Challenges to generate musical visualizations

Rute Moura[1] *, **Giordano Cabral**[1] , **Jader Abreu**[1] , **Mychelline Cunha**[1] **Horhanna Almeida**[1]

[1]Mustic† Centro de Informática/Universidade Federal de Pernambuco
Av. Jornalista Aníbal Fernandes, s/n, Cidade Universitária(Campus Recife) – 50.740-560, Recife, PE

rmam2@cin.ufpe.br,grec@cin.ufpe.br,jaoa@cin.ufpe.br, msh@cin.ufpe.br, hao@cin.ufpe.br

Abstract. *The technological acceleration of recent years has allowed for significant advances in data processing and the study of Music, an art with richly expressive and communicative information. In this article, we bring reports of experiences and contributions in the treatment of musical data extracted from digital MIDI and sound files, for the development of systems that generate musical visualizations.*

1. Introduction

Music enchants and fascinates humanity for a long time. It is capable of arousing feelings and emotions, it contains abstract data of subjective characteristics that have an innate complexity to be measured and analyzed.[1] The explanations of its concepts in Musicology literature are based on dense and long technical texts of music theory with few explorations of visual resources, deepening the study of musical information as an object of communication, and using very few computational resources available to facilitate the apprehension, perception, and analysis of its elementary concepts.

Given the computational capacity in data processing today, the ease of access to open data increased the power of the music industry through Peer-to-peer (P2P) sharing and access to collections of musical works through digital files such as MP3, WAV, MIDI. However, due to the large volume of existing compressed raw data, there are often computational complexities in the treatment and extraction of characteristics from musical information to make them visually communicable.

Within this context and considering the representational challenges of Music, members of the research group Mustic-CIn/UFPE, seek research projects to assist in the visual communication of musical information. Using software resources generating useful applications, enabling technology to be a means to help professionals and experimenters in the area of Music in their creative processes and to expand access to musical information.

2. Visual Music

Computational Music Visualization is an interdisciplinary area that brings together researchers and applications from several other related areas such as Music, Computer Science, Information Visualization, Design, and their specialties. Thus, this large area can be categorized by its types of applications, such as symbolic visualizations focused on the study of structural and elementary information in music, which can be analytical as in Cabral [2], animated and interactive as in Cantareira [3] and Silva [4] communicating musical data in real-time. They can also be used as a tool for music education Hein[5] and creation of compositions [6]. The category of sound visualizations, which focuses on the audiovisual processing Jacomé [7] and Scordato [8] which develops Ianixx, for the creation of artistic performances.

Performing exploratory research on the Music Visualization area, we synthesized its definition with the diagram shown in figure 1.

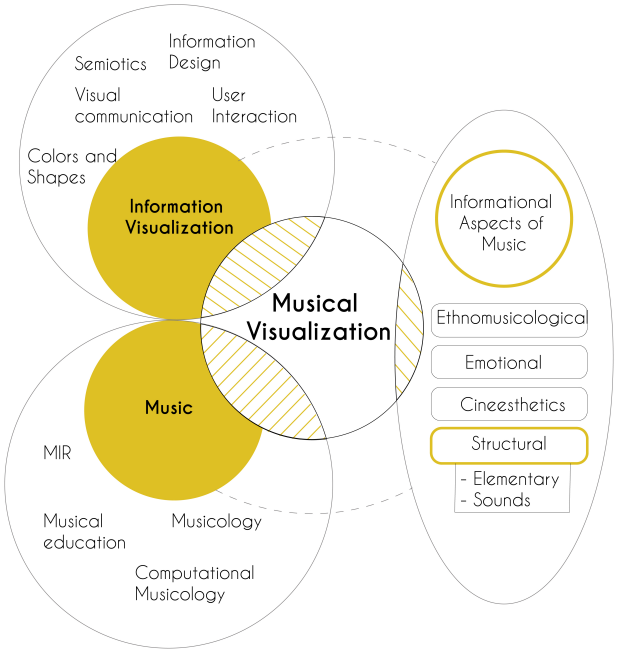

Figure 1: Visual Music

Given the breadth of application of musical visualizations in various areas, we consider it as an intersection and, through new technology, it allows to improve communication, study, and understanding of music as information and as a sign [9]. The MIR (Music Information Retrieval) area, for example, has well-established works in the context of developing a set of tools for computationally extracting musical information, such as Music21 [10]. However, the development of musical visualizations still has limitations, mainly in relation to multi-representational challenges, Futrelle [11]. Considering that this is an area that is still maturing, we seek to carry out studies in relation to different ways of representing music. Therefore, we emphasize that to create a musical visualization, we must consider its informational aspects, that is, which musical information will be visually communicated.

*Supported by FACEPE.
†https://mustic.cin.ufpe.br/

3. Related Work

Over the last few decades, several works have been deepening research in areas such as Music Information retrieval (MIR) and Semantic Audio Analysis (SAA), whose objectives are, respectively, to retrieve information from musical data and analyze information in audio [12].

An example of relevant work in this area of study was the development of Pure Data, the free visual programming software for computer-supported interactive music works developed by Miller Puckette [13], which expanded the possibility of experimenting with data processing computational music.

Based on Miller Puckette's work, LibPd was created, a library that makes Pure Data an embeddable tool in any [14] application. Since then, many works started to use Libpd to create and manipulate musical and audio information, making Libpd one of the most used libraries in the area by authors such as Wilcox[15] Brinkmann [16] Konrad [17] Hathway [18]and Steps [19].

More recently, other libraries and toolkits have been explored for such work, such as MatLab, well-known mathematical analysis software that has developed a toolkit for MIR [20].

Another great revolution was the emergence of the Musical Instrument Digital Interface (MIDI), which made musical communication possible with the robust [21] protocol. This protocol has evolved over the years allowing for better communication between instruments and computers. Recent works try to facilitate the use of this tool through a better human-computer interface for creation and interaction with MIDI Machado [22] and Amorim [23]. Another example is applications that help experimentation and create new rhythms from rich interfaces through the MIDI protocol, as in Wilson [24].

On the other hand, interdisciplinary studies deepen discussions in the Musical Visualization area, which carry out research to make musical information more visual and understandable to the eyes. As in Vieira's work [25], which uses dynamic visualizations to understand tones with parameters of graphic animations, with variations in volumes and colors according to the musical energy, also seeking to realize emotional aspects such as warm colors are attributed to joyful moments and cool colors for tragic music moments. The *Musanim* [26] is also a practical example of dynamic visualization, as is the *Da Capo* [4] which uses MIDI symbolic data inputs, to allow interaction with controllers through controllers musical instruments generating musical visualizations. Or even in tools like MusicVis [27], which through the *input* of MP3 files or microphone capture performs the processing of musical data generating a real-time visualization marking the visual elements in the graphic space of the browser. system user, analyzing the entire musical sequence according to timbres and sound frequencies.

Considering all this state of the art, this article tries out the use of some technologies found in related works, such as Libpd, and uses symbolic data input from MIDI files to explore musical information generation and visualization in interactive applications.

4. Developing Music Visualizations

In this article we will describe the development of two software, *Espectromusic* and *Mandrit*, which we developed through the processing of musical data, with the extraction of MIDI file characteristics and transformation of its information into structured formats to obtain dynamic and static visualizations that visually communicate musical and sound information through graphical results.

4.1. Espectromusic

The *Espectromusic* is a multiplayer game in which the player aims to overcome a digital labyrinth created and manipulated in real-time by music captured by a microphone. Creating a tournament dynamic, as shown in the figure 2 with collections performed during user validation processes. Engaging interaction and competitiveness between the listener and the opposing musician. And it can also be used as a complementary projection to the musical performance according to the variation of volumes and frequencies of what is played and which forms the spectrum of obstacles.

Figure 2: Validations Espetromusic

Designed as a platform game, it has two main scenes, the first with audio input, in other words player 1 interacts with a predetermined song, as a tutorial stage interacting with the *ball* through the keyboard controls. aim to cross the obstacles until the next stage. In the second phase, you can interact directly with player 2, the musician, considered the generator of obstacles when playing the music, raises the spectrum and with his instrument can build obstacles preventing player 1 from moving forward with his *ball*.

4.1.1. Extraction of sound characteristics

When we developed the entire graphical interface of the game by Unity, we received the musical data captured in

real-time by the microphone and transformed it into dynamic visualizations with a graphic area determined by the elevation of bars according to the sound spectrum, having as input what the musician instrumentalist plays.

To process the sound data captured via the microphone, we used the *LibPd* library as shown in the flowchart of figure 3, which allows input and control over the sound, transforming Pure Data (PD) into an embeddable library in the Unity IDE. PD allows you to process raw audio from audio drivers and MIDI drivers, which makes it possible to use musical and sound information in mobile phone applications, game systems, and web pages.

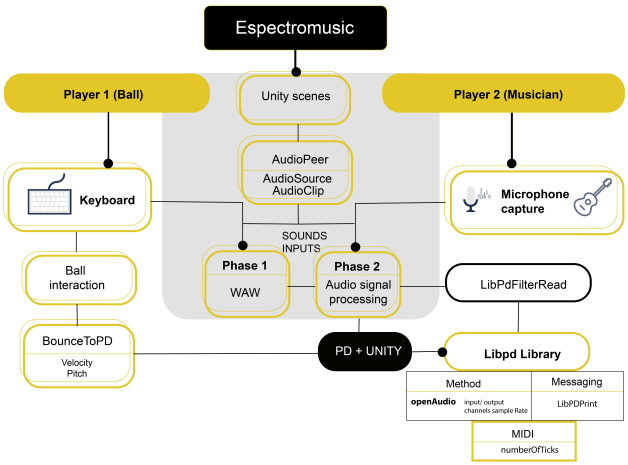

Figure 3: MIDI input microphone processing

Therefore, it was necessary to develop a unity script that uses a fundamental frequency and adds to the variation of the ball's speed in the scenario, taking the pitch of a certain frequency and sending it to the scene through LibPd, in a cyclic exchange that passes it on to the PD to synthesize In real-time. For this real-time integration of Unity and PD to occur, the programs must be open synchronously for the game to run.

We use the *audiopeer* methods that extract values from the spectrum by calling the *audiosource* script, which is the component that reproduces the music, in the first phase in WAW format, and in the second phase, we obtain musical information captured via a microphone, and apply to the objects of the game scene. In addition, *audioclip* complements the processing, having the function of storing a sample of audio data, which has variations in the spectrum generated about intensity, referring to the amount of energy, height according to the format of the sound oscillations, and timbre determined by frequency, with this information being shown at each frame and generating obstacles.

It is important to note that the LibPd library works internally through MIDI messages, such as note-on, integer values of the channel, pitch, and velocity. And it sends a control change event to the PD. So, we have a script called *LibPdFilterRead* that opens the microphone door when we enter the second phase of the game and initializes PD by taking the Unity sample rate and calculating the number of ticks.

We emphasize that during the development of *Espectromusic* we also explored audio processing, to apply reactive sound effects to the *ball* object which the player manipulates according to speed variation, creating sounds through visual programming with PD, as shown in figure 4.

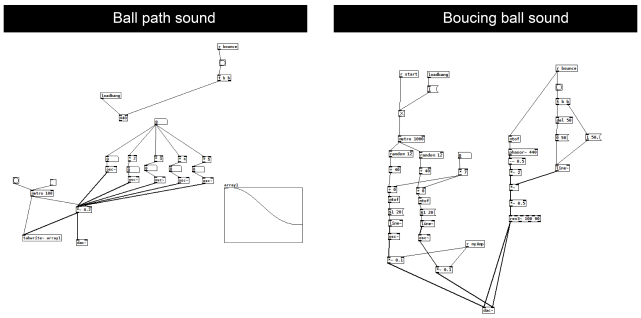

Figure 4: Visual Sound Programming with PD

In the PD graphic environment, we create sound objects, called "patches", we apply two characteristics to *ball*, in (I) the sound of its path, triggered by a certain MIDI frequency, with speed and height controls, which vary according to input values determined by the oscillators. As in II, the sound of the *ball* bouncing, simulating its touch on the game's floor plan, but with random variations of frequency values.

These files synthesize a continuous and pleasant sound according to the received MIDI frequencies, and with the use of the converter *dac* they transform the digital to analog, generating a sound output. It is important to emphasize that this PD graphic environment format for sound creation made it easier for the project developers to carry out the manipulation of digital audio data, experimenting, listening, and visualizing sound waves, spectra, frequencies, and amplitudes.

4.1.2. Challenges in data interference

With the development of *Espectromusic*, we faced some computational challenges about audio interference in raw data capture, due to microphone transmission and real-time data acquisition, which depending on the environment also transmits noise or information that we had no computational control. In addition to the deviation of data by the PD inputs, due to the use of multiple inputs, both from a complete music file and the sound applied in the interaction of player 2 with the *ball* having variable pitches according to the speed of the movement.

In addition to these aspects of musical data, we highlight the need for tools for musical game development that perform dynamic integration and manipulation between the Unity IDE and the PD. Because despite the LibPd library, which has been a great alternative in our case, it still had several computational limitations, for example, the game only works with Unity3D's 32bit editor.

Another difficulty was to develop sounds that were pleasing to the ears by manipulating the oscillators and perceiving according to the speed of the ball. Needing a thorough job of defining the ideal frequency and the limits of variations, as well as synchronizing the sending of data in real-time. So we consider that there are also gaps about the domain of tools and technical specifications in the choice of sounds, in the attribution of geometric shapes, as we need to carry out in-depth studies in the manipulation of the 3D area and construction of the scenario. That's why it made it easier to have specialist members in Music, Design and Computer Science accompanying and developing the project together, and thus we obtained visual and interactive results with *Espectromusic*.

4.2. Mandrit

Knowing the various applications and potential of software resources - for processing, processing, and manipulating data from digital MIDI files -, we developed the*Mandrit* system: musical visualization generator for rhythm analysis, as shown in figure 5, it draws three types of graphic models to visually communicate rhythm and demonstrate its micro information through the rhythmic signature of a musical work.

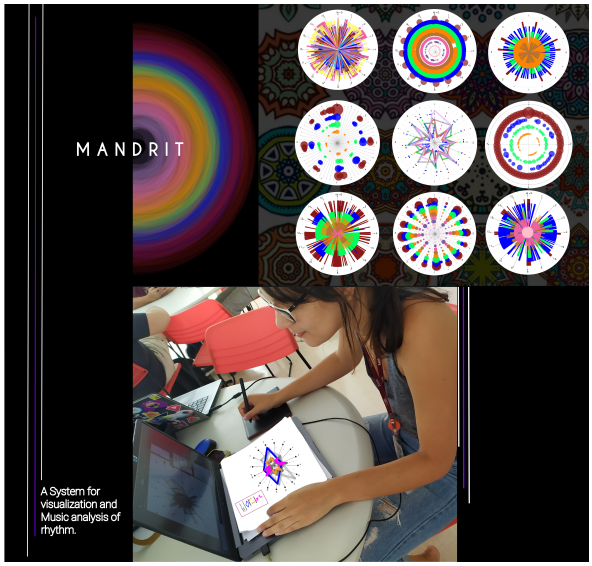

Figure 5: Musical Rhythm Analysis

With its visualizations, *Mandrit* proposes a musicological study of musical analysis processes and Auditory Scene Analysis (ASA)[28] to measure musical perceptions with the aid of graphical representations, using qualitative and quantitative assessments.

With the collections of feedbacks, we realized as well as [29] that the results show that direct associations of musical perception by the association of colors as a representative entity of the instrument, and also the participants make correlations based on graphic visual structures that are largely determined by context, and their abilities to understand and perceive music associating musical visualizations.

The elementary information of the song chosen to develop visualizations was the rhythm. Realizing the scarcity of analytical and computational research for the study and analysis of rhythm, its micro information that visually communicates variations, musical signatures, polyrhythms, differences, and similarities of musical works.

4.2.1. Extraction of rhythmic characteristics from MIDI

In this case study, the process of exploring and choosing rhythmic information to generate a musical visualization is carried out through tutorials and collection of feedbacks with specialists in music and the area of Graphic Expression and Design. Therefore, we emphasize that to build a musical visualization, we first need to define the properties that we want to communicate visually according to the needs of potential users. Therefore, with *mantrit* aiming to generate visualizations that emphasize the signatures of the music time signature. Seeking to facilitate the analysis of intervals, pulses, patterns, and regular or irregular movements of bars. Therefore, we performed the extraction of these characteristics through the mandrit algorithm, described by the flowchart of figure 6. The rhythm signatures, which we generate, are based on a volume of data extracted from MIDI files, extracting the events of the measure, its metrics, and counting the number of notes of each instrument.

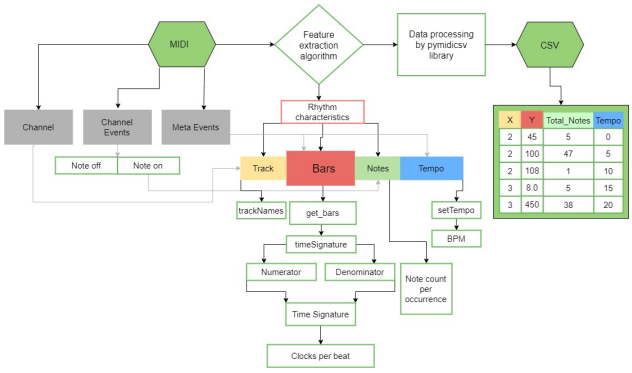

Figure 6: Fluxograma Mandrit

With it, in addition to performing the specifications of the characteristics of the rhythmic element, obtaining its metadata, we also created a database with a collection of over 60 songs, different genres, and with simple metrics and enabling the analysis of bars. We structure and store musical sequences according to track variables, time signature, and amount notes. To develop the extracted code, we determine tasks to develop with these following steps:

(1) Read MIDI file in type 01 format.
(2) Declare measure data for each song with the get_measure function.
(3) Extract meta-event from bar signature.
(4) Calculation of bar division
(5) Count the amount of "Note on" notes.

(6) Generate the matrix with three elements: X (tracks), Y (Measure subdivision) and Total_of_Notes (Amount of notes).

To carry out these specifications, we access the raw data of a musical sequence, and collect the information from the *tracks* by accessing the channels of the MIDI file, counting the occurrences of the number of notes only by the *noteon*, and considering respectively each temporal occurrence as a function of the measurement metrics. The transformation of these raw data and metadata was done through the library Py_Midicsv [30] which has instructions determined by the type 1 format protocol of a MIDI file, in which all data are separated by track. With this implementation, we store data in a spreadsheet format with structured and organized information, facilitating the generation of *Mandrit* views. We apply mathematical and computational functions aided by software resources, developing the algorithm described in figure 7.

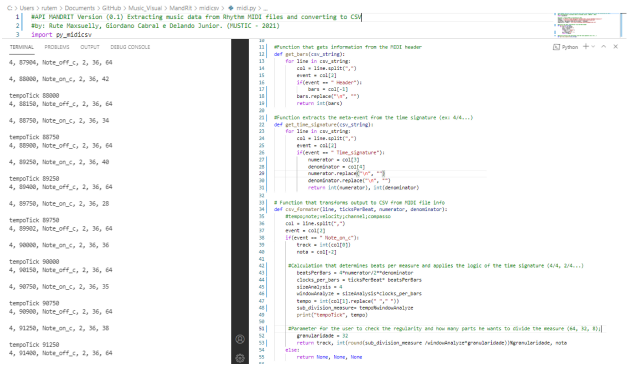

Figure 7: MIDI Rhythm Extraction

We emphasize that we extract only some information from the midi file, making specifications for the elementary rhythm data to be returned, the parameters of "tracks" are also called tracks, the signature of the measure and "Notes". In other words, these were our fundamental variables to store in numerical data of the musical sequence and represent a synthesis of the rhythm of each musical work to be analyzed.

Based on the structured data generated by extracting rhythm characteristics, we also developed an algorithm that generates the *Mandrit* graphics, as shown in figure 8. Where we assign rhythmic information applying to graphic shapes - bubbles, lines, dots, colors -, in addition to using polar mathematical functions, that is, it has an angular orientation according to a given rotation.

The generation of static graphs was the result of tests seeking to assist in the priority display of the amount of notes as a function of temporal occurrence, so *mandrit* generates three types of graphs: Polar Bubble Chart, Polar Radar Chart, and Radial Collumn. All of them have the same principle in the representation of rhythm in a metaphor to clockworks with their polar and cyclic visualizations, however, we experience in their visual approaches the variations of graphic forms attributed to musical information.

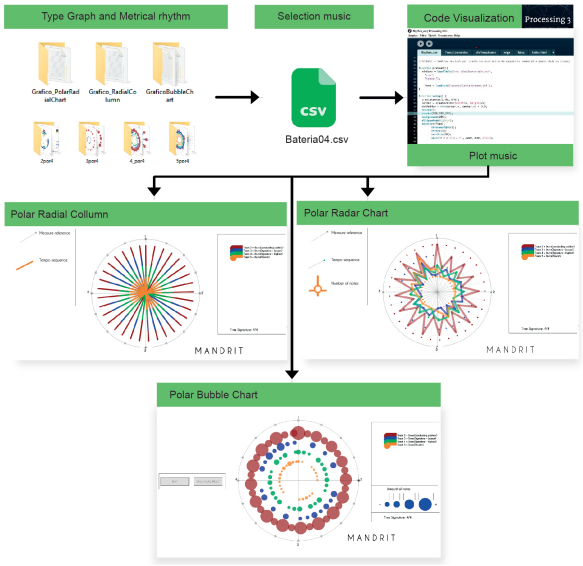

Figure 8: Plotting rhythmic visualizations

We considered a set of fidelity tests performed accompanied by visualizations, which were very useful in the process of understanding and perceiving the organization of musical data in the representation. The expert contributors, in addition to bringing provocative *feedbacks* and qualitative impressions, also contributed to the musical analysis by noticing elements that were not visible before, such as the similarities between aspects of musical performance about the instruments of a musical work.

4.2.2. Challenges generations of Mandrit Views

In every experimental process of constructing musical visualizations, we face computational challenges about the treatment and extraction of rhythm characteristics. A first limitation is that *Mandrit* only extracts rhythmic data from MIDI files in type 1 format, due to the data structure we need to have the tracks organized and separated to facilitate musical analysis. However, because of the productions available online, we often find files that complicated the processing and resulted in graphic pollution, as in the figure 9.

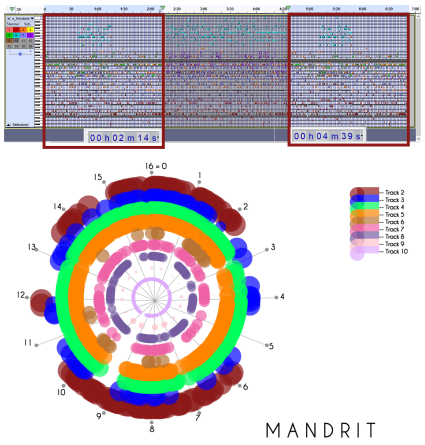

Figure 9: MIDI production challenges

When we plotted the music, we considered the amount of accumulated information incomprehensible, and when we checked the musical production, we realized that the producer gathered excerpts from the song ”A Felicidade, Toquinho” in its initial and final parts in conjunction with Bossa style randomness, which resulted in a file of 06 minutes and 53 seconds, with a complex and accumulative amount of musical data. For this reason, the plotted view does not show the rhythm signature correctly demarcated, and it also has no reference to its instruments in the legend, which limits the reading and communication of the rhythm.

Still, on the aspects of independent productions, we highlight that we often find files without track descriptions, or their musical sequences completely disorganized. This made it difficult to plot the views, even to apply parameterization and generalization of graphic models. On the other hand, all this also demonstrates the musical artistic singularities and complexities. Like for example in the song ”Take five, Dave Brubeck”. It has a complex 5/4 time signature and so we experimented with plotting in different arrangements, as shown in the figure 10 and soon noticed inconsistencies in reading files from different MIDI bases.

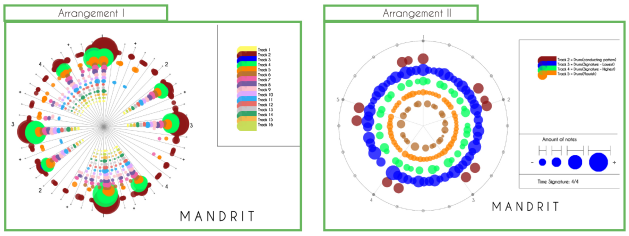

Figure 10: Arrangements

As we observed in ”Arrangement I”, it generated a more complex visualization with a greater number of tracks (tracks) being incomprehensible its fundamentals since it is a piece of quinary music, that is, with a complex five-pulse bar. In the MIDI file that we printed the ”Arrangement II”, with the information much more organized and structured, it is possible to obtain this signature, through the improvements we made implementing a generalization in the length of the song according to a granularity applied to the values in the extracted data, which can range between 64, 32 and 8 in granularity.

That is, these values can be changed by the user who generates the visualization, depending on what they need to conceive or understand in the musical analysis of the work.

Through the oscillations, and variations in the counting of the number of notes demonstrated during the course, we understand that these differences in the aspects of expressiveness are notorious with different productions and files of the same music. This context is explained by the music production process, which is relative according to each musician, demonstrating in the different graphic results their distinct expressiveness, despite retaining a signature in the most striking poles of the rhythm.

There were also some limitations regarding the generation of static views, due to graphic pollution resulting from the accumulation of temporal information plotted in a two-dimensional graphic area. That’s why we bring modifications in the reference of the tempo with the grid to the movement and orientation of the clock and a metaphor to the rhythmic time represented by an animated cursor as its pointer to help in the accompaniment. In general, there are different challenges and specific gaps that arise during the process of extracting information from raw data, and there are also several possibilities to visually represent and communicate information with of this, we learn the advantages and disadvantages of using certain graphic forms and to make choices according to each need, always adapting to the objectives centered on the users.

5. Guidelines for developing Music Visualizations

As a result of these case studies and in the development of *Espectromusic* and *Mandrit*, several lessons were learned about the process of building music visualization systems, mainly regarding the selection, extraction and generation of representation. With the creation of prototypes we were able to obtain musical parameters, visual structures and identify challenges and limitations to visually communicate musical information.

Thus, in addition to the technical records described in this article, we also provide guidance on the challenges of exploring this interdisciplinary area, which is Computational Visualization Music. Designing 5 guidelines that can help in the process of creating a system to generate musical visualizations, they are:

(1) Conducting similar research, user-centered technical and participatory observation to explore musical information to be communicated visually.

(2) Categorization and selection of visualization types to develop.

(3) Exploration of the creative process aided by computational resources, to generate automatic visualizations through the extraction, treatment, filtering of musical data and application in *visual structures*.

(4) Use of a Design process based on cyclical processes of ideation, prototyping, and evaluation with *feedbacks* collection and fidelity tests.

(5) Communication and sharing of graphic results to strengthen the community.

We consider these to be crucial steps to achieve the objective of visually communicating musical information, through we built an experimental protocol model based on *Design Thinking* composed of targeting cards and methods applicable to graphic experiments, enabling a systematically mapped view of automatic generation of musical visualizations. In addition to broadening the discussion, and unraveling gaps in the quest to facilitate and com-

plement communication between professionals and experimenters in the field of music. Transforming computer interfaces and tools as a means to make musical information accessible and understandable.

6. Final Considerations

With these experience reports, we highlight the importance of interdisciplinary research as a means to expand the application of technology in the music field, narrowing the limitations and connecting professionals from fields that can complement each other, generating useful and interesting results. For this, it is necessary to emphasize that the follow-up with specialists in the area of music and Information Visualization, in continuous cyclical evaluation processes with fidelity tests, were essential to achieve graphic results and obtain improvements in each session and experimentation round.

Despite this, communication between developers and musicians is still a gap in the extraction and transformation process in prioritizing data to be visually reported. Because there are technical dependencies required due to the need for music theory studies to flow this exchange.

Acknowledgement

To the Collaborators of the Mustic research group: Ricardo Sholtz, Delando Júnior, Flaviano Dias, Felipe Calegário e FACEPE.

References

[1] José Fornari. Da música à musicologia, January 2019.

[2] Giordano Cabral and Robert Willey. Analyzing Harmonic Progressions with HarmIn: the Music of Antônio Carlos Jobim. page 12, 2007.

[3] Gabriel Dias Cantareira, Luis Gustavo Nonato, and Fernando V. Paulovich. MoshViz: A Detail+Overview Approach to Visualize Music Elements. *IEEE Trans. Multimedia*, 18(11):2238–2246, November 2016.

[4] Mateus Bastos da Silva. Da Capo: Uma exploração das representações da música, 2019.

[5] Ethan Hein. *Designing the Drum Loop: A constructivist iOS rhythm tutorial system for beginners*. PhD thesis, Steinhardt School of Culture, Education, and Human Development, New York University, 2013.

[6] Delfina Malandrino, Donato Pirozzi, and Rocco Zaccagnino. Visualization and music harmony: Design, implementation, and evaluation. In *2018 22nd International Conference Information Visualisation (IV)*, pages 498–503, Fisciano, Italy, July 2018. IEEE.

[7] Jarbas Jacomé. Sistemas Interativos de Tempo Real para Processamento Audiovisual Integrado, 2007.

[8] Julian Scordato. Composing with Iannix. In *Proceedings of the Fifth Conference on Computation, Communication, Aesthetics and X.*, volume V, page 389, Lisboa, Portugal, 2017.

[9] Eero Tarasti. Los signos en la historia de la música, historia de la semiótica musical. *Semiótica Musical*, pages 15–71., 2008.

[10] Michael Scott Cuthbert and Christopher Ariza. music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data. page 7, 2010.

[11] Joe Futrelle and J. Stephen Downie. Interdisciplinary Research Issues in Music Information Retrieval. *Journal of New Music Research*, 32(2):121–131, June 2003.

[12] Bozena Kostek. *Perception-based data processing in acoustics: applications to music information retrieval and psychophysiology of hearing*, volume 3. Springer Science & Business Media, 2005.

[13] Miller Puckette et al. Pure data: another integrated computer music environment. *Proceedings of the second intercollege computer music concerts*, pages 37–41, 1996.

[14] Peter Brinkmann, Peter Kirn, Richard Lawler, Chris McCormick, Martin Roth, and Hans-Christoph Steiner. Embedding pure data with libpd. In *Proceedings of the Pure Data Convention*, volume 291. Citeseer, 2011.

[15] Dan Wilcox. Pdparty: An ios computer music platform using libpd. In *Proceedings of the Pure Data Convention*, 2016.

[16] Peter Brinkmann, Dan Wilcox, Tal Kirshboim, Richard Eakin, and Ryan Alexander. Libpd: Past, present, and future of embedding pure data. In *Proceedings of the Pure Data Convention*, 2016.

[17] Markus Konrad and Klaus Jung. *Analysis of audio synthesis possibilities on mobile devices using the Apple iPhone and iPad*. PhD thesis, Ph. D. Thesis, HTW Berlin, 2011.

[18] Joe Hathway. Creating a digital musical instrument for children's education and expression. 2021.

[19] Leonardo Porto Passos, Leonardo Arruda, and José Fornari. Pure data for pure audio games.

[20] Olivier Lartillot, Petri Toiviainen, and Tuomas Eerola. A matlab toolbox for music information retrieval. In *Data analysis, machine learning and applications*, pages 261–268. Springer, 2008.

[21] David Miles Huber. *The MIDI manual*. Sams, 1991.

[22] André Machado. *Tradutor de arquivos MIDI para texto utilizando linguagem funcional CLEAN*. PhD thesis, Universidade Federal de Uberlândia, Uberlândia, 2001.

[23] ANDRÉ AMORIM and JOHANNES KJELLBERG. A step out of the grid-interaction with touch-based alternative rhythm programming layouts in drum machine user interfaces. 2020.

[24] Scott Wilson. Patterning: The iPad drum machine that wants us to think in circles, 2015.

[25] Marco Filipe Ganança Vieira. *Interactive music visualization: implementation, realization and evaluation*. PhD thesis, 2014.

[26] Stephen Malinowski. Music animation machine renderers, 2012.

[27] Van Huynh and David Comberg. MusicVis a web application for visualizing sound., 2017.

[28] Albert S. Bregman and Stephen McAdams. *Auditory Scene Analysis: The Perceptual Organization of Sound. The Journal of the Acoustical Society of America*, 95(2), February 1994.

[29] Gregor Strle e Matev vz Pesek e M. Marolt. Towards user-aware music information retrieval: Emotional and color perception of music. In *Emotions and Personality in Personalized Serviços*, 2017.

[30] John Walker. MIDICSV, 2004.