Thor Magnusson (gb)
Faculty of Arts, University of Brighton
Senior Lecturer

ixi lang: A Constraint System for Live Coding

1. Introduction

In the late 1990s a new performance practice appeared in the more experimental venues of the musical world, where performers would step onto stage with a rather strange musical instrument, the laptop. These performance contexts, in pubs and clubs, were primarily designed for pop or rock bands. Instead of locating themselves behind the mixer, where the best sound is normally to be heard, they placed their equipment on the stage, typically on a table, and presented some rather refreshing and novel musical worlds. Whilst the audience appreciated the texturally sophisticated world of sound these instruments were capable of, the performance aspect of the music suffered. What were these musicians actually doing behind these screens on the stage?

A decade later some solutions had evolved, addressing this lack of coupling between the performer's gestures and the sound emitted by the speakers. One of them is VJing. By analysing the sound signal – typically through Fast Fourier Transform Analysis or even OSC messages sent from the sound generating software – the VJ is able to generate visuals that connect and represent the sound in endless interesting, yet arbitrary, ways. Another solution is represented by a field often called NIME (New Interfaces for Musical Expression), with university courses and conferences dedicated to the investigation (see www.nime.org). Here various interfaces have been designed that allow the performer to use her body, in a manner inspired by acoustic instruments, to control a digital sound engine. The third response to the problem of the exclusiveness of computer music performance is live coding.

2. Live Coding

Live coding needs no introduction, but as a summary it comes with an imperative that performers project their screens such that the audience is able to participate in the musical creation. And this should be done from a clean slate where the code is designed in real-time. A dedicated forum exists for practitioners (www.toplap.org) and various papers have been written with topics that range from general introductions (Collins et al. 2003), to live coding in specific systems (see Wang & Cook 2004; Rohrhuber et al. 2005; Sorensen 2005), or live coding as artistic practice (Nilson 2007; Sorensen & Brown 2007).

A typical problem for the live coder is the high level of expertise required for such performance (Nilson 2007). Very few performers are able to exhibit those skills without consistent dedication to practise (Sorensen and Brown 2007). Although I have long been fascinated by certain virtuosic live coders, it seemed to me that such incorporation of dexterity strives against the primary rationale of the mechanical computer; namely the automation of rote tasks and the augmentation of mental capacity.

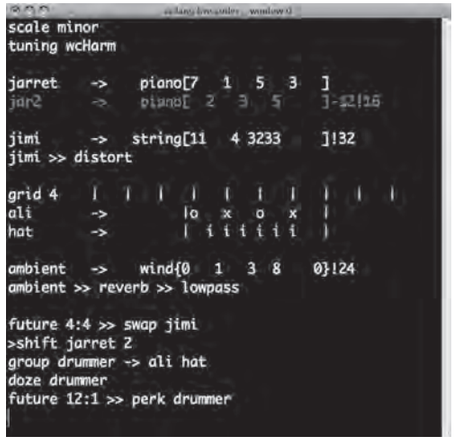

Fig. 1: A screenshot of an ixi lang session.

3. Design Rationale

From this perspective, I attempted to design a musical live coding language that would free performers from having to think at the level of computer science, allowing them to engage directly with music through high-level representation of musical patterns. Most importantly, the language should be easily understandable by the audience who would be able to follow each step of the performance, given a little bit of imagination in terms of interpreting language features and functions.

The ixi lang was intended to address a problem in live coding involving slow buildup times and lack of musical constraints. Too much freedom can confuse the performer. The goal was to be able to create a tune with rhythm and melody within a few seconds from the performance starting. The language should also be understandable to non-programmers who would be able to follow clearly the performer's train of thought.

4. Ixi Lang Functionality

The ixi lang has three modes of musical notation that can be generated and synchronised in real-time: melodic, percussive and concrète (sample based). These musical patterns are created in the form of identifiable agents whose performance can be adjusted through various methods (e.g., shifting notes, transposition, reversing, inversing, scrambling). Figure 1 shows a text document that serves as the code input window. The code is both as updated representation of the score (it can change according to the user's design of algorithms) and a direct instruction to the system's play mechanism (the score itself).

The ixi lang clearly affords a certain limited set of musical activities. It provides a scaffold for externalising musical thinking and through its simplicity attempts to ease the live coder's cognitive load. As a live coding system it goes further than most common live coding environments in providing a simple, high-level platform for musical improvisation. However, this is at the cost of possible expression, as height (in terms of abstraction) will always impede freedom.

5. Conclusion

The ixi lang was devised to address specific problems common in live coding performance, such as slow and laborious build-up, incomprehensibility, and difficulty in making simple musical structures. It provides the performer with a very high-level language where musical structures can be set up in a matter of seconds using a syntax that is intuitive and easily understandable to audience. With user comments such as (Magnusson 2010):

- "Wonderful to break free from the rigid time line approach."
- "The audience can immediately participate in the performance. The language is general and simple. At times funny to watch."
- "A release from the paralysis of choice! Still, I would love to be able to become more proficient with SC so as to tailor the environment to my needs."

With responses like these, I find that the project has succeeded in fulfilling the original aims.

References

[1] Collins, N.; McLean, A.; Rohrhuber, J. & Ward, A. (2003). "Live Coding Techniques for Laptop Performance," in Organised Sound vol. 8 (3), pp. 321–30.

[2] Magnusson (2010). "ixi lang: A SuperCollider Parasite for Live Coding" in SuperCollider Symposium 2010.

[3] Nilson, C. (2007). "Live Coding Practice" in NIME 2007 Proceedings.

[4] Rohrhuber, J.; de Campo, A. & Wieser, R. (2005). "Algorithms Today: Notes On Language Design for Just In Time Programming" in ICMC 2005 Proceedings.

[5] Sorensen, A. (2005). "Impromptu: An interactive programming environment for composition and performance," in ACMC 2005 Proceedings.

[6] Sorensen, A. & Brown, A. (2007). "aa-cell in Practice: An Approach to Musical Live Coding" in ICMC 2007 Proceedings.

Wang G. & Cook P. (2004). "On-the-fly Programming: Using Code as an Expressive Musical Instrument," in NIME 2004 Proceedings