Proceedings of the 7th International Conference on Live Coding (ICLC 2023)
April 19th - 23rd – Utrecht, Netherlands

Reproducible Musical Analysis of Live Coding Performances Using Information Retrieval: A Case Study on the Algorave 10th Anniversary

Georgios Diapoulis
Chalmers University of Technology, University of Gothenburg
georgios.diapoulis@chalmers.se

Martin Carlé
Ionian University, Hub of Art Laboratories
mc@aiguphonie.com

ABSTRACT

We present a reproducible music information retrieval (MIR) study on 133 performances from the 10th anniversary of Algorave. Our aim in this paper is to provide a reproducible framework for computational analysis of musical performances. Here, we present a tool for analysing acoustical characteristics and for visualizing the musical structure from performances of one algorave event. Our musical analysis of the live coding performances highlights the musical diversity within the live coding community to a broader scientific audience. At the same time, we expect that the algoravers will gain insights on their own musical practices through the computational analysis of the musical structure of their performances. In concerning ourselves with reproducibility, our intention is to motivate more researchers to analyse musical practices of other under-represented music communities. As a basic tool for reproducibility we construct a pipeline for analysing performances using Python within a Jupyter notebook. To make this reproducible on different computers we wrapped the whole workflow setup into a docker image. We represent the results of our analysis as a series of plots of different kinds. These plots present both overviews of the entire repertory in compact form, and comparisons of individual pieces in more detail. In learning one can use such visualization as a means for raising awareness on one's evolution of the musical outcome. In performance this visualization can be developed to a real-time and possibly an interactive tool which informs the coder about the musical outcome of a live set on-the-fly. Finally, we reflect on how and to what extent such MIR studies can provide valuable insights in live coding performance practices, while also considering the limitations faced when dealing with such large parameter spaces in human machine musicianship.

1 Introduction

An algorave party aims at celebrating the act of live coding by hosting and promoting live music events across the world. Live coding is a performance practice where a domain specific programming language is used for music-making (Collins et al. 2003). Typically, live coders share their screen with the audience with the intention to make the process of music-making technically transparent to the listeners. Moreover, an algorave party is a political act in physically displacing academic music concerts from lecture halls to the dancefloors of nightclubs. During a live coding performance we allow ourselves to fail (Knotts 2020). Uncertainty and instability during a performance is common and the audience welcomes technical errors and programming mistakes during performance (Armitage 2018). These are attitudes also shared in other improvisation contexts.

Here, we present an analysis of recordings from the 24 hours stream of Algorave 10, retrieved from the Internet Archive[1]. The event was shortly announced only a few days before the happening by Alex McLean on the mailing list of TOPLAP[2] and the TOPLAP's chat server. The 144 available slots were filled within the next days and 133 recordings were retrieved from the archives. The aforementioned welcoming ethos of the live coding community was present vibrantly in the chat of the streaming provider and on other channels (e.g. TOPLAP Discord). All five continents were represented and a broad variety of performance practices and systems were exhibited during the 24 hours live streaming.

Our approach aims at a reproducible framework for music information retrieval of the Algorave 10th anniversary. The goal is to offer a framework that live coders can easily use to conduct musical analysis of performances and gain further insights into decoding a live set. We start by presenting the background of algorave parties and provide generic statistics of events taking place so far. We continue with the specifics of the Algorave 10th Birthday Party and offer descriptive statistics of the performances. Furthermore, we discuss our musical structure analysis from a cognitive MIR perspective and introduce a visual representation of the analysed musical forms. Finally, we review the structure of the reproducible framework of this study.

2 Background of algorave acts

The term algorave has been presented as a descriptor of a musical genre welcoming newcomers and underrepresented groups (Armitage 2018). Diversity and the live coding ethos are particularly important to the community. Accordingly, women live coders have been actively present since the founding act of TOPLAP ("TOPLAP ManifestoDraft," n.d.), albeit still forming a minority in the early days. Today, there are several bands, workshops, algorave events, TOPLAP nodes, and research programs (Sicchio 2014), where women are leading figures. For instance, the Hydra environment[3] for visual live coding, widely used in algoraves, has been developed by Olivia Jack.

The main impact of algoraves is to provide an incentive for people to get together and celebrate live coding dance music. As dancing involves the body ontological relations between concepts and sounds are further emphasised. In particular, drawing relationships between thoughts and sounds in electronic music is not a given thing (see algorithms in TOPLAP manifesto). There is a long discussion about whether electronic sounds can have an embodied meaning (Dahlstedt 2018), especially when the sound source has been disconnected from the musical instruments, transferred to electronic circuits, or reproduced through loudspeakers.

2.1 Historical accounts of algorave parties

An algorave is typically an event where people get together to enjoy live coding acts while dancing to the beat. The first algorave was in London in 2012 (Collins and McLean 2014). Since this kick-off event, many physical and virtual algoraves have been organised worldwide with the help of the algorave official website ("Algorave," n.d.). The 10th anniversary of the first algorave party was celebrated worldwide with numerous physical events and a 24 hours live stream. Each performance had a time slot of 10 minutes which is typically considered a short performance, just within the limits of being a challenging endeavour (Baalman 2020). This indicates how well the community has progressed since its beginnings in the early 2000s and suggests a grown maturity of the available live coding tools.

The Algorave page[4] lists 322 past events since the first algorave in 2012, and one future event that will accompany the ICLC 2023 conference this year. A crucial aspect of algoraves is that the events take place in a physical venue where people can get together and dance. A total of 21 out of the 322 events have been reported as being 'online' or on 'the internet'. The pandemic was a turning point for online events. Before, only two live-streamed sessions were reported. The first event was a hybrid algorave live-streamed from Sheffield in 2016 and the second was the 5th anniversary in 2017. The Algorave 10th anniversary live stream took place between 2022/03/19 and 2022/03/20, and was accompanied by several local events celebrating the occasion with the motto "weareten".

Collins and McLean (2014) presented the first study on algoraves. They show an estimate of how many people were dancing at each event out of a total of 18 algoraves, from 2012-03-17 to 2014-04-26. While it can be difficult to keep an objective account of how many people are dancing on algoraves, it is interesting to see the audiences' engagement with dance raising over the years. A way to approach this question would be to perform some video analysis on any recordings from algoraves and provide an estimate of the overall overt movement of the participants.

[1] https://archive.org/details/toplap
[2] https://forum.toplap.org/t/algorave-turns-10/1882/42
[3] https://github.com/hydra-synth
[4] https://algorave.com/ – Accessed 2022-12-12

2.2 Descriptive statistics on the 24 hours stream of the Algorave 10

The 10th anniversary had international coverage with 13 physical and virtual events. The events are mainly located in Europe and North America. Indicatively, in 2021 there were 15 live events in total, and in 2022 the total number of events is 28 so far. The motto "weareten" is explicitly written in several announcements. It is important to notice here that live coding in performing arts is about to become 20 years old, whereas the expression of live coding as dance in nightclubs is only 10 years old.

Throughout the 24 hours happening, the "Algorave 10th Birthday Party" was accompanied by a smooth and continuous live streaming of heterogeneous performances. Given the decentralized and flat organization of the community, we would like to draw special attention to the well-coordinated organization and execution of the event. Inevitably, different time slots served different time zones better. This led a clustering of locations within the 24 hours streaming. The event's coverage was worldwide and engaged musicians from all five continents (Africa, America, Australia, Asia, and Europe). Table 1 shows the number of performances per continent. There was one cross-continental performance between Barcelona and Toronto (slot #104) and one worldwide performance (slot #17)[5], where several live coders spontaneously entered Estuary[6] and started an on-the-spot improvisation.

EU	NA	SA	AS	OC	AF	Total
63	27	20	20	2	1	133

Table 1: The number of performances in Algorave 10 per continent. (EU: Europe, NA: North America, SA: South America, AS: Asia, OC: Oceania, AF: Africa).

Various programming languages and performance setups across the live coders can be noted. Table 2 accounts for the occurrences of the different live coding languages. The total sum of the noted systems equals the number of performances in the current analysis. Several performances used hybrid systems with more than one programming language. As annotated, we report the main language used during the performances. It is furthermore important that many systems also incorporated live coding visuals. In particular, Hydra was employed in many performances. Many video descriptions do not explicitly state the programming language used. Thus, Table 2 should be seen as the overall tendency of the performers' preferences for generating sound. Several performance setups also incorporated hardware synthesizers and other equipment. Most notably, a "game boy(girl)" by pulu, but also musical instruments, like electric guitars. Visual languages and any hardware controllers of musical instruments used during performances are not reported in our annotation.

LANGUAGE	Annotated
Tidal Cycles	52
SuperCollider	18
Sonic Pi	14
FoxDot	8
Orca	7
CSound	1
Chuck	1
MAX/MSP	1
Other	8
N/A	11
—————	————
TOTAL	121

Table 2: Annotated counts on the participants preferred music-making software programming environment.

[5] http://ten.algorave.com – Accessed 2022-05-10

[6] https://archive.org/details/algorave-10-equinox-open-jam

3 Characteristics of musical form

We begin by presenting theoretical backgrounds for the musical form of a performance from the perspective of music cognition. Then we discuss the parameters of the musical form and continue to build on this knowledge in order to construct a circular representation showing the temporal evolution of a music performance. Lastly, we discuss the benefits and limitations of this approach.

Musical form depends on our memory capacity. Musical memory operates on three main timescales, the so-called echoic memory for sounds with a duration of less than 0.5s, the short-term memory (STM) for sound events between 0.5-8s and the long-term memory (LTM) for sound events that occur on larger timescales, typically more than 15s (Snyder 2000). These different levels of organization surface differently when listening, imagining and playing music. For instance, during a music performance, the musician has to be attentive to both, the micro-structure of a musical phrase, and also to larger musical segments. In the context of a so-called canonical live coding performance (Roberts and Wakefield 2018), the coder is mostly applying changes to the musical outcome on larger timescales (15s and more). Few live coding environments may enable the musician to apply fast changes, like ixi-lang (Magnusson 2011) which was developed with the goal to make code modification within less than 5 seconds possible. A typical MIR architecture extracts low-level acoustical features. These features share some similarities with our auditory perception. The formation of pitch, for example, takes place within 10ms when listening to the acoustic environment. Above this level, our perceptual capacities group events together in order to structure perceptual boundaries (Bregman 1994). Indicatively, the gestalt principles, like the principle of proximity, similarity and good continuation are excellent examples of perceptual organisations (Schnupp, Nelken, and King 2011).

Several MIR architectures make use of the knowledge taken from the workings of our music cognition. For instance, computational segmentation of sounds is a typical example of a MIR task that employs music perception and cognition knowledge. Musical structure analysis uses it to identify different sounds, music excerpts or even songs. How different segments get organised is also important. Accordingly, pulse and meter are the first steps towards organizing motifs and themes. That is how we achieve periodic groupings of adjacent and nonadjacent elements (Parncutt 1994). Studying musical form requires to take both low-level and high-level music percepts into account. A typical cognitive approach of MIR begins with the low-level features to predict higher-level features. Consequently, a musical form can be described by two family characteristics: primary parameters, including pitch, rhythm and harmony, and a secondary parameters which include loudness, tempo, event density and timbre characteristics, among others (Snyder 2000). In this study we are combining these approaches. We start by extracting low-level acoustical features to synthesize higher-level musical structures and incorporate pitch-based and rhythm-based features to account for the musical form of a live coding performance.

3.1 Musical structure analysis in MIR

Musical structure analysis (MSA) in audio-based MIR focuses on objective descriptors of the signal. This is a necessary step when attempting to understand what the main acoustical characteristics of a composition are. Qualitative descriptors or human annotations typically complement these objective ones. The latter may correspond to information like labelling musical sections, such as verse, refrain, or annotating onsets and offsets of musical events, and so on. Furthermore, current state-of-the-art analyses are based on novelty, repetition and homogeneity (Klapuri, Paulus, and Müller 2010). Some of these approaches use similarity-based representations, such as MFCCs similarity matrices, to represent the temporal evolution of the audio. Studies on MSA have shown that the main challenges are related to subjectivity, ambiguity and hierarchy (Nieto et al. 2020). A typical example of "subjectivity" relates to how human annotations are collected. For instance, a noisy data set is a crowd-sourced annotation strategy. Respectively, ambiguity is related to the subjective percepts of the same annotator. Thus, when listening to a sound excerpt repeatedly, the annotator may be undecidable for a task such as music segmentation.

3.2 Primary and secondary parameters of musical form

3.2.1 Primary parameters

Primary parameters exhibit proportional relationships between them (Snyder 2000, 195). For instance, we do categorizations of pitch intervals, and we make subdivisions of rhythmic structures. The same does not apply to loudness, and pitch is independent of loudness. We perceive roughly the same pitch quality regardless of how loud a sound is. Pitch emerges when sound events range between 10-100ms and demonstrate stable perceptual qualities. For instance, we can easily understand whether a pitched sound has a higher pitch than another. Tonality and harmony are pitch-dependent, and when concerning music, we often discuss the fundamental note of a sequence. Sometimes an exact boundary between primary and secondary parameters is unclear. The mel-frequency scale is a pitched-based perceptual scale which describes how humans perceive equally distanced pitch intervals. A common acoustical descriptor is computed on the mel-scale. The so-called mel-frequency cepstral coefficients (MFCCs) have been used in many applications, like music genre classification, similarity measures, spectral envelope and speech recognition.

Rhythm is defined when "two or more events take place within the length of short-term memory" (Snyder 2000, 159). Rhythm emerges from repetition of sound events where pulse and event density are particularly important. In the context of musical structure, rhythm is expressed in hierarchical organizations and we learn and categorize different rhythms based on cultural characteristics, music training and musicianship.

3.2.2 Secondary parameters

Secondary parameters may be seen as by-products of primary characteristics. For instance, the spectral centroid, an acoustical feature which can describe the brightness of a sound, may be a by-product of pitched events. Brightness has a typical range of 2000-4000Hz which can be an important characteristic for periodic and transient signals. Pulse and event density are fundamental components of our rhythm perception. Pulse demonstrates an endogenous periodicity that is not strictly periodic but exhibits temporal fluctuations (Large and Snyder 2009). Beat depends on rhythmic complexity and has been suggested to play an important role in the perception of groove. Meter exhibits hierarchical organization and depends on the pulse, although there are cultural variances (Holzapfel 2015).

3.3 Musical form in live coding

Little research has been done on musical form and its characteristics in live coding. To some extent, this is reasonable as live coding is mainly an improvisational practice, free of rigid form characteristics. Recently a study examined two different aspects of visualisations during live coding practice (Dal Ri and Masu 2022), namely as a linear representation of events and as a density plot. The study suggests that the two approaches can complement each other. Magnusson examined how visual notation relates to code representation and how this is reflected in the temporal evolution of musical events. He also discussed how this knowledge was used for developing Threnoscope (Magnusson 2015).

4 Circular visual representation for STM windows

Given the above mentioned definition of rhythm focusing on our musical memory, we here present a circular visualization for structural representations of music performances. Our motivation is to represent performances of unequal lengths. Although all performances in this study had approximately a duration of 10 minutes, they all were still slightly different to each other. Therefore we present a circular representation to demonstrate the evolution of the Algorave 10 performances and we divide the circle into an arbitrary number of equally-distanced slices. Each slice has a maximum duration similar to the upper limit of our STM which is 8 seconds. The circular plot can represent both, pitch-related and rhythm-related characteristics combined. It may be used to represent the temporal evolution of one feature at a time. This is based on the assumption that each performance can be divided into an arbitrary amount of discrete segments. In this specific case, we divide the circular representation into 72 segments. As a result, the perceptual continuity between segments is imposed forcefully but it nonetheless facilitates our aim to communicate the temporal evolution of highly diversified performances. This is because several of the gestalt principles, like the principles of good continuation, common fate and similarity, are applied to the proposed visualization. The segments are equally-distanced from each other and their values are local descriptors which were calculated based on the mean feature values. Any other global descriptor can also be applicable, like higher-order statistical moments. Each segment is visually represented with a slice, as shown in Figure 1. An alternative approach to transforming all performances to equal lengths would need to involve dynamic time warping or zero padding. Dynamic time warping is a computationally expensive technique for our use case and doesn't serve well the reproducibility goals of the present study. Zero-padding could be useful for computations in the frequency domain. So, the circular representations here may be seen as an alternative approach to a frequency-domain representation, though they are not equivalent and should not be confused. Furthermore, we average real numbers per segment (feature values) and plot them in a circular representation to demonstrate their evolution over time.

In Figure 1 we map pitch- and rhythm-based features. Essentially, the plot is a counter-clockwise representation of the temporal evolution of a 10 minutes recording. The shift into the performer's perspective is done by representing the time as chunks (slices) of STM windows. The plot requires a three-dimensional representation. Time is the first dimension and two more dimensions show the feature values. Here, we show one pitch-based and one rhythm-based feature. Specifically, time is shown as a function of angles/radians (from $0\% - 100\%$), the pitch-based feature is shown on the colourmap (spectral centroid) and the rhythm-based feature is a function of the radius (onsets strength). The features are extracted using the Librosa library in Python (McFee et al. 2015).

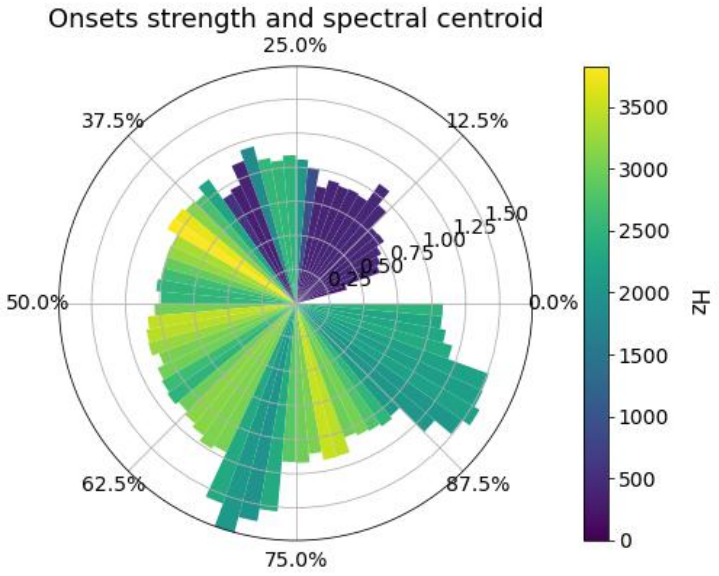

Figure 1: *Circular representation of a performance. The time axis is represented in radians, here shown in percentages. Each slice on the rose polar plot represents one STM window* $(0.5 - 8s)$*. The length of each slice shows the onsets strength, which may be seen as an indication of the density of the events. The heatmap shows the range of values for the spectral centroid.*

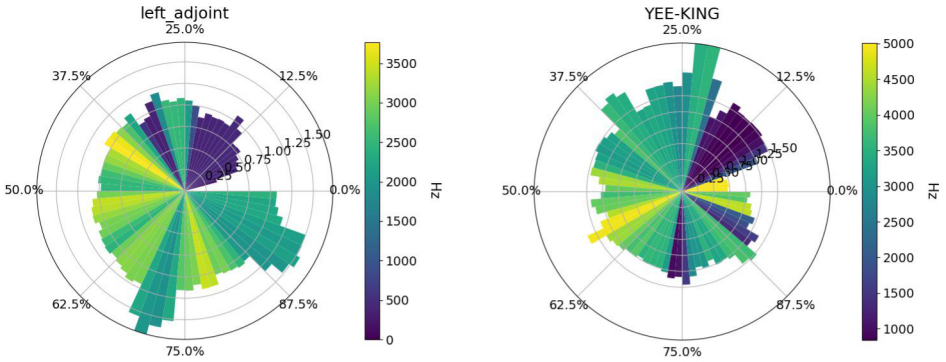

Figure 2: *A comparison of two performances, by left_adjoint and YEE-KING. The dimensions are the same as those in Figure 1. Please notice that the range of values for the acoustical features of onsets strenght and spectral centroid differ between the two performances.*

4.1 A closer look on two live sets from Algorave 10

Figure 2 shows two performances from Algorave 10, by left_adjoint[7] and YEE-KING[8]. The two cases were selected because they demonstrate a rich musical evolution over time. The plots show the same acoustical features as Figure 1, onsets strength on the radius and spectral centroid on the heatmap. Both performances begin with no sound during the first seconds of the performances. This is represented accurately with no sound events in the case of the performance by left_adjoint. In contrast, in YEE-KING's plot, some high-pitched background noise is adequate and is shown with high values of the spectral centroid, an indicator of the brightness of a sound and a relatively small density of sound events.

Both performances show a similar evolution in the first two minutes (see from 0% - 20%) with a low density of musical events and relatively low brightness values. During this period, left_adjoint conducts a so-called 'blank slate' live coding while experimenting with muffled synth and drum sounds. Later (~25%), some brighter drum sounds, like cymbals, are introduced, and the indicator of event density (onsets strength) is slightly increased. At around 40% of left_adjoint's performance, there is a climax in the brightness of the musical outcome caused by multiple cymbal sound events. Similarly, YEE-KING's performance demonstrates a climax on the onset strength in the region of 25% of the total performance time. In the second half of the performance (50% - 100%), both cases exhibit great amounts of variability

[7] https://archive.org/details/algorave-10-left-adjoint
[8] https://archive.org/details/algorave-10-yee-king

in the feature values. In the case of left_adjoint, two overt climaxes are shown of the event density indicator which is surrounded by variations between mid and high-frequency values for the spectral centroid. In the case of YEE-KING performance, the variations on the indicator of event density are subtle but there are significant variations on the spectral centroid.

5 Reproducible analysis of the Algorave 10 performances

Reproducibility with regard to both, information retrieval and analysis, has been realised by employing docker. In addition, the original videos were retrieved from the Internet Archive and placed in an online git repository for large files[9]. Since any changes on the git-lfs repository can be transparently tracked, the latter practice suits reproducibility better than retrieving the videos from the Internet Archive. However, a drawback of the git-lfs solution is that a configuration file with unique references to the original material is required. Another one it that repository provides put a limit on the download bandwidth which in our case means that after seven full downloads per month slower downloads are to be expected. We still consider this approach the better practice but remain open to any feedback from the community.

The `Dockerfile`, the recipe to build a container image, is made available in a GitHub repository[10]. The analysis presented in this study can be reproduced using two methods: i) run the prebuilt image from docker hub[11] which is the recommended method, or ii) build the image from the Dockerfile yourself[12]. There are three main steps to reproduce our results. The first step is to retrieve the recordings of the Algorave 10 performances from the mentioned git-lfs repository. We provide a Python script for the video-to-audio conversion. But the audio files are also readily available in mp3 format with as sampling rate of 22050 Hz. The second step extracts the acoustical features from the audio files, while the third step analyses and plots the acoustical features.

At the end of the building process of the docker image, the user can access a Jupyter Lab notebook demonstrating the descriptive statistics and three python scripts that render the plots of this study.

5.1 Acoustical feature extraction

The acoustical feature extraction is taking place during the building process of the docker image while the extracted features are stored in a data file. This file can then be accessed within the Jupyter Lab notebook.

For each performance we extracted 9 time series of acoustical features and one global descriptor for the estimated tempo: three rhythm-based features (onsets strength, tempo and pulse) and seven pitch-based features or feature vectors (20 MFCCs, spectral centroid, spectral roll-off, spectral flatness, spectral bandwidth, 7 features of spectral contrast and a pitch estimator of the fundamental frequency of a window). The pitch estimator showed many missing values ($\sim 40\%$ of the windows).

We mapped each feature to the circular representation as a function of time. Corresponding to the STM window of 8s, we computed a mean within each of 72 segments (as the actual low-level analysis window is 2048 samples). Finally we plotted each feature for 121 performances (as shown in Figure 3). Specific performances were excluded from the analysis as outliers. There were two main criteria for identifying outliers: i) performances which did not begin at the scheduled time, and ii) performances that showed low values of the onsets-related acoustical feature. The two main rhythm-based features, pulse and onsets strength (an indicator of event density) show similar distributions for every performance. Some pitch-based features, such as the spectral roll-off and the spectral bandwidth, show correlations. The two main rhythm-based features are also correlated with each other.

5.2 Estimated tempos of the algorave performances

Figure 4 show a histogram and a box plot of tempos for all the performances. The average tempo is 122 ± 18 bpm and is matching the so-called preferred tempo. The tempo range is $81 - 161$ bpm. The performances had an average duration of $574 \pm 42s$. This average duration results in an approximate value of $8s$ for each one of the 72 slices, which matches the upper limit of our STM.

[9] https://github.com/gewhere/algorave10-large-files
[10] https://github.com/gewhere/iclc2023
[11] https://hub.docker.com/r/algorave10/iclc2023
[12] Building the image from the Dockerfile require several hours. In contrast, the prebuild docker image runs within few seconds.

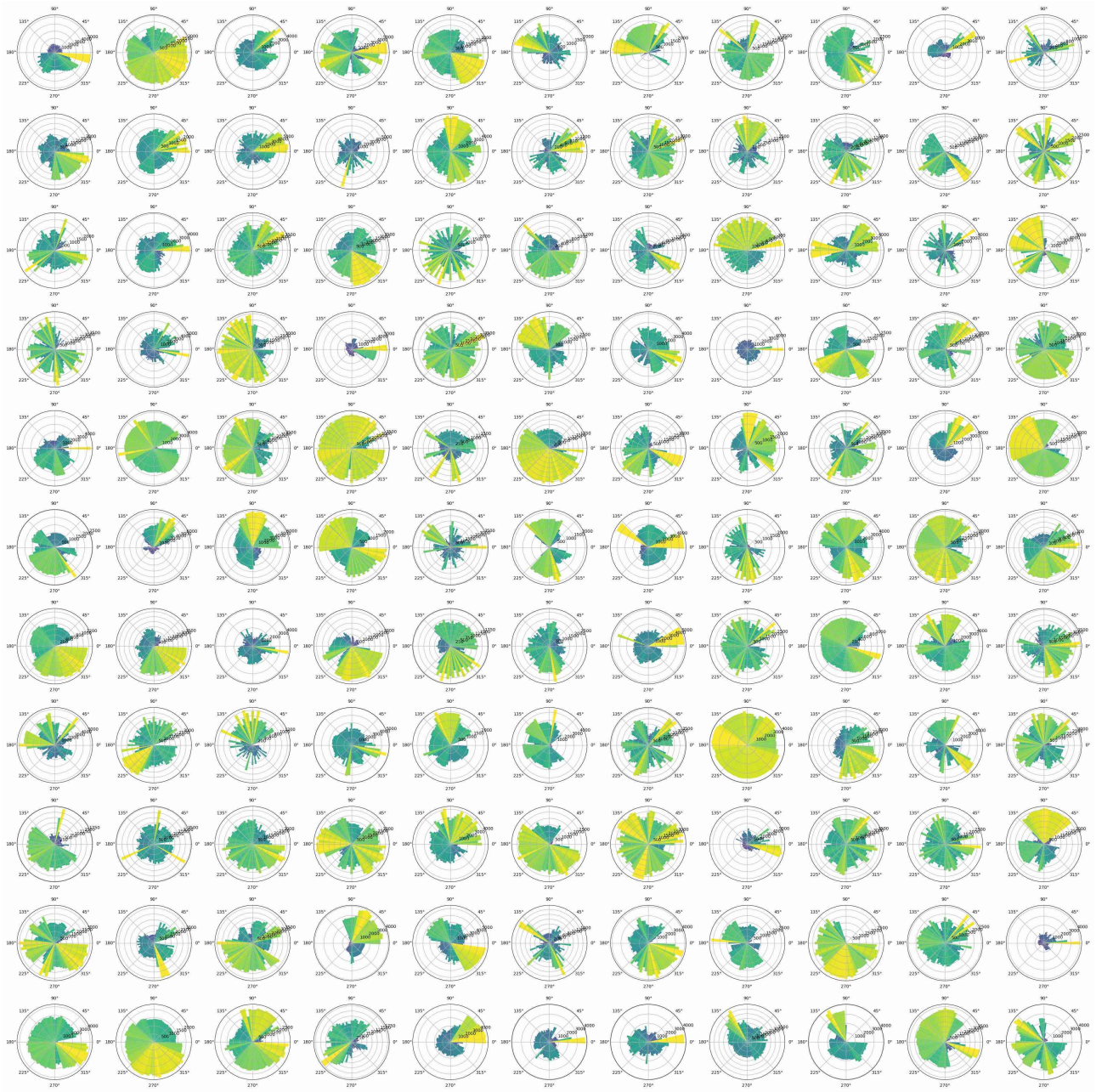

Figure 3: *Polar plots for all 121 live coding performances. The plots are two dimensional, in contrast to Figure 1, and show the temporal evolution of the spectral centroid. Every plot is normalized for each performance individually.*

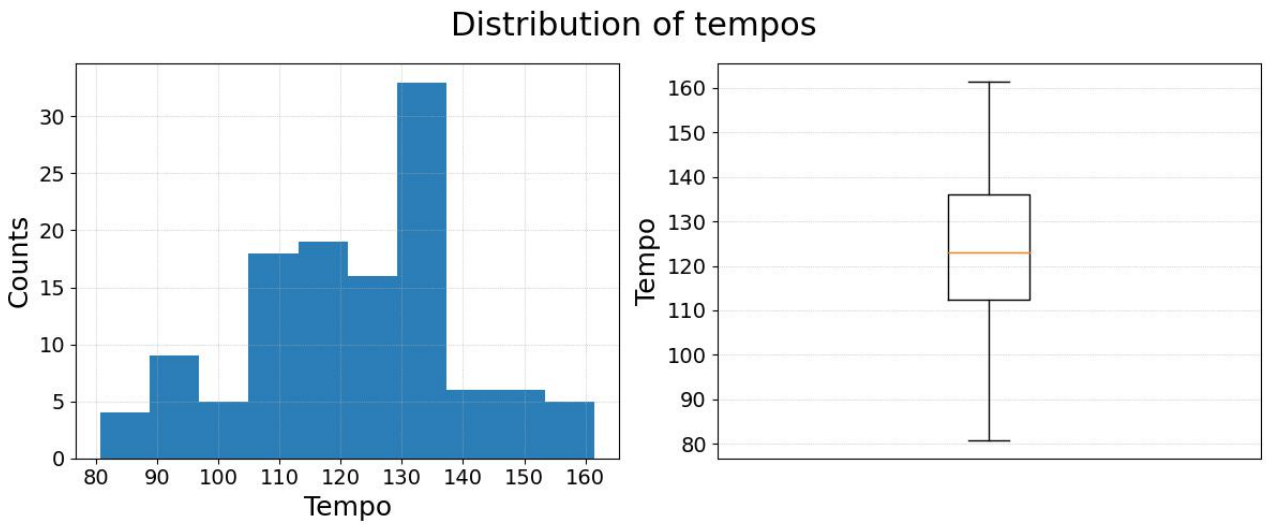

Figure 4: *Tempos for all the 121 performances. Left hand side shows a histogram of the tempos and the right hand side shows a box plot with the median and the 1st and 3rd quartiles.*

6 Discussion

The present study attempts to supply a transparent process for systematic computational analysis of live coding performances. As accessibility is an important value for the live coding community, we provide a reproducible analysis based on docker that launches a jupyter notebook for the computational analysis of the study. The raw data set which was retrieved from the Internet Archive has an audio-visual format from all the recordings carried out during the Algorave 10, 24 hours streaming. We copied the Internet Archive Algorave 10 collection to a git-lfs repository resulting in a transparent and computationally robust reproducibility method. As a beneficiary side-effect, this method offloads computational resources from the Internet Archive. Our study reveals first insights into the audio data set. To access further details of the live coding experience, one may look into computer vision analyses of the video material and how the outcome correlates with the acoustical features.

The live coding community acknowledges that there is no school for learning how to live code (Nilson 2007). Consequently, studying the musical structures of different performances can enable valuable comparisons of the musical forms created in live coding. To this end, the study offers a useful tool for studying the musical structure of live coding performances and provides a testbed for exploring more than the musical outcome. In particular so, as the video data are also made available, the basis for a multimodal information retrieval study is ready at hand. It is important for us to communicate our findings with the community and make the respective insights available to everyone interested in live coding practices. For this reason, we paid special attention to a quick and economic reproducibility of our study and offer a template to instantly start out with the MIR.

The produced diagrams allow for quick perception of similarity. For instance, the circular representation makes it easy to identify where a climax in the musical features occurs during a performance. In turn, this enables to visually recognise whether two performances share similar strategies in the evolution of the musical structure. We regard this as our main contribution and we also see potential for interactive applications. This circular visualization can convey valuable information of the evolution of cognitive processes and is used to communicate both, low-level and higher-level musical structures. For instance, the plots may be used to display perceptually significant changes on the level of our STM or to display larger musical segments, such as motifs and themes. To achieve this, it may be as easy as to highlight a quarter of the circular diagram. The capacity to easily spot out perceptually relevant information, such as climaxes, is an important feature, since a live coding session is a cognitively demanding task while resources are scarce during a performance.

Moreover, the proposed circular representation can be a valuable tool in teaching and practising live coding. For instance, it is acknowledged that cognitive load is high during a live coding performance (McLean 2014), but live coding is also seen as a technique which may sharpen attentional capabilities (Nilson 2007). While learning how to live code, the performer has to cope with many hard mental operations that may counteract the perceived flow during a performance (Nash and Blackwell 2014). Hence, an interactive visualization of the evolution of a live coding session may be useful, especially in the case when practising live coding. Such an interactive application could be seen as an informative real-time prop that reminds the coder about the stationary evolution of the acoustical features.

Besides the concrete case of circular diagrams, the study offers a playground for exploring similarities and differences between performances, live coding systems and languages. Further analyses including the video material can help to examine how programming language notations are related to musical structure. Such analyses can be valuable for making informed decisions in developing live coding idioms and languages.

Information retrieval studies in live coding are limited. Xambó, Lerch, and Freeman (2018) has proposed a theoretical framework for information retrieval studies in live coding focusing on real-time processes. Tools for such real-time applications have been available for over a decade (Collins 2011) and they overlap to some extent with technologies of musical agents (Xambó 2021). The approach we present here differs from such real-time and performative applications in focusing on an offline system for analyses of live coding musical performances. Such a reproducible testbed may help us to identify patterns during a live coding performance, ultimately leading to interactive tools for real-time applications. Indicatively, we envision an interactive application for our proposed circular visual representation of the musical form which might become a tool of descriptive notations (Magnusson 2015). More applications may come out of this engagement with a similar reproducible approach.

The diversity and the ethos of the community encourage open practices and transparent procedures. Likewise, accessibility has been addressed in live coding (Skuse 2020; Vetter 2020). Design and display technologies have been reported as key facilitators for disabled musicians. Whether the circular representation presented here may facilitate learning and interactive explorations in this direction is an open question. We would like to point out that we regard the circular representation as equally useful for both, disabled and non-disabled musicians. For example, in the case of visually impaired and blind musicians, we can imagine a tangible shape-changing interface for representations of the musical structure.

7 Conclusion

We present an information retrieval study of the Algorave 10, 24 hours live stream. A total of 133 performances with a maximum duration of 10 minutes each were retrieved from the Internet Archive and mirrored to a git repository, based on git for large files. We present a reproducible study using the docker containerisation infrastructure as to offer a playground for the creative exploration of information retrieval techniques applied to live coding performances. The computational analysis is based on audio data but the original recordings also include video data which can provide further possibilities for in-depth analyses of the performances. On the basis of our engagement with the dataset, we can provide descriptive statistics about the Algorave 10 and a circular visual representation for the musical form of live coding performances. This visual representation was constructed by considering aspects of music perception and memory. We expect that such visual representation will be useful for live coders and may have applications in teaching and practising musical live coding.

8 Acknowledgements

This contribution has been partially funded through the financial support of the project "ΔΗΜΙΟΥΡΓΙΚΟΣ ΚΟΜΒΟΣ ΤΕΧΝΩΝ *MIS :5047267*" code 80504, ΕΣΠΑ 2014-2020, ΕΠΑνΕΚ; HAL (Hub of Art Laboratories), co-financed by Greece and the European Union and implemented at the Ionian University, Corfu.

References

"Algorave." n.d. https://algorave.com.

Armitage, Joanne. 2018. "Spaces to Fail in: Negotiating Gender, Community and Technology in Algorave." *Dancecult: Journal of Electronic Dance Music Culture* 10 (1).

Baalman, Marije. 2020. "Marije Baalman performing 10 minute live coding challenge at Creative Coding Utrecht." https://marijebaalman.eu/projects/code-livecode-live.html. 2020.

Bregman, Albert S. 1994. *Auditory Scene Analysis: The Perceptual Organization of Sound.* MIT press.

Collins, Nick. 2011. "SCMIR: A SuperCollider Music Information Retrieval Library." In *ICMC*.

Collins, Nick, and Alex McLean. 2014. "Algorave: Live Performance of Algorithmic Electronic Dance Music." In *Proceedings of the International Conference on New Interfaces for Musical Expression*, 355–58.

Collins, Nick, Alex McLean, Julian Rohrhuber, and Adrian Ward. 2003. "Live Coding in Laptop Performance." *Organised Sound* 8 (3): 321–30.

Dahlstedt, Palle. 2018. "Action and Perception: Embodying Algorithms and the Extended Mind." In *The Oxford Handbook of Algorithmic Music*, 41–66. Oxford University Press.

Dal Ri, Francesco Ardan, and Raul Masu. 2022. "Exploring Musical Form: Digital Scores to Support Live Coding Practice." In *NIME 2022*. PubPub.

Holzapfel, André. 2015. "Relation Between Surface Rhythm and Rhythmic Modes in Turkish Makam Music." *Journal of New Music Research* 44 (1): 25–38.

Klapuri, Anssi, Jouni Paulus, and Meinard Müller. 2010. "Audio-Based Music Structure Analysis." In *ISMIR, in Proc. Of the Int. Society for Music Information Retrieval Conference*.

Knotts, Shelly. 2020. "Live Coding and Failure." *The Aesthetics of Imperfection in Music and the Arts: Spontaneity, Flaws and the Unfinished*, 189.

Large, Edward W, and Joel S Snyder. 2009. "Pulse and Meter as Neural Resonance." *Annals of the New York Academy of Sciences* 1169 (1): 46–57.

Magnusson, Thor. 2011. "The Ixi Lang: A Supercollider Parasite for Live Coding." In *ICMC*.

———. 2015. "Code Scores in Live Coding Practice." In *Proceedings of the International Conference for Technologies for Music Notation and Representation, Paris*. Vol. 5.

McFee, Brian, Colin Raffel, Dawen Liang, Daniel P Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. 2015. "Librosa: Audio and Music Signal Analysis in Python." In *Proceedings of the 14th Python in Science Conference*, 8:18–25. Citeseer.

McLean, A. 2014. "Stress and Cognitive Load." *Collaboration and Learning Through Live Coding*.

Nash, Chris, and Alan Blackwell. 2014. "Flow of Creative Interaction with Digital Music Notations."

Nieto, Oriol, Gautham J Mysore, Cheng-i Wang, Jordan BL Smith, Jan Schlüter, Thomas Grill, and Brian McFee. 2020. "Audio-Based Music Structure Analysis: Current Trends, Open Challenges, and Applications." *Transactions of the International Society for Music Information Retrieval* 3 (1).

Nilson, Click. 2007. "Live Coding Practice." In *Proceedings of the 7th International Conference on New Interfaces for Musical Expression*, 112–17.

Parncutt, Richard. 1994. "A Perceptual Model of Pulse Salience and Metrical Accent in Musical Rhythms." *Music Perception* 11 (4): 409–64.

Roberts, Charlie, and Graham Wakefield. 2018. "Tensions and Techniques in Live Coding Performance."

Schnupp, Jan, Israel Nelken, and Andrew King. 2011. *Auditory Neuroscience: Making Sense of Sound*. MIT press.

Sicchio, Kate. 2014. "Hacking Choreography: Dance and Live Coding." *Computer Music Journal* 38 (1): 31–39.

Skuse, Amble. 2020. "Disabled Approaches to Live Coding, Cripping the Code." In *Proceedings of the International Conference on Live Coding*, 5:69–77.

Snyder, Bob. 2000. *Music and Memory: An Introduction*. MIT press.

"TOPLAP ManifestoDraft." n.d. https://toplap.org/wiki/ManifestoDraft.

Vetter, Jens. 2020. "WELLE-a Web-Based Music Environment for the Blind." In *Proceedings of the International Conference on New Interfaces for Musical Expression. Birmingham, United Kingdom*, 701–5.

Xambó, Anna. 2021. "Virtual Agents in Live Coding: A Short Review." *arXiv Preprint arXiv:2106.14835*.

Xambó, Anna, Alexander Lerch, and Jason Freeman. 2018. "Music Information Retrieval in Live Coding: A Theoretical Framework." *Computer Music Journal* 42 (4): 9–25.