Software to Improve MIDI Controller Usability

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

William Litton Haslet

Spring, 2023

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Briana Morrison, Department of Computer Science

ABSTRACT
MIDI controllers are a popular tool for music creation, but their potential usefulness is limited by shallow and cumbersome configuration options. To improve the usability of MIDI controllers, I propose a MIDI customization software. The software acts as a virtual controller that translates and transmits incoming MIDI messages in accordance to the configuration provided by the user. The software is developed as a desktop application using NodeJS, enabling it to run on multiple operating systems with ease. The completed software enables real-time extension of MIDI controller functionality using a simple software interface. The music creation process is made more efficient by expanding MIDI control possibilities. In the future, the software can be mechanically improved with user testing to streamline the user experience and aesthetically improved with the addition of an attractive user interface.

1. INTRODUCTION

Alongside computers, the standardization of Musical Instrument Digital Interface (MIDI) in 1983 may have been the key that unlocked the potential of digital technology in music creation. MIDI is a digital communication system originally created to act as an interface between synthesizers, allowing one to play notes and adjust parameters on the other (The MIDI Association, n.d.). MIDI communication involves actions like sending a "note on" message when a key is pressed or sending "continuous control" messages to adjust volume or other parameters when a knob is turned. Today, MIDI enables communication between hardware instruments, computers, and music software, fully integrating the power of digital technology into the music creation process.

A common MIDI tool is the MIDI controller, which acts as an output source of MIDI messages to control connected hardware instruments or music software. MIDI controllers are often designed with the appearance of a keyboard synthesizer, with the primary control being a piano keyboard and extra controls like knobs, faders, and buttons being added for additional functionality. Musicians are able to play the MIDI controller in a similar way to hardware instruments and route the output to whatever hardware or software supports it. With the additional control knobs mentioned previously, it becomes possible for artists to expressively control not only hardware instruments, but also software instruments that would otherwise have no tactile control surface. A way to change the MIDI messages output from the controller using software would streamline MIDI controller usage and expand its uses.

2. RELATED WORKS

There is little relevant research literature for the topic of MIDI controller usability, but there exists software similar to what I propose. One example is Midi Shift (Haute Technique, n.d.), which focuses on accessibility and ease of use by making the MIDI modification simple to set up and activate. My proposed software will build further upon the customization possibilities with the goal of providing greater flexibility to the user.

In addition to MIDI shift, real-time MIDI remapping solutions are built into some music software like Ableton Live. Ableton Live calls this customization control surface scripts and automatically routes MIDI messages depending on the software state (Ableton, n.d.). With my proposed software, user will be able to switch between multiple software or hardware instruments outside of a single software.

3. PROPOSED DESIGN

MIDI controllers are most effective when designed for a specific use case. However, the wide range of hardware and software available make generic MIDI controllers a necessary balance of convenience and cost for many users. Most hardware and software instruments will consistently support MIDI messages output by any controller, but a limitation of the control possibility of MIDI controllers arises when the controller is used between multiple instruments. The expected input MIDI message values will often be different, meaning that additional controls such as knobs and buttons will not function as the user expects or desires. Customization of MIDI controller output is often inconvenient and limited in its potential for complex customization. Both of these problems can be solved with software that acts as a middleman for MIDI messages and allows the user to modify and route the input to a hardware instrument or music software.

The usability and flexibility problems of MIDI controllers can be solved with user created presets of MIDI message modification rules that can be effortlessly switched between. Presets are a concept that many MIDI controllers already implement, but the limitations of these implementations are numerous. Controller or manufacturer specific software is often needed to modify the presets, the modification possibilities are limiting, and switching between the presets is cumbersome. By moving the preset features into a separate software, these problems can be solved. Controller specific software will not be needed because my proposed software will modify the incoming MIDI messages of any controller, so modifications can be centralized to one software. By processing and outputting an entirely new MIDI message based on the user's modification rules, there are no limits to ways the controller can be configured. The preset switching process will be streamlined by allowing the user to define any input as a dedicated preset switching function, allowing them to change presets with a single action.

The system to make these MIDI controller improvements possible will be structured as a simple flow from input to modification to output. The MIDI inputs will be configured by the user, allowing them to route any MIDI controller or MIDI output source to the modification software. For each of these devices, a list of modification rules will be created by the user. The input MIDI messages will be sorted based on their message type and compared against the appropriate rules to determine what should be output. Finally, the correct output message is created and sent into the MIDI output that the user has selected for the preset. Alternatively, the input MIDI message could output nothing, and instead modify the internal state of the software, with actions like switching the active preset, so that later MIDI messages will be modified by the new preset.

The modification rules the user creates will be made up of two parts: the input filter and the applied modification. With the input filter, the user can choose what MIDI input source and MIDI message type is affected by that modification rule. With the applied modification, the user can set the output message type and the message value, or set the message value by processing the input value with simple mathematical rules.

To improve the flexibility of controllers, the option to use inputs to change the software state instead of generating MIDI output will be implemented. The most important of these state changing inputs will be for preset switching, which will be built with a set of rules dedicated to changing the active preset.

The software will be developed with NodeJS and JavaScript to enable rapid prototype

iteration and ease of deployment to all operating systems. The MIDI input and output will be handled using a NodeJS package called node-midi that is a wrapper for the RtMidi C++ library.

4. ANTICIPATED RESULTS

I expect my proposed software to improve the usability, efficiency, and flexibility of MIDI controllers. The usability will improve because the software will bring all the customization options the user needs into one place. Usability will also increase due to the speed with which my proposed software will allow the user to switch between presets, making a direct improvement to the efficiency of using a MIDI controller between multiple output sources. The flexibility improvement is a result of the freedom in MIDI message modifications that my software will allow the user to create. The possibility of advanced configuration will have the effect of increasing the number of actions a single MIDI controller can perform, making it more efficient.

5. CONCLUSION

MIDI controllers are a valuable hardware tool for music creation, but their limited customization options hold back their full potential. With my proposed MIDI modification software, any incoming MIDI messages can be automatically translated into the user's preferred message based on the modification rules the user creates. By moving the customization options from the hardware to software, the user can edit their MIDI controller output with much less effort and test their changes in real time. By providing an easy way to create and switch modification presets, a single MIDI controller can be used across multiple instruments or applications with ease.

6. FUTURE WORK

The largest step toward releasing a version of this software to the public would be making an efficient and attractive user interface. Without a well-designed user interface, the time saving benefits of editing MIDI outputs with software instead of hardware would be lost. The ideal user interface would give users the immediate access to all the features they need to modify their presets without over filling the screen with information. The software would also need to be tested extensively, for both stability and usefulness, in order to have a polished finished product to release.

REFERENCES

Creating your own Control Surface script. (n.d.). Ableton. Retrieved October 27, 2022, from https://help.ableton.com/hc/en-us/articles/206240184-Creating-your-own-Control- Surface-script

MIDI Shift. (n.d.). Haute Technique. Retrieved October 27, 2022, from https://hautetechnique.com/midi/midi shift/

The MIDI Association. (n.d.). *MIDI History: Chapter 6-MIDI Is Born 1980-1983*. MIDI Association. Retrieved October 27, 2022, from https://www.midi.org/midi-articles/midi- history-chapter-6-midi-is-born-1980-1983