JuxtaMIDI: Using Data Visualization to Pinpoint Mistakes in MIDI Practice Recordings

Jeremy Grifski and Stephen Wu

Abstract— When a musician wants to practice their instrument, they often have to rely on their peers or an instructor to help them isolate mistakes in their technique. As an alternative solution, we are proposing a system to answer the following question: how can we leverage data visualization to pinpoint mistakes in music data? For the sake of scope, we have chosen to focus on MIDI recordings.

Index Terms—Music, Data Visualization, MIDI.

1 Introduction and Motivation

Music is a profession and hobby enjoyed by many people. Unfortunately, the field hasn't received a lot of attention from the technology community. To this day, musicians still practice their instruments with little to no benefit from technology.

One area of music that could really benefit from a technological upgrade would be practice. After all, practice is usually something that occurs alone without a lot of feedback. Without access to an instructor, musicians may find it difficult to self-assess their abilities. They could all benefit from some sort of tool to help pinpoint their mistakes.

In this project, we built a data visualization dashboard which can be used to compare practice MIDI files with professional MIDI files. The goal is to isolate areas in the practice file which are most unlike the professional file for the sake of improvement.

2 Related Work

While there was plenty of motivation for the project, we still needed to lay the ground work for the project. In particular, we had to come up with some research questions, design goals, tasks, and metrics which we could use to map out our implementation.

2.1 Research Questions

As mentioned previously, the major research question we looked to address is the following: how can we leverage data visualization to pinpoint mistakes in MIDI practice recordings?

Naturally, this question raises several underlying questions such as:

- What are practice areas and quantifiable data (pitch, tempo, etc.) that we can gleam from MIDI recordings?
- What are the most effective ways of visualizing those practice areas?
- What are our options in analyzing MIDI files to visualize MIDI events in a useful manner for musicians?
- How useful is comparing MIDI recordings via velocity, sustain, and note frequency over time graphs
- Can we algorithmically generate useful automated feedback from analysis of these MIDI recordings and graphs

In an effort to pinpoint mistakes, we wanted to find the best ways to represent our musical data, so the user would see value in the tool.

- *Jeremy Grifski is a student at The Ohio State University. E-mail: grifski.1@osu.edu.*
- *Stephen Wu is a student at The Ohio State University. E-mail: wu.2719@osu.edu.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

2.2 Design Goals

At a high level, our goal was to construct a dashboard split into two panes: the file pane and the graph pane.

The file pane should contain a list of active MIDI files which are each given a color for encoding purposes. That means the dashboard should be able to support about 20 simultaneous MIDI files due to the limits of color perception. This should be more than enough considering the practicality of comparing that many recordings.

Each file in the file pane should be able to be selected for viewing purposes in the graph pane. When unselected, the file's background color should be neutral. When selected, the file's background color should mirror its color in the graph pane.

Meanwhile, the graph pane should contain several graphs:

- Notes versus Time (master graph)
- Notes versus Frequency
- Velocity versus Time
- Sustain versus Time

As a stretch goal, each graph should be connected with the master graph for filtering purposes. When a section of time is selected in the master graph, all other graphs should be updated to reflect the new subsection of data. This would allow a user to hone in on specific mistakes.

In addition, graphs should contain tooltips which will highlight areas with the highest amount of mistakes. These tooltips should include high level notes to assist the user in understanding the data.

Finally, the dashboard should be extended to include realtime recording and sheet music comparison.

2.3 Tasks and Metrics

In order to verify the success of the project, we tracked several tasks in GitHub:

- MIDI File Upload
- MIDI File Pane
- Notes versus Time Graph
- Notes versus Frequency Graph
- Velocity versus Time Graph
- Sustain versus Time Graph
- Mistake Analysis for Tooltips
- Realtime Recording
- Sheet Music Rendering and Comparison

Each of these tasks were broken down into smaller tasks as they all need to be designed, prototyped, and tested.

As for verifying that our design is good, we tested it on users of varying musical abilities. For less experienced individuals, we had them watch us interact with the tool through a demonstration where we collected feedback. For experienced musicians, we had them play a song to generate a MIDI file, then we asked them to indicate any mistakes they felt they made. Finally, we compared their personal insight to the tool.

2.4 Implementation

Ultimately, JustaMIDI ended up being a web-based tool built entirely as a static website using just JavaScript, CSS, and HTML.

2.4.1 Dashboard Overview

In terms of general design, we went with a dashboard approach using CSS grids. In other words, we divided the space into four columns by percentage of screen width from left to right: 9%, 43%, 43%, and 3%. Likewise, we divided the space into two rows by screen height from top to bottom: 60% and auto.

Each dashboard element was then assigned some range of grid points that they could occupy. For example, the MIDI selection interface was given the entire first column. Likewise, the pane selection interface was given the entire last column. Meanwhile, the note graph was given the center two columns of the top row while the remaining graphs split the bottom row of the same columns.

2.4.2 MIDI Selection Interface

As mentioned, the entire first column was dedicated to the MIDI selection interface. This interface consisted of a single MIDI upload button and any number of MIDI file items.

The upload button works just like any file upload interface. When a file is uploaded, a callback function is executed. In this case, we parse the midi file, and pass that data to the respective graphs panes to be rendered. At the same time, we generate a MIDI file list item which allows us to interact with the loaded file in various ways such as:

- Changing its name
- Toggling it on/off
- Deleting it
- Playing/pausing it

In addition, it is at this point that we assign the file a color for encoding purposes in each graph. To aid in encoding, we color the background of the file element, so it matches any data associated with the file in the various graph panes.

2.4.3 Notes Over Time Pane

In the top pane, we render the notes over time graph. Each note is encoded in three visual channels: color, position, size. The color gives us the track mapping, so it should match whatever color the file was assigned in the MIDI selection pane. Meanwhile, the position encodes two pieces of information: note and time. Finally, the size encodes note duration.

When looking at this graph, it is helpful to imagine a piano along the y-axis. Lower pitches are near the top of the graph while higher pitches are closer to the bottom of the graph. Meanwhile, the x-axis gives us the duration of each note from the time it begins to the time it ends. Time is given as generic time units which gain meaning given a tempo. Finally, each note has a tooltip which gives information like note, start time, duration, and velocity.

To generate these features, we use a combination of open-source libraries, but the primary tool is D3 which allows us to create graphs from Scalar Vector Graphics (SVG). In particular, the notes over time graph is generated by extracting the note information from all active MIDI files and plotting the results against the axes as rectangles with the appropriate color.

2.4.4 Note Frequency Pane

In the bottom left pane, we render the note frequency graph. Here, we filter the MIDI file data, so we end up with a collection of notes. Each note corresponds to a number which we dynamically map to their respective letters using a lookup table (LUT). Using this mapping, we're able to generate a histogram of notes.

Just like the notes over time plot, we also use D3 here to handle the bulk of the heavy lifting. At a high level, we use the note counts to plot rectangle against two axes: frequency and note. The frequency axis is scaled by the maximum note frequency over all active tracks. Meanwhile, the x-axis contains a direct mapping of all unique notes over all active tracks. We then sort that axis by the note frequency of the primary track.

2.5 Environments

To complete this project, we required the following software:

- JavaScript: a web-based programming language
- D3.js: a data visualization library
- MIDI.js: a MIDI processing library
- GarageBand: a MIDI editing tool
- GitHub: a version control and project management tool
- Travis CI: a continuous integration tool for testing

With this software, we were able to build and test the entire system.

3 Design Method

4 Results and Analysis

Evaluation was done in two phases, an initial prototype on March 7, 2019 and an updated prototype on April 6, 2019. Feedback was gathered from both phases to evaluate and improve the product.

For the updated prototype evaluation, we shared the tool with one of our advisors. Unfortunately, they hadn't gotten around to the tool in time due to vis paper deadlines. As a result, we decided to review the tool ourselves.

4.1 Prototype Design

The initial prototype aimed to accomplish several of the major tasks mentioned previously. In particular, the following tasks were implemented:

- MIDI File Upload
- MIDI File Pane
- Notes versus Time Graph
- Notes versus Frequency Graph
- Velocity versus Time Graph

In addition, we added a few extra interactivity features during our own iterative process:

- MIDI File Color Mapping
- MIDI File Playback
- MIDI File Renaming
- MIDI File Toggling

With the prototype read to go, we began the evaluation process by presenting it to the class.

4.2 Peer Feedback

In general, feedback was positive. However, there were a few things that needed improvement in the original prototype. The following list contains the feedback from our peers:

- Colors
  - Too closely related (light blue and dark blue)
  - Yet, calming
- Velocity plot
  - Difficult to see differences in curves
  - Plot may not be capturing appropriate data (summing velocities)
  - Look into normalization
- Target audience
  - Musicians don't care about time steps - they want beats and measures
- Interactivity
  - Playback should animate graphs

4.3 Musician Feedback

In addition to the in class feedback, we also chose to evaluate the JuxtaMIDI platform ourselves. In particular, Stephen leveraged his intermediate piano playing abilities to generate the following list of comments:

- Inherent limitations:
  - Online, generated MIDI files are often lacking when compared to an actual song:
    * Song may have a constant velocity for every note.
    * Key signatures, tempo, and sustain may be missing.
  - Recorded MIDI files have some of these issues and more:
    * Velocity is obtained, but this is not exactly reflective of playing on an acoustic piano, which is what would be typically used in performance. Keyboards, especially cheaper ones, are simply less expressive than acoustic pianos.
    * If the user is off-time by even a beat, the whole song gets offset. If the user corrects their time and becomes on-time, this issue is resolved. This means the user needs to be playing to a metronome and correct any time lost, which adds additional requirements on their playing.
  - Preprocessing:
    * Key signatures and tempo need to be manually added if needed for tool (it isn't in the current stage).
    * The MIDI file also needs to be edited to remove any wait in the beginning.
- Benefits:
  - Where this tool thrives is helping identify insights that can't be simply heard or seen in a recording.
  - Users can upload and play their recordings.
  - The velocity curve is probably the most novel item, since other tools (like GarageBand) provide the features of the Master Note graph.
    * Users can follow the sheet music and quickly see how they played different sections, e.g. if they played one forte section louder than another, or if they mistakenly played one forte section as piano.
- Potential:
  - This tool would greatly benefit from adding measure count. Since sheet music often comes with measure numbers, users could cross-reference their playing with the measure number.
  - Exploring and solving the tempo offset issue would make this tool much more useful.
  - Some additional outlier detection or analysis would also help.
  - Since musicians are used to sheet music, having a sheet music overlay with different colored notes per track would be immensely helpful.
    * This would require key signature, tempo, and a really good sheet music generator

4.4 Design Changes

To address some of the feedback, we decided to look into the following changes:

1. Adding a time marker to show where the user is in the song
2. Changing velocity plot to be more expressive
3. Reordering the color array so additional tracks have drastically different colors
4. Adding option for user to specify tempo, changing time to measure

For the final implementation, we completed #1 and #2 for the updated prototype.

4.5 Production Implementation

Ultimately, the final production dashboard was extended to include a marker to illustrate playback. The marker is colored based on which track was playing.

5 Discussion

6 Conclusion

It may seem odd to want to think of music in a visual way, but we feel our system will have a positive impact on musicians who want to improve their practice sessions.