SOUND CHOREOGRAPHY <> BODY CODE

ALEX MCLEAN

ICSRiM, School of Music, University of Leeds
United Kingdom

a.mclean@leeds.ac.uk

KATE SICCHIO

Lincoln School of Performing Arts, University of Lincoln
United Kingdom

kate@sicchio.com

Keywords: Choreography, Live Coding, Dance, Hacking, Feedback

A performance work, *Sound Choreography* <> *Body Code* is introduced, which connects live coding and live choreography in a feedback loop of influence. Both the practice and the discussion of the work raise issues in coding, choreographic scores, and live performance through an exploration of feedback, interpretation and technological mediation of sound and movement in live coding environments. It suggests a model for interpretation of scores that allows for different approaches to scoring and interpretation in live performance settings and proposes how mutable scores differ in the instructional language within Sound Choreography <> Body Code.

1. INTRODUCTION

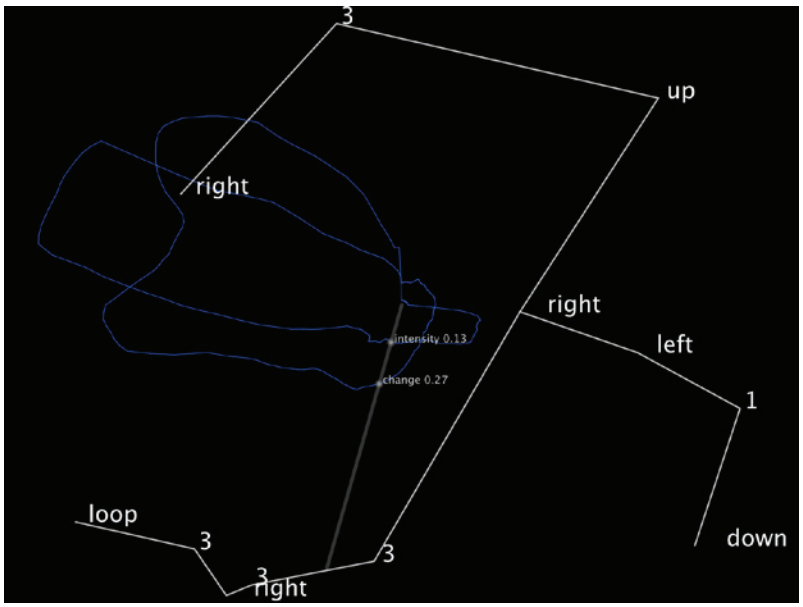
In the following we consider the movements of choreography and of computer programming together. The discussion centres around *Sound Choreography <> Body Code* (SB<>BC), a collaboration between the present authors, which combines choreography and computer programming in a live performance artwork. To date the piece has been presented three times, at Audio:Visual:Motion in Manchester UK, at Thursday Club at Goldsmiths in London UK, and at Hack Circus at Site gallery, Sheffield UK. The following paper begins by examining the relation between code and choreography as explored in performance. We then describe our own collaboration, centring on the feedback loop created between body and code. We conclude with discussion of the experience of programming, the role of mutable notation in performance, and code as a tool, a language or environment.

2. SOUND > CHOREOGRAPHY > BODY > MOVEMENT > CODE > SOUND

This performance follows individual works, by Sicchio in the relationship between choreography and code (e.g. Sicchio, 2014), and by McLean on designing programming languages for expression in live performance (e.g. McLean, 2013). To achieve confluence from these two practices, we needed to connect aesthetic and technical aspects on both sides, and achieve balance between them. The solution we arrived at maintains a clear distinction between choreography/dance on one side, and code/music on the other, but creates a technological connection between them, via their notations. As a result the music is not coded for the dancer, and the dancer does not move to the music; but still a feedback loop is created that passes through the body and code, via machine listening and computer vision (see Fig. 2).

The piece begins with both performers simultaneously creating live action (dancing, typing), and with projections of both code-scores within the performance space. A diagrammatic score is followed by the dancer (Sicchio), with a small series of instructions that include directions (right, left, up, down, loop, if) and numbers (1-3) that are connected into an acyclic diagram (i.e. one that forks but does not reconnect), according to the minimum spanning tree. The dancer has a set series of gestures that are then

organised and performed based upon how the instructions and numbers are connected and continually reconfigured. However, as the performance progresses, the diagram becomes much more complex. The number of instructions fluctuates over time on a scored pattern that peaks at a point of complete overwhelm for the dancer and returns back to a simpler form to end the performance.



The movement of the dancer is tracked by a Microsoft Kinect via a 'patch' made with the Isadora software, detecting the location and shape of the dancer's body in space. From this two floating point values from 0 to 1 are calculated; one representing the position of the dancer along the horizontal axis, and the other derived from the height/width ratio of the bounding box of their body when viewed from the audience. The latter represents an axis from standing, to crouched (or with arms apart), to lying down (along the X axis). These data are sent via the Open Sound Control network protocol to McLean's live programming environment, to intervene in the code (as described below). This motion tracking provides one of the two points of contact between the movement and the sound, which forms the feedback loop.

Fig. 1 The Sound Choreographer, showing instructions right, left, up, down and numbers, connected in a minimum spanning tree. The grey line extending from the centre sweeps around, clock-like, through a single cycle during the performance, and the two blue shapes show "intensity" and "change" graphed over time using polar coordinates. Intensity gives the number of instructions, and change the size of each movement that is made in response to sound onsets.

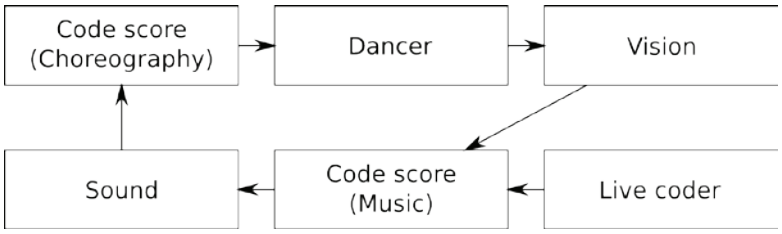


Fig. 2 A diagram illustrating the SC<>BC feedback loop of influence from choreographic score, through the dancer, their movements picked up by computer vision, fed into the live code environment, which produces sound which feeds back into choreography. The live coder is outside this loop.

The danced movements translate into movement within code, in particular that of Texture, a visual live coding environment (McLean 2011; see Fig. 3). Texture is visual in a sense that is stronger than in conventional visual programming languages. That is, the syntax is based on Euclidean distance, whereas in many other visual systems such as Max/MSP or PureData, the programmer makes connections between functions manually (Left-right position is syntactically significant for execution order in Max/MSP, but this is true also of purely text-based languages). Therefore, as the function assigned to Sicchio's position on stage moves around McLean's screen, it interferes and disrupts the running code. Because Texture only connects type-compatible values to functions, the resulting program is always syntactically correct. In practice, this means that program is never disrupted to the point that it falls silent.

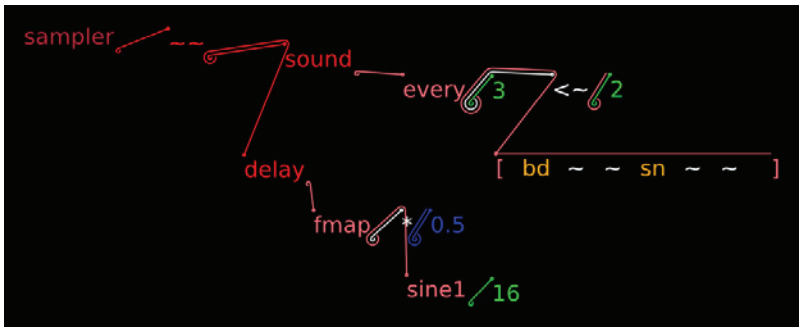


Fig. 3 The visual programming environment Texture. Words are automatically connected (and re-connected) based on proximity and type compatibility (McLean 2011). A version in development, used in Site gallery Sheffield, visualises patterns flowing between functions.

The second point of contact between the choreography and code is via machine listening. The Sound Choreography software performs audio onset detection on the sound produced from Texture, and the words within the choreographic diagram move in response. This can result in connections switching between words, whenever the movement results in a different minimum spanning tree. In this sense, the choreographic structure is dancing more directly to the rhythm of the sound than the human dancer.

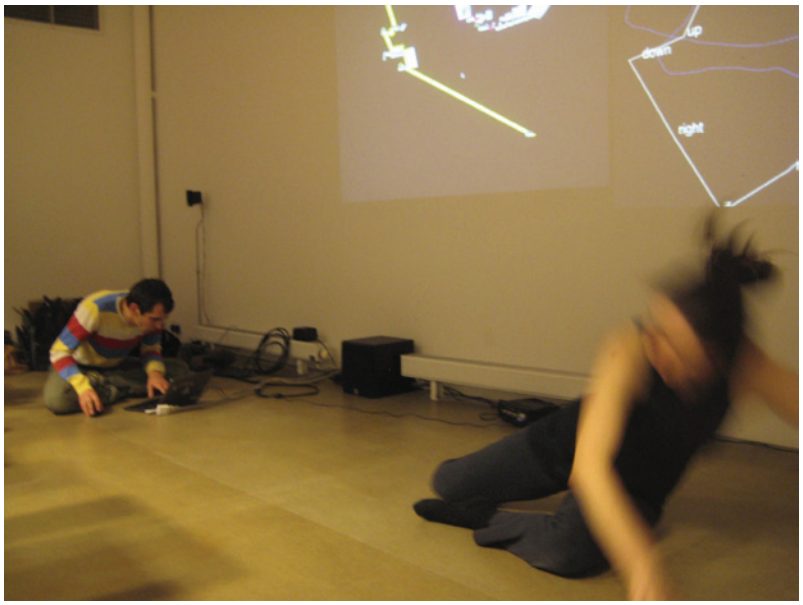


Fig. 3 Still from performance at Site gallery, Sheffield.
Photo credit: Susanne Palzer

These two points of contact create a loop of continuous influence from the body, into the code, into the sound, into the choreography and back into the code (see Fig. 2). This technological feedback loop is instrumental in bringing the piece together; during performances to date, both performers were focused on their individual code-scores, rather than the overall composition of the piece. In particular, Sicchio has not been conscious of developments or changes within the sound as she is dancing and the effect of the sound on her movement is not noticeable by her within the performance. The technology becomes the choreographer in this sense, and is organising the interactions, rather than the performers sensing each other in that moment. Whether there is a subconscious level of interaction between code and movement is an open question.

3. CODE AND CHOREOGRAPHY – MUTABLE PERFORMANCE

Other projects have recently explored the interface between choreography, live art and live coding practices (Cocker, 2014; Collins, 2011). There are also direct connections to the work of Fluxus artists such as Higgins (1961) found in this spectrum of interpreting scores. Higgins discusses this idea of the interpretation of the score, stating “All markings on the face of the notation are to be used or deliberately ignored for the production of movements and sounds according to any consistent system

which the individual performer devises for the notation he is using" (Higgins, 1961).

When considering the relation between choreographer and computer programmer, there are three distinct elements at play. Firstly there is the score, which defines some organising principles behind the work. Then there is the notator, who writes the score. Finally there is the score-follower, who carries out the actions defined by, or otherwise organised within the score. In the following we focus on the interrelationships between these elements.

A key point to consider in comparing live choreography with live code is the extent to which the notator dictates the score, and the score constrains the score follower. This reflects two distinct purposes: the use of tools for creating specific outcomes in performance versus creating an environment for exploration in performance. The result is a spectrum of possibilities, where at one end lies mutable scores, or scores open for interpretation by people and machines, through languages. The other end of this contains coded, fixed 'objects' or fixed/coded people reproducing an ideal performance each time the piece is executed.

In terms of "Sound Choreography <> Body Code" two instructional languages are used (one for movement and one for sound) and they address these concerns differently. While the Sound-choreographer for the movement is a mutable-open score, Texture is a mutable-closed score. For example, while certain aspects of the choreography are scored within the performance, there are others that are not explicit within the score. The gestural movements that are repeated throughout are set by the performer, but the amount of repetition, location in space and order of the gestures are determined through following the score. As the score changes, the dancer may change their own system of interpretation. With Texture, the human-technology interface is reversed; the notator is human, and the score-follower is an electronic computer. Accordingly, the score is expressed and followed unambiguously, although the performance is still indeterminate as the score is continuously changed by the notator.

As the choreographic score becomes more complex over time, the dancer eventually has to accept that they cannot perform it accurately, and that they are failing to interpret the score in its entirety. The dancer is therefore compelled to change the way that she interprets the

score, for instance focusing on sections, or a looser reading of it. So while there is some set movement vocabulary that is not notated, the interpretation of the system and language is still mutable within a live performance. Instead, the syntax of Sound Choreographer is fixed and based on proximity (as it is tracked by the kinect), and semantics is open to the performer's interpretation.

This model for scoring performance also may be useful in examining coding practices in general. Creating tools versus environments within software may demonstrate various ways in which coding may be approached. Within "Sound Choreography <> Body Code" there is a conversation between disciplines, code and people. It creates a language between these elements and therefore an environment to be explored. When considering code in this way, it becomes less utilitarian and more expressive. The software in this work may in a sense be considered useless; it does not offer utility as a tool, but instead provides a connection between practices. This allows body and code to resonate through notation, but only through difficult struggle.

ACKNOWLEDGEMENTS

This work was developed in conjunction with the Live Notation project (funded by the Arts and Humanities Research Council).

REFERENCES

Cocker, E. (2014). *Live notation - reflections on a kairotic practice.* Performance Research Journal 18 (5).

Collins, N. (2011). *Live coding of consequence.* Leonardo 44 (3), pp. 207-211.

Collins, N., A. Mclean, J. Rohrhuber, and A. Ward (2003). *Live coding in laptop performance.* Organised Sound 8 (03), pp. 321-330.

Farnell, B. (1996). *Movement Notation Systems*, pp. 855-879. New York: Oxford University Press.

Higgins, D. (1961). Museum of Modern Art. Museum label for artist, *Graphis 82*, New York, 20 December 2013.

McLean, A. and Wiggins, G. (2011). *Texture: Visual notation for the live coding of pattern.* In Proceedings of the International Computer Music Conference 2011.

McLean, A. (2013). *The Textural X.* In Proceedings of xCoAx2013: Computation, Communication, Aesthetics and X.

Sicchio, K. (2014). *Hacking Choreography: Dance and Live Coding.* Computer Music Journal 31 (1), pp. 31-39.