Physical interaction with automated music composition platforms

Jonas Abernot[1] and Simon Hickinbotham[2]

[1]INRIA, Lyon, France
[2]Department of Computer Science, York center of complex system analysis, University of York, UK

The generation of computer music can be thought of as similar to the process of compiling any computer program: a series of coded instruction are read by some interpreter, and a time-based sequence of actions are perfomed, in this case the generation of sound. Composition in these languages can be done off-line, or during the performance. Live coding is the use of text-based computational languages to generate music in real time. The flexibility of the coding language is paramount, giving the coder the freedom to express the music as she wishes. However, since the creation of coding languages is a highly specialised task, it is difficult to create a language that is optimised for use by the layman rather than by the specialsied coder.

One of the problems with learning live coding languages is the real-time nature of the composition process. One can get caught up in the act of composition, and stick to coding styles that do not fully exploit the faclities that the language provides – it is difficult to RTFM whilst in the middle of a composition. Our recent development of an evolution-based live code generator [1] offers a solution to this problem, since mutations on patterns can change them in ways that do fully exploit the facilities of the language because the mutations are built by applying the grammar of the language to the parse tree.

In addition to the evolution of live coding patterns, our *commensal computing* system uses a second evolutionary process to dynamically classify accelerometer data from physical movements of dancers into commands for the evolutionary pattern generator [2]. This linkage between dynamic classification of music and dynamic generation of new live coding patterns opens up new areas for real-time composition. However, since the range of possible configurations of the system is huge, we first need to consider how best to integrate these controllers. This contribution gives a brief description of this new composition space, and suggests the most fruitful early avenues for research.

Implementation

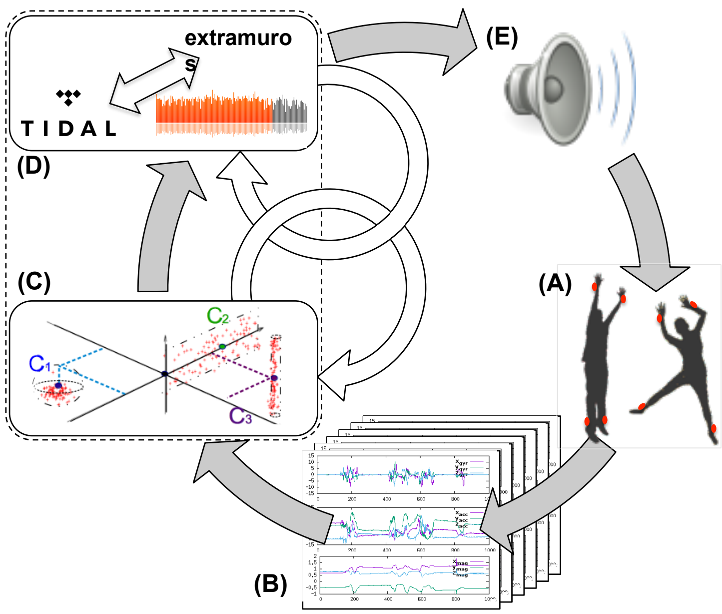

Figure 1: Architecture of the instrument, from [2]

Our evolutionary system involves two entities: The 'host' which evolves the musical patterns, and the 'symbiont', which evolves control signals based on the physical movements of the performer. Figure 1 shows how the system is organised. (A) Data produced by the performer(s) are captured by multiple sensors, producing a high dimensional temporal signal (B). The signal is then processed by the "symbiont" (C) that performs subspace-clustering on the input data. Clusters are then used as symbols to feed the host (D). Thanks to this dimension reduction the host is then able to process them to produce music (E), which provide a feed-back to the performer(s).

Our initial experiments with this system have revealed several distinct challenges that need to be addressed in order to further develop the tool: usability, creativity and composability. The 'symbiont' system was developed such that each cluster center corresponsed to a particular sound sample. When a move is assigned to a cluster, the sample is played repeatedly, but gradually fades out. This allows a pleasing interaction between moves and sounds to be developed during the performance. However, the control of the host's evolutionary capabilities is necessarily more dynamic: not only must existing patterns have moves associated with them, but the performer must somehow indicate which patterns are fittest, and which patterns are to be used as the basis of further mutations. Since we want to resist the temptation to hard-code the association of particular moves with particular controls, more work is needed to determine how best to map movements to a more complex control scheme.

Given these observations, we present a new method for associating the symbiont's evolved cluster centres with the control architecture of the host. This is the first implementation of the Creative Systems Framework described in [3], which has been achieved via a coupling of move association and compositional control through the evolution of evolution (evoevo).

References

[1] S. Hickinbotham and S. Stepney, "Augmenting live coding with evolved patterns," in *LNCS 9596: Evolutionary and Biologically Inspired Music, Sound, Art and Design*, pp. 31–46, Springer, 2016.

[2] J. Abernot, G. Beslon, S. Hickinbotham, S. Peignier, and C. Rigotti, "A commensal architecture for evolving living instruments," in *1st Conference on Computer Simulation of Musical Creativity*, pp. 1–16, https://csmc2016.wordpress.com/, 2016.

[3] G. A. Wiggins and J. Forth, "Computational creativity and live algorithms," in *Oxford Handbook on Algorithmic Music* (A. McLean, ed.), ch. 15, pp. 266–290, Oxford: Oxford University Press, 2016.