Guitaptop - Adding Expression to Laptop Generated Music

Oriol Nieto
Java Music Systems Final Project
Music and Audio Research Laboratory
New York University

December 6, 2010

Abstract

We present a new instrument called Guitaptop that synthesizes music by making use of an accelerometer and a mouse trackpad. This instrument will enhance compositions with richer expression by creating music by tilting the laptop and sliding fingers on the trackpad instead of just pressing buttons. This instrument can create different recordings by putting them into different tracks. It also allows to save the composition as a wave file and to play some background beats to get inspiration while creating a new piece. Future work should be focused on creating an Applet out of it to have an online version of the instrument and to be able to choose from different sounds, and add other new features.

1 Introduction

For a long time computers have been used to synthesize music [9], [4]. Unfortunately, the expressive part of the general input devices of the computers (*e.g.* keyboard or mouse) are limited. Multiple devices have appeared to overcome this problem and to make computers more expressive (just to name a few: Reactable [5] or Radio Baton [6], but please refer to the NIME conference for plenty of other new devices for musical expression [7]). Some

of these devices used to use the MIDI port [1], however nowadays we usually find them with the more versatile USB connection.

One big issue of these new devices is their high prices in the market, so that the general public loses interest in them because generally they are not willing to make an investment to create more expressive music. Some of them will more likely use their last generation phones to create music [11]. As laptops are getting smaller and more powerful, nowadays we can even fin laptop with built-in accelerometers. Since this component comes with the laptop, we decided to make us of it in order to create a new instrument that we present in this paper: The Guitaptop. This instrument makes use of the built-in accelerometer to make your laptop a more expressive music synthesizer.

Outline The remainder of this article is organized as follows. Section 2 gives a report on the detailed description and design of the instrument. The implementation details are described in Section 3. The artistic analysis and whether the instrument does what it has to do can be found in Section 4 Finally, Section 5 gives the conclusions and future work.

2 Detailed Description

We wanted our instrument to be easy to set up and play with it. So the very first time it loads, just by pressing the key 'q' sound will start coming out of the speakers. By tilting the laptop and by making gestures with the mouse, this note will be shaped in many different ways, becoming much more expressive.

In Figure 1 there is a screenshot of the Guitaptop. In this screenshot there are two tracks (in the middle of the picture, two buttons for each track, one for playing the track and the other one for deleting it). On the top of the figure there are the recording and playback controls. In the bottom there are the sample controls and the timer.

So in the beginning, there will be no tracks. If we want to begin a new recording of a track, we have to press "Start Recording", and when we are done, we press "Stop". A new track will be added. At any time we are

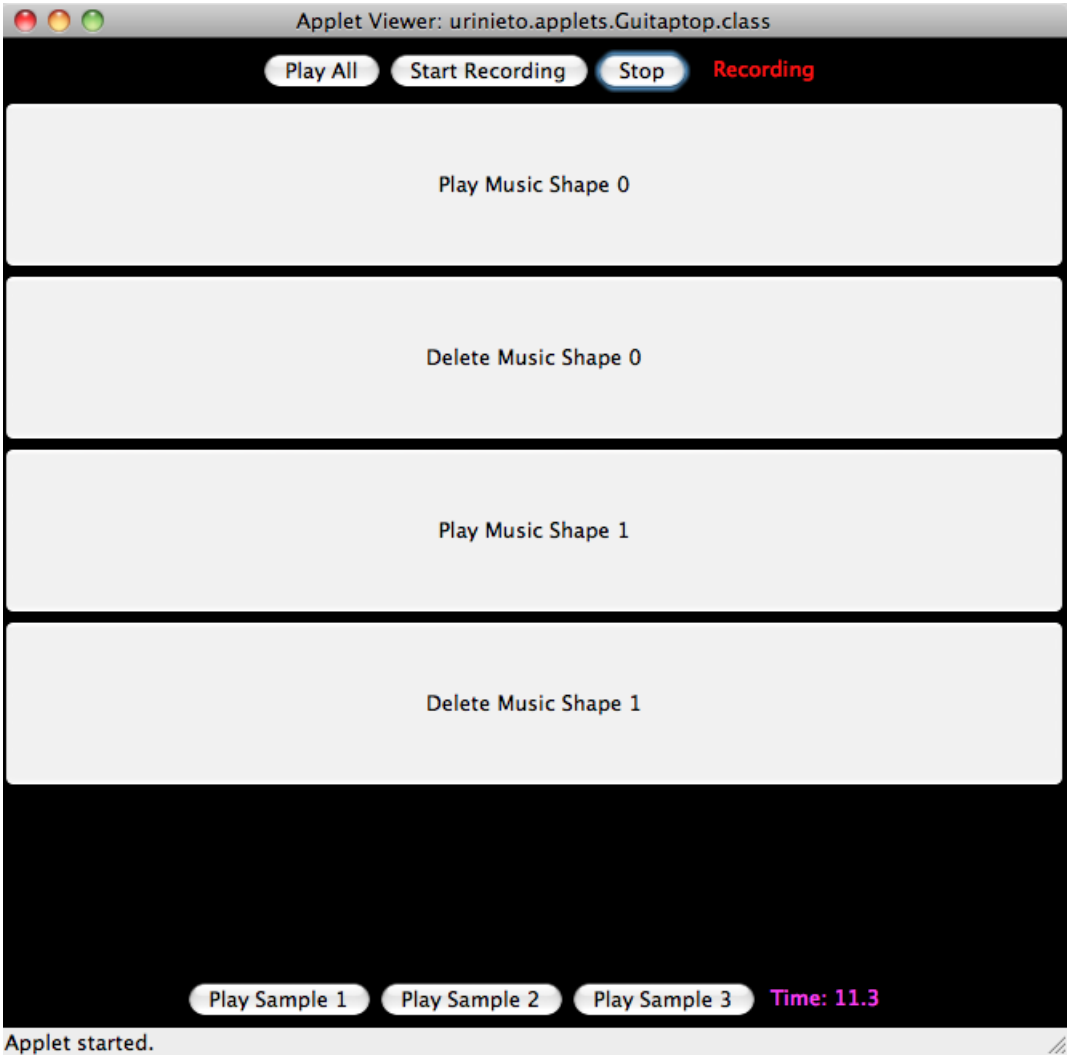

Figure 1: Screenshot of the Guitaptop. On the top we can find the Recording and Playback Controls. In the middle we can find the different tracks (if any. In this case there are two different tracks). In the bottom part there are the sample controls and the timer.

recording a red "Recording" message will appear on the top right side of Guitaptop. By pressing "Play All" we will hear the same as "Play Music Shape 0", since right now there's only one track. But if we add more tracks

(by pressing "Start Recording" again), we will hear all of them at the same time when "Play All".

At every time we record or play any of the tracks (or all of them), the timer will be constantly updated.

Finally, at anytime we press the "Play Sample" buttons, one different background beat sample will start to play (independently if we're recording or not). This might help us to find some inspiration to our piece.

3 Technical Description

For the implementation of the Guitaptop we have used JSyn [2] and JMSL [3] (both written in Java) and for the accelerometer part we have used the Sudden Motion Sensor (SMS) by Apple [10]. We used a MacBook Pro with the Snow Leopard version of OSX.

The first part was to make the SMS available from the Java source. We used the open source libraries called Unimotion [8]. These libraries were not updated for the Snow Leopard version of OSX, so we had to tweak them in order to make them work. Once they worked in Objective-C / C, we made a Java Native Interface (JNI) library out of them, so that we could call them from a Java application.

Once we could access the accelerometer information from Java, we could start developing the actual Guitaptop. First of all, we developed an *Accelerometer Notifier* structure, so that we could have a callback **every 100ms**. We chose this number because we don't want to overkill it, and updates every 100ms give enough resolution for working with the accelerometer in real time without wasting CPU cycles. The source code for that can be found in files *AccelerometerEvent.java* and *AccelerometerNotifier.java*.

The Guitaptop implementation is mostly found in *Guitaptop.java*, and we added the Accelerometer callback in there. The Guitaptop class is actually an *Applet*, and it uses JSyn and JMSL technologies to synthesize and organize the music and instruments. It also makes use of the KeyListener and the MouseListener technologies available from the Java library. Once

we have set up the JSyn Engine and the MusicDevices, it is trivial to set up a prefabricated Instrument and make it sound by pressing a key, and using KeyListener methods of Java to do so. But now we want to record this note and be able to play it back later on. The procedure is as follows:

- After "Start Recording" is pressed, we store the start time in a variable.
- If the 'q' key is pressed, then we add silence from when the "Start Recording" was pressed and now. Now we will keep adding MusicShapes every time there is a change in the accelerometer or in the trackpad (this is done in the accelerometer callback).
- When 'q' key is released, we add the final MusicShape with its current time, and update the silence time, so that we will add silence between notes if we add another note.
- Repeat until "Stop" is pressed.
- When "Play All" is pressed, set time to zero and reproduce the previously stored MusicShapes.

So now we have a basic methodology to store an Instrument and play it back. But now we want to add more tracks to it. To do it we make use of the *Vector* object from Java to store an array of Tracks. We will add a new track every time "Start Recording" is pressed. We also store an array of play and remove tracks buttons, so that we will be able to play or remove a specific track. Since the class Vector is a dynamically allocated array, we can easily remove and add elements to it, so it is very convenient in this case. So now, we will have a custom Note (source found in *GTNote.java*) that will be updated every time there's a note to be added to the current Track.

As for the Instrument part, the source code is found in *GTInstr1.java*. It is basically two different oscillators (Sine and SawTooth) multiplied, and passed through a lowpass filter. The frequency of the Sine is controlled by the 'z' component of the accelerometer, and the cutoff frequency is controlled by the 'x' component. The frequency of the Saw is controlled by the movements of the mouse.

The beats from the Samplers are extracted from the Logic Pro Loops Library, and they are being read from a custom URL [1]. They are played using the *SampleReader16V1* and *SynthSample* classes.

Finally we added a *JMSLMixerContainer* so that the compositions can be easily stored in a WAV file. This is added to each Instrument of all the MusicShapes.

4 Artistic Analysis

The live sound is a good synthesized sound that allows a lot of expressivity and even emotion when applied to the changes in the tilt and in the mouse trackpad. The experience becomes more evolving when used with one of the different samples to play a beat in the background.

One of the issues is the clicks that we hear when the playback is on. We should rewrite the methodology to store changes, or add a better ASDR filter, since for some reason the ADSR wasn't being applied to the recorded MusicShapes. However, after a while, this surprised me as a pleasant feeling, and made me go inside the instrument even more. Sometimes, when there are too many tracks, a clipping starts to occur, but sometimes this clipping makes thing more interesting.

5 Conclusions and Future Work

We have presented a new instrument called Guitaptop that makes use of the accelerometer and the mouse trackpad. This makes the Guitaptop a synthesizer with a great amount of expressiveness. The instrument also successfully records into different tracks and is able to bounce everything to a wav file.

The Guitaptop is intended to be an online instrument, and that's why eventually it will be a part of an online Applet. However we couldn't do it by now, since it takes too long to deploy a JNI library as an Applet (we have to certify the applet in order to do so to start with). It is a process that we

[1] http://urinieto.com/NYU/JavaProject2010/sounds/{GT1.wav, GT2.wav, GT3.wav}

will do in the near future.

When recording to a wav file, it would be good to record the background beats as well. Right now it was not an easy task to do, and we stuck to the JMSLMixerContainer class.

In the future it would be good to add more instruments to the Guitaptop. Right now there's only one. With new instruments new shapes and sounds would be easily explored.

Also in the future the clicks of the recordings (mentioned in Section 4) should be removed.

Another nice feature to add in the future would be a cool visualizer, that would go with the sound. Or maybe a visual sequencer for all the music shapes. We envision it as aesthetic and easy to use.

Finally we would love to have some Signal Processing into the Guitaptop. Something like Delay, Reverb, Chorus, Distortion, ... We think that Guitaptop would be much improved with these Signal Processing add-ons.

References

[1] International MIDI Association. MIDI: Musical instrument digital interface specification 1.0. *North Hollywood: International MIDI Association*, 1983.

[2] Phil Burk. Jsyn - a real-time synthesis api for java. In *Proceedings of the International Conference on New Interfaces for Musical Expression. Univ. of Michigan, Ann Arbor, USA*, 1998.

[3] Nick Didkovsky and Phil Burk. Java music specification language, an introduction and overview. In *Proceedings from the 2001 International Computer Music Conference*, 2001.

[4] Sergi Jordà. Improvising with computers: a personal survey (1989-2001). *Journal of New Music Research*, 31:1–10, 2002.

[5] Sergi Jordà. Interactive music systems for everyone exploring visual feedback as a way for creating more intuitive, efficient and learnable instruments. In *Proceedings of the Stockholm Music Acoustics Conference*, 2003.

[6] Max Matthews. Radio-baton and improvisation modes. In *Proceedings of the 1997 International Computer Music Conference - Thessaloniki Greece*, 2003.

[7] NIME: New interfaces for musical expression, http://www.nime.org.

[8] Lincoln Ramsay. Unimotion: unified motion detection for apple portables, http://unimotion.sourceforge.net/.

[9] Curtis Roads. *The computer music tutorial*. MIT Press, 1996.

[10] Apple: Sudden Motion Sensor, http://support.apple.com/kb/ht1935.

[11] Ge Wang. Designing smule's iphone ocarina. In *Proceedings of the International Conference on New Interfaces for Musical Expression. Pittsburgh*, 2009.