

# Challenges to generate musical visualizations

Rute Moura<sup>1 \*</sup>, Giordano Cabral<sup>1</sup>, Jader Abreu<sup>1</sup>, Mychelline Cunha<sup>1</sup> Horhanna Almeida<sup>1</sup>

<sup>1</sup>Music<sup>†</sup> 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 enchanting 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 Music-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.

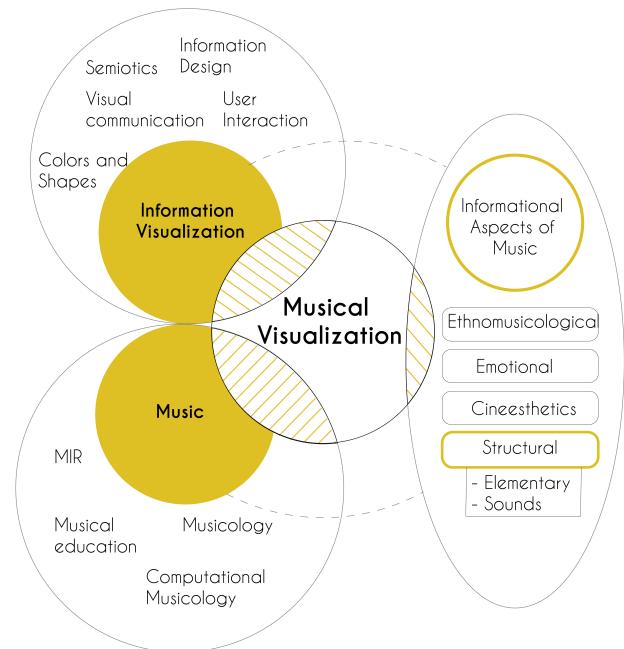


Figure 1: Visual Music

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.

<sup>†</sup><https://music.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

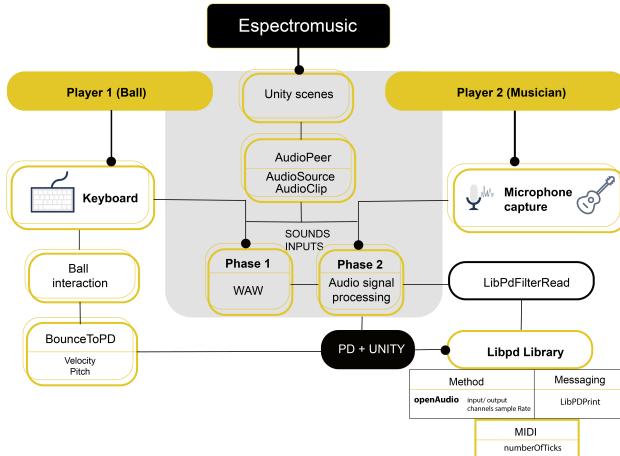
Designed as a platform game, it has two main scenes, the first with audio input, in other words player 1 interacts with a predetermined song, as a tutorial stage interacting with the *ball* through the keyboard controls. aim to cross the obstacles until the next stage. In the second phase, you can interact 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**

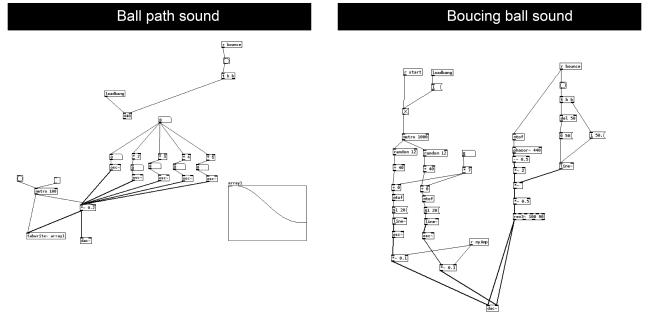
Therefore, it was necessary to develop a unity script that uses a fundamental frequency and adds to the variation of the ball's speed in the scenario, taking the pitch of a certain frequency and sending it to the scene through LibPd, in a cyclic exchange that passes it on to the PD to synthesize In real-time. 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, *audio-clip* 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

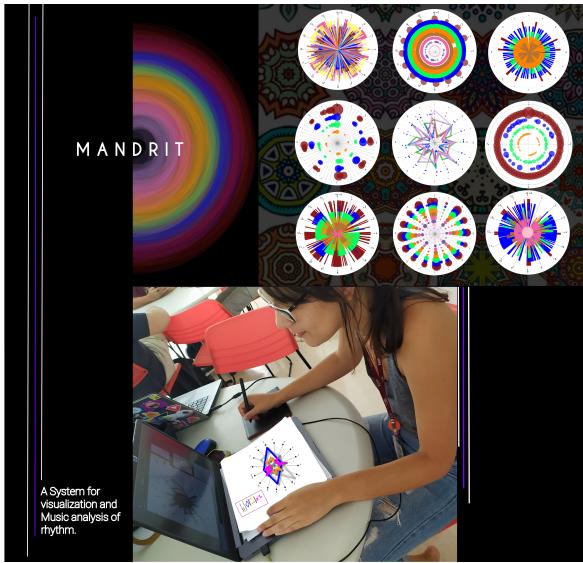
With the development of *Espectromusic*, we faced some computational challenges about audio interference in raw data capture, due to microphone transmission and real-time data acquisition, which depending on the environment also transmits noise or information that we had no computational control. In 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**

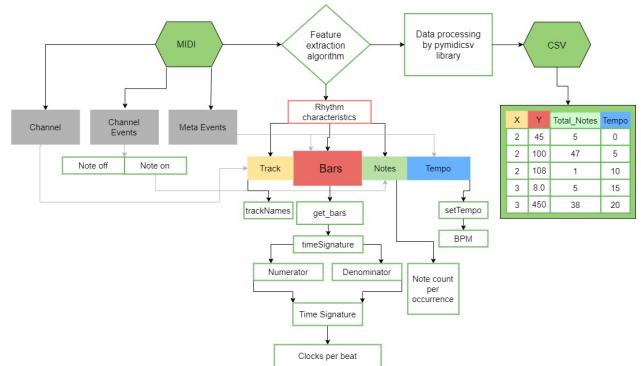
With its visualizations, *Mandrit* proposes a musical study of musical analysis processes and Auditory Scene Analysis (ASA)[28] to measure musical perceptions with the aid of graphical representations, using qualitative and quantitative assessments.

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 *mandrit* 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.

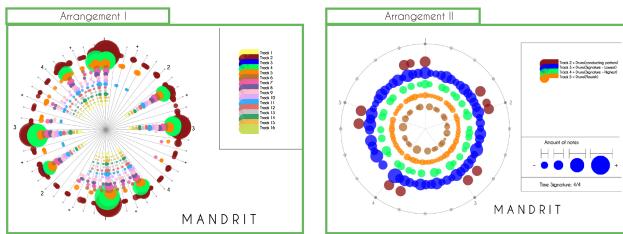
(6) Generate the matrix with three elements: X (tracks), Y (Measure subdivision) and Total\_of\_Notes (Amount of notes).

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.

```
C:\Users\rober...>cd Documents\GitHub\Music_Visual + Mandrit> python < midi.py >
1: 144:0000: Extracting tracks and measure subdivision files and converting to CSV
2: 145:0000: Note_Polarized_Graphical_Carta
3: 146:0000: Note_Radial_Cartoon
4: 147:0000: Note_BubbleChart
5: 148:0000: Note_Score
6: 149:0000: Note_Song
7: 150:0000: Note_Temporal
8: 151:0000: Note_Quantitative
9: 152:0000: Note_Harmonic
10: 153:0000: Note_Metric
11: 154:0000: Note_Rhythmic
12: 155:0000: Note_Metrical
13: 156:0000: Note_Metric
14: 157:0000: Note_Metric
15: 158:0000: Note_Metric
16: 159:0000: Note_Metric
17: 160:0000: Note_Metric
18: 161:0000: Note_Metric
19: 162:0000: Note_Metric
20: 163:0000: Note_Metric
21: 164:0000: Note_Metric
22: 165:0000: Note_Metric
23: 166:0000: Note_Metric
24: 167:0000: Note_Metric
25: 168:0000: Note_Metric
26: 169:0000: Note_Metric
27: 170:0000: Note_Metric
28: 171:0000: Note_Metric
29: 172:0000: Note_Metric
30: 173:0000: Note_Metric
31: 174:0000: Note_Metric
32: 175:0000: Note_Metric
33: 176:0000: Note_Metric
34: 177:0000: Note_Metric
35: 178:0000: Note_Metric
36: 179:0000: Note_Metric
37: 180:0000: Note_Metric
38: 181:0000: Note_Metric
39: 182:0000: Note_Metric
40: 183:0000: Note_Metric
41: 184:0000: Note_Metric
42: 185:0000: Note_Metric
43: 186:0000: Note_Metric
44: 187:0000: Note_Metric
45: 188:0000: Note_Metric
46: 189:0000: Note_Metric
47: 190:0000: Note_Metric
48: 191:0000: Note_Metric
49: 192:0000: Note_Metric
50: 193:0000: Note_Metric
51: 194:0000: Note_Metric
52: 195:0000: Note_Metric
53: 196:0000: Note_Metric
54: 197:0000: Note_Metric
55: 198:0000: Note_Metric
56: 199:0000: Note_Metric
57: 200:0000: Note_Metric
58: 201:0000: Note_Metric
59: 202:0000: Note_Metric
60: 203:0000: Note_Metric
61: 204:0000: Note_Metric
62: 205:0000: Note_Metric
63: 206:0000: Note_Metric
64: 207:0000: Note_Metric
65: 208:0000: Note_Metric
66: 209:0000: Note_Metric
67: 210:0000: Note_Metric
68: 211:0000: Note_Metric
69: 212:0000: Note_Metric
70: 213:0000: Note_Metric
71: 214:0000: Note_Metric
72: 215:0000: Note_Metric
73: 216:0000: Note_Metric
74: 217:0000: Note_Metric
75: 218:0000: Note_Metric
76: 219:0000: Note_Metric
77: 220:0000: Note_Metric
78: 221:0000: Note_Metric
79: 222:0000: Note_Metric
80: 223:0000: Note_Metric
81: 224:0000: Note_Metric
82: 225:0000: Note_Metric
83: 226:0000: Note_Metric
84: 227:0000: Note_Metric
85: 228:0000: Note_Metric
86: 229:0000: Note_Metric
87: 230:0000: Note_Metric
88: 231:0000: Note_Metric
89: 232:0000: Note_Metric
90: 233:0000: Note_Metric
91: 234:0000: Note_Metric
92: 235:0000: Note_Metric
93: 236:0000: Note_Metric
94: 237:0000: Note_Metric
95: 238:0000: Note_Metric
96: 239:0000: Note_Metric
97: 240:0000: Note_Metric
98: 241:0000: Note_Metric
99: 242:0000: Note_Metric
100: 243:0000: Note_Metric
101: 244:0000: Note_Metric
102: 245:0000: Note_Metric
103: 246:0000: Note_Metric
104: 247:0000: Note_Metric
105: 248:0000: Note_Metric
106: 249:0000: Note_Metric
107: 250:0000: Note_Metric
108: 251:0000: Note_Metric
109: 252:0000: Note_Metric
110: 253:0000: Note_Metric
111: 254:0000: Note_Metric
112: 255:0000: Note_Metric
113: 256:0000: Note_Metric
114: 257:0000: Note_Metric
115: 258:0000: Note_Metric
116: 259:0000: Note_Metric
117: 260:0000: Note_Metric
118: 261:0000: Note_Metric
119: 262:0000: Note_Metric
120: 263:0000: Note_Metric
121: 264:0000: Note_Metric
122: 265:0000: Note_Metric
123: 266:0000: Note_Metric
124: 267:0000: Note_Metric
125: 268:0000: Note_Metric
126: 269:0000: Note_Metric
127: 270:0000: Note_Metric
128: 271:0000: Note_Metric
129: 272:0000: Note_Metric
130: 273:0000: Note_Metric
131: 274:0000: Note_Metric
132: 275:0000: Note_Metric
133: 276:0000: Note_Metric
134: 277:0000: Note_Metric
135: 278:0000: Note_Metric
136: 279:0000: Note_Metric
137: 280:0000: Note_Metric
138: 281:0000: Note_Metric
139: 282:0000: Note_Metric
140: 283:0000: Note_Metric
141: 284:0000: Note_Metric
142: 285:0000: Note_Metric
143: 286:0000: Note_Metric
144: 287:0000: Note_Metric
145: 288:0000: Note_Metric
146: 289:0000: Note_Metric
147: 290:0000: Note_Metric
148: 291:0000: Note_Metric
149: 292:0000: Note_Metric
150: 293:0000: Note_Metric
151: 294:0000: Note_Metric
152: 295:0000: Note_Metric
153: 296:0000: Note_Metric
154: 297:0000: Note_Metric
155: 298:0000: Note_Metric
156: 299:0000: Note_Metric
157: 300:0000: Note_Metric
158: 301:0000: Note_Metric
159: 302:0000: Note_Metric
160: 303:0000: Note_Metric
161: 304:0000: Note_Metric
162: 305:0000: Note_Metric
163: 306:0000: Note_Metric
164: 307:0000: Note_Metric
165: 308:0000: Note_Metric
166: 309:0000: Note_Metric
167: 310:0000: Note_Metric
168: 311:0000: Note_Metric
169: 312:0000: Note_Metric
170: 313:0000: Note_Metric
171: 314:0000: Note_Metric
172: 315:0000: Note_Metric
173: 316:0000: Note_Metric
174: 317:0000: Note_Metric
175: 318:0000: Note_Metric
176: 319:0000: Note_Metric
177: 320:0000: Note_Metric
178: 321:0000: Note_Metric
179: 322:0000: Note_Metric
180: 323:0000: Note_Metric
181: 324:0000: Note_Metric
182: 325:0000: Note_Metric
183: 326:0000: Note_Metric
184: 327:0000: Note_Metric
185: 328:0000: Note_Metric
186: 329:0000: Note_Metric
187: 330:0000: Note_Metric
188: 331:0000: Note_Metric
189: 332:0000: Note_Metric
190: 333:0000: Note_Metric
191: 334:0000: Note_Metric
192: 335:0000: Note_Metric
193: 336:0000: Note_Metric
194: 337:0000: Note_Metric
195: 338:0000: Note_Metric
196: 339:0000: Note_Metric
197: 340:0000: Note_Metric
198: 341:0000: Note_Metric
199: 342:0000: Note_Metric
200: 343:0000: Note_Metric
201: 344:0000: Note_Metric
202: 345:0000: Note_Metric
203: 346:0000: Note_Metric
204: 347:0000: Note_Metric
205: 348:0000: Note_Metric
206: 349:0000: Note_Metric
207: 350:0000: Note_Metric
208: 351:0000: Note_Metric
209: 352:0000: Note_Metric
210: 353:0000: Note_Metric
211: 354:0000: Note_Metric
212: 355:0000: Note_Metric
213: 356:0000: Note_Metric
214: 357:0000: Note_Metric
215: 358:0000: Note_Metric
216: 359:0000: Note_Metric
217: 360:0000: Note_Metric
218: 361:0000: Note_Metric
219: 362:0000: Note_Metric
220: 363:0000: Note_Metric
221: 364:0000: Note_Metric
222: 365:0000: Note_Metric
223: 366:0000: Note_Metric
224: 367:0000: Note_Metric
225: 368:0000: Note_Metric
226: 369:0000: Note_Metric
227: 370:0000: Note_Metric
228: 371:0000: Note_Metric
229: 372:0000: Note_Metric
230: 373:0000: Note_Metric
231: 374:0000: Note_Metric
232: 375:0000: Note_Metric
233: 376:0000: Note_Metric
234: 377:0000: Note_Metric
235: 378:0000: Note_Metric
236: 379:0000: Note_Metric
237: 380:0000: Note_Metric
238: 381:0000: Note_Metric
239: 382:0000: Note_Metric
240: 383:0000: Note_Metric
241: 384:0000: Note_Metric
242: 385:0000: Note_Metric
243: 386:0000: Note_Metric
244: 387:0000: Note_Metric
245: 388:0000: Note_Metric
246: 389:0000: Note_Metric
247: 390:0000: Note_Metric
248: 391:0000: Note_Metric
249: 392:0000: Note_Metric
250: 393:0000: Note_Metric
251: 394:0000: Note_Metric
252: 395:0000: Note_Metric
253: 396:0000: Note_Metric
254: 397:0000: Note_Metric
255: 398:0000: Note_Metric
256: 399:0000: Note_Metric
257: 400:0000: Note_Metric
258: 401:0000: Note_Metric
259: 402:0000: Note_Metric
260: 403:0000: Note_Metric
261: 404:0000: Note_Metric
262: 405:0000: Note_Metric
263: 406:0000: Note_Metric
264: 407:0000: Note_Metric
265: 408:0000: Note_Metric
266: 409:0000: Note_Metric
267: 410:0000: Note_Metric
268: 411:0000: Note_Metric
269: 412:0000: Note_Metric
270: 413:0000: Note_Metric
271: 414:0000: Note_Metric
272: 415:0000: Note_Metric
273: 416:0000: Note_Metric
274: 417:0000: Note_Metric
275: 418:0000: Note_Metric
276: 419:0000: Note_Metric
277: 420:0000: Note_Metric
278: 421:0000: Note_Metric
279: 422:0000: Note_Metric
280: 423:0000: Note_Metric
281: 424:0000: Note_Metric
282: 425:0000: Note_Metric
283: 426:0000: Note_Metric
284: 427:0000: Note_Metric
285: 428:0000: Note_Metric
286: 429:0000: Note_Metric
287: 430:0000: Note_Metric
288: 431:0000: Note_Metric
289: 432:0000: Note_Metric
290: 433:0000: Note_Metric
291: 434:0000: Note_Metric
292: 435:0000: Note_Metric
293: 436:0000: Note_Metric
294: 437:0000: Note_Metric
295: 438:0000: Note_Metric
296: 439:0000: Note_Metric
297: 440:0000: Note_Metric
298: 441:0000: Note_Metric
299: 442:0000: Note_Metric
300: 443:0000: Note_Metric
301: 444:0000: Note_Metric
302: 445:0000: Note_Metric
303: 446:0000: Note_Metric
304: 447:0000: Note_Metric
305: 448:0000: Note_Metric
306: 449:0000: Note_Metric
307: 450:0000: Note_Metric
308: 451:0000: Note_Metric
309: 452:0000: Note_Metric
310: 453:0000: Note_Metric
311: 454:0000: Note_Metric
312: 455:0000: Note_Metric
313: 456:0000: Note_Metric
314: 457:0000: Note_Metric
315: 458:0000: Note_Metric
316: 459:0000: Note_Metric
317: 460:0000: Note_Metric
318: 461:0000: Note_Metric
319: 462:0000: Note_Metric
320: 463:0000: Note_Metric
321: 464:0000: Note_Metric
322: 465:0000: Note_Metric
323: 466:0000: Note_Metric
324: 467:0000: Note_Metric
325: 468:0000: Note_Metric
326: 469:0000: Note_Metric
327: 470:0000: Note_Metric
328: 471:0000: Note_Metric
329: 472:0000: Note_Metric
330: 473:0000: Note_Metric
331: 474:0000: Note_Metric
332: 475:0000: Note_Metric
333: 476:0000: Note_Metric
334: 477:0000: Note_Metric
335: 478:0000: Note_Metric
336: 479:0000: Note_Metric
337: 480:0000: Note_Metric
338: 481:0000: Note_Metric
339: 482:0000: Note_Metric
340: 483:0000: Note_Metric
341: 484:0000: Note_Metric
342: 485:0000: Note_Metric
343: 486:0000: Note_Metric
344: 487:0000: Note_Metric
345: 488:0000: Note_Metric
346: 489:0000: Note_Metric
347: 490:0000: Note_Metric
348: 491:0000: Note_Metric
349: 492:0000: Note_Metric
350: 493:0000: Note_Metric
351: 494:0000: Note_Metric
352: 495:0000: Note_Metric
353: 496:0000: Note_Metric
354: 497:0000: Note_Metric
355: 498:0000: Note_Metric
356: 499:0000: Note_Metric
357: 500:0000: Note_Metric
358: 501:0000: Note_Metric
359: 502:0000: Note_Metric
360: 503:0000: Note_Metric
361: 504:0000: Note_Metric
362: 505:0000: Note_Metric
363: 506:0000: Note_Metric
364: 507:0000: Note_Metric
365: 508:0000: Note_Metric
366: 509:0000: Note_Metric
367: 510:0000: Note_Metric
368: 511:0000: Note_Metric
369: 512:0000: Note_Metric
370: 513:0000: Note_Metric
371: 514:0000: Note_Metric
372: 515:0000: Note_Metric
373: 516:0000: Note_Metric
374: 517:0000: Note_Metric
375: 518:0000: Note_Metric
376: 519:0000: Note_Metric
377: 520:0000: Note_Metric
378: 521:0000: Note_Metric
379: 522:0000: Note_Metric
380: 523:0000: Note_Metric
381: 524:0000: Note_Metric
382: 525:0000: Note_Metric
383: 526:0000: Note_Metric
384: 527:0000: Note_Metric
385: 528:0000: Note_Metric
386: 529:0000: Note_Metric
387: 530:0000: Note_Metric
388: 531:0000: Note_Metric
389: 532:0000: Note_Metric
390: 533:0000: Note_Metric
391: 534:0000: Note_Metric
392: 535:0000: Note_Metric
393: 536:0000: Note_Metric
394: 537:0000: Note_Metric
395: 538:0000: Note_Metric
396: 539:0000: Note_Metric
397: 540:0000: Note_Metric
398: 541:0000: Note_Metric
399: 542:0000: Note_Metric
400: 543:0000: Note_Metric
401: 544:0000: Note_Metric
402: 545:0000: Note_Metric
403: 546:0000: Note_Metric
404: 547:0000: Note_Metric
405: 548:0000: Note_Metric
406: 549:0000: Note_Metric
407: 550:0000: Note_Metric
408: 551:0000: Note_Metric
409: 552:0000: Note_Metric
410: 553:0000: Note_Metric
411: 554:0000: Note_Metric
412: 555:0000: Note_Metric
413: 556:0000: Note_Metric
414: 557:0000: Note_Metric
415: 558:0000: Note_Metric
416: 559:0000: Note_Metric
417: 560:0000: Note_Metric
418: 561:0000: Note_Metric
419: 562:0000: Note_Metric
420: 563:0000: Note_Metric
421: 564:0000: Note_Metric
422: 565:0000: Note_Metric
423: 566:0000: Note_Metric
424: 567:0000: Note_Metric
425: 568:0000: Note_Metric
426: 569:0000: Note_Metric
427: 570:0000: Note_Metric
428: 571:0000: Note_Metric
429: 572:0000: Note_Metric
430: 573:0000: Note_Metric
431: 574:0000: Note_Metric
432: 575:0000: Note_Metric
433: 576:0000: Note_Metric
434: 577:0000: Note_Metric
435: 578:0000: Note_Metric
436: 579:0000: Note_Metric
437: 580:0000: Note_Metric
438: 581:0000: Note_Metric
439: 582:0000: Note_Metric
440: 583:0000: Note_Metric
441: 584:0000: Note_Metric
442: 585:0000: Note_Metric
443: 586:0000: Note_Metric
444: 587:0000: Note_Metric
445: 588:0000: Note_Metric
446: 589:0000: Note_Metric
447: 590:0000: Note_Metric
448: 591:0000: Note_Metric
449: 592:0000: Note_Metric
450: 593:0000: Note_Metric
451: 594:0000: Note_Metric
452: 595:0000: Note_Metric
453: 596:0000: Note_Metric
454: 597:0000: Note_Metric
455: 598:0000: Note_Metric
456: 599:0000: Note_Metric
457: 600:0000: Note_Metric
458: 601:0000: Note_Metric
459: 602:0000: Note_Metric
460: 603:0000: Note_Metric
461: 604:0000: Note_Metric
462: 605:0000: Note_Metric
463: 606:0000: Note_Metric
464: 607:0000: Note_Metric
465: 608:0000: Note_Metric
466: 609:0000: Note_Metric
467: 610:0000: Note_Metric
468: 611:0000: Note_Metric
469: 612:0000: Note_Metric
470: 613:0000: Note_Metric
471: 614:0000: Note_Metric
472: 615:0000: Note_Metric
473: 616:0000: Note_Metric
474: 617:0000: Note_Metric
475: 618:0000: Note_Metric
476: 619:0000: Note_Metric
477: 620:0000: Note_Metric
478: 621:0000: Note_Metric
479: 622:0000: Note_Metric
480: 623:0000: Note_Metric
481: 624:0000: Note_Metric
482: 625:0000: Note_Metric
483: 626:0000: Note_Metric
484: 627:0000: Note_Metric
485: 628:0000: Note_Metric
486: 629:0000: Note_Metric
487: 630:0000: Note_Metric
488: 631:0000: Note_Metric
489: 632:0000: Note_Metric
490: 633:0000: Note_Metric
491: 634:0000: Note_Metric
492: 635:0000: Note_Metric
493: 636:0000: Note_Metric
494: 637:0000: Note_Metric
495: 638:0000: Note_Metric
496: 639:0000: Note_Metric
497: 640:0000: Note_Metric
498: 641:0000: Note_Metric
499: 642:0000: Note_Metric
500: 643:0000: Note_Metric
501: 644:0000: Note_Metric
502: 645:0000: Note_Metric
503: 646:0000: Note_Metric
504: 647:0000: Note_Metric
505: 648:0000: Note_Metric
506: 649:0000: Note_Metric
507: 650:0000: Note_Metric
508: 651:0000: Note_Metric
509: 652:0000: Note_Metric
510: 653:0000: Note_Metric
511: 654:0000: Note_Metric
512: 655:0000: Note_Metric
513: 656:0000: Note_Metric
514: 657:0000: Note_Metric
515: 658:0000: Note_Metric
516: 659:0000: Note_Metric
517: 660:0000: Note_Metric
518: 661:0000: Note_Metric
519: 662:0000: Note_Metric
520: 663:0000: Note_Metric
521: 664:0000: Note_Metric
522: 665:0000: Note_Metric
523: 666:0000: Note_Metric
524: 667:0000: Note_Metric
525: 668:0000: Note_Metric
526: 669:0000: Note_Metric
527: 670:0000: Note_Metric
528: 671:0000: Note_Metric
529: 672:0000: Note_Metric
530: 673:0000: Note_Metric
531: 674:0000: Note_Metric
532: 675:0000: Note_Metric
533: 676:0000: Note_Metric
534: 677:0000: Note_Metric
535: 678:0000: Note_Metric
536: 679:0000: Note_Metric
537: 680:0000: Note_Metric
538: 681:0000: Note_Metric
539: 682:0000: Note_Metric
540: 683:0000: Note_Metric
541: 684:0000: Note_Metric
542: 685:0000: Note_Metric
543: 686:0000: Note_Metric
544: 687:0000: Note_Metric
545: 688:0000: Note_Metric
546: 689:0000: Note_Metric
547: 690:0000: Note_Metric
548: 691:0000: Note_Metric
549: 692:0000: Note_Metric
550: 693:0000: Note_Metric
551: 694:0000: Note_Metric
552: 695:0000: Note_Metric
553: 696:0000: Note_Metric
554: 697:0000: Note_Metric
555: 698:0000: Note_Metric
556: 699:0000: Note_Metric
557: 700:0000: Note_Metric
558: 701:0000: Note_Metric
559: 702:0000: Note_Metric
560: 703:0000: Note_Metric
561: 704:0000: Note_Metric
562: 705:0000: Note_Metric
563: 706:0000: Note_Metric
564: 707:0000: Note_Metric
565: 708:0000: Note_Metric
566: 709:0000: Note_Metric
567: 710:0000: Note_Metric
568: 711:0000: Note_Metric
569: 712:0000: Note_Metric
570: 713:0000: Note_Metric
571: 714:0000: Note_Metric
572: 715:0000: Note_Metric
573: 716:0000: Note_Metric
574: 717:0000: Note_Metric
575: 718:0000: Note_Metric
576: 719:0000: Note_Metric
577: 720:0000: Note_Metric
578: 721:0000: Note_Metric
579: 722:0000: Note_Metric
580: 723:0000: Note_Metric
581: 724:0000: Note_Metric
582: 725:0000: Note_Metric
583: 726:0000: Note_Metric
584: 727:0000: Note_Metric
585: 728:0000: Note_Metric
586: 729:0000: Note_Metric
587: 730:0000: Note_Metric
588: 731:0000: Note_Metric
589: 732:0000: Note_Metric
590: 733:0000: Note_Metric
591: 734:0000: Note_Metric
592: 735:0000: Note_Metric
593: 736:0000: Note_Metric
594: 737:0000: Note_Metric
595: 738:0000: Note_Metric
596: 739:0000: Note_Metric
597: 740:0000: Note_Metric
598: 741:0000: Note_Metric
599: 742:0000: Note_Metric
600: 743:0000: Note_Metric
601: 744:0000: Note_Metric
602: 745:0000: Note_Metric
603: 746:0000: Note_Metric
604: 747:0000: Note_Metric
605: 748:0000: Note_Metric
606: 749:0000: Note_Metric
607: 750:0000: Note_Metric
608: 751:0000: Note_Metric
60
```

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 sig-

nature 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. *Semiotica 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 inter-college 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.