Project Proposal

# Description

Many modern electronic keyboards have pitch and mod wheels. These wheels allow a musician to bend the pitch of the notes being played or to add vibrato. These functionalities give musicians the option to play notes with frequencies outside of the normal keyboard vocabulary, which allow musicians to be more expressive in their playing. However, many keyboards do not have these wheels and as such, they are limited in what they can play.

For the final project of MUMT 620, I propose to create a way for a keyboardist to apply the effects of pitch and mod wheels without having the actual pitch and mod wheel hardware. Specifically, the bottom 8 notes of the keyboard (which are seldom used by keyboardists) will be re-mapped.

To create the functionality of the pitch wheel, the bottom 5 white keys (A-B-C-D-E) will be used. The distance between each of these notes will represent a quarter tone, so if a user plays a note with their right hand and then plays “A-B-C-B-A”, the note will rise up 1 semitone, and then go back to its original note. If a user plays “E-D-C-B-A”, the note will fall a whole tone. These transitions will be blended together, to avoid discrete notes being heard during the pitch bend.

To create the functionality of the mod wheel, the bottom C# and D# keys will be used. By playing these notes alternately (as if to play a trill), the “volume” that a musician plays these keys will map to the intensity of the vibrato. If a musician plays a note in their right hand and then begins to trill lightly between the bottom C# and D#, this will cause a faint vibrato. By trilling louder and louder (that is, by pushing the keys harder), this will cause the vibrato to grow in intensity. To be clear, this does not affect the volume of the note being played, just the vibrato.

# Methodology

To complete this project, I will have to use some software and/or write some code to re-map the keys of the keyboard. This software will receive the MIDI output of the keyboard and then run an algorithm to determine what sound to produce.

I believe I could use Max for this project, although I hope to talk to Professor Wanderley and see what he recommends. Alternatively, I believe I could use a simple Python script to write the necessary code. Specifically, libraries such as Python-Midi [1] and Mido [2] provide APIs which allow a Python script to interact with MIDI messages. The general flow of this program would be as follows:

* Read MIDI input
* If one of the bottom 5 white keys are being played:
  + Perform calculations to determine pitch bend
* If the bottom C# or D# is being played:
  + Perform calculations to determine vibrato
* If any other keys are being played:
  + Apply the pitch bend & vibrato effects to the notes being played
* Output audio

Because both the pitch bend and vibrato change over time given a constant input, state variables will need to be used to determine what calculations must be done to produce the desired effects.

# Goals and Novelty

The goal for this project is to create a way of easily applying pitch bend and vibrato without the pitch bend and modulation wheels. This problem has been encountered before and other solutions exist. For example, Andrew P. McPherson augmented a piano with optical scanner hardware to detect continuous piano gestures, which allowed for applications such as pitch bend [3]. Similarly, McPherson et al. applied capacitive touch sensors to a piano to provide continuous pitch control [4]. A widely available solution to this problem is the Roli Seaboard, which has multiple sensors on each key and allows a user to control the pitch, amplitude and timbral variation of the music [5]. Finally, a more unique solution to this problem was presented by Kilian et al., in which electromyography was used to measure muscle activity and map this information to vibrato [6].

These are all valid solutions, yet there all have one major caveat. Each solution requires additional hardware to apply pitch bend or vibrato. There does not currently exist a way to apply these effects using only the hardware provided by a simple electric keyboard. Therefore, the novelty of this project is to provide a software-only solution which anyone can use at no cost.

The idea to change the mapping of a keyboard to accomplish this goal is a unique idea, although alternative piano mappings for specific applications have been done before. For example, Feit and Oulasvirta created an alternative mapping to allow a piano keyboard to be used for text entry on computers [7]. My project will provide another way that alternative keyboard mappings can be used to communicate information other than individual notes.

# References

|  |  |
| --- | --- |
| [1] | G. Hall, "vishnubob/python-midi," 8 July 2015. [Online]. Available: https://github.com/vishnubob/python-midi. [Accessed 4 October 2021]. |
| [2] | O. M. Bjorndalen, "mido/mido," 5 July 2021. [Online]. Available: https://github.com/mido/mido. [Accessed 4 October 2021]. |
| [3] | A. P. McPherson, "Portable Measurement and Mapping of Continuous Piano Gesture," in *Proceedings of the International Conference on New Interfaces for Musical Expression, pp. 152--157*, Daejeon, Republic of Korea, 2013. |
| [4] | A. P. McPherson, A. Gierakowski and A. M. Stark, "The Space between the Notes: Adding Expressive Pitch Control to the Piano Keyboard," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Paris, France, 2013. |
| [5] | R. Lamb and A. N. Robertson, "Seaboard: a new piano keyboard-related interface," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, Oslo, Norway, 2011. |
| [6] | A. Kilean, J. Karolus, T. Kosch, A. Schmidt and P. W. Pawel, "EMPiano: Electromyographic Pitch Control on the Piano Keyboard," in *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, Yokohama, Japan, 2021. |
| [7] | A. M. Feit and A. Oulasvirta, "PianoText: redesigning the piano keyboard for text entry," in *Proceedings of the 2014 conference on Designing interactive systems*, Vancouver, BC, Canada, 2014. |