SoundSync

Authors: Caleb Lille, Rohan Raavi, Sanjana Shriram
Affiliation: Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*— In music performances, distractions often disrupt the flow of musical expression. One prominent challenge is page flipping. Currently, solutions include foot pedal page turners and human assistants. However, performances can be disrupted by foot pedal devices, while human page turners can obstruct the musician's view and cause physical sheet music to fall. SoundSync uses an eye tracking camera and a microphone to capture user's gaze and audio input to autonomously turn a page. Through a decision logic program, these inputs determine the best window for turning pages with 95% accuracy.**

***Index Terms*— Audio Alignment, Score Following, Eye Tracking, Gaze Tracking, Music, Digital Music, Sheet Music.**

1 INTRODUCTION

Musicians face a variety of distractions that can deduct from the beauty of the music they are making. A common problem musicians face is turning a page in a manner that doesn't detract from the music. With the advent of technology and digital music displays such as tablets, page turning technology is changing.

Page turning technologies have been on the rise in the music industry. The most widely available options include a foot pedal page turner and a physical human page turner. A foot pedal page turner is a device where the musician presses the foot sensor to turn a digital page of music. It uses a Bluetooth connection to a digital tablet and can be a convenient page turning solution for musicians.

However, current technologies have some problems. For example, the foot pedal page turner requires a loud foot tap to signal a page turn. This can be distracting during a performance. In some cases, the foot pedal does not work and can require a second tap. A human page turner can obstruct the player's view and introduces the risk of dropping physical sheet music during a performance.

SoundSync is a digital page turning system that utilizes visual and audio inputs to determine when to flip the page. The system takes in eye position and audio through an eye tracking camera and microphone. Both input streams are fed into different models that track where the user is located in the music. These results will be fed into a decision logic program that will decide when to flip the page. Digital sheet music will be displayed on a Windows laptop. Lastly, SoundSync will be using a Google Board to handle all the data processing. With a focus on accessibility and inclusivity, SoundSync aims to provide an inclusive streamlined music making experience for all musicians.

2 USE-CASE REQUIREMENTS

SoundSync was designed with accessibility as a top priority. Music is present in every community, and the goal of this project is to make music more accessible to all. The social implication of SoundSync is that more people of varying backgrounds can play music without the fear of missing a page turn. The resources that most people have access to have also been considered when designing SoundSync. As of 2019, 73% of adults in the United States owned laptops or personal computers[6]. With accessibility in mind, laptops come out ahead of alternatives like iPads and tablets.

Because SoundSync exclusively uses audio and visual inputs, it is accessible to those who cannot operate a foot pedal page turning device or similar technologies. Since it's fully digital, SoundSync is less disruptive than traditional methods such as loud foot pedals and physical page turning all while guaranteeing that the musical ambiance remains undisturbed.

2.1 Sensor Input Use Case

The system must be able to take in audio and visual inputs. The visual input comprises of tracking the user's eye gaze on the screen. The eye tracking and head tracking model must identify patterns where the user wants to turn the page by staring at the end of the page and/or head gestures.

The audio model takes in the user's audio input to track the user's current position in the music, while also being aligned with the pregenerated MIDI File. Users should be able to play correctly 90% of the time. The system must be robust enough to handle occasionally wrong notes and wrong tempo.

Both sensors should have a failure rate of less than 1% where a failure is defined as the system not detecting a given input. These requirements stem from the inclusive design of our system. A user's hands are occupied while making music and foot pedals may be inaccessible. Therefore, the system will rely on visual and audio inputs exclusively.

2.2 Frontend Use Case

The system will indicate to the user where they are in the score with a cursor that can be toggled. The display will be easy and intuitive to use, with buttons to upload PDFs and MIDI files. The system also displays page turns with a quick but useful animation of a page turning, and turns the page accurately. During calibration, there must be a running video feed of the user so they are aware if

they're out of frame. The cursor must toggle when pressed 100% of the time.

Page flipping is the most important requirement; the page flip success rate must be at least 95% to justify using this technology over existing solutions.

This requirement is rooted in the principle of accessibility for social impact. The frontend is what a user directly interacts with, so it should be as easy to use as possible to encompass a wide range of people.

2.3 Hardware Use Case

The hardware was designed with quality of life in mind. Moreover, SoundSync will have intuitive operating features, such as a start button, and an override page turn button. In terms of power, the system must be powered by a battery pack to make the system portable and compact, ensuring a non-distracting user experience. Additionally, the system must be operable for a maximum rehearsal session of 4 hours. The hardware components excluding the display should weigh no more than 5kg.

This requirement was established to improve portability. With a focus on intuitive controls and simple hardware setup, the user experience will seamlessly integrate with the music making process.

By prioritizing these features, SoundSync aims to create a user friendly, adaptable, and seamless platform for musicians while promoting accessibility.

3 ARCHITECTURE AND PRINCIPLE OF OPERATION

SoundSync's physical structure consists of a Tobii Eye Tracker 5 camera mounted to a personal digital display being used to read digital music. A microphone will sit near the user's instrument to record sound. The Google Board will be mounted near the base of the stand, and the battery pack will be on the bottom of the stand. The battery pack powers the Google Board, and the board is connected to both the microphone and the laptop. The eye tracking camera will be connected to the laptop through USB-C.

On a high level, the Google Board will be connected to the Tobii Eye Tracker 5, microphone, and display. The system incorporates two models: a visual eye tracking model, and an audio alignment model. The visual model will take in the filtered eye tracking data and determine from a solely visual perspective, whether or not to turn the page. The audio alignment will take the processed audio signal and try to align the stream with an uploaded MIDI file. The models running on the Google Board will capture, detect, and classify the input streams. These actions will then be sent to the frontend, where the page will either be flipped or not.

Refer to Figure 1 for a block diagram of the system architecture. Figures 2 and 3 explicate the frontend and backend subsystems.

4 DESIGN REQUIREMENTS

4.1 Eye Tracking Requirements

Our specifications revolve around keeping the eye tracking component accurate and precise. To address the visual sensor input use case requirement, the system's margin of error is low and allows the system to outperform current solutions.

The first requirement is that the eye tracking field of view must be 14 cm x 20.5 cm because this is the minimum field of view necessary to register a human face. Another requirement is keeping the eye tracking accurate and precise to one bar. Therefore, a 15" display requires an accuracy of 4.0 cm to ensure we are within a single bar. A precision of 1.5 cm, which corresponds to the height of a bar, keeps the eye tracking within the correct line. These two requirements warrant that the system accurately and precisely follows the player's gaze. On top of this, eye tracking filters will be used to increase the precision from the original distribution of points provided by the Tobii Eye Tracker 5. These filters will include saccade detection, which aims to accurately adjust for the small jitters in the eye when a user stares at one single point. These jitters create a larger distribution of where the camera thinks the user's eye is looking. This filter aims to dial the accuracy and precision back to the 4.0 cm and 1.5 cm mentioned earlier. Other eye tracking filters include outlier detection to filter out points that are unlikely to be where the user is looking. This technique can also be applied in situations when the user glances away from the sheet music.

4.2 Audio Requirements

In order to address use case requirements for sensor inputs and frontend, the backend must ensure accurate page turning. Specifically, we're looking at the time distance between where the music is and where the model thinks it is. The accuracy requirement is up to 1 beat in either direction, forward or backward. This requirement ensures that our audio accuracy is close enough to avoid most audio alignment and syncing issues.

To address the frontend use case requirement, segmentation should be deployed to reduce latency. Segmentation involves dividing an audio sequence into smaller pieces in order to process them individually. This technique is essential to guarantee real time page flipping.

To address the sensor input use case, the audio model must be robust enough to operate accurately despite 1 wrong beat per sequence of 10 beats. The user must perform 90% of the notes accurately within a segment of music.

Another requirement designed to address the frontend use case requirement is frequency filtering and SNR. The audio component should ensure that frequencies being picked up are within the plausible frequency range of the user's instrument and that the SNR is greater than 25dB.

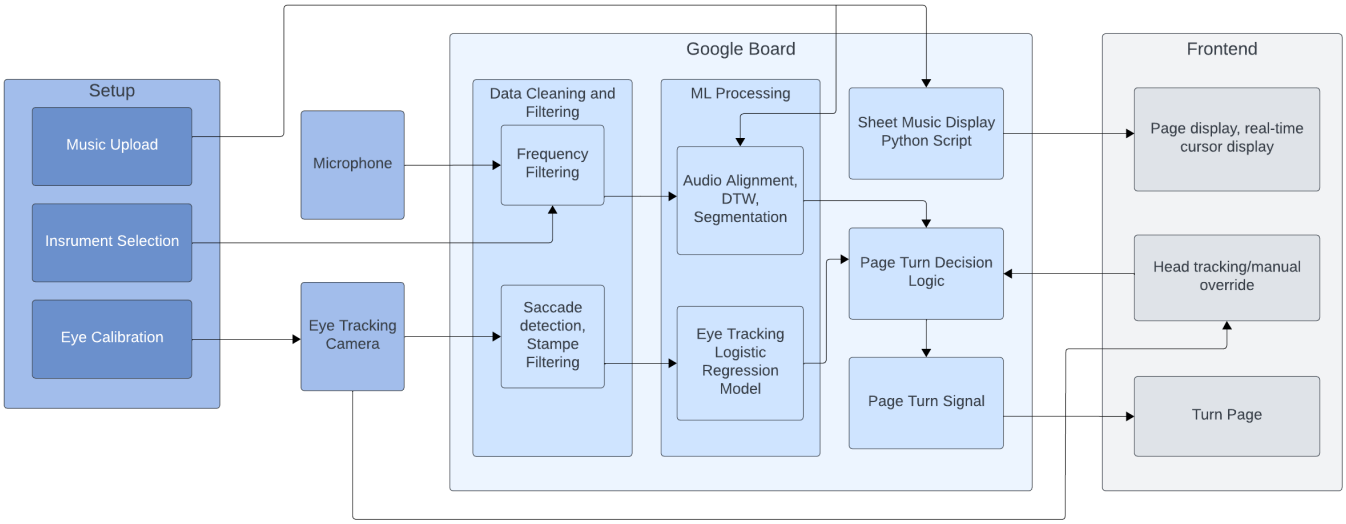

Figure 1: Block diagram of frontend, backend, and hardware components interacting. The setup stage shows what happens before the user can begin playing music. The Google Board section details all the processing occurring with data collected from the microphone and camera peripherals. Finally, the frontend demonstrates key features of the completed application and how it responds to real time used inputs.

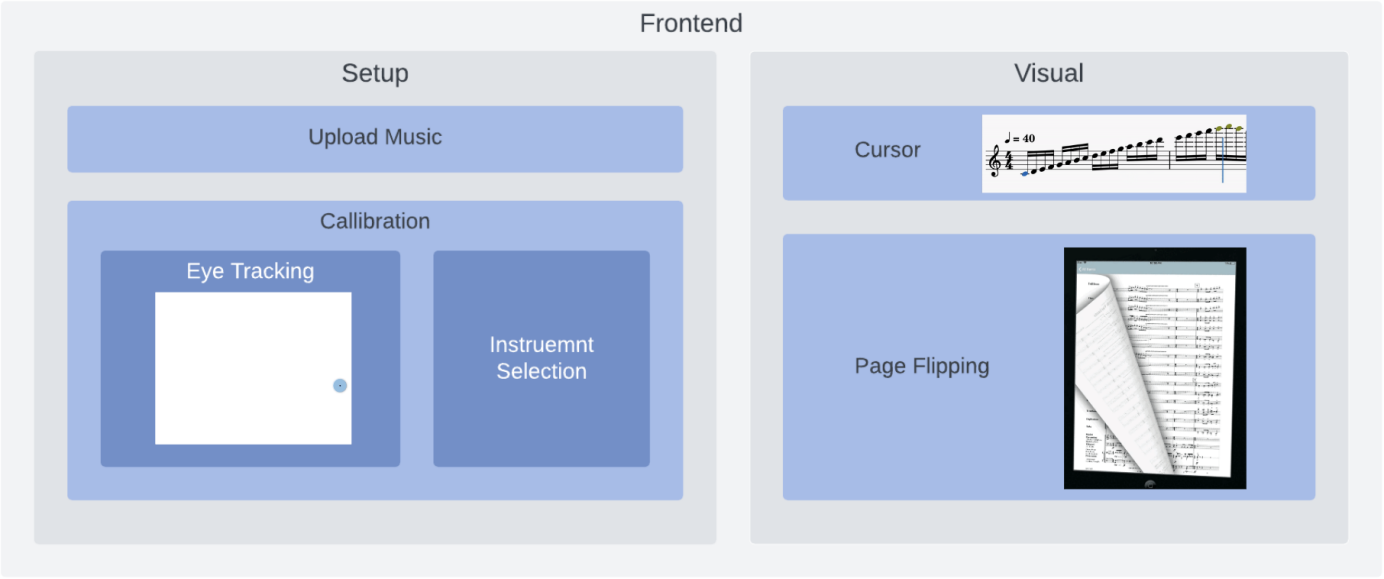

Figure 2: Frontend subsystem diagram. The frontend is structured for ease of use. At the beginning, the user is directed to an "Upload Music" screen where they will upload both their sheet music and a MIDI file of the music. After that, the user will begin calibrating their eyes. By focusing on each corner of a page of sheet music, the camera will map the location of the eyes to a location on the screen. The user will also select their instrument which will trigger frequency filtering in the backend. Once these setup steps are complete, sheet music will be displayed, and the user can begin playing music. There will be a moving real time cursor and short page flip animations to simulate turning a real page while giving the user time to adjust to the new sheet. Parts of these figures are adapted from [4] and Tobii.

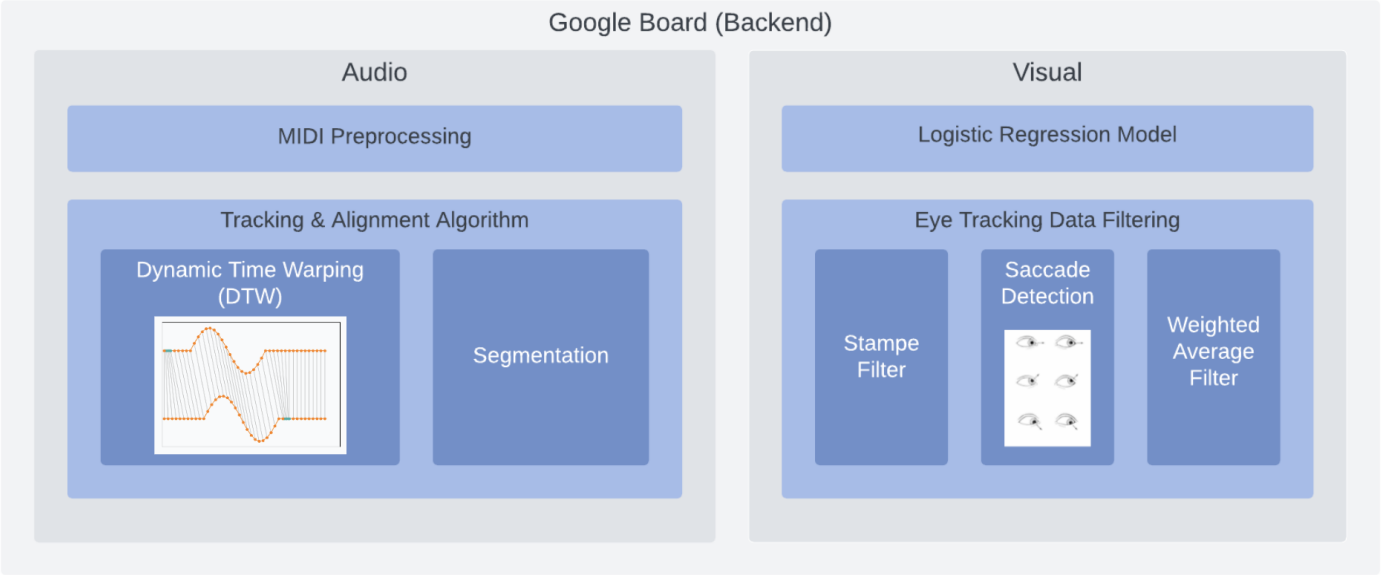

Figure 3: Backend subsystem diagram. A Google Board runs both audio and visual algorithms. On the audio end, a preuploaded MIDI file is segmented into finite time sequences and is examined to extract information on tempo throughout the piece. The segmented parts of the MIDI file are time warped with the live audio to align the user's playing. On the Visual end, data from an eye tracking camera is filtered and run through a logistic regression model to determine whether to flip the page. Parts of these images are reproduced from Vision Therapist Resources and [7].

4.3 Hardware Requirements

All hardware used for the design must improve the quality of life for the user. First, the battery life should last for the entirety of a long practice session; four hours should account for most cases. Therefore, the power budget will be 8 Watt-Hours and the Google Board satisfies this because it has a maximum power consumption of 2 Watts[3].

The size of the SD card must be large enough to account for the models for both eye tracking and audio alignment, uploaded MIDI files, and PDFs of the sheet music. Therefore, we chose a 256GB SD card to account for any additional storage we may not have foreseen.

Furthermore, the whole system must be portable and consist of small components: a Google Board, lapel microphone, and power bank can fit in the palm of a hand. The last hardware requirement is the inclusion of a backup override button. This is necessary in failure cases to ensure functionality of the system and achieve the 95% page turn success rate.

4.4 Low latency

Processing delay for both these models must be short and accelerated to avoid unnecessary lags within the system. SoundSync's processing latency should be no longer than 500 ms. 500 ms refers to a quarter note at 120 BPM, which will be the fastest tempo that our system will work with. Additionally, the combined model sizes should not exceed 256GB. These design requirements address the frontend use case requirement of having a high speed processing system.

5 DESIGN TRADE STUDIES

5.1 Jetson vs Google Board

The Google Board is a board designed for onboard machine learning, therefore, the laptop will not be running any computation and only serving as a display. Therefore, having the processor run ML inferences quickly and efficiently is a requirement. Importantly, the board needs to run for a sufficiently long time using low power as this system is powered by a single precharged battery pack.

The competing board to the Google Board is the Nvidia Jetson board. This comes in two varieties: the Jetson Orin Nano and Jetson Nano. The Jetson Nano reports an AI performance of 0.5 TOPS at a reported wattage of 5-10W[5]. However, the Google Board runs 0.5 TOPS at 0.25W[3]. This eliminates the Jetson Nano as a viable option. The Jetson Orin Nano, on the other hand, runs 20TOPS at 7-10W[5]. Although this board surpasses the Google Board's max performance of 4TOPS, the Google board's compensates by running at 2W[3]. This means although the Jetson Orin Nano would handle the computation power, the high power requirement of the board would cause the session duration to decrease significantly and violate the battery life use case requirement. Another aspect is the design of the Google Board compared to the Jetson boards. The Google Board is designed for machine learning and can compute inferences up to an order of magnitude faster than traditional CPUs which are found on the Jetson Boards. Having this fast computation speed at low power perfectly fits our design requirements of low power and low latency.

5.2 Camera

The Tobii Eye Tracker 5 camera is the foundation of retrieving high quality eye tracking data. This camera, designed for eye tracking, can retrieve pupil data within 0.5 degrees compared to 2-5 degrees for a conventional webcam[2]. The camera also has built-in head tracking which when used with the existing model can help account for non-linearity in the music by giving the user a manual override method. The FaceTrackNoIR API will be used in conjunction with the camera to improve head tracking. Although conventional webcams can be programmed to have high quality head tracking, adding an additional peripheral would result in a less compact design. Furthermore, Microsoft tested the accuracy and precision of the predecessor to the Tobii Eye Tracker 5 under different lighting environments. Because these cameras are similar, the baseline data extracted from Microsoft is likely comparable to the Tobii Eye Tracker 5 camera under artificial and natural lighting conditions. Therefore, we have confidence in the accuracy and precision of this camera to meet our design requirements.

5.3 Microphone

A lapel microphone is vital for staying in line with the compactness use case. A lapel microphone is a microphone used in TV shows and documentaries that is clipped onto the presenter. This microphone works very well at picking up sounds at a short distance and not very well at picking up sounds from a farther distance. Although boom microphones and podcast microphones can lead to slightly clear signals, the size and/or price of these microphones make them incompatible with our project. The lapel microphone also has the ability to be clipped onto either the user's collar to pick up breaths or on the stand to get a clearer sound of the instrument. This versatility allows the design to alter slightly based on the microphone placement.

5.4 Frontend Choice

There were a couple options for the display: making a React webpage, having a Python program run locally to display the application, or coding an iPad app.

Developing an iPad app proved to be difficult as nobody had experience with Swift or app development experience. Hosting a React webpage proved challenging in the real time aspect of it. Real time data processing would require optimizing the app's performance to handle continuous streams of data. These problems could pose to be especially challenging due to limited processing power and memory. Integration with an iOS app also requires adhering to iOS SDK guidelines and managing resources to prevent the app from slowing down and failing to be real time. The 15" screen was variable as well across different tablets. For a digital music display, a standardized minimum screen size is necessary to ensure readability is maintained.

While our frontend developer has experience with React web applications, several challenges prevented us from ultimately choosing React as our frontend framework. Our priority is the user experience, which we want to be in real time. Because React is a client-side framework, it may be difficult to handle the continuous stream of data from the peripherals. Furthermore, integrating specialized hardware like the Tobii Eye Tracker 5 and Google Board would require careful handling of low level interactions and device specific protocols. Overall, integration and compatibility is non-trivial.

Locally hosting a frontend in Python was the most reliable way to guarantee real time communication and easy integration with peripherals and hardware. This helps remove a connectivity step to the Google Board which helps ease of implementation. Python also has an extended set of libraries that are compatible with MIDI files, the Tobii Eye Tracker 5, and audio in general.

5.5 Displays

The current design for the system has a 15" laptop screen to act as the display placed on top of a musical stand. Although laptops are more bulky than a tablet, 73% of Americans own a laptop which increases accessibility[6]. Furthermore, using a laptop as a display is much simpler as an iPad would require a developed app, while the laptop can simply display the camera feed given by the Google Board. Furthermore, because pro models of Tobii Eye Trackers are outside of our budget, we will be developing for the more accessible Tobii Eye Tracker 5. This camera requires the developing through the Steam Engine API on Windows OS. It is important to note that the laptop will not be computing, but rather the laptop is just running software that extracts data points from the camera and sends it to the Google Board, thereby acting as a display.

5.6 Override Conditions

Override conditions may be necessary to satisfy accurate page turning. With override conditions, users can communicate directly with the system and manually turn a page. If a user uses override conditions they can mitigate any risk associated with using the system's autonomous components. In case of failure, there is still a way to mitigate the risk of not having the page turn at all. This approach provides the same functionality of existing apps today, and therefore isn't viewed as a significant negative.

5.7 Rehearsal vs Performance vs Practice

The scope of whether the user is performing or practicing drastically changes the design of the system. Designing for performance requires the system to have extremely robust page turning with the ability to distinguish and separate polyphonic music. This is because a wrong or inappropriately timed page turn may cause the user to lose

focus and can lead to mistakes that jeopardize the quality of the performance. Musicians are also playing in environments with several other musicians, often in similar frequency ranges as their own instrument. For the scope of our project, we will be designing the system for private rehearsal. This allows for a lower successful page turning rate as the stakes are not as high. However, because private rehearsals have the user starting and stopping frequently, the system will need to compensate by being able to have the user able to set where they would like to reset to. This feature will be implemented in the frontend as a drag-and-drop feature. Because the size of each bar and their location on the page is standardized, this feature will easily track where the user would like to set this reset marker.

6 SYSTEM IMPLEMENTATION

6.1 Audio Alignment

Audio alignment is grounded in Dynamic Time Warping. DTW is an algorithm aimed at minimizing the Euclidean distance between two finite time sequences. This finite sequence assures the boundary conditions requirement for DTW is achieved. The monotonicity requirement is achieved because the music never needs to rewind so the time sequences always progress forward in time. The continuity condition is satisfied as we will not be skipping notes within the time sequences. The warping window condition refers to making sure the frequencies fall within a certain range which will be calculated based on the user's instrument. Once all of these requirements are satisfied, the algorithm will return the offset of the two time sequences in milliseconds.

6.2 Eye Tracking & Head Tracking

The eye tracking and head tracking will have a hardware component and a machine learning component. The hardware component is the Tobii Eye Tracker 5. This camera is designed for eye tracking and head tracking. It gives great starting precision data to work with and apply various data filters on. The machine learning component takes in the data and runs it through a logistic regression model. At the beginning stage of the system, there will be a calibration step. In this stage, the user must follow a moving tracker dot, which sets up the Tobii Eye Tracker 5.

6.3 Frontend

The sheet music and moving cursor will be displayed on an interactive webpage. This webpage will not be deployed, but rather locally hosted on a laptop. The webpage will be written via a Python script with the use of python libraries, such as TKinter. The webpage will have 4 pages: the start page, the calibration page, the instrument selection page, and the sheet music display page. The start page will simply display a brief description of SoundSync and include a Start Button. This button will lead to the calibration page. Here, eye tracking calibration occurs. The instrument selection page, displays instrument options for the user to choose, which begins the frequency filtering in the backend. The instrument selection also asks the user to upload a MIDI file of the music that they are playing, preferably generated from MuseScore. The last page displays the sheet music, the togglable cursor, and the page turns.

7 TEST & VALIDATION

7.1 Audio Tests

The first audio test is a signal integrity test that will be performed using a pure 440Hz audio input. Instruments such as violins have audio inputs at harmonics, therefore a non harmonic pure tone will be fed into the microphone for this test. Once filtered and processed, the SNR of the signal picked up by the microphone must be above 25 dB.

The second test is page flipping at multiple tempos and varying musical structures. Custom composed pieces will encompass a range of musical beats, structures, and notes. The beats will cover note lengths ranging from an eighth note to a whole note. Structures will include repeats within and across pages. Finally, notes will live within the range of G3 - E6. Not all notes will be tested, however, the highest and lowest notes will be. The following tempos will be tested: 60BPM, 90BPM, and 120BPM. This range accommodates most beginner repertoire. For each of these variations, the page must flip within the last measured bar of a page.

The third test is checking the scaling of the time complexity through various segmentation sizes of the audio. In DTW, an audio stream is divided into small time segments. The audio alignment model latency will be measured across different segmentation sizes, such as length of 1 beat, length of a 1 quarter note, 1 eighth note, and 1 sixteenth note. These tests will output the optimal segmentation size for the audio alignment model.

Another test that can be run is the system's robustness as the environment changes. For example, the system should operate near ideally in a closed room with padding on the walls to absorb excess noise. However, we want to make sure our system also works in a room with open windows or ajar doors. The system under conditions where noise is voluntarily added by the user should still meet all use case requirements. Therefore, we will be testing the system in various locations to ensure different types of practice locations do not affect the performance of the system.

For unexpected user edge cases such as stopping prematurely or repeating sections randomly, the system is expected to operate normally without any unexpected behaviors.

7.2 Eye Tracking Tests

Eye tracking tests focus on verifying that the distribution of data points obtained by the Tobii Eye Tracker 5 stays within one bar. The predicted size of this bar is 1.5 x 4.0 cm for a 15" display. If the data point distribution does not meet this requirement, the size of the bar will need to be adjusted. To test this, filtered camera data will be plotted and the distribution of data points will be measured against the size of the bar. Other metrics to test the robustness of eye tracking involve repeating the same test with 3 different tempo markings at 60BPM, 90BPM, and 120BPM. We anticipate that accuracy and precision will change as a function of the tempo since users must scan the page faster to continue playing at tempo. Users will also sightread at least two different pieces of varying difficulty. The changes in the distributions of points will be recorded and measured to ensure that they remain within the dimensions of the bar.

7.3 Integration Tests

Integration testing aims to measure and verify the improvement that eye tracking adds to the existing system.

User testing will consist of two violinists. They will be using the system twice: once with just audio and once with both audio and eye tracking. Users won't be told which system they are using, and will be asked to rate the timing of the page turns on a scale of 0-10. These metrics will be used to understand the improvement of eye tracking to the user experience.

The full system latency will be evaluated and compared with the audio only system latency. These results will be graphed and mathematically inspected to quantify improvement.

7.4 Power Tests

We intend to test the power of each component individually by noting the decrease in power of the battery in 20 minutes. Components such as the microphone do not come with a datasheet where power is listed and hence will be manually tested for power consumption. The component predicted to take the most power is the Google Board. This board's power consumption is a function of the amount of operations per second. Therefore, testing of the Google Board's power will be done where the Google Board is completing 0.5 up to 2 TOPS. We will then run a power test of the whole system and make sure the total power is approximately the summation of each individual component.

7.5 User and Frontend Tests

Users will test the system by playing multi-paged repertoire. Their feedback will be on a scale of 0-10 to quantify how accurately they believe the system flipped the page. The goal is to achieve an average of 9.5 out of 10 across 10 trials with at least two different users.

An important test to run is the speed of the page turning animation. This will importantly be an animation, because a sudden jump from page to page may disorient the user. Therefore, we want to find the ideal page flip animation speed that both feels natural to the user and quickly displays the next page. This may be a function of tempo as during a faster piece, the user may want a faster animation speed. We will test the ideal animation speed for 60, 90, and 120 bpm. Then, for other tempos, the animation speed will be extrapolated from these three data points.

Additionally, the override controls will be thoroughly tested. The system has two page turning override features: a head gesture and an external button. If the digital page turn fails, the user can turn their head to the right or left to turn the page right or left, respectively. If this override fails, the user can press an external button, which will flip the page. Users will test the override conditions and give feedback on a scale of 0-10 to quantify how satisfied they are with this feature. These override controls must be tested to ensure that they are reliable.

8 PROJECT MANAGEMENT

8.1 Schedule

Refer to Figure 4 in the Appendix.

8.2 Team Member Responsibilities

The components for this project fall under 3 categories: frontend, visual, and audio. The audio tasks include testing the best placement for the microphone, ensuring the microphone signal is clear and has little to no noise, and processing the signal for DTW. The visual tasks consist of writing the code that interfaces with the Tobii Eye Tracker 5, filtering the data points to improve precision, and compiling the data into a relevant vector for the logistic regression model. The frontend tasks comprise of building a user interface that shows the eye tracking calibration, allows the user to select their respective instrument, and displays the sheet music with a cursor that follows exactly where the user is playing. The tasks will be primarily divided with Sanjana tackling the frontend and eye tracking, Rohan working on the hardware and integration with decision logic algorithms, and Caleb solving audio alignment problems. This is tentative and plans to shift these responsibilities temporarily in preparation for big exams are in place.

8.3 Bill of Materials and Budget

Refer to Table 1 in the Appendix.

8.4 Risk Mitigation Plans

The major risks in this project involve the integration of the eye tracking model into the self-sufficient audio alignment system. Eye tracking data is notoriously noisy and

challenging to work with. Pattern recognition and rapid or darting eye movements could pose a significant risk to our algorithm. Musicians often look away from their music entirely to focus on conductor cues, therefore making it more difficult to understand and locate where the eyes are on a page. To mitigate these inconsistencies present in eye tracking, the system should function well exclusively using audio for score following.

9 RELATED WORK

Systems that follow a player and flip the page automatically were first built by Dr. Roger Dannenberg at Carnegie Mellon University. These systems then went on to become a system known today as SmartMusic. Other people have built similar systems such as Andreas Arzt who showed that DTW could be run on segmented sequences of the MIDI file and live audio for real time audio processing and score following[2].

SoundSync differentiates itself from existing technologies by studying how eye tracking can be combined with audio alignment to provide extended functionality for users. Our system will also perform audio processing in real time, whereas many existing technologies perform post processing on completed audio streams to determine where a user is located in the music.

10 SUMMARY

Soundsync will be taking in visual and audio inputs to determine when to digitally flip a sheet of music. The system will be using the Tobii Eye Tracker 5 camera, a Lapel clip-on microphone, and the Google Board. The eye tracking camera is for tracking the user's eye gaze and head gesture and feeding this data into our visual ML model. The microphone will take in the user's audio input and feed it into our audio alignment program. The Google Board takes care of all of the processing for our system and interacts with our display. The display will be hosted locally on a Windows Computer, which will display the sheet music and page turns. SoundSync is designed to be accessible to those who cannot operate existing page turning machines and for those who are looking for a seamless and non-distracting music experience.

The system will be challenging to implement, especially designing the decision logic to turn the page by weighing the data from the visual model and the audio model. Furthermore, eye tracking will be tricky because musicians tend to look away from the music to watch the conductor or other players for cues.

SoundSync is designed for accessibility and aims to promote inclusivity in the music space.

Glossary of Acronyms

- API - Application Programming Interface
- BPM - Beats Per Minute
- DTW – Dynamic Time Warping
- MIDI - Musical Instrument Digital Interface
- ML – Machine Learning
- OS - Operating System
- SDK - Software Development Kit
- SNR - Signal to Noise Ratio
- TOPS - Terra Operations Per Second

References

[1] Andreas Arzt, Gerhard Widmer, and Simon Dixon. "Automatic Page Turning for Musicians via Real-Time Machine Listening". In: Jan. 2008, pp. 241–245. DOI: 10.3233/978-1-58603-891-5-241.

[2] Anna Bánki et al. "Comparing online webcam- and laboratory-based eye-tracking for the assessment of infants' audio-visual synchrony perception". In: *Front. Psychol.* (2021).

[3] Google. 2023. URL: https://coral.ai/products/dev-board/#description.

[4] JWPepper. 2023. URL: https://www.jwpepper.com/sheet-music/eprint-digital-sheet-music.jsp.

[5] NVIDIA. 2023. URL: https://developer.nvidia.com/embedded/jetson-modules.

[6] Pew Research Center. *The Demographics Of Device Ownership*. Tech. rep. 2022. URL: https://www.pewresearch.org/internet/2015/10/29/the-demographics-of-device-ownership/.

[7] Romain Tavenard. "Introduction to Dynamic Time Warping". In: (2021).

[4] [5] [3] [6] [1] [2] [7]

11 Appendix

Table 1: Bill of materials

Description	Model #	Manufacturer	Quantity	Cost @	Total
Tobii Eye Tracker 5	0005	Tobii	1	$298.53	$298.53
Google Coral Dev Board	0000	Google	1	$144.24	$144.24
Lapel Microphone	0000	Amazon	1	$73.83	$73.83
Battery Pack	0000	Charmast	1	$38.37	$38.37
256GB SD Card	0000	SanDisk	1	$0.00	$0.00
Switch	0000	Amazon	1	$10.98	$10.98
IC Buttons	0000	OTTO	4	$0.00	$0.00
Total					$565.95

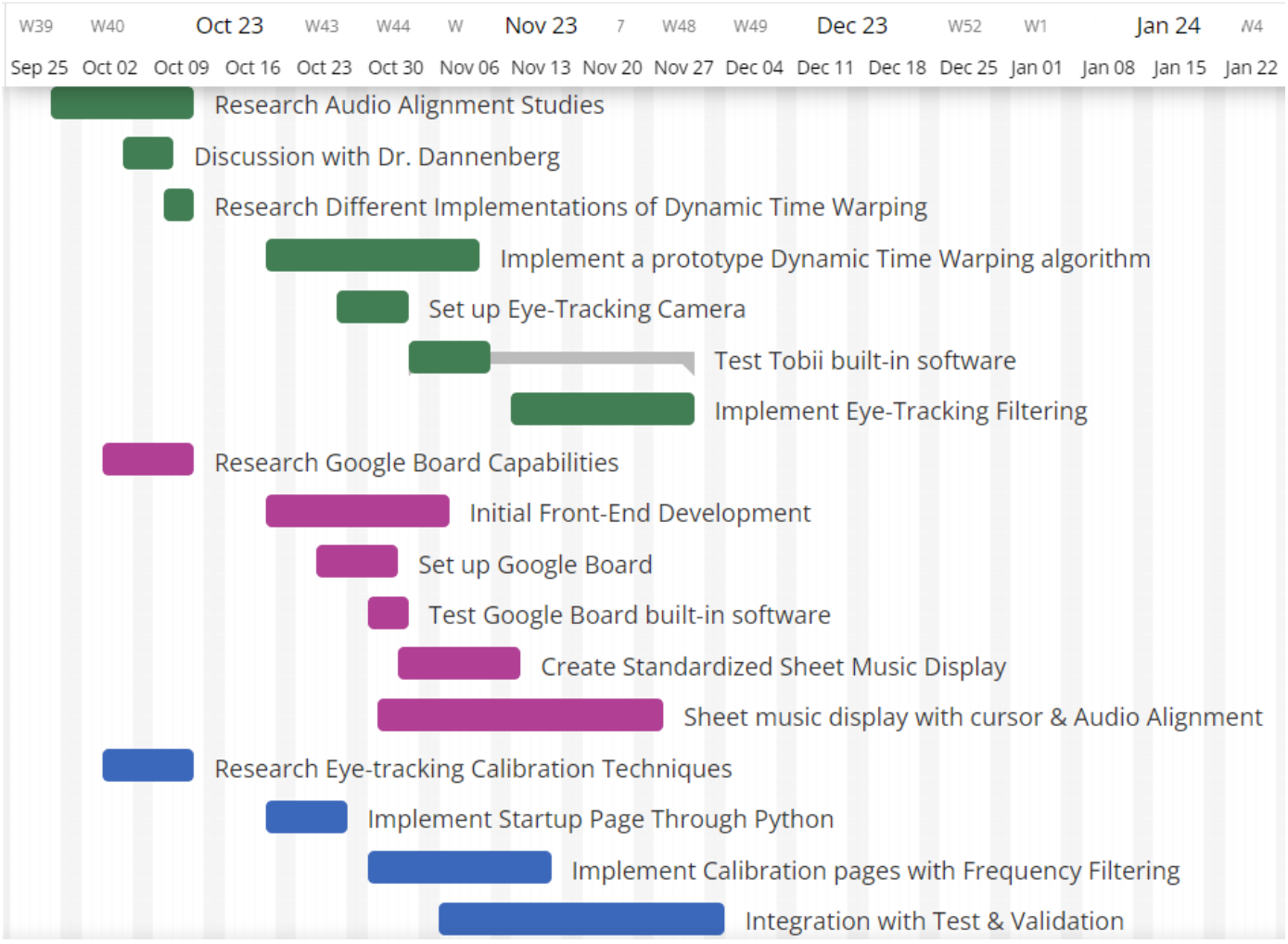

Figure 4: Gantt Chart Diagram. The green tasks (Caleb) are audio, the pink tasks (Rohan) are hardware, and the blue tasks (Sanjana) are display and eye tracking. The division of labor is divided to optimize parallel development and provide slack for integration of different components. The tasks were divided based on areas of expertise or interest and accounted for breaks and holidays.