Reflections On Learning Live Coding As A Musician

May Cheung

LIVECODE.NYC

Introduction

After spending many years as a musician from a jazz performance background, I was excited to try a new method of creating music. Having been active in the live code community in New York (livecode.nyc) for almost a year, it became apparent to me that my process in creating musical structure, harmonic ideas and melodic ideas were different than live coders who came from a programming background. Using the live code environment Sonic Pi (based on the programming language Ruby, created by Dr. Sam Aaron) as a reference, this poster presents the parallels and differences between the experience of live coding music versus the experience of performing and playing music. In rare occasions, the dichotomy of the two worlds converge as we present the work of Anne Veinberg, a renowned classically-trained pianist who uses a MIDI keyboard as an interface to code through a program created by composer-performer Felipe Ignacio Noriega.

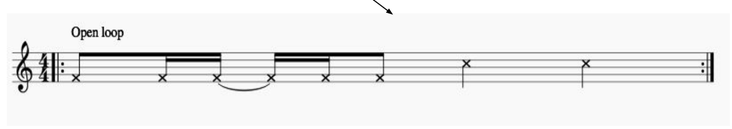
Differences

1. **Visual differences**

Visually, code is expressed abstractly by using the modern Roman alphabet, numbers and keyboard symbols. Written music, however, has its own hierarchical structure by using notes, bar lines, symbols and numbers. In live coding, one must "think in the language" so to speak. In Sonic Pi specifically, there are two ways to code notes: using MIDI note numbers (61 means C#, for example) or by scientific designation (A2, A3 for example). The following excerpt illustrates these visual differences, which result in the same rhythmic pattern:

```
live_loop :clap do
  sample :tabla_ke1
  sleep 0.25
  2.times do
    sample :tabla_ke1
    sleep 0.125
    sample :tabla_ke1
    sleep 0.25
  end
  2.times do
    sample :tabla_tas1
    sleep 0.5
  end
end
```

Translates to:



2. **Multitasking**

One can "play" multiple instruments at the same time without physically moving towards each respective instrument on stage. One can change instruments at the stroke of a few keys. Contrast this to a live music show gig where a truckload of instruments would have to be driven to a venue. Also, from a mixing standpoint, one can also manipulate the reverb ("`with_fx :reverb`") in real time whereas a mixing engineer would be in charge of the overall sound using a soundboard at a live music show.

```
live_loop :ok do
  use_synth :tri
  with_fx :reverb, mix: 0.6, room: 0.7 do
use_synth_defaults amp: 0.75, attack: 0.4, decay: 0.25, sustain: 0.6, release: 0.2
play choose([ :a6, :g6, :c6, :e6, :d6, :b6])
sleep 3
  end
end
```

TYPE OF SYNTH

REVERB MIX

ADSR

CHOSEN NOTES TO RANDOMIZE

DURATION OF NOTE + REST

Live coding allows for multi-track experimentation whereas songwriting in its most traditionally western definition is usually a one-dimensional experience. Even if we compare a live coding session to using Garageband, for example, we would not be able to randomize notes or pan left to right automatically. Also, coding encourages us to think of music in a linear way with the use of formal, artificial languages - artificial in that they are constructed by individuals, as opposed to stemming from an organic cultural process that comes as natural languages do (McLean 2011, 17)

3. **Practical application**

I discovered the ease of coding rhythms in Sonic Pi versus writing them on paper, practicing and executing them. I felt more inspired to discover world rhythms and test them out using code. For example:

```
s = 0.125

live_loop :claveone do
  sample :bd_tek
  sleep 4 * s
end

live_loop :clavetwo do
  sleep 3 * s
  sample :sn_zome
  sleep 3 * s
  sample :sn_zome
  sleep 2 * s
end
```

This is a Reggaeton beat; "claveone" is a basic four-on-the-floor bass drum pattern. "clavetwo" is the clave pattern that juxtaposes the bass drum with a 3+3+2 rhythm, which is also called a "tresillo" rhythm, stemming from Caribbean roots. Imagine other world beats one could replicate: Hungarian, Gamelan, Indian tabla beats, for example.

4. **Artist-Audience relationships**

A look into Artist-audience relationships between live coding and performing music reveal that human connection is lost in Algorave settings. there is hardly any introduction, eye contact or human connection between the artist and the audience. Simply, the live code artist gets on stage, plugs an HDMI cable into their laptop, opens it up, finds their code, and presses a button to run their code. After setup and initiation of the code, the audience gazes at the projected code while heavy drum samples or noise blasts away through sub woofers. Since live coding sets last anywhere between 15-40 minutes, the audience is subjected to a longer performance before applauding (the irony is that this is also the case in classical music concerts) which therefore demands a longer attention span from the audience. In a conventional pop or singer-songwriter set, songs are typically 2-5 minutes, maximum. Less attention span is required from the audience and there is more respite between songs for both the performer and audience member. The human connection in conventional performances are more prevalent and therefore, could serve as inspiration to the live code music community even though the medium live coders use are artificial languages. As TidalCycles creator Alex McLean states "Computers give us privileged access to the digital realm, but we must not lose sight of the analogue, because humans experience and interact with the world as an amalgam of both." (McLean 2011, 18)

Parallels

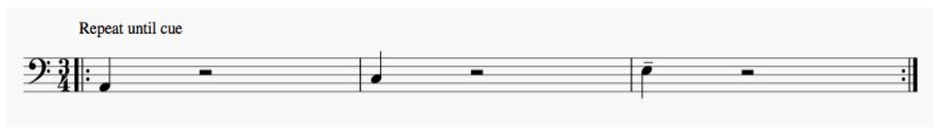
Through my own experiences in live coding and performing music, I discovered two major similarities between the two:

1. The act of associating meaning to symbols or syntax

The figures below illustrate the associations I make when coding in Sonic Pi. These "live loops" are like bars of music enclosed with repeat signs:

```
live_loop :letsgo do
  use_synth :dpulse
  play :a3
  sleep 3
  play :c3
  sleep 3
  play :e3
  sleep 3
end
```

Translates to:



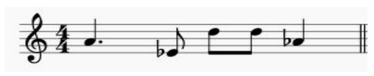
These translations from code to music, or vice-versa, make the process of live coding easier by association as a musician. More often than not, I hear a rhythm or melody before I code it.

2. Method of preparation

I use the same method to prepare for a live code show or a musical performance. I practice, I memorize and I test. If I am about to do a live code show, I practice executing my whole Sonic Pi set at least five times over several days leading to the performance from memory. Then on the day of the show, I test myself to see if I have retained the process and flow of my set, starting from a blank "buffer". Once I have the code memorized, I take calculated risks and improvise from there. My training as a musician has taught me that the act of practicing is important, especially when it comes to learning new languages: spoken, written, read or coded. I assume that once I attain fluency in Ruby, which is the language that Sonic Pi is based off of, I would not have the need to rely on memory to regurgitate a live set on a blank slate. As Fabia Bertram notes, in her paper, 'Learning Elementary Musical Programming with Extempore: Translating Arvo Pärt's *Fratres* Into Live Code Snippets', "the student will finally have a coding performance that can stand on its own while having practiced or improved, or even acquired new programming skills." (Bertram 2014, 3). I must note that not all coders perform this way. Many live coders have their code in order and ready to go before going up on stage. They can change numbers, variables, or comment out sections as they go along during their performance which can prove to be helpful and less stressful.

Convergences

Broadly speaking, the "cross-pollination" between musicians and live coders rarely happens. However, after attending my first ICLC conference in Morelia in 2017, I met other musicians who were immersed in the world of live coding, namely, classically-trained pianist Anne Veinberg, who presented her piece CodeKlavier in which she combines live coding with piano performance. In her performances, she uses an 88-key MIDI keyboard as an interface to send commands to a program that coder and composer Ignacio Noriega created. Veinberg, who obtained her Bachelor in Music from the University of Melbourne and Master in music at the Conservatory of Amsterdam, plays the piano in various capacities - from classical performances to live coding music, sending commands to her laptop while playing specific sequences or singular notes. With an acoustic-MIDI piano, specifically a Yamaha Disklavier, Veinberg plays a series of notes that translates into code through a program created by fellow live coder and composer Felipe Ignacio Noriega. For example, if she plays this passage on the piano:



the word "`hello`" would appear quickly on the line of code on the projector screen. The result is astounding, as the audience can see the code being "typed" by a piano - not the conventional way of coding as one would expect.

Conclusion

The possibilities and sheer accuracy of what live coding brings to the world of music is necessary and important for the evolution of music itself. Live coding as a musician enriches one's experience of creating music in that the possibilities are infinite and that it helps to further one's creative potential. A musician's prior knowledge makes learning live coding easier with the discipline and structural understanding involved in both coding and performing. With this prior knowledge, we find that the structures between code and musical notation are similar as well as the method that comes with preparing for a show. On the contrary, we also find that music notation gives way to interpretation, whereas code acts as specification; the act of live coding requires little if any physical movement between instruments; musical structure and organization of ideas are developed differently; calculated randomization makes the impossible (by musical notation standards), possible. Also, the disparity in artist-audience relationships between live code settings and music performance is apparent as live code performances lack human interactions. Finally, we find the union of the live code and music performance world made possible by musicians such as Anne Veinberg as she combines piano performance with code, furthering the future potential of music technology and music creation.

References

Bertram, Fabia. 2014. "Learning Elementary Musical Programming with Extempore: Translating Arvo Pärt's *Fratres* into Live Code Snippets." University of Cologne, Germany.

McLean, Christopher Alex. 2011. "Artist-Programmers and Programming Languages for the Arts." PhD diss., University of London.