METRIC INTERWEAVING IN NETWORKED DANCE AND MUSIC PERFORMANCE

Iannis Zannos
Ionian University, Department of Audiovisual Arts
zannos@gmail.com

Martin Carlé
Ionian University, Department of Audiovisual Arts
mc@aiguphonie.com

ABSTRACT

The present work explores the potential of networked interaction through dance. The paper reports the conception and initial implementation steps of an ongoing project, with public presentations planned for June and October 2018. The project's main objective is to extend the interaction paradigm of live coding through intimate coupling to human body movement using wearable devices. Our target is a performance involving dancers in separate locations, whose movements are tracked with magnetic and acceleration-based sensors on wireless wearable devices. In this way, two or more dancers performing concurrently in distant locations can jointly create a performance by sharing data measured by the sensors. Inspired by traditional african music practices, where several musicians play on one instrument creating interlocking rhythmic patterns, and by research on rhythmical interweaving (epiplokē) of ancient greek metrical theory, we use the data to modulate the metric patterns in the performance in order to weave rhythmic patterns. We discuss the design choices and implementation challenges for a performance.

The second main objective is to develop a prototype that demonstrates the use of literate programming and reproducible research practices with open source tools and evaluates the advantages of such techniques for development as well as for dissemination and ultimately educational purposes. We develop new tools and workflows using EMACS and org-mode as a platform for both documentation and development on an embedded wearable device made with CHIP-PRO. We show the benefits of using this environment both for documentation and for streamlining and speeding up the development process.

1. INTRODUCTION

1.1 Networked Live Coding with Gestures

The background for the present research lies in three distinct fields: Gestural Interfaces for Music Performance, Networked Music Performance, and Live Coding. By attempting to address all these fields at once in a novel experimental setup, we develop alternative approaches which may provide accessible albeit constrained solutions to some of the fundamental problems facing each field. The solutions are informed by the limited technical resources available and the need to develop a framework that can be implemented and adapted easily but also works within the requirements of a live performance. By necessity, the system presented is not a general solution, but a specialized medium for performance through manipulation of rhythmical patterns within a predefined metrical framework. However, it can be argued that this is a new type of instrument suitable for networked performance.

1.2 African Traditional Music, Ligeti, and Metric Interweaving

The idea for the instrument was given by the rhythms of african traditional music, and especially cases where several musicians collaborate to create a metric pattern conceived as a single musical entity from parts contributed by several performers[1]. Such music has already been received in the west by avant garde composers such as György Ligeti[2].

a.

b.

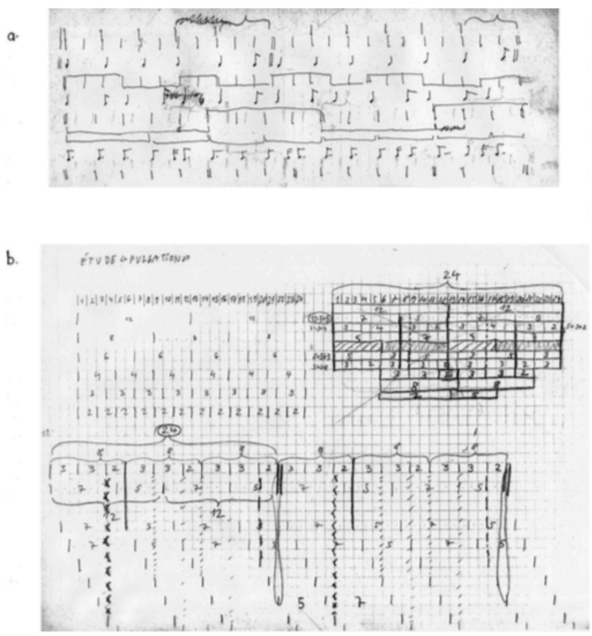

Figure 1. Györgi Ligeti's sketches of metric interlocking, from [2].

Furthermore, the play with complex metric patterns preoccupies one of the most prominent subgenres in the practice of Live Coding, that of the Algorave[3], [4]. In Algoraves, metric patterns for dance are defined and generated by writing code. In our experiment, we invert that relationship, by letting dancers define the metric patterns. By substituting dance movements instead of coding as performance-shaping medium, the performative aspect becomes more readily visible, while the range of influence over the sound diminishes. The gestural vocabulary of actions influencing meter is far more limited than that of program code. Thus, this is an exercise in minimalism, which presents some challenges and also allows us to return to situations similar to the origins of collaborative song and (pattern) weaving cultural practices.

In addition to the performance element, the present experiment stresses the collaborative aspect by letting dancers in different locations jointly form the rhythmic pattern of the performance. This collective aspect has been recently the subject of research connecting coding with weaving as collective cultural processes[1].

In the present research we draw upon archaeological research regarding the relationship of song and weaving in early Indo-European cultures, and use some of the patterns outlined there to design the metric pattern generation code[5].

1.3 Literate Programming, Reproducible Research, and Shareable Design

An additional driving factor behind the conception of the present project is the open source movement and its role in education. We wanted to create wearable devices based on low-cost open source hardware and software, and furthermore to see how we can share this technology and know how with students. As data acquisition unit, we chose Chip Pro, an ARM based device running Linux. Chip Pro[2] is an embedded device that can be programmed over a wifi network using Docker as an online virtual application packaging platform for deployment[3]. In addition to using EMACS and org-mode, a tool already established in the field of literate programming and reproducible research [6], we explore the potential of new cloud virtualization technologies in the sharing of code and applications online. Our first challenges in this domain were to package the complex scripts used to configure and load applications via Docker in order to be able to share them. Packaging furthermore permitted us to break the scripts down into modules which can be compiled and loaded independently of each other. As a result, development cycles became dramatically shorter, because we were able to reduce the build and load times by one order of magnitude, since we rebuilt only the code changed in a small module and uploaded that while keeping the remainder of the system intact.

2. SYSTEM OVERVIEW

The performance takes place concurrently on several different sites, connected via internet. On each site, there is one performer/dancer, provided with a wearable sensor device `W`. The sensor device consists of a Flora 9-DOF sensor by Adafruit, which measures the ambient magnetic field and the accelleration and orientation of the sensor, and a CHIP-PRO embedded computer, which receives the measurements of the sensor via an I2C bus and relays it to a computer via WiFi. The computer receives the data from the local performer as well as from all other performers, and sends the local performer's data to all other sites. On the computer, a program written in SuperCollider synthesizes the music for the performance based on the data received from all performers (See Figure 2).

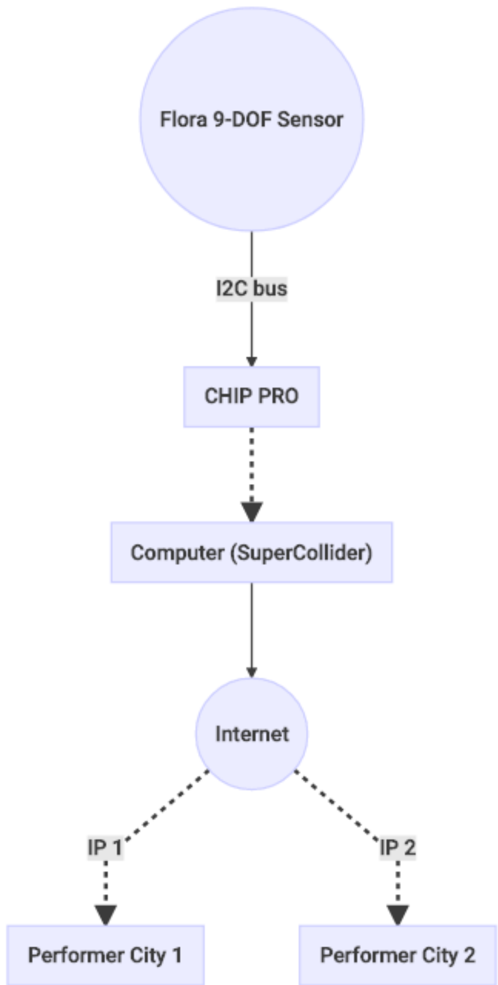

Figure 2. Architecture of the system.

[1] See the kairotic and penelope projects (http://kairotic.org/ and https://penelope.hypotheses.org/)

[2] See https://getchip.com/pages/chippro

[3] See https://www.docker.com/

[4] See: https://penelope.hypotheses.org/630 "Weaving Tidal-Cycles pattern in a TC-1 loom".

[2] See https://getchip.com/pages/chippro

[3] See https://www.docker.com/

3. METRIC PATTERN GENERATION: DESIGN

3.1 Collaborative rhythm creation and metric weaving through dance

The guiding principle behind the design of the collaborative definition of metric patterns by the performers is based on the the phenomenon of "inherent rhythm" observed by Kubik in his study of east and central african instrumental music [7]. In this study, Kubik posits that "*In some East and Central African instrumental music, [...] musicians playing together [...] produce rhythmic patterns, which are not perceived by the listener as they are actually played by the musicians. Instead of this he hears a conflict of other rhythms, which are not played as such but arise in his imagination.*" [7], p. 33. (See Fig. 3).

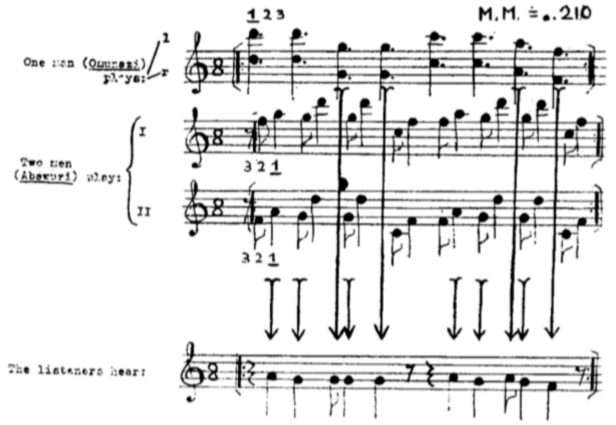

Figure 3. Metric interweaving in African Music, [7], p. 33.

Similar phenomena are also known in western music practices such as the hocket of medieval polyphony or virtual polyphonic lines in baroque monophonic playing. Bregman uses similar examples to explain the phenomenon of stream segregation in Auditory Stream Analysis [8], p. 18f. This emergence of overall patterns from individual lines that differ from each other in precisely controlled manner has given rise to comparisons with the weaving of image patterns in textiles [9]. McLean has used his TidalCycles software for Live Coding to weave textile patterns in a residency project, which he reported in Penelope's project colloquium[4]. In his report on this project, McLean remarks: *"Others commented on how they disliked the TC-1, because it was inaccurate, slow and 'distanced' yourself from the weave because you didn't create the shed directly. But this really felt like making techno music – you don't use acoustic instruments directly in the same way, but for me this puts more emphasis on feeling the sound — or in this case the fabric – itself as it emerges from the machine."* Our approach starts from a similar point of view, namely to regard the indirectness involved in using simple motion and magnetic sensors as control devices and relayed to remote sites as a creative aesthetic challenge in the performance and to devise an interaction mechanism that encourages performance with this medium. In our search for such mechanisms we were guided by the remarks of Giorgio Fanfani's contribution to the Penelope Colloquium, where he points out the importance of the concept of rhythmical interweaving (epiplokē) in ancient Greek metrical theory as *"a paradigm for pattern generation – every geometric or figurative motif emerging on the fabric resulting from an ordered crossing and combination of discrete elements, the threads, regulated by numerical relationships*.[5] In fact, in his study on epiplokē, Cole (1988) posits that certain types of metrically ambiguous anacreontic verses should in fact be regarded as part of a 'cyclically recurring pattern'. This creates a link to the cyclical meter of african dance (and dance in general), thereby opening up the possibility of seeing analogies if not affinities between these genres of poetry and dance.

A study of Antony Tuck's paper on the relationship between Patterned Textiles and the Origins of Indo-European Metrical Poetry (2006) led to some characteristic examples that seemed most fitting as a basis for our design [10] pp. 542, 543. As shown in Figure 4, ancient weaving patterns can be encoded as regular sequences of black and white (top and bottom) nodes in a rectangular grid.

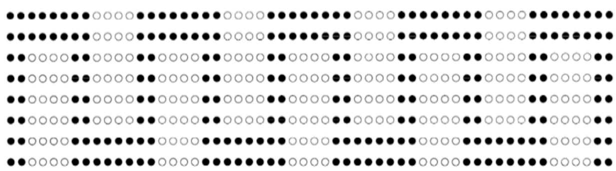

Figure 4. Fret patterns in ancient indoeuropean weaving, after [10] p. 543.

The pattern can be coded numerically by rows of numbers denoting the sequence of black and white dots (top and bottom threaded knots) on each row, from bottom to top, as follows:

```
2, 4, 8, 4, 8, 4, 8, 4, 8, 4, 2
2, 4, 8, 4, 8, 4, 8, 4, 8, 4, 2
2,4,2,4,2,4,2,4,2,4,2,4,2,4,2,4,2,4,2
2,4,2,4,2,4,2,4,2,4,2,4,2,4,2,4,2,4,2
2,4,2,4,2,4,2,4,2,4,2,4,2,4,2,4,2,4,2
2,4,2,4,2,4,2,4,2,4,2,4,2,4,2,4,2,4,2
8, 4, 8, 4, 8, 4, 8, 4, 8
8, 4, 8, 4, 8, 4, 8, 4, 8
```

This representation closely resembles a rudimentary loop-sequencer as known from early synthesizer technology and current sequencing tools. Instead of conveniently using a common midi controller, we chose to provide the performers with wearable movement tracking sensors,

[4] See: https://penelope.hypotheses.org/630 "Weaving TidalCycles patterns at a TC-1 loom".

[5] See: https://penelope.hypotheses.org/614

and ask them to shape the metric patterns through dance. This poses constraints to the available vocabulary of control actions and limits the directness and accuracy of control, but requires from the performer intuitive affinity through continuous use.

3.2 Design of a gestural control vocabulary

Our approach to designing the principles of metric encoding is inspired by metric notation of ancient Greek poetry. This employs two symbols short (u) and long (-) to build a vocabulary of metric units (*"feet"*), consisting of a few syllables each. Metric patterns do not have an explicit fixed phonetic counterpart, that is to say, a position marked with the sign of a short syllable (u) just as one marked as long (-) can take on the sounds many different syllables depending on the poem or verse. Following this principle, we define metric units as cycles or sequences whose components are abstract symbols, where each symbol may manifest itself aurally by choosing from a set of sounds. In other words, each symbol does not represent a kind of sound but a more generic category at the syntactic or colotomic level. Thus, the most fundamental level of our vocabulary defines a mechanism for playing sequences of beats as abstract symbols as well as ways to specify different sounds or actions corresponding to each symbol. The most basic element in a performance is a sequence of beats, where each beat may initially either be unassigned or assigned to a symbol.

The simplest case scenario starts with a set of beat lines beating regularly and synchronously:

```
o o o o o o o o o...
o o o o o o o o o...
o o o o o o o o o...
```

In order to permit incremental construction and refinement of patterns, we organize the beat lines into cycles of limited length, for example 12 beats. The lenghs of cycles may vary in the course of the performance depending on actions of the performers as explained below:

Each performer act on one beat line from this set, and may modify it by performing several types of actions on it:

1. Assign a symbol to a beat in a cycle of the line.
2. Modify or inflect the type of sound corresponding to a symbol on a beat.
3. Displace the train of beats in time by one beat or fraction of a beat.
4. Modify the length of the cycle of beats.
5. Modify the tempo of the beat line (the time interval between individual beats).
6. Modify the sound quality produced by the beat line by changing parameters of a filter applied to the sound output of the line.

The gestural control possibilities afforded by a single sensor unit with 3 types of 3-dimensional sensors are limited. We are not using any gesture recognition algorithms to differentiate a repertory of gestures, but use peak detection and value thresholds on the acceleration sensor to detect assignment events corresponding to single beats (point 1 above) and use continuous values from the other two sensors for continuous modification of parameters (point 6). Adding more categories of modification is most readily done by adding more sensors to the system at different parts of the body. However at this stage in the project we concentrate on gaining familiarity with and evaluating the basic aspects of behavior of the system, before proceeding to enrich its behavior any further.

4. IMPLEMENTATION

4.1 Wearable hardware interface

4.1.1 Chip Pro prototype

The Chip Pro is a System-on-Module (SOM) made by Next Thing Co and based on the based on the ARM Cortex-A7 CPU and capable of running Linux. However, because it is designed for low cost and low power consumption, it has only 256MB of RAM. It is thus not practical to run Debian Linux on the Chip Pro, as it is on the Raspberry Pi. For this reason, Next Thing Co provides an alternative suite of tools to support development on Chip Pro. The suite is named Gadget and consists of a command line tool and a lightweight version of Linux named GadgetOS. GadgetOS is based on buildroot (https://buildroot.org), a tool for generating embedded Linux systems, which relies on the well-known makefile language. GadgetOS is at early stages of development and thus documentation and libraries, while available, are not yet fully mature for general use and require much low-level work to configure. We based our development on org-mode, a package running on the EMACS editor which supports Literate Programming and Reproducible Research methodologies by executing blocks of source code embedded in plain text and optionally combining these blocks into external files to create standalone programs. Using these tools, we developed our own Integrated Development Environment (IDE) in order to be able to better automate, modularize and document the development process. Our experience with this environment is based on the work done to enable data acquisition from the Flora 9 sensor via the I2C bus. The drivers and python libraries provided by Adafruit - the manufacturer of Flora - were made for Debian Linux running on the Raspberry Pi. Thus the first port for Chip Pro was also on Debian. However, Debian is deprecated by Next Thing Co, because it demands too many resources to run efficiently on Chip Pro. Thus, we had to port the Adafruit libraries to GadgetOS. This required modifications and fixes to the original code. More importantly, we had to remove all unused residual objects from the resulting Docker images, because otherwise they would be too large to fit on Chip Pro. The build process took approximately 12 minutes and the shrinking process ("squash") approximately 20 minutes. In order to avoid having to repeat these processes at each time that we wanted to modify the program, we isolated those parts which needed further modification and placed them on a separate Docker module, which we could build and load separately. The smaller submodule required less than 30 seconds to build and load as opposed to the 30 minutes required to build, compress and load the entire system. Thus we could continue working on the parts that needed further modification much more efficiently. This functionality was implemented using the babel facility of org-mode on EMACS.

4.1.2 Update on the status of CHIP-PRO, alternative platforms

Since the original submission of the present paper, Next Thing Computing, the company that created the CHIP-PRO has declared bankruptcy. Thus, it is necessary to look for alternative platforms to continue this work in the long term. For Linux-based work these would be Raspberry Pi – which recently issued a Docker implementation – an Bela mini. An alternative which covers the functional requirements of the project but does not implement docker is Adafruit's Feather Huzzah ESP8266 wifi development board. This is programmable with micropython. We used it to develop our prototypes for experimentation in the performances.

4.2 Mapping input data

4.2.1 Calculate orientation from gyroscope angular data

We based our wearable controller on the Flora 9-DOF sensor by Adafruit[6] and used the Chip Pro embedded computer[7] to sample its data and transmit them wirelessly to the onsite computer. Both of these are low cost devices (Flora: 19.95$, Chip Pro: 16$, available with Development kit at 48$). The Flora sensor is a combined Accelerometer/Gyroscope/Magnetometer and provides 3 sets of measurements in three dimensions each (x-y-z), i.e. a total of 9 data values per sample frame. The Flora contains a type of cheap MEMS sensors which are commonly used in smartphones and other devices requiring orientation information such as visual reality headsets and drones. The orientation data from the *Flora* are measurements of the angular rate of movement obtained directly from the gyroscope, and thus they require further processing in order to be translated in absolute x-y-z coordinates in 3D space. We used a python library by Jean Rabault for this translation.[8] We subsequently discovered a sensor unit by Bosh (Adafruit's BMO055, ca 40$) which filters the data on-board and combines measurements by its built-in sensors to provide more reliable position and movement data. Our prototypes for the performances used this sensor.

4.2.2 Peak detection

Peak detection is a simple method for triggering events such as setting the states of a beat. We apply peak detection on accelerometer data, thereby detecting deliberately abrupt movements as triggers. Distinguishing the main direction of the acceleration vector may further be used to differentiate the choice of action intended. SuperCollider provides a reliable PeakFollower generator which is applicable on control rate signals carrying the signal from the input vector:

```
PeakFollower.kr(in: 0, decay: 0.999)
```

To filter out unwanted peaks below a given amplitude, the PeakFollower may be used conjointly with an amplitude follower passed through a comparison binary unit generator (<) acting as gate.

4.3 Representation of rhythmic patterns

To code and coordinate the rhythmic patterns we employ *sc-hacks*, a library for live coding by the first author. Two features of the library play a key role in the implementation of the metric pattern scheme described above:

1. The *Notification* class, implementing an improved version of the Controller pattern for assigning function callbacks to messages. This makes it possible to connect any two objects *receiver* and *sender* with an arbitrary function to be executed by the *receiver* object when it receives a message from the *sender* object. Using this pattern, it is possible to associate any symbol that is emitted in the course of a stream of beats to any action when received by any player object. It is thus possible to encode beat sequences as patterns with a custom play type which emits the symbol of the beat and lets any number of player/receivers translate the symbol to sound in their own way. Beat synchronization for multiple players is thus guaranteed.
2. One of the basic features of the gestural interaction design outlined above is the modification of beat patterns during the performance. The built-in pattern playing object in SuperCollider does not give access to the playing patterns data, and thus it cannot modify patterns while they are playing. *sc-hacks* defines an *EventPattern* and *EventStream* class which make it possible to modify patterns while they are playing.

5. NETWORK REQUIREMENTS

5.1 Networking Prerequisites for Performance Sites

In order to ensure that data connections work at short notice, it is required that each site possess a static IP address connected to the internet via UDP on the port number used by SuperCollider (57120). The local computer is connected to the internet via Ethernet.

[6] https://www.adafruit.com/product/2020
[7] https://getchip.com/pages/chippro
[8] See https://folk.uio.no/jeanra/Informatics/QuaternionsAndIMUs.html and https://folk.uio.no/jeanra/Informatics/quaternions.pdf for an explanation of the algorithm. The library is available on github: https://github.com/jerabaul29/IntegrateGyroData.

5.2 Latency Issues and Synchronization Strategy

We tested and benchmarked communication between Athens and Corfu using SuperCollider's internal local time facility, and arrived at a latency of under 30 milliseconds. While this is surprisingly small, it can be expected that latency will increase when performing across larger distances. This could prove damaging if the signal for changing the state of a beat arrives at a remote site at a time when the performance has moved to the next beat. A countermeasure for that would be to use a universal counter for beats shared between all performance sites, and to send beat-specific commands together with the number of the corresponding beat. However this would require one of two possible precautions to preclude that a command arrives at a remote site too late to act on the beat it was meant for: Either code in the system a latency interval between the instant of recognizing the command and the beat to which it is assigned, thereby making sure that the signal will reach both the local and the remote systems before the beat to which it is addressed, or schedule missed beats to change their state when they are reached at the next cycle in a cyclic beat pattern.

6. CONCLUSION

This project is the first step of the research group at the Audiovisual Arts Department of the Ionian University in the direction of genuine embedded wearable computing devices. Compared to recent work on Raspberry Pi, development on the Chip Pro proved to be much more demanding and time-intensive. On the other hand, the perspectives opening for further projects based on this platform are very promising. A number of projects are being envisaged already, ranging from augmented instruments using small wearable sensors to distributed environmental sensing projects for ongoing compositions. The work on this platform has made such projects much more accessible both economically and technically. The most significant insight is the realization that globe-wide projects involving sensors can be carried out with limited means and relative ease, first since the Chip Pro hardware used is not only low-cost but also small and light enough to be shipped anywhere without extra expenses, and second because Chip Pro software can be updated remotely over the Internet, enabling remote maintenance and tweaking of projects.

Acknowledgments

This project is conducted within the framework of the European Project EASTN-DC. Many persons and institutions were involved in the project, and we list here only those known at the moment of submission for publication. We are grateful for the collaboration of the MANTIS center of the University of Manchester and in particular for Ricardo Climent for his support of the performance in Manchester, to Fiori Anastasia Metallinou for support of the performance at the Athens National Observatory, Vasilis Agiomyrgianakis for technical support in Athens, Stella Dimitrakopoulou as the dancer in Athens, Marilena Georgandzi for wearable construction advice, to Ilia Katsaridou and Makis Stergiou for coordinating the performance and for technical support in Brussels, to Kiyoshi Furukawa of the Tokyo University of the Arts for providing spaces and technical support at the Intermedia Art Department in Tokyo, to Satoru Takaku of Nihon University for enlisting the help of dancers for the performance in Tokyo, and to Nathaniel Virgo of the Earth-Life Science Institute for correspondence and advice which helped the project conceptually.

7. REFERENCES

[1] G. Kubik, “The Structure of Kiganda Xylophone Music,” in African Music, Vol. 2 No. 3, 1960, pp. 6-30.

[2] M. Scherzinger, “Györgi Ligeti and the Aka Pygmies Project,” in Contemporary Music Review, Vol. 25, No. 3, 2006, pp. 227-262.

[3] N. Collins and A. McLean, “Algorave: Live Performance of Algorithmic Electronic Dance Music,” in Proc. Int. Conf. on New Interfaces for Music Expression, London, 2014, pp. 355-358.

[4] K. Burland and A. McLean, “Understanding live coding events,” in International Journal of Performance Arts and Digital Media, Vol. 12 No. 3, 2016, pp. 139–151.

[5] A. Tuck, “Singing the Rug: Patterned Textiles and the Origins of Indo-European Metrical Poetry,” in American Journal of Archaeology, Vol. 110, No. 4, 2006 pp. 539-550.

[6] L. Stanisic, A. Legrand, and V. Danjean, “An Effective Git And Org-Mode Based Workflow For Reproducible Research,” in Operating Systems Review, Association for Computing Machinery, 2015, 49, pp. 61-70.

[7] G. Kubik, “The Phenomenon of Inherent Rhythms in East and Central African Instrumental Music,” in African Music, Vol. 3. No 1. 1962, pp. 33-42

[8] A. Bregman, Auditory Scene Analysis: The Perceptual Organization of Sound. MIT Press, 1994.

[9] A. McLean, “The Textural X,” in Proc. xCoAx2013: Computation Communication Aesthetics and X, 2013, pp. 81-88.

[10] T. Cole, Epiploke: Rhythmical Continuity and Poetic Structure in Greek Lyric. Harvard University Press, 1988.