Groovebox: Exploration of drum groove generation from hand percussion

Anonymous Author(s)

Figure 1: Two synthesisers on a table, 2019.

Abstract

This paper introduces Groovebox, a percussion DMI that takes an active role in the creation and performance of music. Users and the device interact in a call-and-response format, similar to some jazz improvisation styles, where both entities are inspired by the other's performance while still contributing creatively to the piece.

Keywords

ML, Generative Music, DSP, Onset Detection, Variational AutoEncoder

1 Introduction

Digital Music Instruments (DMIs) are often limited to responding to user input. A keyboard, for example, takes user key presses and translates that into either MIDI messages or sound. The same is true for electric drum kits, digital synthesizers, and many more. This means that any musical creativity comes solely from the player (occasionally inspired by algorithms built into the DMI, for example chord machines or arpeggios), which is often desirable if said player has a clear musical idea, but doesn't afford any opportunity for collaborative improvisation (like would happen when playing with others). The player's technical ability is also often a limiting factor in musical creation in these contexts, as virtuosity and control often only comes with long hours of practice, as is the cost of the DMI, which is often high enough that the number of people that can afford to use it is severely limited.

We propose *Groovebox*, an instrument designed with the goal of being intuitive, expressive, and accessible. In broad terms, it is a percussive DMI that allows users to improvise in call-and-response fashion, dynamically responding to user input and enabling the exchange of musical ideas between the player and the device. It is designed to be primarily used in the context of performance, as there is some variability to the responses it forms based on the player's input.

To play the instrument, the user drums a rhythm on the top plate with their hands, and the VAE model generates a corresponding drum beat, which will be played in a loop over MIDI until another hand-drummed sequence is played. The user can also apply pressure to the top plate in different ways to modulate the outputed drum beat in different ways, for example increasing the volume when more pressure is put in.

In this paper we explore the effectiveness of our onset detection algorithm and the generative model, and qualitatively evaluate the system as a whole.

NIME '25, June 24–27, 2025, Canberra, Australia

2 Background and Prior Art

DMIs have been an active field of research and innovation for decades [11]. With the increasing technological capabilities afforded by faster processors, more storage, and other technological innovations, more and more people in both the academic and commercial world have explored the ways computers can be used to make music. Computer-based technologies (e.g. digital synthesizers) use computer programs to synthesize sounds. The first of these was the Fairlight CMI, which was a keyboard with an on-board computer [11] [20]. Through the years, computer-based musical devices have evolved and diversified to include Digital Audio Workstations (DAWs), drum machines, sequencers, and so on [20]. Many of these have an emphasis on portability and ease of integration into home studios or stage set ups. For example, the OP-1 is a popular portable synthesizer and DAW (among other things) with a particular emphasis on having a small form factor [2], and many ways to integrate computer or other equipment via MIDI and USB interfaces. The Novation Launchpad is another popular MIDI controller with compact design and easy integration with a computer running many major DAWs [1]. More experimental DMIs also exist, for example "The Bean," which uses pressure applied to points on a compound curve to play notes, somewhat like a harp [16].

Robotic musical instruments, which can be defined as "sound-making device that automatically creates music with the use of mechanical parts, such as motors, solenoids and gears," have also had a large amount of development in recent years [17]. These have included player pianos, robotic drummers, violinists, cellists, drummers, robotic bagpipes, and more [17]. Many of these robots are performers which play an instrument, but some act as a musical interface. One example of this is robotic systems that react to human dance [27].

The study of Human-Robotic Interaction (HRI) and its sibling Human-Computer Interaction (HCI) have been popular for decades [22] [26]. Within the context of musical systems this has taken the form of wearable e-textile interactions, gesture-sensor-based interactions, mobile-device interactions, laptop-based interactions, and more [29] [24]. For haptic interfaces (which focus on the sense of touch), the interpretation and communication of emotion (referred to as "affective haptics") is of particular interest [28]. Affective haptic interfaces can include force and touch feedback, and data collected for affect detection may be intensity and duration of interaction, mode of interaction, and so on [10]. Another area of study of emotion communication between humans and some sort of electronic or electromechanical interface (especially within the context of DMIs) is parameter mapping, which is the mapping of various inputs to system outputs [15]. An example of this could be increased pressure on a pad leading to louder volume. These physical interactions can broadly be broken up into two categories: ergotic and non-ergotic, where ergotic interactions are considered to have work exchanged between the user and the system, and there the system responds with proportional haptic feedback [6]. An example of an ergotic system might be a drum, where work is done by hitting the top membrane is the vibration of the instrument. Ergotic interactions can be further broken down into three sub-interactions: excitation, where energy is put into the system (e.g. hitting the drum), modification, where the sound is altered (e.g. applying pressure to the drum head to change the pitch), and selection (e.g. choosing which drum to hit at a drum kit) [6]. Some consider musical instruments to be ones where sounds are produced by ergotic gestural interactions [6]. As such, when designing a DMI it is important to consider whether or not one wishes to make the instrument ergotic or focus on the user interacting with it from a purely intellectual and symbolic standpoint, and the impact that will have on the broader user experience.

As stated earlier, there are barriers to entry for any instrument. This can be not only technical difficulty, but also cost, and many instruments fall short of the needs of users with disabilities [12]. As such, accessible DMIs have been created with varying goals from trying to create low-cost open-source devices to better server a wider base of musicians and lower the monetary barrier, and the field of adaptive music has grown in pursuit of designing instruments that serve a broader community of musicians with diverse abilities [12].

As digital systems have grown able to play a more active and complex role in music, some researchers suggest that some digital musical systems can be treated as their own "behavioral objects" and that the traditional musical paradigm must be re-framed to include these behavioral objects and the ways they can interact with their environments [5]. Practically this means considering that some digital systems might be "composed" in a similar way to one composes parts of a musical piece (for example in the case of modular synthesizers), exhibit behaviors in response to some internal or external state, and have various interactions with any human (or non-human) actors in the broader musical system. Another manifestation of this can be seen in generative music systems, where the digital system itself is a compositional actor within the process of music composition [5]. Such systems are their own field of study unto themselves, and generally have a similar structure: they may have some overall narrative structure, external interaction, or other input that informs the composition, which consists of melody, harmony, rhythm, and timbre, which are often encoded as a series of notes [14]. Often these systems have been built using systems like MaxMSP or Pure Data [5], but in recent years machine-learning has been an increasingly popular tool for generative music, arrangement, and orchestration. In particular Variational AutoEncoders (VAEs), which take an input, compress it down into a latent-space representation, and then reconstruct the input from the latent-space representation have shown to be well-suited for this task [25]. This is because the latent-spaces can be shaped during model training to resemble human-creatable representations of information, from which the model's encoder can generate something that looks like it could have been from the training data [25]. Google Magenta's GrooVAE model was trained on drum groove MIDI data and had its decoder essentially distill the input into a monophonic rhythm, from which it then was trained to recreate the original drum groove [13]. This means one could then input a tapped rhythm, and get a passable drum groove out of the model [13].

3 Requirements

Groovebox was imagined as a percussive instrument that can be struck or otherwise played with the hands, be intuitive to play without much prior expertise, compact and modular enough that it can be thrown in a bag and set up anywhere quickly and easily, and have a low enough cost of production that it is accessible to most musicians.

3.1 Gestural Interaction and Input

We envisioned gestural interactions to include both percussive striking with the hands and sustained pressure on the top plate, so the device must have some method of detecting both percussive onsets, sustained presses and swipes along the 2D plane of the top plate. This would necessitate a reliable method of differentiating the two. Force or touch feedback is often necessary for ergotic interaction, and so the system must have some method of providing haptic feedback to user interaction. Additionally, the system should detect where percussive interactions are performed on the surface of the device to allow for variation in produced sound.

3.2 Visual Design

Two characteristics of design are discoverability (is it possible to figure out what actions are possible and how to perform them with an object) and understanding (how is the object supposed to be used, what do all the different controls do, etc.) [23]. For the system to be intuitive, it should have both of these, so the user must be able to understand how to interact with it by it's shape, form, and visual design The mappings between gesture and system output must also be easily understood through the device's shape, form, coloring, and so on.

As such we felt it appropriate to draw inspiration from other hand-percussion instruments (which often have a rim and membrane) by having a visually distinct rim and and center area. At the same time, however, we also wanted to differentiate the instrument enough from preexisting acoustic drums to invite the exploration of different types of gesture and interaction.

3.3 Portability

For the device to be easily portable, it must be small enough to fit in a common backpack. As such, it must have a footprint of no more than 1 square foot and a height of no more than 3 inches. The device must also be self-contained and require no more than a single USB cable for power and MIDI data transmission, so that it can be set up in different locations.

3.4 Use Context

The device is designed to be used on a tabletop, alongside other DMIs such as synthesizers. It should easily integrate with an arbitrary home studio ecosystem, or be easy to "slot into" more temporary setups like on stage.

3.5 Cost and Development Materials

To ensure the device can be used by a broad user-base, the device must be under $75 USD to fabricate. To further this goal and ensure the device is as accessible as possible, the source code and build plans are open-source and free.

4 Design

4.1 Top Plate

The top plate of Groovebox is a triangular sheet with rounded corners sitting on top of three piezoelectric pressure sensors, allowing localization of pressure placed on the plate as well as vibration sensing. Paper pressure sensors [18] [19] were also considered and investigated, but not selected due to the time investment required to create quality sensors reproducibly.

The top plate is split into three colored zones to play the kick drum, snare drum, and hi-hat or ride (selected by toggle switch), enabling users to play simplified 3-note drum beats which will inform the two bar, fully orchestrated response from the model. The rest of the drum kit is broadly categorized to fit within these three categories: cymbals and other related metals with the ride/hi-hat zone, snare and toms with the snare zone, and kick drum. In order to detect which zone the user played, the use of separate plates per zone was considered, but to simplify the design we opted to use a single top plate with positional interpolation from piezo sensors.

To manufacture this, we 3D printed a rim with edges, see Figure 2, in which a laser-cut acrylic triangle sits as seen in Figure 3. This also serves to provide a visually-distinct "rim" and "head" to the plate, similar to the rim and head of an acoustic drum like a snare or tom drum. Acrylic was chosen so that LED lights mounted inside the enclosure would shine through, allowing us to dynamically color sections of the head around the rim to provide visual feedback and visually separate the zones of the top plate. Colored marker is also used on the bottom of the top plate to diffuse the light from the LEDs and ensure the three zones are always visually distinct, even when the device is off.

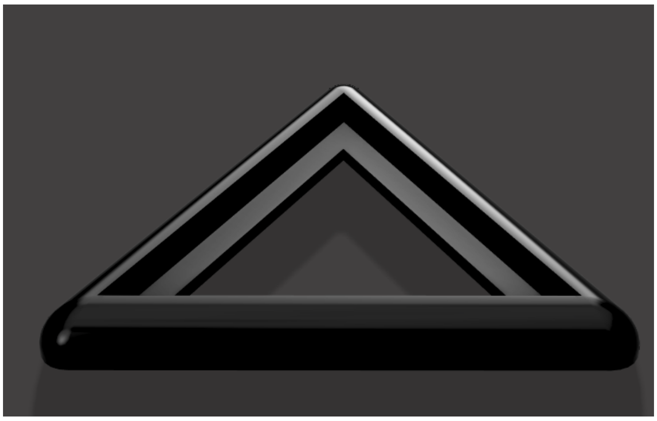

Figure 2: 3D rim for top plate

4.2 Player Feedback

Feedback is provided both through haptics, visuals, and sound. For haptic feedback, a vibration motor is mounted in the center of each zone. These motors provide short-decay vibrations of taps to mimic snares (like one would find in a cajon) and also vibrate the currently-playing groove (each zone vibrating as drums within it's category is played).

Visual feedback is provided by LED lights inside the enclosure. The lights mounted in the bottom two corners (mapped with kick and snare) flash the current tempo the device has inferred from the user-tapped sequence, and those in the ride/hi-hat zone use color to show the selected instrument.

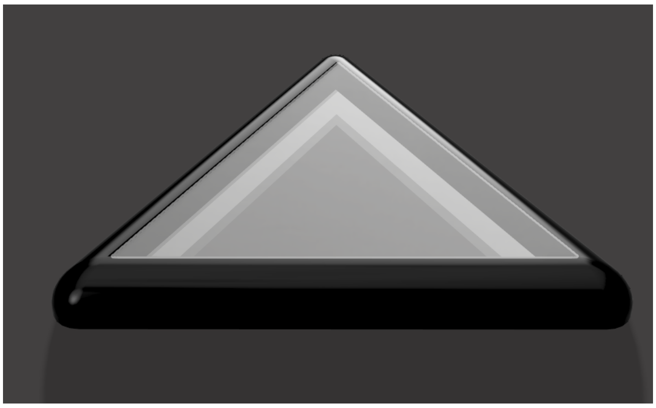

Figure 3: Top plate

Auditory feedback is provided by a surface transducer mounted to a side plate of the device, which mechanically vibrates the plate within the auditory frequency range, thereby producing sound from the plate.

4.3 Gesture Mapping

A capacitive touch sensor on the side of the device (mounted at the corner of the ride/hi-hat section) is used to toggle between whether this zone is mapped to ride cymbal or hi-hat.

When additional force is applied and swiped along one of the sections of the top plate, the velocity of all notes related to the mapping of that section can be increased or decreased. For example, this gesture could be used in the hi-hat zone of the top plate to not only affect the volume of the hi-hat but also of cymbals.

Hits within each zone are mapped to a note for the zone's associated sound (kick, snare, and ride or hi-hat). Since force is often associated with volume [6], presses within each zone with swipes towards or away from the top plate's center can be used to affect the volume of the instrument's group (e.g. a press with swipe towards the center in the hi-hat zone would reduce the volume of the hi-hat and all other cymbals. This is facilitated by reducing the velocity of the MIDI notes outputted by the system and reducing the volume of these sounds created by the device.

Tempo is stored in global state within the device and inferred from tapped sequences in the center of the device.

4.4 Device Enclosure

A triangular box (with 8-inch-long sides) was laser-cut from acrylic, along with three small square stands on which to mount the top plate. This box houses the electronics, lights, and power circuitry. A piece of rubber is placed under each mounting square to isolate the top plate from other vibrations in the environment, and to give the top plate some ability to move as the user presses into it.

4.5 Electronics

The initial iteration of Groovebox used an Arduino Uno to detect onsets and communicated with a host computer for additional processing and groove generation. This, however, did not meet the requirement of being self-contained so we opted to migrate to a single-board computer to handle all processing and VAE inference on-board the device. The Raspberry Pi 3B and Jetson Nano were considered as we had them readily available, and between

Figure 4: Device with enclosure and top plate

the two the Raspberry Pi was selected due to the relatively low overhead of the machine learning model used.

This Raspberry Pi reads the signal from the piezoelectric sensors, performs signal processing to determine the type of interaction, performs the VAE inference to generate a groove from the user input, and handles groove modulation based on user input.

The Raspberry Pi runs directly on the power of the USB cable used to connect it to a computer, over which the final MIDI data is also transmitted.

4.6 Onset detection

The tap/hit detection system consists of a piezoelectric sensors placed between the top plate and acrylic mounting squares. These sensors utilize the piezoelectric effect to generate small voltages when force is applied, which is read by the microcontroller's ADC and passed through the onset detection pipeline. The original design contained a pipeline of filters and moving averages to remove noise and normalize the input, but this came at the cost of processing time as well as complicating the ability to process sustained presses. In the current software, the raw piezo data is used by the program to determine onsets and offsets. The program will assume that the piezo reads zero when not pressed, and extremely high when pressed on. This behavior can be achieved through careful physical assembly.

The system is designed to have three software processes, the orchestrator process, the signal processing process, and the inference process (see figure **??**. The signal processing process detects user gestures and locations (via spacial interpolation and other signal processing algorithms discussed in 4.6) and passes them along to the orchestrator process. The orchestrator process is responsible to managing system state (e.g. tempo, current groove), assembling user-drummed sequences, coordinating with the other two processes, applying modulations to the system output, and outputting the final MIDI data. The final process, the GrooVAE process, generates a groove from the user-drummed sequence and passes that back to the orchestrator.

4.7 Drum Groove Generation

Google Magenta's Tap2Drum model [13] is the model used in this system. It is a Variational sequence-to-sequence Autoencoder (VAE) of their proposed GrooVAE family, which has three main objectives: humanization, infilling, and generation. This specific model is trained such that the latent space encodes the rhythm. A drum groove can therefore be encoded as a rhythmic sequence, and any rhythmic sequence can be decoded as a full drum groove. The model, dataset, and trained weights are publicly available [13].

This model is retrained to use a latent space consisting of rhythmic patterns of four notes (kick, snare, hi-hat, and ride). Additionally, the data processing is changed so that the model outputs a fully-orchestrated version of the next two bars in the recording rather than just a fully-orchestrated version of the same bar, so that there is some level of prediction of what musical ideas come next. Using the dataset published by [13] target data of this shape is created by processing each groove, and the model is retrained on this. The resulting model is an extension of the Tap2Groove module presented by [13], where multiple notes are considered but the other effects of orchestration and infilling are still achieved.

Other models considered were Google Magenta's MusicVAE [25], which is another VAE where latent space manipulation is utilized to shape the model's encodings to "look" like something producible by a human, namely MIDI, and Meta's MusicGen Model [7], which generates audio based on audio tokens (which they obtain from audio samples using the EnCodec model proposed by [9]).

5 Implementation

Unfortunately, many of the subsections did not get fully integrated, and the system didn't end up being fully functional. Many sub-systems like the onset detection, lights control, haptic feedback control, sound output through the side-mounted transducer, and basic control software worked independently (see videos in Appendix A).

5.1 Circuit Design and PCB Fabrication

The fabrication of a physical circuit to connect all the parts was critical, since the number of integrated components was large enough that a breadboarded solution would not have fit within our size constraints. A perfboard circuit was hand-wired to test the designed schematic, which was used to test and develop light and haptic feedback control (figure 5).

Because this circuit had many tightly-packed wires, the circuit created was not very clean and rather fragile. For this prototype wires were stacked one on top of the other, and for the most part current ratings of wires was not super considered since the assumption was that the final circuit would have high-current power lines. Additionally, the connection between the Raspberry Pi and this circuit was not very good (it was designed to be a "hat" which sat on top of the GPIO pins), likely due to low-quality female header pins and/or not enough splay tension created when soldered.

After realizing the original plan of using the Voltera One in the lab was found to be infeasible (due to the absence ink or an ink head), many attempts were made at milling a double-sided PCB for this design on a CNC router (see figure 6). A design was created in KiCAD, and after many failed attempts to get the software in the Carbide ecosystem working (which was the CNC's manufacturer), pcb2gcode [4] was used to create gcode files and Candle [8] was used to control the CNC as per a couple tutorials [21] [3]. Due to a broken bit being the only on-hand bit

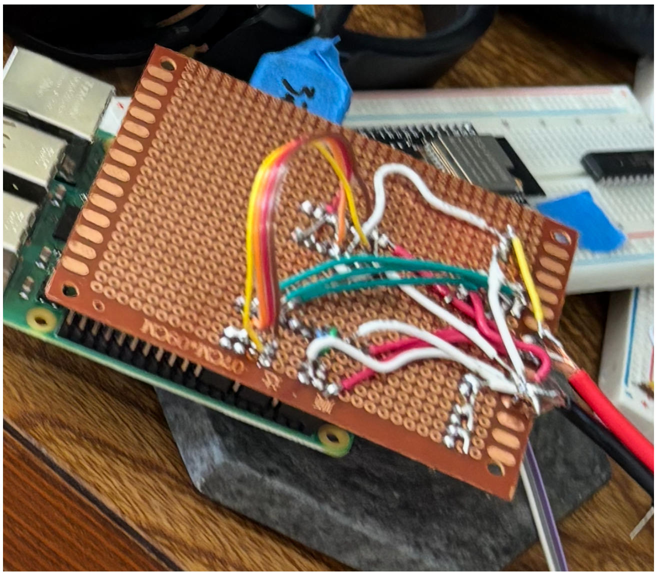

Figure 5: Prototype circuit on perfboard

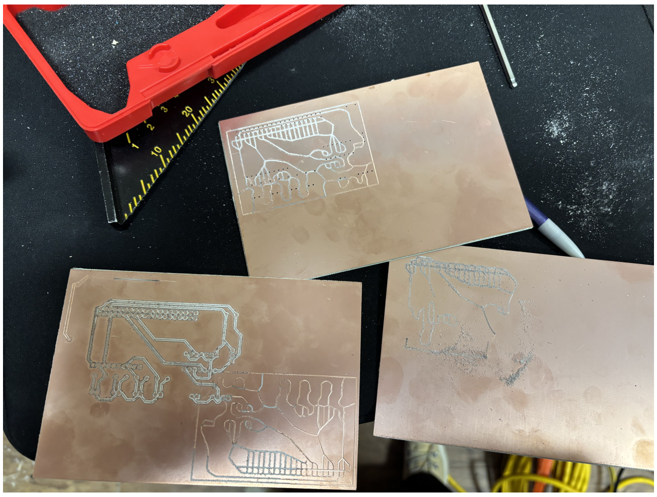

Figure 6: Failed PCBs

small enough to cut traces, it was used to plow into the copper plating of the board, which led to more pressure than would be ideal (causing bowing in regions of the board near the center, and as a consequences not full-depth cuts). Improved results were gotten by using Voronoi routing, which instead of cutting out individual traces instead isolates maximal regions connecting endpoints, leading to less cuts and oftentimes thicker traces. The two types of boards can be seen in figure 6.

Unfortunately, due to issues with reading piezo inputs on the prototyped board, a prior perfboard piezo hookup circuit was used to read piezo inputs, and the circuit was used to power the other subsystems like sound. We unfortunately ran out of time to integrate the lighting and haptic systems, but due to the nature of the perfboard prototype circuit needing to sit on top of the Raspberry Pi, the other piezo board and ADC would not have been able to connect to it had it been in place. This meant not all the boards would fit in the enclosure, and unfortunately the system ended up having to be demonstrated out of it (see figure 7).

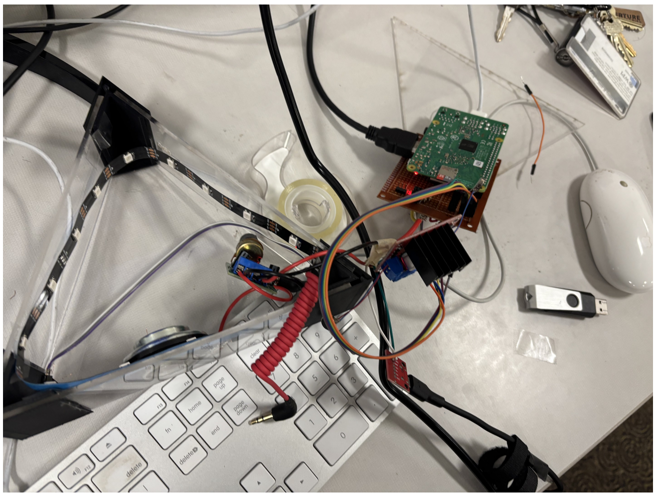

Figure 7: Open device due to multiple boards being used.

5.2 Machine Learning Model

The updated machine learning model was created, trained, and ready to use. Unfortunately it wasn’t integrated onto the Raspberry Pi due to time constraints. Hyperparameter search was successfully used to isolate hyperparameter values that produced acceptable results, however a more robust search would have yielded better performance. Modifying the model’s architecture to have more recurrence (and therefore ability to reckon with time) would have also improved importance, but this was not implemented due to time.

5.3 Software Systems

Tap tempo, location-based onset detection, and user input recording were all implemented. The next steps would have been to integrate communication with a machine learning inference process and add support for extended gestures, but this was unfortunately not done due to us running out of time.

6 Reflections

The biggest obstacle that was encountered during this problem was time. Many tasks depended on other tasks being done, and many people’s progress depended on the success of other people’s tasks (which were not always straightforward). One example of this would be the fact that PCBs didn’t get made meaning the integration of other parts didn’t happen.

If we were to do this again, we would have put together a comprehensive breakdown of the work that needed to be done, sequence the work by identifying what depended on other tasks being done, and set hard deadlines for each task assigned to specific people instead of the softer deadlines (like "the end of the week" sent in a Discord channel) that we had.

Additionally, we would have ordered parts much much sooner, and planned to cut PCBs much earlier (like week 3). In week 1 we also should have verified we had all the tools and parts needed to perform all our tasks (e.g. ink for the Voltera One, V bits for the CNC).

Finally, the last point of contention we had was not having a solidly-defined idea of the interaction between the user and machine until the end of week 5. If we were to do this again, this would have been figured out week 1, and as a result we would have realized that the changes in the machine learning model were necessary much sooner in the term, allowing for more time to train, tune, and test different architectures.

On the other hand, the performance of the model, sound system, and many individual functionality of many subsystems came together relatively painlessly, which was a definite success. Our prior experience with using logic level shifters for controlling lights and other modules informed our decision to use on in this system, and as a result there were virtually no issues controlling the lights or motors.

References

[1] Launchpad | Novation.
[2] OP–1 field.
[3] Rapid PCB Prototyping with KiCad and a CNC Router | Peter Todorov Blog.
[4] pcb2gcode/pcb2gcode, Dec. 2025. original-date: 2015-02-17T10:16:56Z.
[5] O. Bown, A. Eldridge, and J. McCormack. Understanding Interaction in Contemporary Digital Music: from instruments to behavioural objects. *Organised Sound*, 14(2):188–196, Aug. 2009.
[6] C. Cadoz. Supra-Instrumental Interactions and Gestures. *Journal of New Music Research*, 38(3):215–230, Sept. 2009. Publisher: Routledge _eprint: https://doi.org/10.1080/09298210903137641.
[7] J. Copet, F. Kreuk, I. Gat, T. Remez, D. Kant, G. Synnaeve, Y. Adi, and A. Defossez. Simple and Controllable Music Generation. *Advances in Neural Information Processing Systems*, 36:47704–47720, Dec. 2023.
[8] Denvi. Denvi/Candle, Dec. 2025. original-date: 2015-07-15T17:46:48Z.
[9] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi. High Fidelity Neural Audio Compression, Oct. 2022. arXiv:2210.13438 [eess].
[10] M. A. Eid and H. Al Osman. Affective Haptics: Current Research and Future Directions. *IEEE Access*, 4:26–40, 2016. Conference Name: IEEE Access.
[11] S. Fasciani. History of Digital Musical Instruments, Aug. 2014.
[12] E. Frid. Accessible Digital Musical Instruments—A Review of Musical Interfaces in Inclusive Music Practice. *Multimodal Technologies and Interaction*, 3(3):57, Sept. 2019. Number: 3 Publisher: Multidisciplinary Digital Publishing Institute.
[13] J. Gillick, A. Roberts, J. Engel, D. Eck, and D. Bamman. Learning to Groove with Inverse Sequence Transformations, July 2019. arXiv:1905.06118 [cs].
[14] D. Herremans, C.-H. Chuan, and E. Chew. A Functional Taxonomy of Music Generation Systems. *ACM Comput. Surv.*, 50(5):69:1–69:30, Sept. 2017.
[15] A. Hunt, M. M. Wanderley, and M. Paradis. The importance of parameter mapping in electronic instrument design, May 2002.
[16] R. Huott. Precise control on compound curves. In *Proceedings of the 2005 conference on New interfaces for musical expression*, NIME '05, pages 244–245, SGP, May 2005. National University of Singapore.
[17] A. Kapur. A HISTORY OF ROBOTIC MUSICAL INSTRUMENTS. *ICMC*, 2005.
[18] R. Koehly. *Fabrication of Sustainable Resistive-Based Paper Touch Sensors: Application to Music Technology.* PhD thesis, Sept. 2011.
[19] R. Koehly, M. Wanderley, T. van de Ven, and D. Curtil. In-House Development of Paper Force Sensors for Musical Applications. *Computer Music Journal*, 38:22–35, June 2014.
[20] R. Kogan. Brief History of Electronic and Computer Musical Instruments. Apr. 2008.
[21] mattwach. Milling Printed Circuit Boards (PCBs) on a Cheap CNC Machine.
[22] B. A. Myers. A brief history of human-computer interaction technology. *interactions*, 5(2):44–54, Mar. 1998.
[23] D. A. Norman. *The design of everyday things.* MIT press, Cambridge (Mass.), rev. and expanded edition edition, 2013.
[24] J. Qi, L. Ma, Z. Cui, and Y. Yu. Computer vision-based hand gesture recognition for human-robot interaction: a review. *Complex & Intelligent Systems*, 10(1):1581–1606, Feb. 2024.
[25] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck. A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music, Nov. 2019. arXiv:1803.05428 [cs].
[26] T. B. Sheridan. Human–Robot Interaction: Status and Challenges. *Human Factors*, 58(4):525–532, June 2016. Publisher: SAGE Publications Inc.
[27] K. Suzuki and S. Hashimoto. Robotic interface for embodied interaction via dance and musical performance. *Proceedings of the IEEE*, 92(4):656–671, Apr. 2004.
[28] P. V. Unma Desai, K. Yamakawa, and K. E. MacLean. A Descriptive Analysis of a Formative Decade of Research in Afective Haptic System Design, Apr. 2023.
[29] A. Xambó. Embodied Music Interaction: Creative Design Synergies Between Music Performance and HCI. In S. Broadhurst and S. Price, editors, *Digital Bodies*, pages 207–220. Palgrave Macmillan UK, London, 2017.

A External videos, phtotos, and source code

Source code, photos, and video recordings can be found at https://github.com/WheatleyTheCore/Groovebox.

Video of haptic motor control working: https://drive.google.com/file/d/1HjcLWOebkLvY0TNgQXAtHfgimb6ieeb1/view?usp=sharing.

Video of lights working: https://drive.google.com/file/d/16L-5HFANtyg8-DlWtUqhRhX5H6bjK1qp/view?usp=sharing