessed May 1, 2021).

[64] Wikipedia entry. *Wikipedia: Duration (music)*. https://en.wikipedia.org/wiki/Duration_(music). 2021 (accessed May 1, 2021).

[65] Wikipedia entry. *Wikipedia: Function (music)*. https://en.wikipedia.org/wiki/Function_(music). 2021 (accessed May 1, 2021).

[66] Wikipedia entry. *Wikipedia: Guitar Hero*. https://da.wikipedia.org/wiki/Guitar_Hero. 2021 (accessed May 1, 2021).

[67] Wikipedia entry. *Wikipedia: Interval (music)*. https://en.wikipedia.org/wiki/Interval_(music). 2021 (accessed May 1, 2021).

[68] Wikipedia entry. *Wikipedia: Key (music)*. https://en.wikipedia.org/wiki/Key_(music). 2021 (accessed May 1, 2021).

[69] Wikipedia entry. *Wikipedia: Metre (music)*. https://en.wikipedia.org/wiki/Metre_(music). 2021 (accessed May 1, 2021).

[70] Wikipedia entry. *Wikipedia: Musical notation*. https://en.wikipedia.org/wiki/Musical_notation. 2021 (accessed May 1, 2021).

[71] Wikipedia entry. *Wikipedia: Musical note*. https://en.wikipedia.org/wiki/Musical_note. 2021 (accessed May 1, 2021).

[72] Wikipedia entry. *Wikipedia: Pitch class*. https://en.wikipedia.org/wiki/Pitch_class. 2021 (accessed May 1, 2021).

[73] Wikipedia entry. *Wikipedia: Pitch (music)*. https://en.wikipedia.org/wiki/Pitch_(music). 2021 (accessed May 1, 2021).

[74] Wikipedia entry. *Wikipedia: Scale (music)*. https://en.wikipedia.org/wiki/Scale_(music). 2021 (accessed May 1, 2021).

[75] Wikipedia entry. *Wikipedia: Sight-reading*. https://en.wikipedia.org/wiki/Sight-reading. 2021 (accessed May 1, 2021).

[76] Wikipedia entry. *Wikipedia: Tempo*. https://en.wikipedia.org/wiki/Tempo. 2021 (accessed May 1, 2021).

[77] Wikipedia entry. *Wikipedia: Time signature*. https://en.wikipedia.org/wiki/Time_signature. 2021 (accessed May 1, 2021).

[78] Wikipedia entry. *Wikipedia: Borrowed chord*. https://en.wikipedia.org/wiki/Borrowed_chord. 2021 (accessed May 10, 2021).

[79] Wikipedia entry. *Wikipedia: Chord names and symbols (popular music)*. https://en.wikipedia.org/wiki/Chord_names_and_symbols_(popular_music). 2021 (accessed May 2, 2021).

[80] Wikipedia entry. *Wikipedia: Tablature*. https://en.wikipedia.org/wiki/Tablature. 2021 (accessed May 2, 2021).

[81] Youtube entry. *Youtube: Chopin, Nocturne in E-flat Major, opus 9 no.2, Piano Solo (animated score)*. https://www.youtube.com/watch?v=--ykTqoQnqI&list=PL3B39D4DC6881CF28&ab_channel=smalin). 2021 (accessed May 2, 2021).

[82] tonejs. *Tone.js*. https://tonejs.github.io/. 2021 (accessed June 8, 2021).

DECLARATION

I declare that this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where states otherwise by reference or acknowledgment, the work presented is entirely my own.

Aarhus, June 2021

Christian Nordstrøm
Rasmussen