Three Prototypes of a Force-Sensing Device for the GuitarAMI System

Hyejin Lee* McGill University
Montreal, Canada
hyejin.lee2@mail.mcgill.ca

Kathleen "Ying-Ying" Zhang
McGill University
Montreal, Canada
kathleen.zhang@mail.mcgill.ca

ABSTRACT

The device proposed in this paper is meant to become part of the greater GuitarAMI ecosystem for augmented instruments. Though the word "guitar" is a prominent part of the name, we consider a force-sensing device to be an asset for any performer and as such discuss three possible builds for this device: two for seated musicians and a third for a standing musician. These three prototypes were constructed using high-load force-sensitive resistors and evaluated using simple effects mapped to their output values. These preliminary use-cases show that these three module builds are promising proof of concept for future development and deployment as musical interfaces.

1. INTRODUCTION

The goal of this project is to provide proof of concept for a force-sensing device that could become part of the GuitarAMI ecosystem. The GuitarAMI is a nylon-string guitar with sensors attached to the body of the instrument, making it an augmented musical instrument [5][6]. The particular device we have propose, however, does not take information from the instrument but the the musician. By creating a force-sensing layer, we wish to gather gestural data from a musician's shift in movement. Thus, it can be used by guitarists, but also by other musicians connected to the GuitarAMI system. The data is by nature time-based and could be an intentional musical gesture, or an ancillary gesture that does not produce sound, but nonetheless accompanies a musician's sound-producing performance gestures [9]. For an example relevant to our evaluation, body posture is an important aspect of expression for cellists, even though it is not sound producing [7]. That is not to say that we are only interested in ancillary gestures. The movements sensed by this particular device are necessarily restricted to the shift of a musician's trunk (with apologies for the under-foot module, which is designed for someone standing). As such, we wanted to pair it with ancillary gestures that may already exist, but the force seat could also be used with musical intention. As such, we have created a preliminary mapping structure for a couple of simple effects (a delay and an octaver) to be used for evaluation. The mapping for the octaver in particular is designed to piggy-back off of a cellist's ancillary gestures. When the cellist leans forward to apply more force to their playing, the amplitude of the synthesized tone also increases. When the cellist leans right to play a low note, the octaver introduces a higher octave accompaniment, and vice versa. In this way, the cellist may play naturally, but also with intention. Eventually, our device will be wirelessly compatible with the Raspberry Pi (more in Section 4, Future Work), but for our initial designs we experimented with three different build types that were implemented in SuperCollider via Arduino Uno.

*

2. METHODS

The creation of the force-sensing seat can be split into three sections: prototyping, which involves the creation of the physical hardware of the device, data acquisition, and mapping this data to a few simple effects.

2.1 Prototyping

Three prototypes were created in order to test three different performance data acquisition methods. Initially, it was proposed that there be a single board that could be placed under the chair of a seated performer or under the feet of a standing one. This evolved into three different modules that could be compared by use-case: two for a seated performer, and one for a standing one. The general design of each module is similar, as each uses four sensors to gather data about force distribution. In fact, the main change is from where these data are gathered.
The current idea is that all three of these modules can be swapped from the same microprocessor.

2.1.1 Circuit

As all modules used the same sensor and microprocessor, the circuit used is very simple. The sensors we used were Tekscan Flexiforce high-load force sensitive resistors (FSRs) that were rated for 100 pounds. These sensors were received as analog inputs by an Arduino Uno.
At first we used the Tekscan recommended circuit for a voltage divider, which is pictured in Figure 1 [8]. Note that the Tekscan recommendation did not include a specific amount of resistance, we experimented until we found a level that seemed to work well for our purposes.
However, this gave us issues with floating values when multiple sensors were connected to the same microprocessor. The first solution we attempted was to add a very small capacitor in parallel with our voltage divider to smooth the data. However, after some debate and input from members of IDMIL, it was determined that the best course of action was to change the circuit to one that would read as constantly high when the pin received no input. The revised circuit is shown in Figure 2.

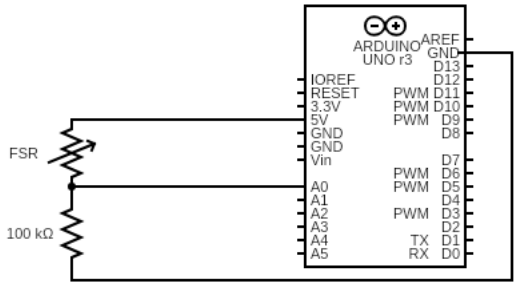

Figure 1: Original circuit diagram (showing one FSR) which lead to floating pins when more FSRs were added.[2]

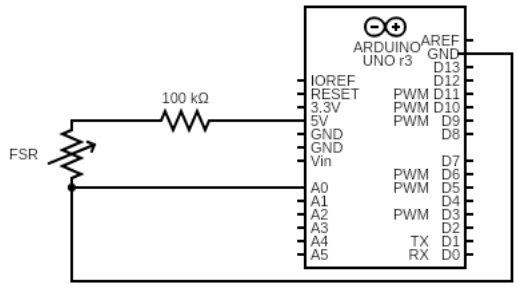

Figure 2: Revised circuit diagram (still only showing one FSR) which employed a pull-up resistor to fix floating pin issue.[2]

2.1.2 *The under-chair module*

This module consists of four separate FSR units that fit under the legs of a chair or piano bench. The idea here is to measure the shift in force of a seated musician. Our main concerns with this particular design was the amount of force that would be concentrated on the relatively small surface area of the FSR. Concerns were even raised that this would break the relatively thin sensor. In order to distribute some of the weight and protect the sensor, it was padded first with craft foam on both sides and then attached to a silicon cup-like structure for both further padding and to keep it from slipping out from under chair legs. A pocket was sewn into foam in order to keep the FSR in place. Figure 3 shows this prototype.

2.1.3 *The small seat module*

An issue we thought of for the under chair module is that many musicians tend to sit on the edge of their seat, in particular cellists and often pianists. In this case, we considered the option of creating a seat that would be more central to a musician's position to more finely measure changes. The small seat measures 30 cm x 17 cm, has an FSR in each corner, and is made of MDF that is covered by a cushion on the top side, while craft foam protects the sensors on the bottom. One element of concern was that the bodies of the sensors do overlap in this use case. Evaluation found that the noise created by this was fairly negligible once a body was seated. Figure 4 presents two views of this module.

2.1.4 *The underfoot module*

Seen in Figure 5, the underfoot module is quite similar to the small seat, excepting that is 43 cm x 30 cm. It is large enough that sensors do not touch at all, and since it is meant to be stood on the top is bare. However, there is still a layer

[2]Created using circuit-diagram.org

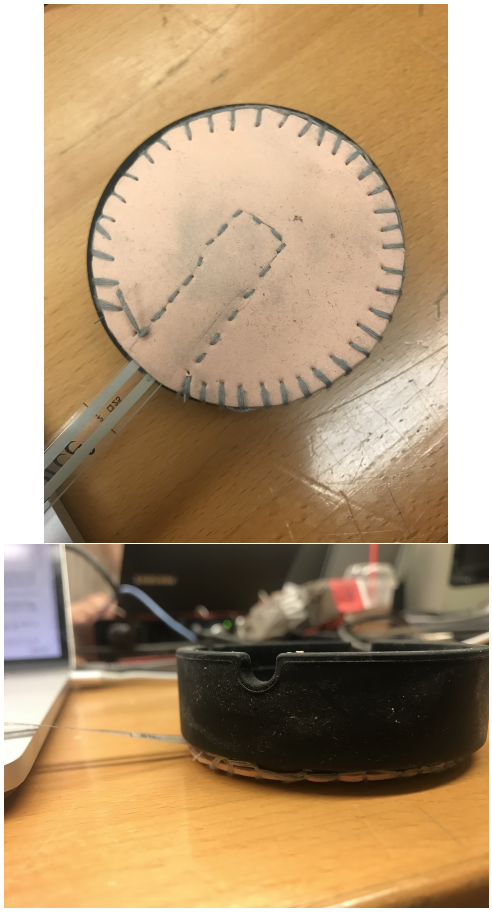

Figure 3: Bottom view and side view of the under-chair module

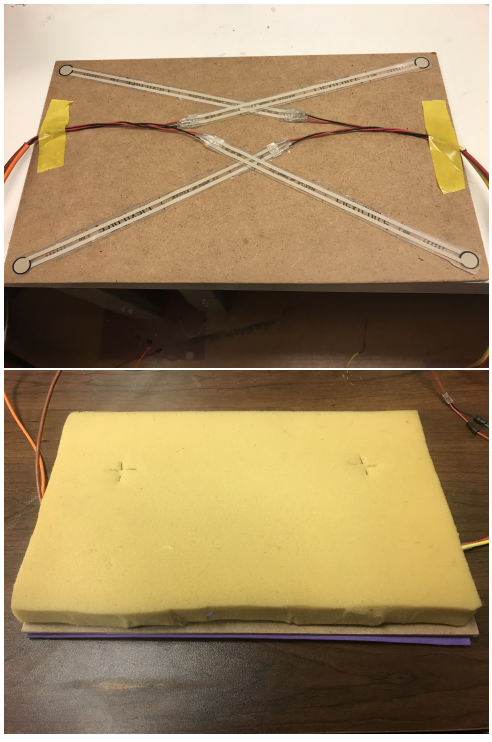

Figure 4: Sensor view and cushion view of the small seat module

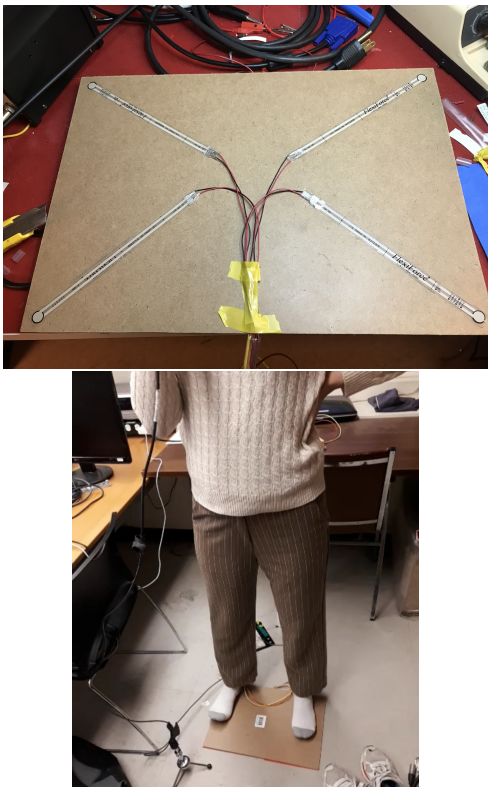

Figure 5: Underfoot module: close-up and with human for scale

of craft foam to protect the sensors from the floor on its bottom.

2.2 Data Acquisition

2.2.1 *Data Transfer*

To transfer our analog data from the Arduino microprocessor to the computer, we installed the Arduino Uno board manager and driver to enable the data acquisition via USB port. Then, we received the four analog values from the specified analog pins (A0, A1, A2, A3) and transformed them into digital values using the AnalogRead function, which is a multi-channel, 10-bit analog to digital converter built into the Arduino system. The values manipulated from this point on were integers between 0 and 1023 that were mapped from the zero to five volts received by the analog pins.

2.2.2 *Data Validation*

As the raw data started from 1023 and decreased as the weight pressure increases, we subtracted 1023 from the initial values and multiplied them by minus one. By importing this operation at the beginning of the program, we could handle the sensor data in a way that increases from 0 with more pressure, which facilitates an easier manipulation of the data.

2.3 Mapping and Effects

2.3.1 *Continuous Mapping Strategy for a Coordinate Plane*

On a plane of X-Y coordinates, we mapped the two parameters—amplitude of delayed signal and delay time—into each axis as illustrated in Figure 6. Four sensors were mapped into each quadrant on the plane. With the input values of each sensor, the change of values in X axis facilitated the function on SuperCollider to modify the amplitude and Y axis to affect the delay time of the sound.
To determine when to operate the functions, we subtracted B (The sum of the sensor 2's value and the sensor 4's value) from A (The sum of the sensor 1's value and the sensor 3's value), and increased or decreased the amplitude by the difference. In the same way, we subtracted D from C and modified the delay time by the value. In this way, we could give a longer delay to the sound when the musician leaned forward and a shorter delay when backward. We also adjusted the sound larger when the musician leaned to the right side and softer to the left side.

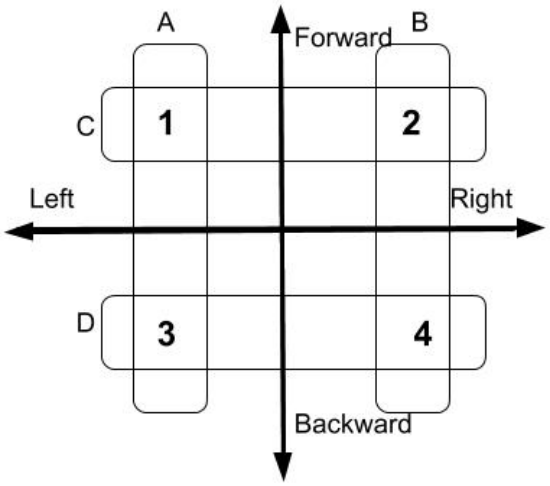

Figure 6: Mapping on a X-Y coordinate plane

2.3.2 *Discrete Mapping Strategy for a Coordinate Plane*

The other mapping strategy we considered was to modify the frequency along with the amplitude to add different sound effects such as octave doubling or chord progression. Values on the X-axis remained the same to affect the amplitude of the sound. However, values on the Y-axis were processed discretely to add a note that is one octave higher or lower, depending on whether the player leans forward or backward. With this mapping strategy, we implemented a subsystem of this project that modifies the above two parameters of a procedurally-generated sound.

2.3.3 *Generation of Sound Effects on SuperCollider*

The software program of this project was implemented in SuperCollider Version 3.10.3 under the GNU General Public License. Two functions were executed simultaneously on SuperCollider to provoke the sound effects.
In the first function, we used the SoundIn object[2] to read the audio input from the external microphone placed adjacent to the player. At the same time, in the main function, we retrieved the four sensor values from Arduino and generated each sound effect in four different conditional branches, each representing the four quadrants on the plane as in Figure 6. Values to change the amplitude and the delay time were sent as arguments from the main function to the first function, and used to synthesize the input signal in the SynthDef class, an inherent class for customized sound synthesis.[4]
Afterwards, the synthesized signal was sent to the output bus, which resulted in the sound being played from the connected speaker. The source code of this project is available at a open-source repository on Github.[3]

2.3.4 *Data Calibration*

The four sensors had subtle fluctuations in their values, which varied from zero to ten without pressure. Thus, the amplitude and delay time repeatedly increased or decreased regardless of the input value when we simply calculated

the subtraction and applied changes to the parameters. To avoid the frequent and meaningless changes, we adjusted the threshold of starting functions during the calibration. We gave changes in amplitude or delay time only when the difference between A and B, or C and D was bigger than 10, which could be further adjusted to detect the input pressure more accurately.

3. EVALUATION

We are currently in the process of both qualitative and quantitative evaluations. Qualitative evaluation was conducted with a cellist use-case and the delay effect for instrument augmentation. Because cello is one of the instruments we had particularly in mind when designing the mapping, it made sense to use it in order to test the seated measures.

3.1 Initial qualitative assessment

Because there were concerns that the prototypes, notably the under-seat module, would even work given the use of FSRs, we are happy to report usable data from all three modules with no clipping of the analog signal from the resistors.

There were instances when, after a long period of testing, one of the sensors would seem to get stuck in a "high" state, however that seems to be a physical issue with the design of the under-seat module in particular, as over time the craft foam discs covering the FSR become compressed. By contrast, the silicone cup seems to have much less by the way of memory, and is less prone to compression while springing back to its original state sooner. As such, the future redesign of this particular module should be made with a similarly robust material, perhaps also over a hard surface.

We found that it was possible for a seated performer to manipulate the delay effect by shifting their weight on the seat. However, we found that our mapping scheme was ultimately disruptive to the musician and will need to be adjusted to be usable. While the concept of creating the effect was to piggy-back off of a musician's existing ancillary gestures in order to create musical meaning, we found that the musician would have to lean unreasonably far in each direction in order to make a meaningful difference. The forward and backward leaning control in particular will need to be adjusted, as cellists and pianists tend to sit very far forward on their seats anyway, so the action of leaning backwards to turn an effect off is especially outside of the normal trunk movement of a seated musician.

We found this effect to be especially bad in the under-seat module where the activating the back sensors would require fully sitting back into the seat of the chair, but even with the much smaller surface area of the small seat module we experienced similar issues.

Thus, we propose the creation of a center (0,0) position more in line with a musician's natural center of gravity and measure forward displacement from that point in order to activate the effect. This would be more in line with how a musician would move to begin with, and would not necessitate leaning all the way back. It could also be used as a displacement point for left and right leaning that could be more subtle.

Although this discussion point is most evident in the application of the seat-based modules, the underfoot module could also do with a serious reassessment, as a similar issue exists where the performer is forced to the margins of the device. Whether a similar re-mapping will be beneficial for this particular application as well will await further evaluation.

3.2 Initial quantitative assessment

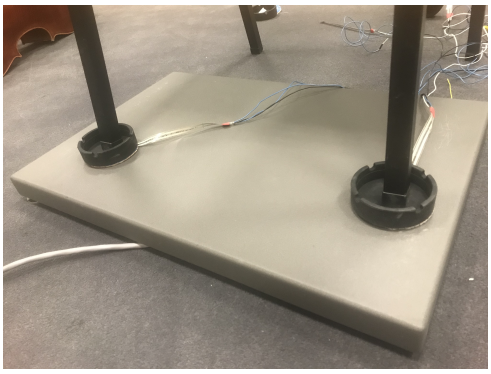

Figure 7: Under-chair module with the Bertec force plate

A pilot quantitative assessment is underway that compares the output of the our device with that of a force plate. Initial measurements have been taken at the Center for Interdisciplinary Research in Music Media and Technology (CIRMMT) using a Bertec Force Plate. Initially, we will attempt to calculate two discrete points of force, using part of the under-chair module as pictured in Figure 7. Data was gathered from the Arduino IDE and from the force plate's digital output, as well as by using Bertec's own data gathering application and force calculations [1]. After calculating the correct data from the force plate's output, a comparison over time (both outputs were time stamped) of data fluctuation as a force is applied to the chair can be mapped.

4. CONCLUSIONS AND FUTURE WORK

Through the design and evaluation process, we learned about transducing a musician's movements to mappable data that would be both musically interesting and possible to control for the performer. Further development of the device would include concluding the evaluation process by trying out the new mapping strategy outlined above, as well as completing our pilot assessment with the information from the force plate. The FSRs have proven to be more resilient and useful than many thought they would be, but they are perhaps not an ideal sensor for this application. As such, we will also be repeating the prototyping process with strain gauges and comparing them our current sensors. If FSRs still prove to be a good design choice, replacing them with a more suitable shape may be the next step after that.

Switching our current microprocessor with one that can communicate wirelessly with the Raspberry Pi is the next critical step in becoming actually integrated into the GuitarAMI ecosystem. Now that we have proof of concept, designing better enclosures is another critical step, as well.

In addition, we could revisit mappings and effects. Placing the octave doubling effect on a pre-generated sound instead of the microphone input as described in 2.3.1 could make it an interesting controller. Future work that employs the pitch detection of the audio input and adds the corresponding octave or chord to it in a real time may result in having more abundant musical effects for the players and their performance. Panning would also be a rather intuitive way for a musician to be able to spatialize their sound, and could even be implemented in quad across the X-Y plane using the current mapping scheme.

References

[1] B. Corporation. *Bertec Force Plates*. Bertec.

[2] E. F. Supercollider tutorial: 20. microphones

and soundin. https://www.youtube.com/watch?v=3vu4UbS2NMw, 2018.

[3] H. Lee. Github repository of the guitarami project. https://github.com/ty1279/guitarAMI, 2019.

[4] J. McCartney. Synthdef, client-side representation of a synth definition. https://doc.sccode.org/Classes/SynthDef.html.

[5] E. Meneses. Guitarami: Development and use of an augmented musical instrument as an artistic creative tool. *NICS Reports*, (18), 2017.

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

[7] J. Rozé, M. Aramaki, R. Kronland-Martinet, T. Voinier, C. Bourdin, D. Chadefaux, M. Dufrenne, and S. Ystad. Assessing the influence of constraints on cellists' postural displacements and musical expressivity. In *International Symposium on Computer Music Multidisciplinary Research*, pages 22–41. Springer, 2015.

[8] Tekscan. Best practices in electrical integration of the flexiforce^{TM} sensor. https://www.tekscan.com/flexiforce-integration-guides, 2018.

[9] M. M. Wanderley, B. W. Vines, N. Middleton, C. McKay, and W. Hatch. The musical significance of clarinetists' ancillary gestures: An exploration of the field. *Journal of New Music Research*, 34(1):97–113, 2005.