ECE 1778 - Final Report

Wednesday, April 15, 2020

Group Name: Notate

Group Members: Kristen Antunes, Yilun Huang, Zihan Zhang

Word Count: 2458

1. Introduction

The rise of accessible music creation tools has introduced a new generation of do-it-yourself musicians. Today's bedroom producers find music notation less and less significant to their practices, with software and technologies that need not require users to read sheet music. While a formal understanding of music notation may not be necessary for personal songwriting, traditional Western notation is still used by the large majority of today's musicians, both classically and non-classically trained. As a result, non-reading musicians are faced with several barriers in order to collaborate with their reading counterparts. For example, a non-reading producer may wish to record a string orchestra for their upcoming release. How can the producer provide their reading performers with the notated sheet music they expect? *Notate* aims to resolve these issues, easing collaboration between reading and non-reading musicians in the transcription of simple monophonic melodies. With *Notate*, users can easily record melodic ideas with their device's built-in microphone, translating this information into a formal notated output to be shared with collaborators.

2. Statement of Functionality & Screenshots from App

Notate functions as a music transcription app, translating audio inputs into sheet music notation output. Users are first directed to the Menu page, with “Metronome”, “Lyrics”, “Record a Song”, and “Scores” page options.

a. Metronome Page

Using the sliding scale shown, users can select the tempo or speed of their song in beats per minute (BPM). This information is imperative to musical performance, and should be relayed to collaborating performers for performance accuracy. After selecting a tempo on the scale and pressing “Play”, users will hear an audible “click” sound in time with their predetermined BPM. While the metronome was originally intended to allow users to select their recording tempo, we determined that changing speed may disrupt recording clarity, with tempos that are either too fast or too slow for accurate pitch and rhythmic detection.

b. Record Page

The Record page features a “Start Record” button, beginning a countdown to record after selection. Users are then expected to sing or perform their melody in time with the screen's visual metronome (as indicated by increasing numbers). The Record page features a visual metronome, rather than an audible click metronome, so as not to interfere with pitch content detection during recording if users are recording without headphones. Once a user has completed their melody, they may press the “Stop Record” button and identified pitches will appear in letter/numerical form (i.e. “A4” for 440 Hz). Users may then select the “Sheet Music” button to take them to their notated output.

While the recording function works properly for vocal input, instrumental inputs may result in incorrect pitch identification. Instrumental timbres frequently highlight different frequencies within the harmonic spectrum, disrupting the information required for pitch detection and resulting in the identification of upper harmonics rather than the fundamental pitch.

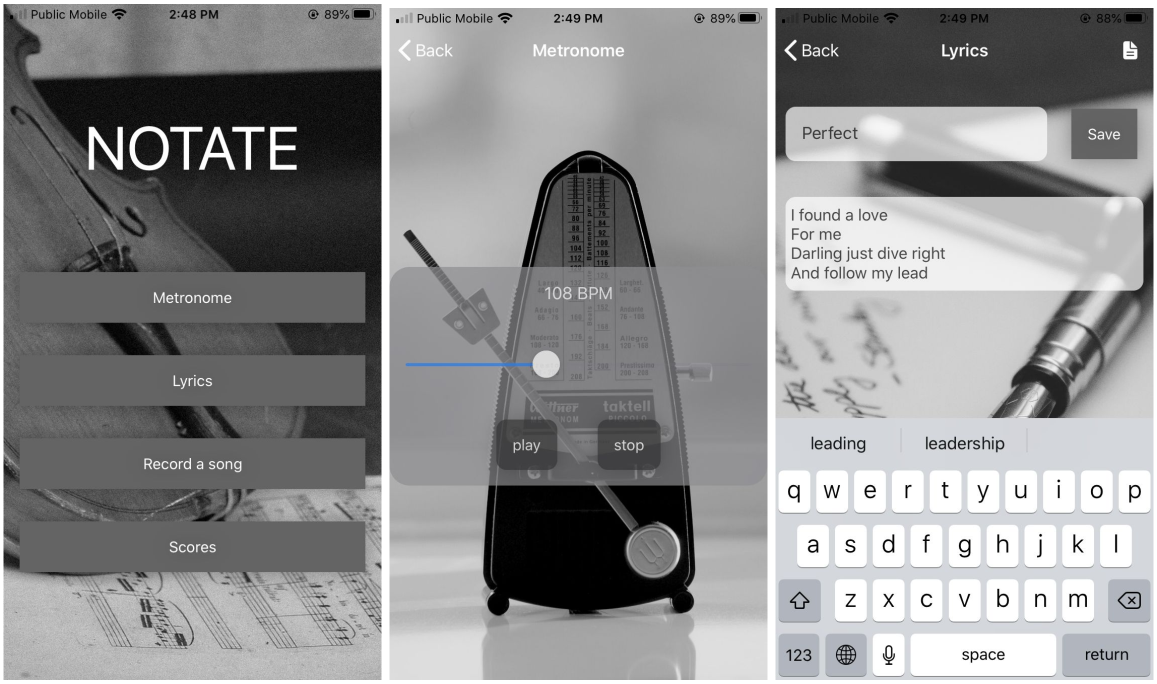

FIG 2.1 Main Menu, Metronome Page, and Lyrics Page

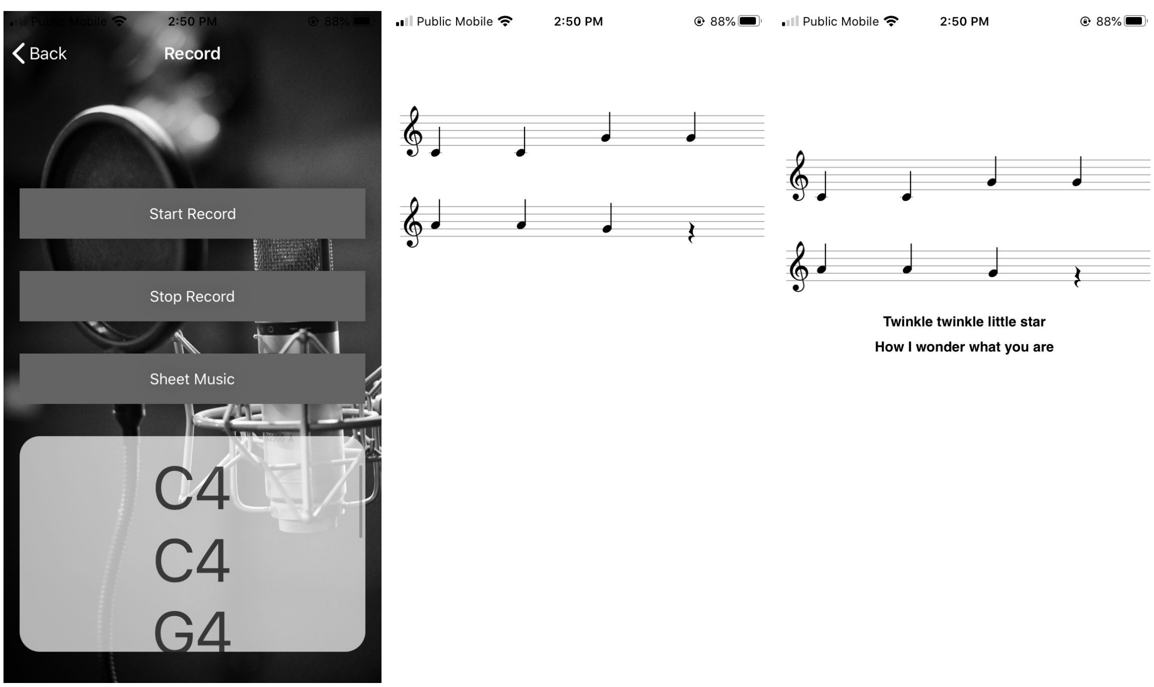

FIG 2.2 Record Page, Notation Output Page and Scores Page

c. Notation Output Page

Upon completing recording, users are directed to the Notation Output page, where their previously recorded melody is presented in standard Western notation form.

d. Lyrics Page

The Lyrics page first invites users to list a title in the top text box. Lyrics are saved locally so long as they have valid titles and include exclusively English characters. Operating independently of Recording functionality, users can write their lyrical ideas both before and after recording audio. After clicking the “Save” button, users can find all previously saved files by clicking the small page icon at the top right corner. From here, users can also select previously saved files to change or edit their lyrics.

e. Scores Page

Upon opening the Scores page, users are presented with a list of all previously saved scores by title, from which they can select to view their creations. Lyrics and Notations that use the same song title are saved together, so that corresponding lyrics appear below notations.

3. Overall Design

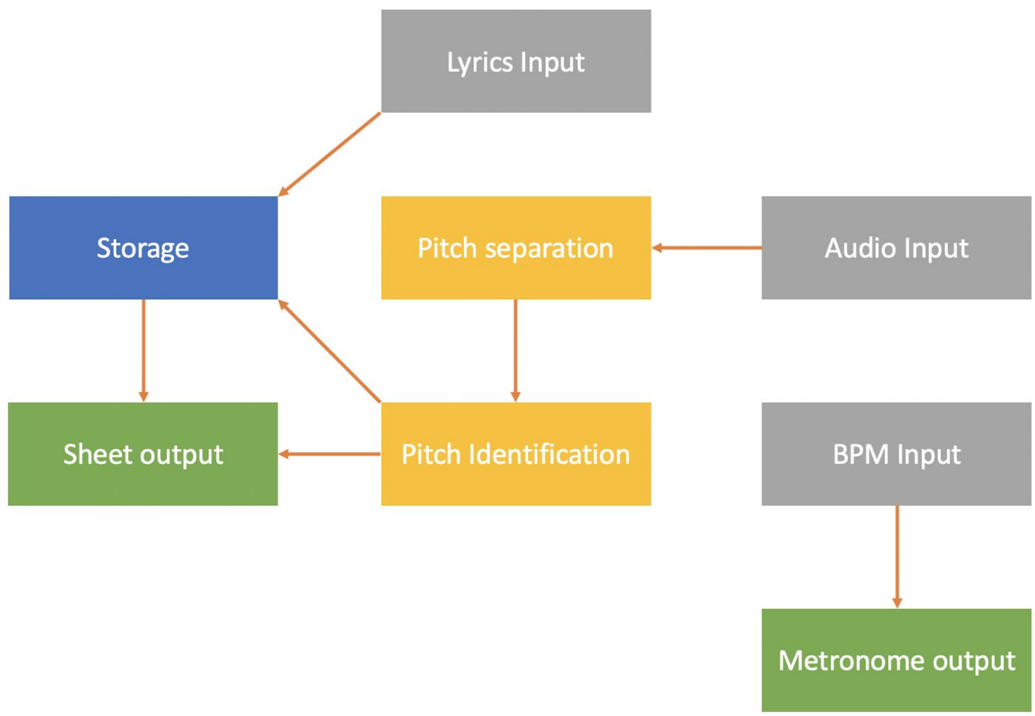

FIG 3.1 Block Diagram

a. Metronome

The metronome functionality is implemented using the Timer object provided by the swift Foundation framework. This object allows the program to trigger events repeatedly based on the time interval given. For example, if the metronome speed is adjusted to 60 BPM, we can set the time interval to 1 second and play the “tick” sound every time the timer fires.

b. Lyrics Input and Storage

Lyrics are saved in text files (.txt) under the “Document” directory in the app container. When a user wants to retrieve their lyrics, the content will be loaded into the Lyrics page for viewing and editing. The old lyrics file will be replaced if the user saves the new lyrics with the same file name.

c. Sheet Music and Scores Page

It became quite a challenge when we wanted to show traditional Western sheet music on an iOS device. Since we did not find any Swift libraries that help draw sheet music, this functionality was built with the help of a server we wrote on our own. The server is written in Python Flask and deployed using AWS Lambda, which lets us run code without provisioning or managing servers. With a JavaScript library named VexFlow included, the server is able to display custom sheet music and present it in the browser.

Once the user wants to see the sheet music, *Notate* sends a json object containing the information of the notes and the lyrics to the server. Then, the corresponding HTML content (sheet music with lyrics) will be presented on the mobile device.

d. Audio Analyzer

The audio analyzer takes an audio file as input and outputs the notation data. The process of analyzing audio is:

1. Split frames. Each frame lasts for 0.1s.
2. Compute noise confidence by DIS function for each frame. Then, delete noise frames.
3. For each note frame, transform signal to frequency domain by short-time Fourier transform (STFT).
4. Transform the signal from frequency domain to log-frequency domain to get candidate note frequencies.[1]
5. Using pitch-detection algorithm to pick the correct note for note[1] frames.
6. Combine note frames to get rhythms. E.g. quarter note/eighth note

---

[1] Gonzalez, S., & Brookes, M. (2011, September). A pitch estimation filter robust to high levels of noise (PEFAC). *2011 19th European Signal Processing Conference*, 451-455). Retrieved April 8, 2020, from https://ieeexplore.ieee.org/abstract/document/7073948

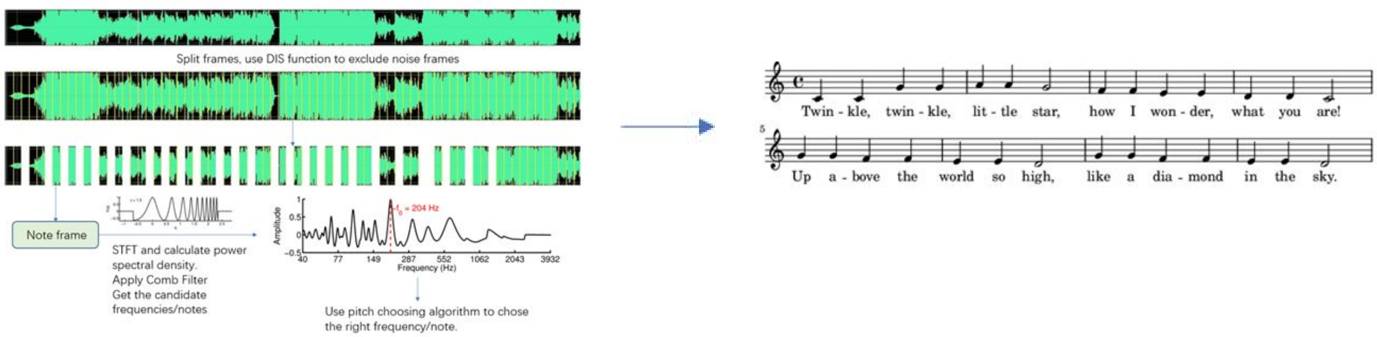

FIG.3.2: Audio analyzing procedure.

4. Reflection

While this course taught our group a significant amount regarding the technical requirements of programming for mobile devices, we also learned the importance of teamwork and prepared planning for such a large scale project. In designing a mobile app, we were forced to consider the way we interact with mobile devices, including the many built-in features that we use without conscious thought, such as the microphone or accelerometer. Recognizing the large variety of features included in mobile devices allowed us to design a better app, taking advantage of the phone's existing capabilities and exploiting them for our app's success. Furthermore, designing an app of this scale required the careful consideration of how users interact with mobile app screens for ease of usage. This includes simple details such as the transitions between app screens and the reactivity of in-app buttons, making the app not only pleasant to look at, but intuitive in use.

In the future, we would look to limit the scope of our app earlier on in development stages. We originally planned to reach advanced rhythmic detection, but came to realize that this task was too complex and time consuming to fit into the schedule of the course. Similarly, we had intended for the app's metronome to determine the speed of recording, but again found this too difficult to implement without constricting pitch and rhythm detection accuracy. Setting these limitations for ourselves from the beginning of our work would have saved us time, improving other functions rather than attempting to build beyond our means. Limiting our scope expectations also directly impacted our time management, as we were left with less time to accomodate for unexpected issues. In the future, we hope to better recognize the complexity of our app earlier on to manage our time and create a buffer for unpredictable problems.

5. Individual Contributions

Kristen Antunes: As a music student, I was able to contribute insights regarding notation literacy to inform the app's creation and judge its accuracy. This included acting as the group's primary test subject, as well as giving feedback on the appearance of the app's notation output. I also helped design and give feedback on the app's UI with my initial Marvel mockups. While the app no longer reflects the creative design of my mockups, these mockups helped inform elements of user experience, such as the progression of app screens from recording to notation. Towards the end of term, I lost access to an iOS device due to the COVID-19 crisis, and was therefore only able to provide feedback based on videos of the app. To assist my team in the completion of tasks, I wrote and designed the majority of our group's presentations and reports.

Yilun Huang: I designed the audio analyzer which transforms the audio input to music notes. To reduce the octave error, I did research and tested several kinds of detection algorithms, helping the app output accurate sheet music. I also made the audio recorder and the text version music note output UI view. This helped Zihan and myself to conduct app testing.

Zihan Zhang: As a programmer, my work focused more on the front-end of *Notate*. I designed UIs for the main menu, the Metronome, the Lyrics page, and the Scores page. I also helped improve user experience in the Record page by helping with the UI design. Furthermore, I was responsible for implementing the functionalities of reading and saving lyrics, playing the metronome, and displaying sheet music. As I have some background in music, I was involved in testing the audio analyzer with my own voice in the later part of the course.

6. Specialist Context

I believe *Notate* holds massive implications for my field of work. The music production industry is filled with talented musicians who often lack the skills or understanding of formal Western music notation. The rise of non-reading musicians has only grown with the increased accessibility of music creation softwares such as Garageband. These softwares often come built into desktop computers, encouraging users to create without the requirement of preexisting musical knowledge. While this lack of knowledge does not inhibit their creativity when working independently, this greatly complicates collaboration when attempting to work with musicians that expect a certain level of music notation literacy. *Notate* could resolve these complexities, serving as an intermediary tool between musicians of different backgrounds to ease the communication ideas and ultimately, music collaboration.

Despite not yet working in the music industry, I see these problems almost every day in my classes, and have even been hired by colleagues to transcribe musical ideas into sheet music notation. An app like *Notate* could not only simplify songwriting between reading and non-reading musicians, but could reduce the need for paid transcription services, saving producers potentially large financial burdens. However, it is important to note that *Notate*'s limited capabilities in monophonic-only notation would not completely render transcription services obsolete. Transcribers, arrangers, and composers will always be necessary in the music industry, particularly when conducting far more complex musical tasks. Yet, this app would certainly lessen the need for notation services in such comparatively small instances as pop song writing, where melodies are often simple enough to be transcribed by the *Notate* app.

Notate would allow non-reading musicians the ease of musical literacy without having to develop a formal understanding, substituting for years of music theory training. Contrastingly, *Notate* could also function as a music theory tool in encouraging users to draw parallels between their auditory musical creations and *Notate*'s visual score outputs. If we were to continue developing the app, *Notate* could function not only as a transcription service, but as an educational tool to teach basic notation concepts to non-reading musicians. Recording a personal melody of one's choice would only enhance the educational experience, as users may be familiar enough with the intended sound of the notation to draw parallels to its visual representation. While I understand that these extended ideas may require significant work to come to fruition, they also serve to comment on *Notate*'s extremely large potential impact on the music industry. Even in its current iteration, *Notate* could help non-reading musicians in understanding the components of music theory notation, such as tempo through metronome usage, pitch, and rhythmic content. While this may not help non-reading musicians develop a formal understanding of music theory, *Notate*'s transcription capabilities in combination with this large potential clearly demonstrate the app's potential for success in today's music industry.

7. Future Work

Notate holds great potential for expansion and improvement for future iterations. First, we would work to improve *Notate*'s pitch accuracy for non-vocal timbres. While the app works extremely well in detecting pitch from voices, instrumental timbres may highlight different frequencies within the harmonic spectrum, disrupting the harmonic frequency information used to help determine pitch. We would like to solidify this pitch content identification as a foundation to improve future iterations of the app.

We would also work to expand upon *Notate*'s rhythmic identification capabilities, with particular regard to advanced and varied rhythmic notation. Currently, *Notate* is only capable of identifying fixed duration rhythms occurring exactly on the metronome beat, where each note in sequence is of the exact same duration. We would like to extend this capability to include varied rhythms, identifying notes of different durations in sequence. For example, a quarter note occurs directly in time with the metronome pulse, while an eighth note would occur directly between metronome clicks. A half note would be held for twice as long as a quarter note, over the duration of two metronome beats. Greater rhythmic variety would expand *Notate*'s capabilities to serve a wider range of musical ideas.

Furthermore, future work should look towards the transcription of polyphonic information. Currently, *Notate* is only able to identify monophonic melodies, defined as a single pitch at a time. Many non-reading musicians play polyphonic instruments such as piano or guitar, where multiple notes are played at once. If *Notate* were able to accurately transcribe this polyphonic information, the app would serve a greater purpose for non-singing musicians as well, being able to notate chord information in addition to exclusively staff notation (pitches and rhythms on a series of lines).

Finally, future iterations of the app may look towards including a means to export notated scores into standard file formats such as .pdf or .jpg. Export functionality would be consistent with *Notate*'s overarching goals, easing the collaboration between reading and non-reading musicians by allowing users the option to easily share their notated scores and lyrics.

8. Statement for Public Posting:

The *Notate* team agrees to the public posting of our final presentation video, source code, and report.

Bibliography

Gonzalez, S., & Brookes, M. (2011, September). A pitch estimation filter robust to high levels of noise (PEFAC). *2011 19th European Signal Processing Conference*, 451-455). Retrieved April 8, 2020, from https://ieeexplore.ieee.org/abstract/document/7073948