Interactive/Responsive Looping and Beat Generation Interface

Task 2: Create an Interactive Music System or Interface
(The realisation is a hybrid of the two)

Written Report

Candidate Number: 19329

Advanced Computer Music: G6003 (UG)

University of Sussex

January 2014

Interactive/Responsive Looping and Beat Generation Interface

1.0 Overview

To realise this project, I decided to implement aspects of both task proposals, to create a system that responded to the user' input information such as pitch and onset data but via a new controllable user interface for live performance. I tackled audio looping, beat generation from onset input and created intelligent response mechanisms to the input. Motivations to design an interface came from being trapped within the limitations of the computer interface, and after being inspired by Jay Silver's "Hack a banana, make a keyboard" Ted talk[1], I was opened to the hundreds of interactions which everyday objects can be used for, to control electronic signals.

Therefore I chose to create an interface that used simple everyday materials such as foil, yoga mats and guitar strings, that would be simple enough and intuitive enough, that new users will instantly understand how the interface works and the interactions it affords.

Overall functionality; The system has three levels/modes. Mode one, 'Looping' takes user's live input and loops it, intelligently expanding or contracting data to reach whole bars. Mode two, 'Beat Generation' takes users' onset data and creates beats from the rhythms of the onsets. Behind these two modes, the system is listening and responding intuitively in the same key, scale and rhythms as the user has created. Mode Three, 'Mute Control' allows users the ability to solo or mute out the three main aspects of this design, live looping, beat generation and automative response.

1.1 The problem space (Why Design an interface for looping and beat generation?)

With many systems out there that already handle looping and interfaces that already allow beat generation such as drum machines. The question arises why tackle this route?
The reason for doing so is that none of these interfaces feedback intelligent responses using the users input data and also they are two separate interfaces which require different interactions. E.g. drum machines are programmed with most commonly fingers, and loop pedals with the feet. Therefore I decided to use the concept of 'convergence' [2] to bring these two different interactions together.

2.0 Music Information Retrieval (Responsive System)

Prior to creating the interface, the retrieval analysis and interpretation of musical data needed to be addressed. Pitch recognition and key detection were implemented into this system and achieved by adapting Nick Collins' OnlineMidi.sc Supercollider code which is based upon key profiles established by Krumhansl and Kessler (K-K profiles) [3]

2.1 Quantization of audio/ Conformity to whole bars

[1] Jay Silver, Ted Talk, 2013, http://www.youtube.com/watch?v=kiUnJ1d8vvw
[2] Rogers, Sharp, Preece, 2011, Wiley & Sons 'Interaction Design', Chapter 1, Convergence.
[3] The most informative paper I've found on these profiles is http://www.theory.esm.rochester.edu/temperley/papers/temperley-marvin.pdf Temperley and Marvin

"One Main aim of interaction design is to reduce the negative aspects (e.g. frustration and annoyance) of the user experience while enhancing the positive ones (enjoyment and engagement)"[4]. In order to adhere to these principles I needed to make sure musical errors such as timing didn't become too much of a problem for performers. For example, its very common to make small errors in timing and so to make sure this didn't become a really big issue, I implemented my own algorithm to analyse the nearest whole bar to where the user had stopped playing would be. This 'conformity or quantization' means that regardless of the user playing too much or too little to make up a bar the algorithm would always take care of it for them so they would always be aligned in whole bars.

This is a form of constraint, a design principle Donald Norman proposed[5] whereby we reduce the usability of certain interfaces in order to keep things simple and easier to use. The whole bar quantization algorithm discussed can be found in the source code of LiveLoopGeneration class, however I discovered recently that using Pbinds the quant: 4 parameter can take care of this for us. However it proved very useful to understand how to carry out bar quantization in situations where I would have to design from scratch.

One of the main benefits of my looping system over other commercial ones however, is the implementation of a feature labelled 'beatstart'. We record the beat the performer decided to punch in and record on, and measure the distance between it and the last whole bar. This allows us to have natural input in creating loop sequences because it doesn't constrict us to always starting at 4 which the quant: 4 does.

2.2 Onset Detection and Quantization to Beat Generation (Turning Onsets into Beats)

Although the quant: 4 method undermined my initial quantization algorithm I had to develop a new, far more complex one for the use of onset detections to be turned into beats. Using the sending of OSC messages at certain rates we can constrict the audio input to be for example at at rate of 1/16th notes or 8hz. However, this means the user has to be absolutely precise with their playing in order to place an onset of a transient directly on the 1/16th count. To resolve this problem I calculated the note durations of sending out OSC messages at a rate of 44100/1024 and determined that 5 values would correspond to a single 1/16th note duration value.

Therefore we can look for every five values being streamed in from user input, if there is an onset detected in this five, then we shall assign an onset to a 1/16th note duration for use in beat patterns.

<u>The First step of the formula:</u>

(1/(Hz*bps))*4

So assuming that bps = 2

44100/1024 = 43.06640625Hz

(1/(43.06640625*2))*4 = 0.046439909297052

[4] Rogers, Sharp, Preece, 2011, Wiley & Sons 'Interaction Design', Chapter 1

[5] Donald Norman, 1988, Basic Book, 'The Design of Everyday Things',

‘0.046439909297052’ = Our note durations for this rate at which we are receiving the audio.

<u>The second step of the formula:</u>

1/16th note duration or 0.25, is the value we wish to use for our beat patterns. So we see how many of our recently discovered note durations fit into 0.25.

0.25/0.046439909297052 = 5.38330078125

Rounded down, we have 5.

The third step of the formula:

The final step of the beat quantization formula, is to iterate over the stores of 0.046439909297052 durations and analyse five at a time.

For each five stored, we see if there is an onset detected.

If an onset has been detected then we store a value with 0.25 duration.
If no onset detected in the five then we store no value with a 0.25 duration i.e a rest.

This advanced algorithm, allows people who are performing with the interface room for error. And as it is rare that people can play exactly perfectly in time, this accommodation reduces any potential frustration and will increase user enjoyment, fulfilling one of the main aims of interaction design outlined in 2.1.

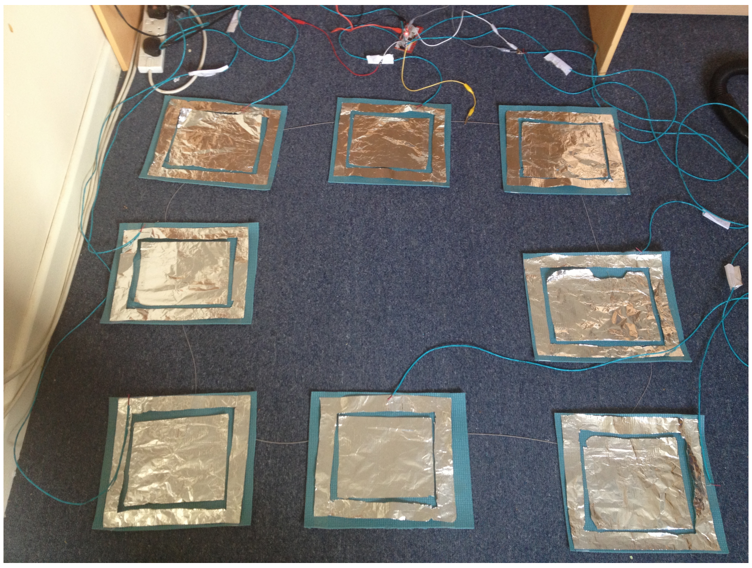
2.3 The benefits of designing such an algorithm for performing

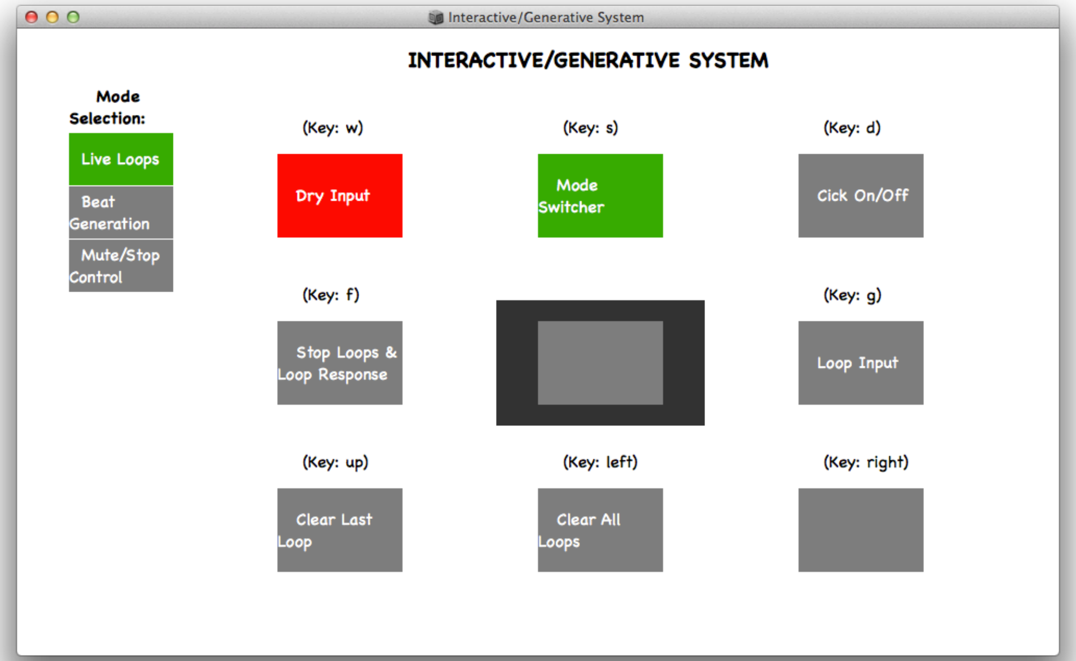
Allowing users the ability to create beat patterns by manipulating their native instrument instead of having to master another type of interaction is a huge advantage this interface holds. As is the ethos of HyperInstruments at MIT [6], we allow the user to continue using the instrument they have spent their time mastering and just create an interface which becomes an extension of that mastery.

Another great benefit of this onset quantization to beat generation algorithm, is that is resolves one of the initial issues discussed early, that we are able to converge interfaces and use this controller not only for looping but as beat generator as well, all from two aspects 1. our feet movement, 2. our instrument movement.

[6] http://www.media.mit.edu/research/groups/1450/hyperinstruments

3.0 Description of the interface


(Fig. 1)



(Fig. 2)

A 3X3 grid of electronic pads was connected to the computer and Supercollider via a Makey Makey [7]. Figure 1, shows the outboard electronic pads the users step on to perform with. And Figure 2, shows the GUI the user also views simultaneously on the computer. The GUI maps identically to the real world which means when users switch

[7] An Arduino like device that makes the computer think a keyboard is connected, but allow us to manipulate what objects cause key presses with breaking out of the board. http://www.makeymakey.com/

mode they see on the computer which pads have changed their functionality by their label and therefore immediately understand that their actions will cause different effects in different modes.

3.1 Interacting with the interface

The user can switch between the modes any point they like with no barriers. The looping mode lets them create multiple sequences of live input that loops in the amount of bars they have been conformed too. The system immediately responds by using the pitch data gathered from these loops and uses it to spit back out.

Whilst in beat generation mode, users make shorter tighter sounds to create more precise onsets for the system to recognize, which it does accurately. Depending on the pad stepped on kick, snare, hihat or tom patterns will be outputted in the rhythm of their onsets. In addition the system will also respond by paying degrees of the key it has detected the user is in. For the kick it plays in sync with the kick rhythm in a low register to mimic that of a bass synth. This following of rhythm in the key is followed by the snare who's partner response synth dots about randomly in the key. The hihat synth counterpart plays in the gaps of the hat hits.

When in the mute/stop control mode the user is able to control all three aspects of this design by muting them whole. This allows for silence, or soloing of certain patterns that the user may wish to focus on. And at any point they can use the dry input pad to improvise over what is being created between them and the computer.

4.0 Conclusion

In terms of the breadth of functionality that the system is able to handle I believe I have certainly achieved in creating an interaction and responsive loop and beat generation interface. It lacks for sophistication from the response synths but the onset quantization to beat generation algorithm is definitely a very highly important and enjoyable feature to use.

The interface itself is also very easy and user friendly as it takes the metaphor of a dance mat which users associate with enjoyment. It could certainly do with improving in terms of materials but the fun of using everyday materials was the inspiration so I'm glad the outcome saw this realised.

The onset quantization beat generation algorithm is something I wish to pursue, using natural language as the next input. Where people will be able to say the word of the instrument in time with tempo and that is what will be outputted by the system.

Sound Examples:
https://soundcloud.com/jamesnapierstuart

Video Demo:
http://www.youtube.com/watch?v=xbXEqcBfqZA&feature=youtu.be