OddString

Real-Time Interaction and Performance

Diego Di Carlo
Computer Engineering
University of Padova
Italia
diego.dicarlo@studenti.unipd.it

Jorge Madrid Portillo
Sound and Music Computing
Aalborg University (Copenhagen)
Denmark
jmadri15@student.aau.dk

ABSTRACT

In this paper we describe the concept, design and implementation of a new device, the OddString for augmenting guitar-like instruments. The purpose is to introduce guitar and bass players with the possibility to both, manipulate the guitar signal and trigger MIDI notes by placing a softpot potentiometer on the back of the fretboard. This mapping allows players to expand the expression of this kind of instruments while preserving the classic expressive elements that can be achieved without the OddString. The new ways of expression are plenty and yet to be discovered by players who desire to enhance or expand their musical expression. OddString is designed to be easily attached/detached to guitars, bass and similar instruments.

Author Keywords

NIME, augmented, guitar, teensy, softpot, bass, arduino, ribbon, piezo, extended, expression

1. INTRODUCTION

There are many ways to augment instruments. Hybrid instruments offer musicians the possibility to expand the control possibilities while maintaining the familiarity and expressiveness of the original instrument. In this sense, the OddString is designed to act upon guitar and bass as a way to convert them into hybrid instruments. We designed the OddString to be easily attached to any instrument of the string family which has a long scale fretbord (enough to fit the softpot potentiometer). The use of this kind of new interfaces is very exciting since it opens a new window for players to explore and create new compositions and performances while maintaining the original sound and features of the instrument.

The philosophy behind the OddString is to let the artists and players interact with digital effects without the necessity to lift their hands from the performing position (one hand in the fretboard and the other resting in the bridge). Not only this, OddString can be used to trigger MIDI notes, thus expanding the sound possibilities of the artist. We think of the OddString as the natural evolutive step for the guitar/bass, since it is acting as an extra "string" the player can use for different purposes. The possibilites are plenty and yet to explore. In our design we focused on two playing modes:

NIME'15, May 31-June 3, 2015, Louisiana State Univ., Baton Rouge, LA.

- **String mode:** In this mode, the OddString acts as an extra physical string-like MIDI controller available to the player on the back of the fretboard.
- **Controller mode:** In this second mode, OddString acts as a standard fader, allowing the player to control all sorts of MIDI mappable controls on a digital audio workstation (DAW).

The mapping, interaction and design of these two modes will be further explored in following sections. For now, let us review the state of the art in section two.

2. STATE OF THE ART

Since the invention of the electric guitar and bass, plenty research and innovation has been made in order to expand the sonic possibilities of players. One of the easiest and functional approaches in enhancing expression have been pedalboards and guitar pedals. These work extremely well since players do not have to change their playing style or even lift their hands from the instrument. In this sense the mapping of pedals is extremely simple and useful. The problem with this sort of interaction is that, it is very limited (most of the time to ON/OFF) and does not necessarily improve the expression of the player, although in some cases it does (whammy, WHA WHA).

In this context, many experimental players and reseraches have been constantly trying to improve and augment guitar and bass in many different ways. For example, by placing accelerometers or WII controllers on some parts of the body [7], players can use natural gestural controls, such as moving the fretboard up and down, to apply some effects. Other approaches such as the one by *Carlos Vamos* [3], consist of attaching directly physical synthesizers or controllers such as the **Alesis Air Synth**[1] or the **Yamaha Motif Rack**. These are very interesting approaches since, the signal is processed on beforehand, and the interaction is usually natural and simple.

The only problem with these approaches is that they are very complex to develop and to attach to the guitar body (usually require electronics knowledge) and the mapping can force the players to change their playing style. Similar approaches like [11] include attaching different sensors like pressure sensors to the back of the guitar. Some other examples for string instruments like the violin, include the usage of many different sensors to interact with, like the Overtone Violin [10]

In the recent months, *MIND* startup [4] has created a prototype, *SENSUS guitar* which also include a similar approach by placing many sensors into the guitar, even a softpot in the back of the fretboard, which proves indeed that it is a good mapping. The main problem with this is that normally players prefer to maintain their own instrument rather than having to adapt to a completely new instrument. In

this sense, OddString is a more natural and accessible way to extend guitar. Finally there have been some prototypes in the last years that aim to create new interfaces for musical expression, such as this **arduino ribbon guitar** [2], which also uses three softpot ribbon potentiometers to produce MIDI notes, as well as other sensors such as a joystick which can map to other things.

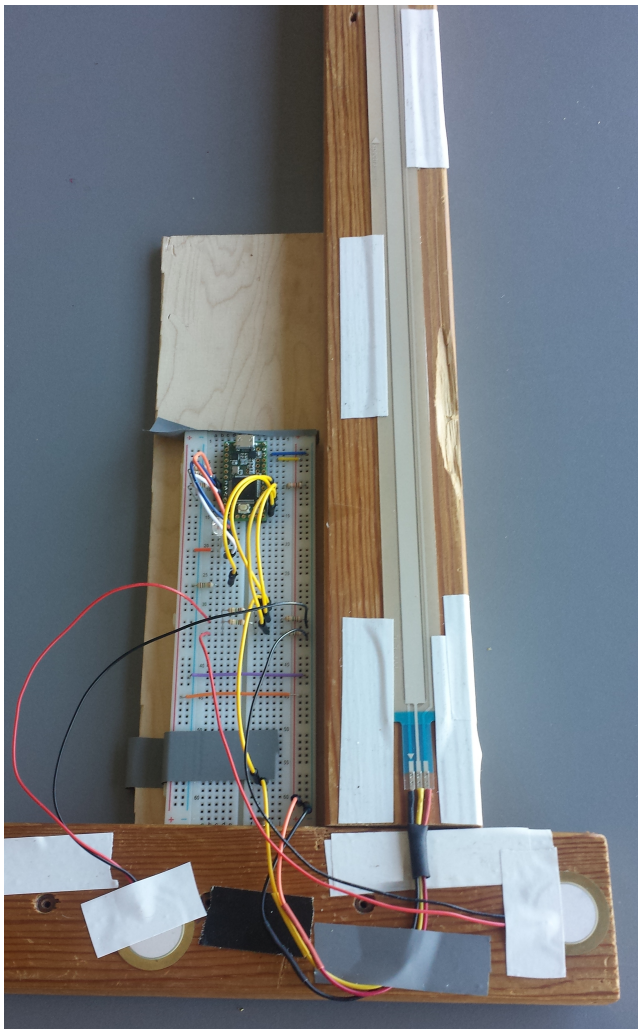

Figure 1: OddString prototype, not attached to the guitar.

3. DESIGN AND INTERACTION

The key concept for the design of the OddString is allowing the player to express as much as possible without changing their playing style. For this reason we decided to use a softpot ribbon potentiometer[1] which in fact resembles a guitar string and place it in the back of the fretboard. Other options considered were using a infrared sensor between the headstock and the lower part of the body in order to track the distance, but this proves inefficient since the player needs to lift the hand from the fretboard, and also this kind of sensors do not work properly under certain light conditions. Having this in mind, to set up our project, we followed **Minuet Framework** [9], a NIME-oriented and user-centred iterative design process in order to design the interaction of the OddString.

[1] `https://www.sparkfun.com/products/8681`. Accessed: 2016-11-06

As main **goal**, it should be designed especially for guitar and bass players to give them the possibility to extend the sounds they make and to control them without learning new difficult techniques.

- **People:** it should be used by every guitar-like instruments player. The role of the audience is very relevant, because this device should add more transparency to the performance of the guitar players that use some effects.
- **Activities:** Triggering MIDI notes and sending control messages via MIDI.
  - **Motivation:** guitars-like instruments players are often limited in using digital effect because they have their both hands most of the time on the instruments.
  - **Collaboration:** augmenting the possibilities of the musicians, it augment the possibilities of a band both in studio and in live performances
  - **Learning Curve:** there should be an initial complexity to become familiar with the device, but it is designed to be controlled using the familiar techniques of the musicians. Moreover there should be a very high ceiling of virtuosity, depending on the skills of the players.
- **Context:**
  - **Music Style:** it should be used in every music genre that fits with the players creativity.
  - **Physical and Social Environment:** the device should be designed to be easyly attached to the instrument and to be independent from the context.

Following the above user perspectives, the **technologies** involved in the specifications of the interaction are the following:

- **Control:** depending on the operation mode, it should trigger MIDI notes or control changes.
- **Mapping:** one ribbon softpot potentiometer with both a discrete (virtual frets) and continuous scale, and one piezo to trigger the actual notes and on other to switch between the two softpot behaviours.
- **Operation Freedom:** there are no limits, because it is just a controller.
- **Embodied Facilitation:** the device is designed to let the musician performs which much freedom as possible without altering its techniques too much.
- **Input:** the sensors are triggered by the contact of the players fingers.
- **Feedback:** following the guitar string metaphors, we designed the device to have visual, tactile and sonic feedback.

Keeping the previous considerations in mind, we now present the actual state of the OddString according to the NIME model presented in [8]. As show in figure 2, our device is acting as a new way of input, which is processed by Teensy and then modifies the signal. The *input*, as described, before, is the **combination** of the piezo sensors plus the softpot ribbon potentiometer. As stated in all the literature related to NIME, the *mapping* component is crucial. In order to achieve a good mapping between input

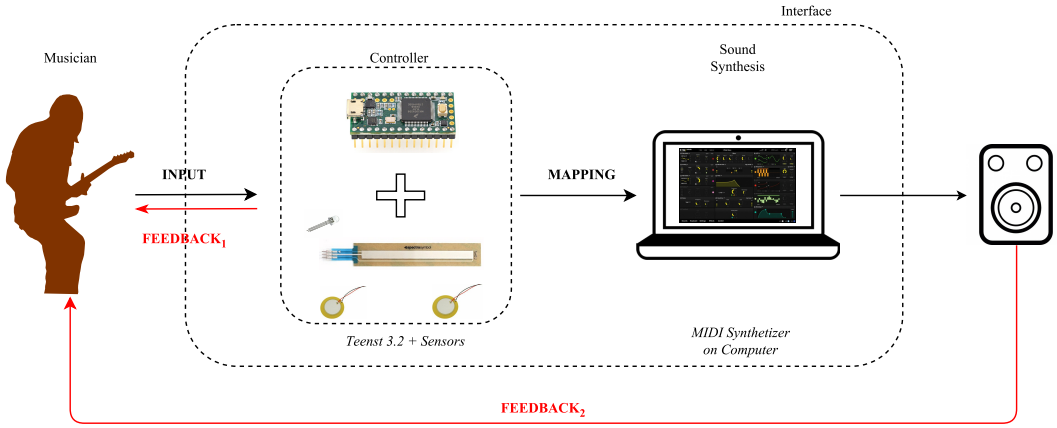

Figure 2: NIME model diagram for the Oddstring

effort, sensor and output sounds, we tried to find the natural metaphors to keep the device as simple as possible for the user. For the string mode, we implemented the behaviour of the sensors so that the device should appear as a guitar string, with a part where a note can be triggered and a part where its pitch is controlled. For the control mode, the distance on the softpot is naturally mapped to the intensity of a preset effect. The visual *feedback* provided by the led helps the player to understand which mode he is using: string or controller mode. Moreover the visual and tactile feedback, with the sofpot and piezos are useful to have a correct and natural understanding of each device part. Finally of course the sound feedback, provided by the overall sound output, both when acting as a string or as a controller.

4. IMPLEMENTATION

In this section we present the materials used for the design, implementation and testing of the OddString prototype. We will also present how the design has been performed to fit the Minuet framework to act as a new interface for musical expression.

4.1 Electronics

For the design of the electronics we used a **Teensy 3.2 USB** [5] development board, 72 MHz Cortex-M4. The Teensy board is very useful since it allows to do fast development compatible with arduino IDE. The Teensy board is connected to two piezo sensors and the softpot ribbon potentiometer as shown in figure 1. Some resistors were used in order to limit current peaks of the piezos and a micro USB cable is used to send the information from Teensy to the computer using **usbMIDI**. A RGB Led is used to provide some visual feedback about the *mode* the user is using (see next section).

4.2 Software

The code implementation has been done using arduino IDE. The **main loop** processes the sensors, determines the frets if the OddString mode is in the string mode, and triggers the notes by sending the midi note to the computer/synthesizer. An overview of the main loop as a flow diagram can be seen under figure 3.

In order to read the softpot sensor, we are taking 20 samples each time, ordering them from lowest to highest, taking the middle values and performing an average. This is not necessarily but it has proven to increase the signal resolution by a small percentage. The case of the piezos is different, the input signal is quite dirty, with plenty of noise. There is also a problem with this, because piezo sensors trigger two times, one when pressed and one when released. In order to solve this issues, we process the piezo signal using an onset detection algorithm, which consists of taking the fft of 64 samples with zero padding, then we calculate the energy by summing the power of two of consecutive fft bins, and finally after a moving average, threshold the peaks with adaptive thresholding [6].

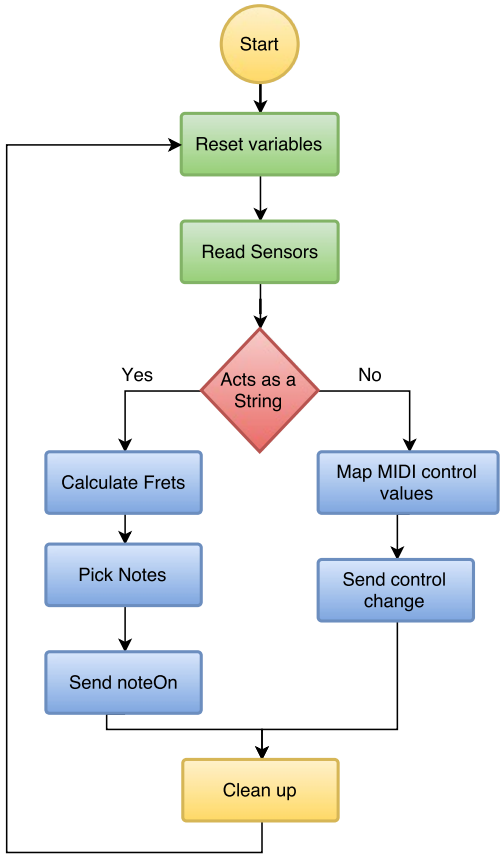

Figure 3: Flow diagram of the main loop

Once the sensors are read, the function **determineFrets** is responsible to take this value and assign it to a particular fret by comparing the softpot value to the stored fret values. This stored values are calculated by calibrating the softpot

on beforehand.

```
for (int j = numFrets; j > 0; j--) {
  int response = false;
  while (!response) {
    int piezoVal = analogRead(piezoPins[0]);

      if (piezoVal > PIEZO_THRESHOLD) {
        int fretVal = analogRead(PIN_SOFTPOT);
        sensorMin = fretVal;
        val = fretVal;
        response = true;
      }
    }
    .
    .
    .
    EEPROM.write(j, val);
  }
```

Listing 1: Code extraction from calibration

4.2.1 Calibration

In order to calibrate the frets of the softpot, we iterate the **number of frets** and assign a integer value when the player press the piezo to indicate the position. Then we store this value in the **EPROM** for future performances. The LED is used to indicate the player when he or she can press the piezo to store the value. A fragment of the calibration code is shown in listing 1

5. CONCLUSIONS

OddString is one of the many steps made towards extending string instruments such as electric guitar or bass. We developed OddString having in mind the difficulties players have when approaching new interfaces, since sometimes it can be quite hard to adapt. For this reason, OddString is designed so that it extends the player capabilities with as few limitations as possible. The implementation of the OddString was simple, the only struggles arised when working with piezos, since the signal is not very reliable. This could be improved by using different kind of sensors like pressure sensors, or some other sort of tactile sensor. The packaging of the OddString is also very important for us, since it should not damage the image of the original instrument. We hope OddString can be used in real live performances by great artists.

6. ACKNOWLEDGMENTS

We would like to thank, Dan Overholt for helping us with the design and interaction, also Francesco Grani for helping us with the electronics and providing us with the Teensy board. We would also to thank Aalborg University CPH for the support and providing the softpot ribbon potentiometer.

7. REFERENCES

[1] Alesis air synth website. `http://www.soundonsound.com/sos/mar03/articles/alesisairsynth.asp`. Accessed: 2016-11-06.

[2] Arduino ribbon guitar. `https://github.com/deanm1278/Arduino-ribbon-synth`. Accessed: 2016-11-06.

[3] Carlos vamos youtube channel. `https://www.youtube.com/watch?v=UOJTMsKR7ag`. Accessed: 2016-11-06.

[4] sensus. `http://www.mindmusiclabs.com/`. Accessed: 2016-11-06.

[5] Teensy. `https://www.pjrc.com/teensy/td_midi.html`. Accessed: 2016-11-06.

[6] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler. A tutorial on onset detection in music signals. *Speech and Audio Processing, IEEE Transactions on*, 13(5):1035–1047, 2005.

[7] N. Bouillot, M. Wozniewski, Z. Settel, and J. R. Cooperstock. A mobile wireless augmented guitar. In *NIME*, pages 189–192, 2008.

[8] E. R. Miranda and M. M. Wanderley. *New digital musical instruments: control and interaction beyond the keyboard*, volume 21. AR Editions, Inc., 2006.

[9] F. Morreale, A. De Angeli, and S. O'Modhrain. Musical interface design: An experience-oriented framework. In *NIME*, pages 467–472, 2014.

[10] D. Overholt. The overtone violin: A new computer music instrument. In *Proceedings of the 2005 International Computer Music Conference*, pages 604–607. Citeseer, 2005.

[11] L. Reboursière, C. Frisson, O. Lähdeoja, J. A. Mills, C. Picard-Limpens, and T. Todoroff. Multimodal guitar: A toolbox for augmented guitar performances. In *NIME*, pages 415–418, 2010.