Brushing Interface - DIY multi-touch interface for expressive gestural performance

Jaehoon Choi
Rensselaer Polytechnic Institute
Troy, New York
USA
jsonchoi@jsonchoi.io

ABSTRACT

This paper presents the design of the *Brushing Interface*, which aims to transform brushing gestures into a genuine and expressive musical/sonic performance. To achieve this, a hardware system consisting of a grid of 216 self-made force sensitive resistor(FSR) sensors and 8 piezo microphones was implemented, which enables high-fidelity gesture tracking and sound production closely tied with brushing gestures. The hardware system, including the sensor itself, was made in a DIY approach, which provides an economical and high-quality design strategy for implementing a multi-touch interface. Moreover, it is combined with a unique gesture mapping strategy that integrates multi-dimensional parameter mapping and continuous gesture tracking, enabling an expressive performance that is highly flexible to configure in various settings.

Author Keywords

NIME, Digital Musical Interface, Gestural Interface, Brushing, Performing, DIY

CCS Concepts

•**Applied computing** → **Performing arts;** •**Hardware** → *Haptic devices; Sound-based input / output;*

1. INTRODUCTION

The gesture is a crucial element in a musical experience. In musical communication, the sensory-motor gesture to the instrument by the performer gets encoded as sound and decoded through the body by the listener.[16] This demonstrates that "musical communication is based on the sharing of neural structures that pertain to movement," which makes it a universal form of expression and creativity.[16] Moreover, the role of gesture in music is not only limited to communication but also fosters musical creativity. David Wessel wrote that a non-fully-intentional bodily interaction with the instrument is crucial for both learning and exploring the "instrument's potential for musical expression."[27]

In electronic music, the design and making of digital musical interfaces are among the most relevant research fields for musical gestures. Unlike acoustical instruments, the physical structure of digital musical interfaces is not bound to sound production.[16, 17] This makes the mapping between the controller and the sound engine a design task[17], and various mapping strategies have emerged in the musical interface research literature. In fact, due to the significance of embodied gestures in music, these designs and mapping strategies suggest various new modes of *musicking*, which makes designing the digital musical interface beyond a technical construction, and an integrated act of musical composition, sound synthesis, and interaction design.[17]. Related to this idea, Perry Cook suggested a design principle in a succinct manner saying "make a piece, not an instrument or a controller."[7]

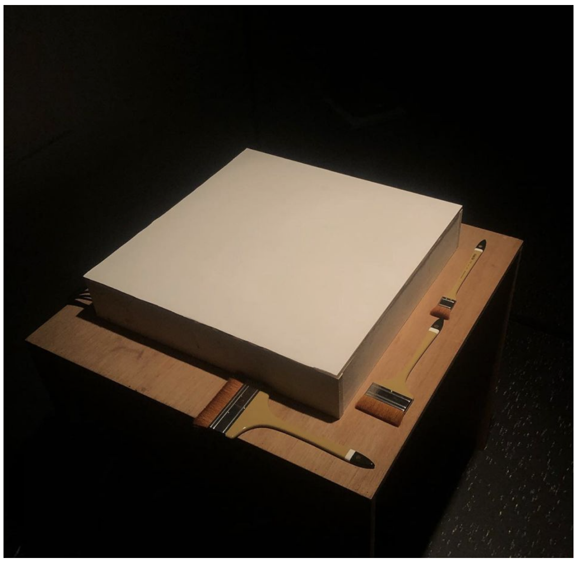


Figure 1: Brushing Interface

The *Brushing Interface* (Figure 1) was made to transform brushing gestures into a genuine musical/sonic expression. Since 2019, I have been working on a series of projects that aims to transform the nuanced materiality of the brush and brushing gestures into a musical/sonic expression. To fully utilize the brushing gestures, a digital interface that can detect the position and pressure of the brushing gesture and its sound in realtime had great potential due to the significance of gesture in musical communication and creativity as mentioned above. The *Brushing Interface* was made from these motives. The work includes a physical interface made in full DIY with a genuine mapping strategy that is flexible, expressive, and well coupled with brushing gestures and the brush's nuanced materiality.

NIME'23, 31 May–2 June, 2023, Mexico City, Mexico.

After the introduction section, this paper is followed by a survey of related works relevant to this project, the design/implementation of the interface, a discussion about the composition and performance that utilizes this interface as a core element, and a brief conclusion with future plans for the project. This paper contributes to the broader NIME community in two ways. First, it provides another DIY method for building a high-dimensional multi-touch sensing interface, which is economical for building and maintenance. Second, it shows a genuine mapping strategy for continuous gestures that is expressive and flexible.

2. RELATED WORKS

Some of the most notable works that expand digital musical interface into an integrated musical practice are Michel Waisvisz's *The Hands*[25] and Laetitia Sonami's *Lady's Glove.*[24] Their works are significant not only because of the instrument's unique assemblage of the interface, mapping, and sound engine, but how they created a performative expression based on their self-made interface. For example, Waisvisz, after finding the setting he liked, stopped changing the configuration of the instrument and focused on developing the performative expression through his gestures, which demonstrates the notion of merging instrument-making and musical performance.[17] The *Brushing Interface* was developed based on this approach, with a motivation to expand and transform brushing gestures into a sonic performance.

There were numerous attempts to connect the act of drawing/brushing with musical expressions through an electronic interface. Some of the early works include the *Oramics Machine* developed by Daphne Oram during the 1960s[18] and *UPIC* (Unité Polyagogique Informatique du CEMAMu) developed by Iannis Xenakis in 1977,[2] which both aimed to implement a sound-drawing machine. However, these projects have relatively less focus on the gestural expressiveness of drawing/brushing motions.

Expanding drawing and brushing gestures into a musical performance has been a vital research and musical interest of the broader NIME community. For example, Sen, Tahiroğlu, and Lohmann developed the *Sounding Brush*, which is an iOS application that enables musicking with "natural gestures of drawing and mark making on a tablet device."[23] Kazuhiro developed a physical interface named *DrawSound* that transforms drawing gestures into sound performance using multi-touch technology.[15] It is implemented by using DiamondTouch multi-touch input interface[9] and two types of custom-made conductive pens. The *Deckle Project*[4] by Choi, Granzow, and Sadler took a slightly different approach from the previous two. Unlike using a pre-existing multi-touch interface[1], the *Deckle Project* built a touch-sensing interface themselves using infra-red object tracking, conductive fabric, magnetometer, and an Arduino. They also added piezo microphones to get the drawing sound so that the performance can be "coupled tightly to mark-making gestures"[4] with the integration of the sensor system.

Deckle Project's bottom-up development approach provides "the artist the capacity to hold macroscopic and microscopic views of the project concurrently," which enables the project to be "in a more aesthetically coherent totality."[19] This methodology was also applied to the *Brushing Interface* project to achieve a similar objective. Moreover, since the *Brushing Interface* aimed for a high-dimension touch sensing capability, making sensors in a DIY fashion was required due to its high cost. Various custom haptic sensors have been researched in the NIME community, such as capacitive sensors implemented with transmitter and receiver electrodes[20] or a metal plate[5], camera-based techniques[8, 13, 14], force sensitive resistors(FSR) using conductive strings[21] or conductive fabric,[10] and etc. For the *Brushing Interface*, Velostat,[11] which is a pressure-sensitive conductive sheet, was used to implement the FSR sensors due to its low cost, stability, and flexibility.

[1]iPad and DiamondTouch

3. INSTRUMENT DESIGN AND IMPLEMENTATION

The main motive for building the *Brushing Interface* was to create an interface that can transform brushing gestures into an expressive sonic/musical performance. To achieve this, the *Brushing Interface* was built with an emphasis on flexibility and expressiveness, which have been implemented both in hardware and software.

3.1 Hardware

The first step of this project was to create a stable FSR sensor for building a high-dimensional sensor grid. The FSR sensor was made by attaching two copper tapes on each side of Velostat,(Figure 2, 3, 4) and each copper tape is soldered to a wire where one of them is connected to the ground and analog input through a resistor, and the other is connected to the power.

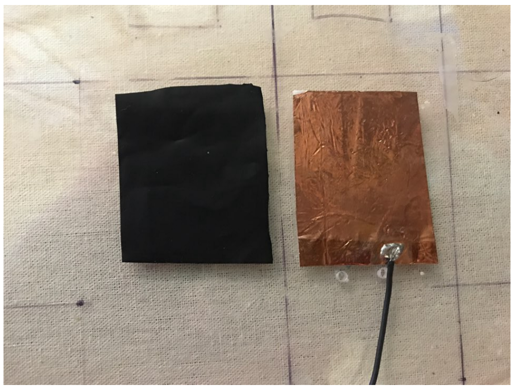


Figure 2: Velostat(left) and bottom layer copper tape

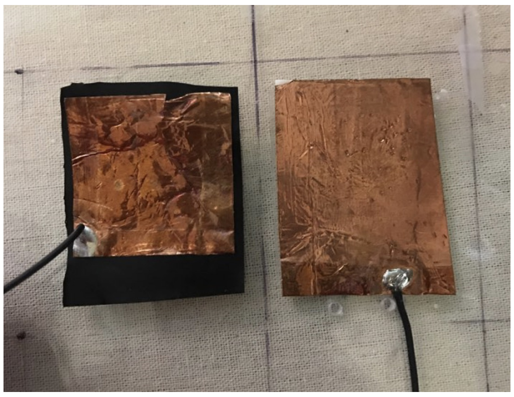


Figure 3: The upper layer copper tape on the Velostat(Left) tape.

This sensor design is affordable to build, solderable, and the size/shape are modifiable. It also showed decent sen-

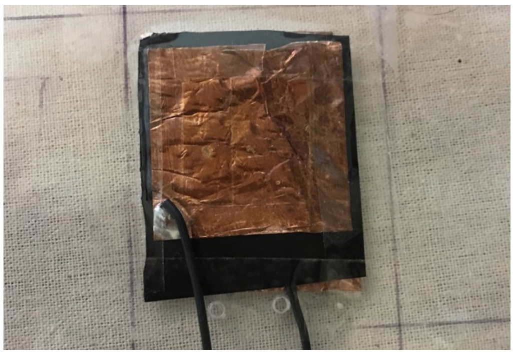


Figure 4: The upper layer copper tape and Velostat taped on the bottom layer copper tape.

sitivity and resolution that is demonstrated in this video.[2] In this demonstration, the Max patch is only receiving the pressure data from one FSR sensor located on the left edge, which is 3cm x 3cm wide. The serial data from the sensor was scaled from 80 - 280 to 0 - 127 MIDI value for convenience. This clearly shows that the sensor has sufficient stability and sensibility for brushing performance.

For additional evaluation, Donneaud and their research group's method was used; stacking coins on the sensor "to measure how a change in weight relates to a change" in output data.[10] Korean 100-Won coins were used for this testing (Figure 5). I measured the serial data output while putting a stack of coins on the sensor and increased the size of the stack incrementally in the range of 10. The output serial data were measured every 30 seconds, for 10 minutes. The serial output value was scaled from 80 - 500 to 0.0 - 1.0. In the plot (Figure 6), the MIN shows the lowest output value among the 30-second timestamps, the MAX shows the maximum among those timestamps, and the AVERAGE shows the average of every timestamp measurement per stack of coins. The linear trendline of AVERAGE was $y = 0.1022x - 0.0712$, $R^2 = 0.9853$, the linear trendline of MAX was $y = 0.0976x - 0.1114$, $R^2 = 0.9795$, and the linear trendline of MIN was $y = 0.1143x - 0.0167$, $R^2 = 0.974$. These results showed decent linearity and consistency of the output data.

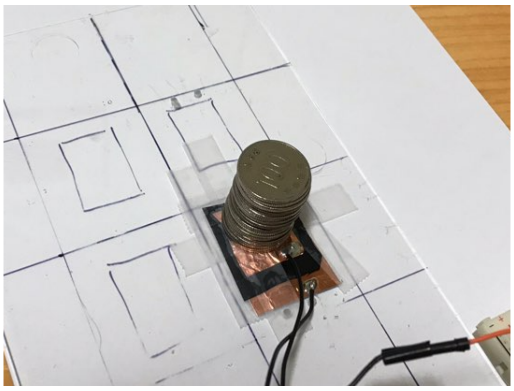


Figure 5: Sensor Evalutation

This sensor design provides an affordable way to build a sensitive, stable, and high-resolution FSR sensor. However, some inconsistencies were observed compared to commercial FSR sensors while I was using and making the *Brushing Interface* for the past few years. The resistance range often

[2]https://youtu.be/AaryPoLBCMA

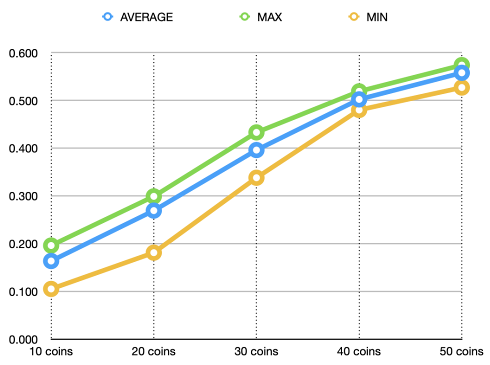


Figure 6: Sensor Evalutation Results

varied due to the slight differences between materials[3], and the DIY manual process of building them also caused some inconsistency among the sensors. Therefore, calibrations had to be done on the software side, and for this project *odot*[3] was used to manage the calibration. However, even though there were some inconsistencies compared to commercial FSR sensors, the immense low cost, solderability, decent sensitivity/resolution, stability, and flexibility in size and shape, make this approach useful for DIY instrument makers. Especially, if the project involves a high number of FSR sensors, this sensor design enables instrument makers to implement it with affordable cost, flexibility, and decent quality.

A Teensy microcontroller[4] was used to process these sensor inputs through Arduino code. To connect 216 sensors with one Teensy board, multiplexers were necessary, and the SparkFun Analog/Digital MUX Breakout — CD74HC4067[5] multiplexers were used to achieve this. This particular multiplexer was useful because it has 16 input channels, which is plenty for this project, and also has an Arduino library[26] that makes the programming part simpler. Custom-designed printed circuit boards(PCB) were used for connections between the multiplexer and sensors (Figure 7). PCBs enhanced the stability of the interface immensely by preventing shortage and lowering the odds of potential damage from external forces. In addition, PCBs provided more visibility to the circuit, which made maintenance and debugging much easier. The PCBs were designed through KiCad (Figure 8) by myself and manufactured from Devicemart [6] which is a third-party PCB manufacturer based in South Korea.

On the top surface of the interface, 216 self-made FSR sensors and 8 piezo microphones were mounted on the acrylic board (Figure 9). This enables realtime detection of the position, strength, and sound of the brush gesture. As mentioned previously from the *Deckle Project*, using the brushing sound from piezo microphones for the performance makes it closely attached to the brushing gestures.[4] Also, as a design aesthetic, electronic components, such as wires, cables, and circuits, were intentionally hidden from the exterior (Figure 1). This clean and minimal design consisting of wood and paper creates more alignment with the brush while emphasizing gestures through its cleanness.

3.2 Software

[3]Velostat sheet and copper tape
[4]https://www.pjrc.com/teensy/
[5]https://www.sparkfun.com/products/9056
[6]https://www.devicemart.co.kr/main/index

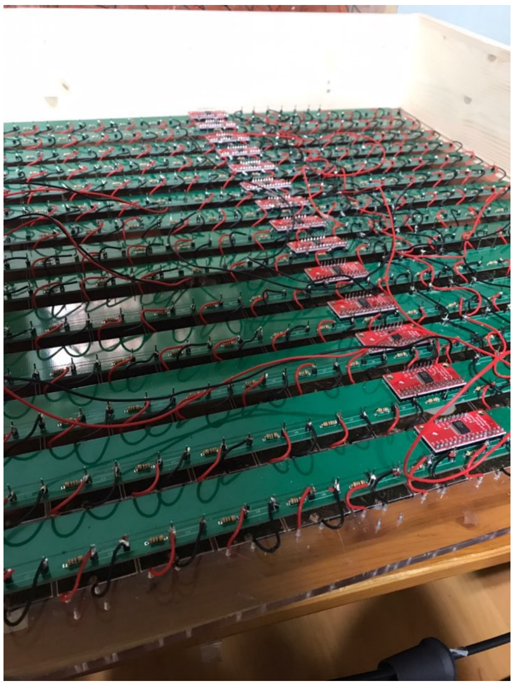


Figure 7: PCBs mounted on the inner part of the interface

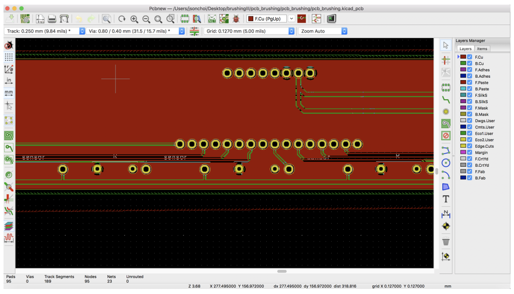


Figure 8: KiCad project screenshot

The software portion of the system was built entirely with Max[7], which mainly functions as the realtime sound engine and the gestural mapping. The sound engine is highly customizable, but it mainly focuses on two modules. One that utilizes the brushing sound from the piezo microphones and applies various audio processes/effects to it, and realtime audio synthesis. The most critical part of the software system is how the sound engine is mapped with the sensor inputs from Teensy. Sensor inputs are sent as serial data to the Max patch, and it is pre-processed through *odot.*[3] Due to its code scripting functionality and OSC(Open Sound Control) based messaging, it can easily pre-process sensor data, such as parsing, assigning variable names, setting thresholds, scaling and etc, in realtime with efficient computation.

The pre-processed data is used for the gestural mapping, which was implemented with *rbfi*(radial basis function interpolator)[12], and *gf* (gesture follower) from *MuBu*[22]. *Rbfi* (Figure 10) is a Max object developed from CNMAT that enables multi-dimensional parameter mapping on a 2-dimensional plane and enables users to continuously interpolate within the parameter space rather than sequencing discretely.[6, 12] The performer can move the mouse point of *rbfi* on the hardware interface by controlling the position of the brush and the strength applied to it. When a

[7]https://cycling74.com/products/max

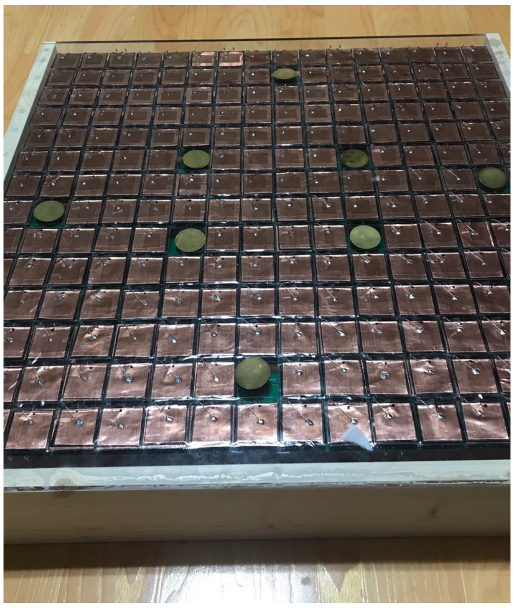
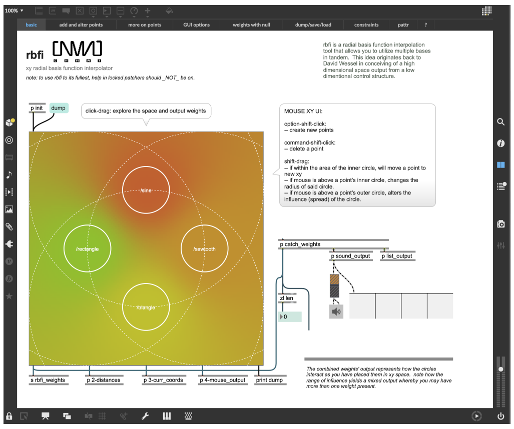


Figure 9: Top surface of the interface before paper attached

certain amount of pressure is applied to the hardware interface through the brush, the Max patch calculates that position and glides the rbfi's mouse-point to that position. Also, the performer can control the speed of that movement by adjusting the amount of pressure they apply to the hardware interface. Due to this tight attachment with the brushing gesture, and the capability to interpolate multiple parameters through a two-dimensional plane, *rbfi* was mostly mapped directly to various parameters of the sound engine.



xy radial basis function interpolator

note: to use rbfi to its fullest, help in locked patchers should _NOT_ be on.

rbfi is a radial basis function interpolation tool that allows you to utilize multiple bases in tandem. This idea originates back to David Wessel in conceiving of a high dimensional space output from a low dimentional control structure.

click-drag: explore the space and output weights

MOUSE XY UI:

option-shift-click:
-- create new points

command-shift-click:
-- delete a point

shift-drag:
-- if within the area of the inner circle, will move a point to new xy
-- if mouse is above a point's inner circle, changes the radius of said circle.
-- if mouse is above a point's outer circle, alters the influence (spread) of the circle.

The combined weights' output represents how the circles interact as you have placed them in xy space. note how the range of influence yields a mixed output whereby you may have more than one weight present.

Figure 10: *rbfi* help file

Gf of *MuBu* provides "the similarity of the performed gesture to prerecorded gestures," and "the time progression of the performance gesture" in a continuous mode.[1] If *rbfi* was mapped with the sound engine, *gf* was mapped to the higher level "states" of the Max patch, such as presets of the sound engine or the mapping of the *rbfi* itself. For example, interactions like changing the mapping setup of *rbfi* when the similarity of the gesture is over 80% with the pre-recorded one over 3 seconds can be implemented.

4. PERFORMANCE

Due to its design and technical implementation, the *Brushing Interface* enables a deeply layered and expressive sound

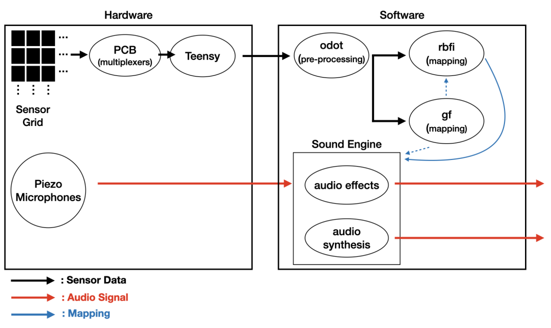


Figure 11: System Diagram Summary

performance through brushing gestures. Brushing gestures can be played in four different performative approaches, using the default brushing sound input, applying audio effects to the brushing sound by moving rbfi's mouse-point position, realtime audio synthesis mapped to the sensor inputs, and changing the higher-level "states" or presets of the patch using *gf* from *MuBu*. Therefore, a performer can use gestures in a mixture of those approaches which makes it expressive and versatile while adding depth and sophistication to the performance.

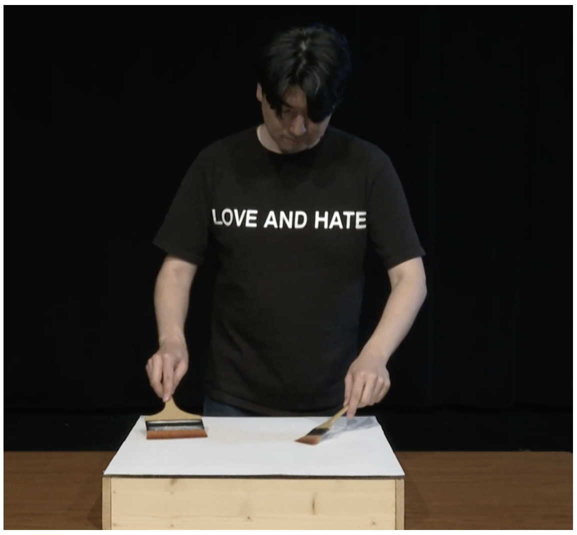


Figure 12: *Drifting* (2022) performance video

Drifting[8] (Figure 12) was the first composition that was made specifically for this interface, and the performative approaches mentioned above were used in this piece. For example, in this composition, the brushing sound input from piezo microphones was convolved by various audio samples such as glass sound, water sound, metal sound and etc. The mix of the original brushing sound and these convolved sounds are determined by the output weight from the *rbfi*. Also, when the brush moves up and down on the side edge of the interface with a certain amount of pressure for an extended amount of time, it shifts into a different setting or section, which was implemented with *gf* from *MuBu*. As this demonstrates, this methodology provides rich expressiveness to electronic music performance, and with the flexible configuration this system is capable of, it has great potential for various new performances.

[8]https://jsonchoi.io/drifting.html

5. CONCLUSION

In this paper, I presented the *Brushing Interface*, which expands brushing gestures into a genuine and expressive musical performance. Through this approach, brushing gestures are mapped in a continuous and multi-layered fashion, which enables a performative and rich gestural expression. This is achieved by implementing gestural mapping with multi-dimensional parameters and higher-level states and presets, which can be highly effective when mapped with various synthesis and audio processing techniques. Also, the physical interface is made in full DIY, including the sensors, which makes the construction process affordable and eases maintenance. For future plans, I aim to explore further to it an interface for improvisational performance. Its richness and versatility can be well suited to an improvisational setting, and this will be explored in all aspects of this project including gesture, sound, and design.

6. ACKNOWLEDGMENTS

This work was supported by Korea Creative Content Agency, Art Center Nabi, and Rensselaer Polytechnic Institute. Also, I would like to thank Matthew Goodheart for his ongoing mentorship on this project and Romain Michon for his help in the early stage of the project on sensor development.

7. REFERENCES

[1] Gesture follower. `https://ismm.ircam.fr/gesture-follower/`. Accessed: 2023-01-09.

[2] R. Bourotte. The upic and its descendants: Drawing sound 2012. In *Proceedings of the International Symposium "Xenakis. The electroacoustic music"(Paris, France: Paris 8 University, 23–25 May 2012).*, 2012.

[3] J. Bresson, J. MacCallum, and A. Freed. o. om: structured-functional communication between computer music systems using osc and odot. In *Proceedings of the 4th International Workshop on Functional Art, Music, Modelling, and Design*, pages 41–47, 2016.

[4] H. Choi, J. Granzow, and J. Sadler. The deckle project : A sketch of three sensors. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, 2012.

[5] J. Chui, Y. Tang, M. Marafa, and S. Young. Solotouch: A capacitive touch controller with automated note selector. In *Proceedings of International Conference on New Interfaces for Musical Expression*, 2013.

[6] CIRMMT. David wessel - designing musical instruments that privilege improvisation. `https://youtu.be/uGASpqTXz4g`, 2012. Accessed: 2023-01-09.

[7] P. R. Cook. Principles for Designing Computer Music Controllers. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 3–6, June 2001.

[8] A. Crevoisier and G. Kellum. Transforming ordinary surfaces into multi-touch controllers. In *NIME*, pages 113–116, 2008.

[9] P. Dietz and D. Leigh. Diamondtouch: A multi-user touch technology. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*, UIST '01, page 219–226. Association for Computing Machinery, 2001.

[10] M. Donneaud, C. Honnet, and P. Strohmeier. Designing a multi-touch textile for music performances. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 7–12, 2017.

[11] A. Dzedzickis, E. Sutinys, V. Bucinskas, U. Samukaite-Bubniene, B. Jakstys, A. Ramanavicius, and I. Morkvenaite-Vilkonciene. Polyethylene-carbon composite (velostat®) based tactile sensor. *Polymers*, 12(12), 2020.

[12] A. Freed, J. MacCallum, A. Schmeder, and D. Wessel. Visualizations and interaction strategies for hybridization interfaces. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 343–347, 2010.

[13] J. Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 115–118, 2005.

[14] C. Harrison, H. Benko, and A. D. Wilson. Omnitouch: wearable multitouch interaction everywhere. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 441–450, 2011.

[15] K. Jo. Drawsound: a drawing instrument for sound performance. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 59–62, 2008.

[16] M. Leman. *Embodied music cognition and mediation technology.* MIT press, 2007.

[17] T. Magnusson. *Sonic Writing: Technologies of Material, Symbolic, and Signal Inscriptions.* Bloomsbury Academic, 2019.

[18] P. Manning. The oramics machine: From vision to reality. *Organised Sound*, 17(2):137–147, 2012.

[19] S. Penny. *Bridging two cultures: towards an interdisciplinary history of the artist-inventor and the machine-artwork.* Hatje Cantz, 2008.

[20] J. Rekimoto. Smartskin: an infrastructure for freehand manipulation on interactive surfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 113–120, 2002.

[21] J.-S. Roh, Y. Mann, A. Freed, and D. Wessel. Robust and reliable fabric, piezoresistive multitouch sensing surfaces for musical controllers. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 393–398, 2011.

[22] N. Schnell, A. Röbel, D. Schwarz, G. Peeters, R. Borghesi, et al. Mubu and friends–assembling tools for content based real-time interactive audio processing in max/msp. In *Proceedings of the International Computer Music Conference*, pages 423–426, 2009.

[23] S. Sen, K. Tahiroğlu, and J. Lohman. Sounding brush: A tablet based musical instrument for drawing and mark making. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 331–336, 2020.

[24] L. Sonami. Lady's glove. `https://sonami.net/portfolio/items/ladys-glove/`. Accessed: 2022-12-25.

[25] G. Torre, K. Andersen, and F. Baldé. The hands: The making of a digital musical instrument. *Computer Music Journal*, 40(2):22–34, 2016.

[26] P. Wasp. Cd74hc4067. `https://github.com/waspinator/CD74HC4067/blob/master`, 2016.

[27] D. Wessel. An enactive approach to computer music performance. *Le Feedback dans la Creation Musical, Lyon, France*, pages 93–98, 2006.