5-2024

Text-to-Speech Animation: Generating Visuals Based on Phonetic Spelling

Colin Tomcak
colintomcak@unomaha.edu

University of Nebraska at Omaha
College of Information Science & Technology
Department of Computer Science
Supervisor: Dr. Harvey Siy

Honors Capstone Report

in partial fulfillment for the degree
Bachelor of Science in Computer Science (Honors Distinction)
in Spring 2024

Text-to-Speech Animation

—

Generating Visuals Based on Phonetic Spelling

Submitted by: Colin Tomcak

Submission date: May 2024

Colin Tomcak
E-Mail: colintomcak@unomaha.edu
B.S. Computer Science

Abstract

Speech is a complex field, and remarkably difficult to master. There exist many groups who could use more tools to help improve their pronunciation of English words. The goal of this project was to create a text-to-speech animation program as an addition to our group capstone, Pronunciation Pal. Pronunciation Pal is a web-based application meant to assist those who are deaf or hard of hearing in improving their pronunciation, likely as an addition to therapy with a speech pathologist. This application uses tools like diagrams and references to help users better understand how to pronounce a word. The goal of this extension was to dynamically generate a 3D facial animation for any word or set of phonemes. This is an addition to the features on Pronunciation Pal that could be helpful for users, especially to the deaf. The project was successful, generating a smooth and reasonably accurate animation for most words, with a virtual face appearing to enunciate the sounds in the given word. It's able to be hosted on our SvelteKit web application and run with minimal resource utilization entirely on the client-side.

Contents

List of Figures **iii**

1 Introduction **1**

2 Background **2**

3 Literature Review **6**

4 Methodology/Approach **7**

5 Results **12**

6 Challenges **14**

7 Future Improvements **15**

8 Conclusion **16**

References **17**

List of Figures

Figure 1: The Main Page of the Pronunciation Pal Application 2

Figure 2: User Account Page, With Some Favorite Words 3

Figure 3: IPA Phonemic Chart (EnglishClub.com, 2024) 5

Figure 4: Image of MediaPipe's Facial Landmarking 10

Figure 5: Diagram Showing the Execution Flow of the Program 12

Figure 6: The Speech Animation Web Page on Pronunciation Pal 13

Figure 7: Additional Animation Screenshot (Logged In) 14

1 Introduction

Various groups, such as people who are deaf or hard of hearing (DHH) typically benefit from communication therapy sessions with speech-language pathologists. Our group capstone project Pronunciation Pal seeks to assist these user groups in improving their pronunciation of English words using various tools such as pictures and speech-sound diagrams. This project is an extension to Pronunciation that is attempting to dynamically generate a 3D facial animation for any word/set of phonemes. In intention, this addition to the web application seeks to help any user improve their pronunciation by animating the facial movements for any word on demand. A screenshot capturing the main page of Pronunciation Pal can be seen in figure 1, and a screenshot of the accounts page can be seen in figure 2.

This project was sponsored by the University of Nebraska Medical Center Munroe-Meyer Institute. MMI is recognized for providing excellent services to individuals with disabilities, including speech and language therapy. Specifically, we communicated with Korey Stading MS, a certified speech language pathologist (CCC-SLP) who has provided speech and language evaluation and therapy services for over 20 years. She was consulted for information in the realm of speech, language, and speech therapy, and provided feedback on the real-word feasibility of this application.

Pronunciation Pal seeks to be a supplementary tool for user seeing to improve their pronunciation, with the application being able to be used anywhere. It's deployed as a web-based application, written in TypeScript, and using the SvelteKit framework. It contains a database of user profiles which also contain word lists that they've favorited, such as words they're currently working on. This is all integrated with this text-to-speech extension, which is hosted on one the application's pages.

As mentioned, this extension seeks to generate 3D facial animations for any given word. This may be a valuable tool for users, especially the deaf and hard of hearing, because it clearly displays the facial movements needed to pronounce a word. The accuracy of this application is highly important, and was one of the most highly focused aspects of

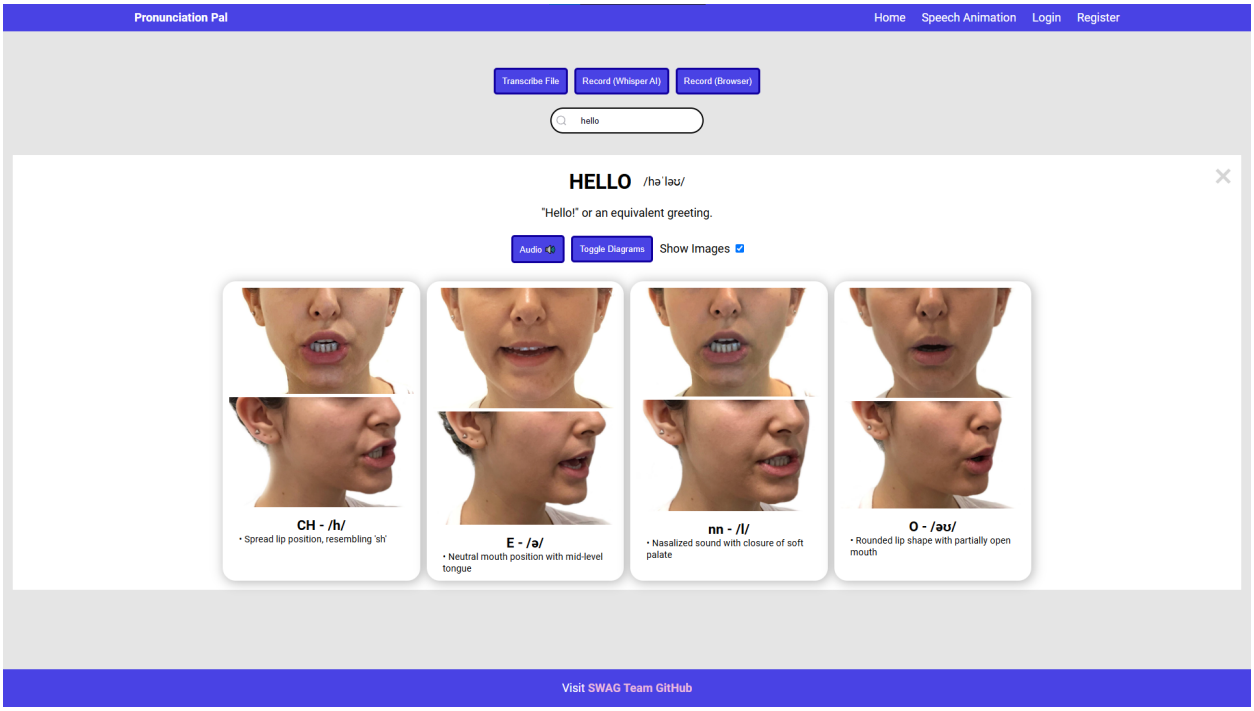

Figure 1: The Main Page of the Pronunciation Pal Application

the extension in its development. The usability with respect to design principles, resource utilization, and eliminating bugs was also considered highly important.

2 Background

The way a word is spelled using the English alphabet often doesn't map to the way it sounds. Simple and common examples of this are "knight", "mortgage", "colonel", and "gnat". Fortunately, this is a well-known and well explored problem, with many fields of research being dedicated specifically to language and speech. Explained here are some important concepts that will be used in this document.

Firstly, phonemes are the smallest units of sound in a language. For example, the sounds /b/, /p/, and /m/ are distinct phonemes. Phonemes can vary between languages, and even within different dialects of the same language. The English we know has 44 phonemes to represent all the sounds heard within the language.

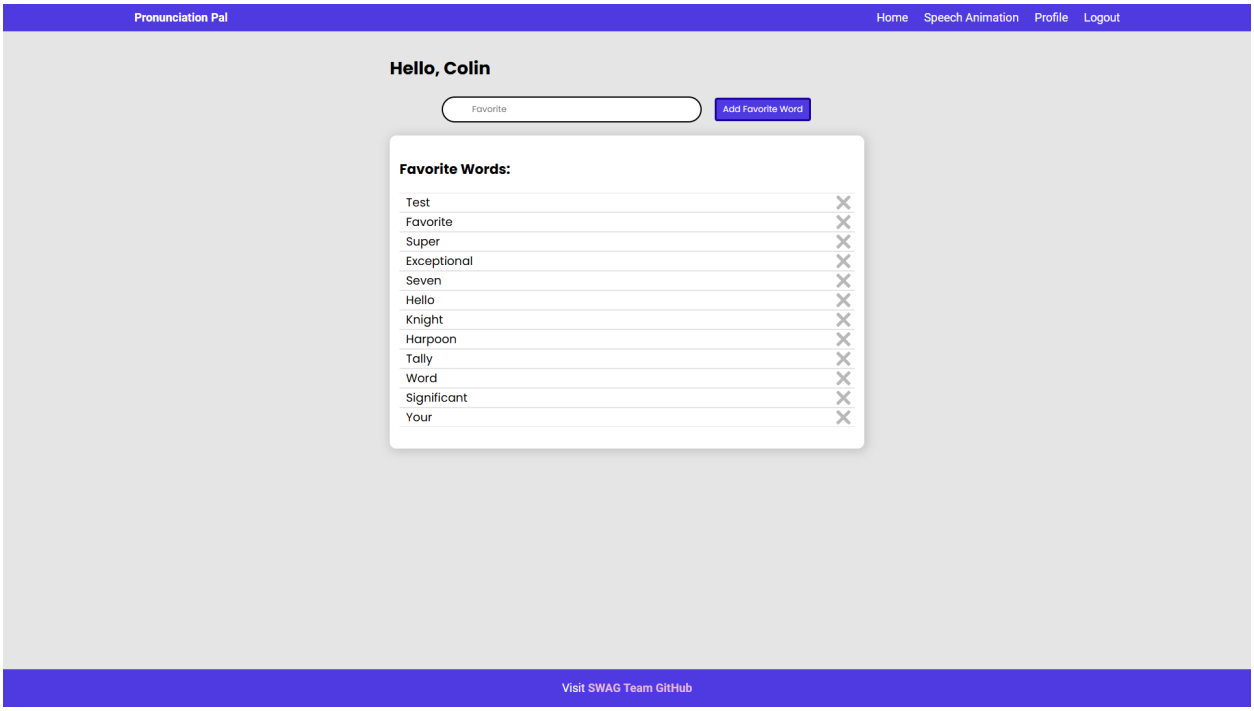

Figure 2: User Account Page, With Some Favorite Words

Graphemes are the written representations of phonemes using a language's alphabet; they compose the orthographic (normal) spelling of a word according to the conventions and rules of that specific alphabet. In English, graphemes can be single letters (like 'a' or 'b'), digraphs (two letters representing one sound, like 'sh', 'th', 'ch', or 'ph'), trigraphs (three letters representing one sound, like 'igh' in "night"), or other orthographic patterns that contribute to its rich but often irregular spelling system. Graphemes don't always correspond directly to phonemes; for example, the letter 'c' can represent different sounds in words like "cat" and "city". Understanding grapheme-phoneme mappings is one of the most essential elements for learning to read and write in any language.

Unfortunately, even understanding graphemes and phonemes, the way certain words sound don't correlate to how they're spelled. For example, the letter 'a' represents slightly different sounds in each word "cat", "car", and "cake". There are also homophones, which are words pronounced the same but spelled differently (to, too, and two), and there are even homographs, which are words spelled the same but pronounced differently ("bow" of a ship

and “bow” that shoots arrows - “bass” the instrument and “bass” the fish). Finally, there are some words that go beyond all of that, whose orthographic spelling doesn’t remotely correlate to its pronunciation (“colonel”).

Due to this, dictionaries usually provide tools to determine how a word should actually sound when it’s spoken. The spelling according to the way a word sounds is called its phonetic spelling. This uses characters/symbols for phonemes that represent a one-to-one correspondence between letters and sounds. This can take different forms. What you’d commonly see in a dictionary is something like this: “unique” with the phonetic spelling being displayed as “yoo-neek”. This is a clear enunciation of the word using graphemes that are relatively objective with regard to how they sound. However, there are also phonetic alphabets, which define specific and definite characters that each map directly to a sound. There are different standards for this; but the most widely used standard is the International Phonetic Alphabet (IPA), which is also the one used in this project. For example, the IPA phonetic spelling for “unique” is /yuˈnik/, and for “hello” it’s /hɛˈloʊ/. A useful table showing examples of the 44 English phonemes can be seen in figure 3. It’s important to note that, occasionally, differing characters are used to describe the same phoneme.

Finally, there are visemes. Visemes are visual representations of phonemes with regard to a person’s face, particularly in the context of lip movements. Visemes and phonemes do not map one-to-one; there can be several phonemes for each viseme. For example, /k/ and /g/, share virtually identical facial positions. Other examples are /b/ and /p/, /v/ and /f/, /n/ and /l/, and /s/ and /z/. Visemes are particularly important in animation and even sign languages, where visual cues can aid comprehension, especially for individuals who are deaf or hard of hearing.

These concepts should cover any terminology used in this paper. This information also serves to remind how intricate language and speech are, highlight the complexities and inconsistencies in the relationship between spelling and pronunciation in English, and to help explain why tools to help with pronunciation are necessary.

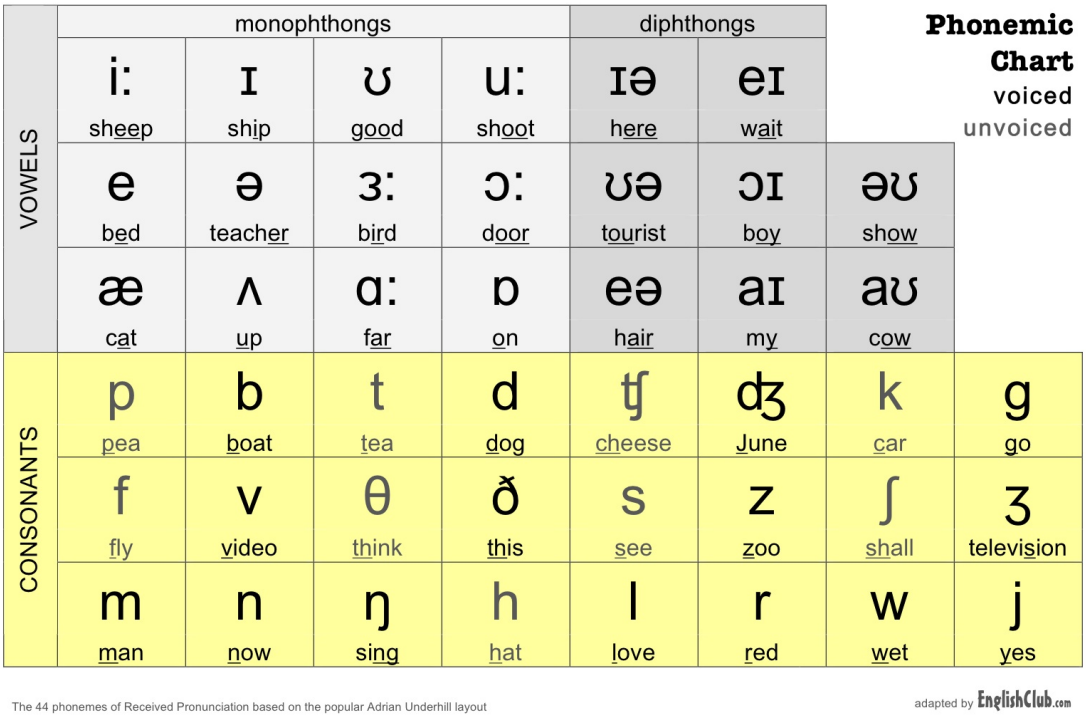

Figure 3: IPA Phonemic Chart (EnglishClub.com, 2024)

3 Literature Review

There are a few different studies that will be briefly referenced here, each addressing the a similar issue this project is. The first is "A Realistic and Reliable 3D Pronunciation Visualization Instruction System for Computer Assisted Language Learning", authored by Jun Yu and Zengfu Wang from the University of Science and Technology of China. This paper proposes a text-driven 3D pronunciation visualization system for computer-assisted language learning. Notably, this is very similar in concept to what this project is attempting to accomplish, but enormously more advanced.

This team used X-ray and magnetic resonance imaging (MRI) images to determine the internal articulatory animation of phonemes, fitting the articulatory shapes to highly detailed mesh models. This data was then used to build a Hidden Markov Model, a type of machine learning model. Along with the use of other complex techniques, articulatory animations could then be produced. The animations produced by this program included both the external appearance of the speech, like the lips, as well as all the internal features, like the tongue. This system was tested and shown to significantly improve the pronunciation accuracy of learners (Yu and Wang, 2016).

The next study is partially similar, where instead of using X-ray or MRI images to capture the articulatory structures inside the mouth, they used ultrasound. The name is "Guided Learning of Pronunciation by Visualizing Tongue Articulation in Ultrasound Image Sequences", written by M. Hamed Mozaffari, Shenyong Guan, Shuangyue Wen, Nan Wang, and Won-Sook Lee from the University of Ottawa. This study investigates the effectiveness of using ultrasound images and videos in conjunction with real-time visual feedback created by using ultrasound on the learner in a guided learning system for pronunciation.

By placing an ultrasound transducer below the chin, visuals of the tongue were able to be seen and recorded. From this, a database of pre-recorded examples of English words were created by native English speakers. To maximize the program's usability for learners, they used image processing techniques to overlay the ultrasound video on top of a feed from

a camera capturing the user's side profile. This overlay technique was also used in real-time on the learners, allowing the learners to easily compare their real-time tongue movements with the prerecorded ones. While using the program, users could search for words from the pre-recorded database, or speak a word and have the database automatically search for a match. The authors determined that the proposed system did in fact help Chinese English learners improve their English pronunciation by helping them match the tongue movements seen in the pre-recorded examples (Mozaffari et al., 2018).

The last study, named "A Deep Learning Approach for Generalized Speech Animation", by Taylor et al., discusses generating speech animations that can synchronize to the sounds within given audio. This uses a deep learning approach, impressively running in real time. Due to the machine learning driven generation, it can also take audio input containing things like singing or foreign languages. The goal of the study was to generate an approach for automated speech animation that was cost-effective, generating accurate and detailed animations at scale. The results of this approach were successful, showing significant performance improvements over previous approaches. However, it was also limited by a couple factors, including limited training data. This model was not used to teach pronunciation, but the results of this study are useful for understanding more about the different technical approaches used to generate speech animations (Taylor et al., 2017).

These studies, specifically the first two, show that speech/articulatory visualization is an effective tool for improving the pronunciation skills of learners.

4 Methodology/Approach

This program makes use of the same approach; the Text-to-Speech Animator extension to Pronunciation Pal enhances the learning effectiveness of the application through visualizing the way words are pronounced to help improve users' articulation.

The internal workings of this program are less complex than they may seem on the surface, and it uses no machine learning. It was implemented first in Python, then ported to TypeScript to run on the Pronunciation Pal web application (using the SvelteKit web framework). Partially because it uses no machine learning, it takes little computing resources to run, generating animations virtually instantly while running entirely on the client-side.

As discussed, every word has a phonetic spelling: the true spelling for how the word sounds. The first task is to get the phonetic spelling for the word the user inputted. This is done by reaching out to an API, specifically this free and open-source service: `https://dictionaryapi.dev/` (meetDeveloper, 2024). If the word exist in the API, we receive the phonetic spelling in International Phonetic Alphabet (IPA) characters.

Each phoneme has a corresponding facial position, so the next task is to map each phoneme in the word to visemes. In English, it's generally agreed there's not more than 11-16 meaningfully different visemes. In the program, there are 15 visemes, each corresponding to one or more characters in the IPA. Here's the description for each viseme in the program:

- "aa": Wide, open mouth shape.
- "CH": Spread lip position, resembling 'sh'.
- "DD": Sudden release of tongue from roof of mouth.
- "E": Neutral mouth position with mid-level tongue.
- "FF": Upper teeth touching lower lip, creating friction.
- "I": Relatively closed mouth with raised tongue.
- "kk": Strong closure of back of tongue against soft palate.
- "nn": Nasalized sound with closure of soft palate.
- "O": Rounded lip shape with partially open mouth.

- "PP": Complete closure of lips with pressure buildup.
- "RR": Slight constriction of tongue with lips slightly apart.
- "sil": Silence or neutral mouth position.
- "SS": Narrow opening between upper and lower teeth, frictional airflow.
- "TH": Tongue between or near teeth, creating fricative sound.
- "U": Rounded lip shape with relatively open mouth.

This viseme selection and naming convention was formed by information generated for virtual reality developers Meta (2024). A dictionary was built to map to each different phoneme to its viseme. A few examples of how phonemes would map to visemes are included here:

- “Home” – Phonemes /hoʊm/ – Visemes [CH, O, PP]
- “Dog” – Phonemes /dɒg/ – Visemes [DD, aa, kk]
- “Hello” – Phonemes /hɛˈloʊ/ – Visemes [CH, E, nn, O]
- “Test” – Phonemes /test/ – Visemes [DD, E, SS, DD]

With the visemes for each word, it was now possible to generate animations for each word. This is how that task was accomplished. Google has a computer vision library called MediaPipe (`https://developers.google.com/mediapipe`). With just the average webcam input, it’s able to landmark 468 different points on a person’s face. A picture showing how the landmarks are distributed on a real face can be seen in figure 4. This library was used in python scripts to capture the facial position for each viseme. These landmark coordinates were simply put into files for each viseme to be used by the program. When the user inputs a word and the visemes are obtained, the program can easily load the necessary data into memory.

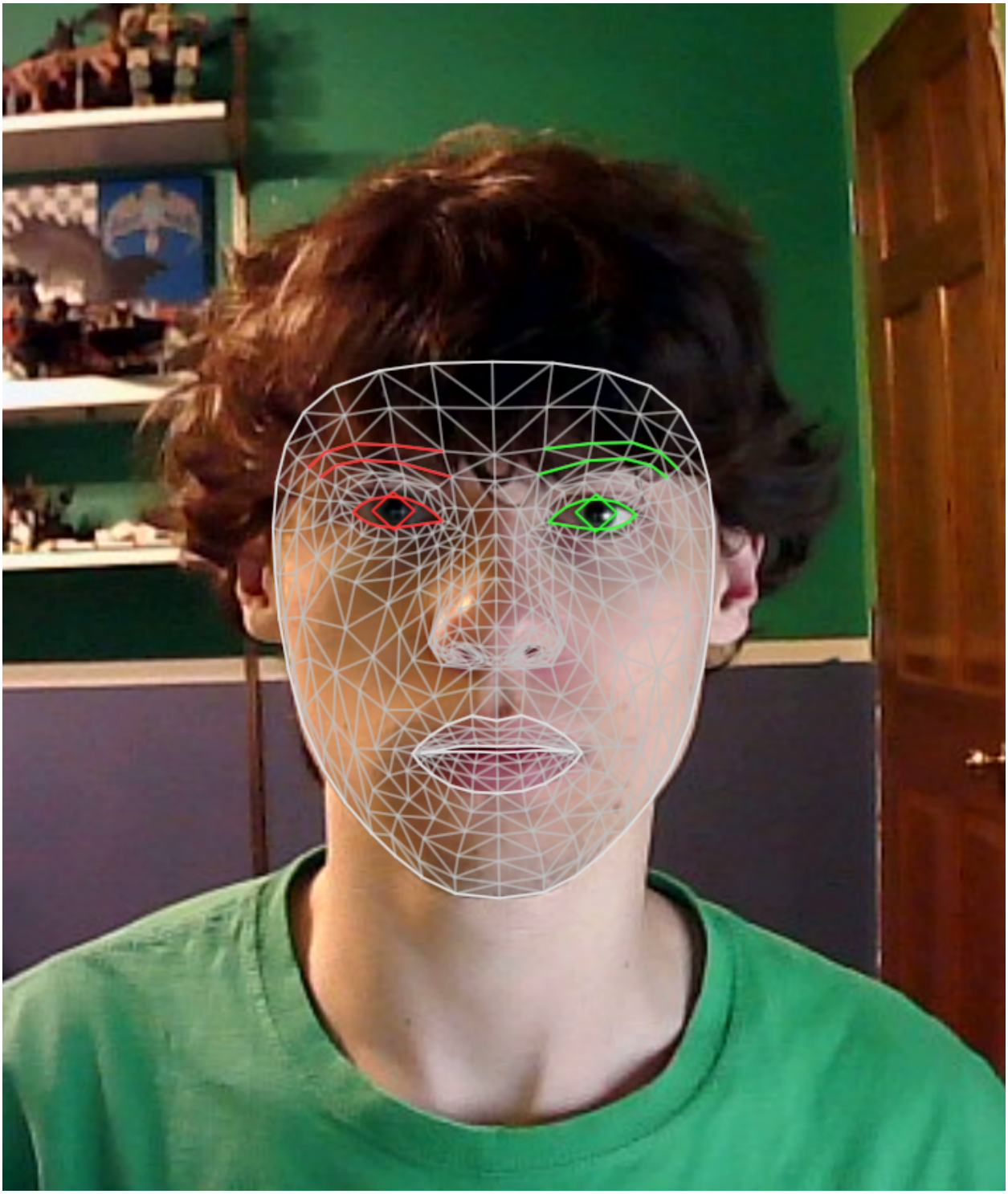

Figure 4: Image of MediaPipe's Facial Landmarking

The next step was to create an interpolation function to generate frames between each viseme. Because the facial structure exists entirely in landmarks, which are 3D coordinates, it was simple to adjust each point in the facial position a certain distance to the next viseme based on the desired number of interpolation frames. In other words, if the user set the number of interpolation frames to a higher number, the facial position would be adjusted a smaller amount every frame to reach the next facial position within that number of frames. The program runs at a constant 60 frames per second, meaning the speed of the animation is tied to the number of interpolation frames between each viseme. Overall, this creates a relatively convincing animation for any given word whose phonemes are available. The number of interpolation frames can be adjusted during runtime as well, without needing to reload anything.

Almost last is the drawing function. The drawing function in the python implementation was built into the MediaPipe library, meaning initially it didn't even have to be implemented. However, upon the port to TypeScript, it was decided to implement a drawing function for easier customization. This included drawing the coordinates, the face mesh connections, and predicting the position of the teeth. Face mesh connections are lines drawn between certain points, like general tessellation for the face, or lines specifically around a feature, like the eyes. This uses face mesh connection arrays from the MediaPipe source code to map out the general facial structure, eyes, irises, eyebrows, nose, and most importantly, lips. The final feature of the drawing function, not contained in the MediaPipe function, was predicting teeth position. The MediaPipe facial landmarking doesn't capture the inside of the mouth, meaning the position of the teeth had to be predicted based on the coordinates of other facial features. This included points like the tip of the nose and the bottom of the chin. This was done, and the teeth were drawn as simple curved lines in the mouth, reflecting the predicted position during the animation of the word.

At this point, there were only a couple more adjustments to be made. One feature that was implemented to help make the animation more accurate was variable phoneme

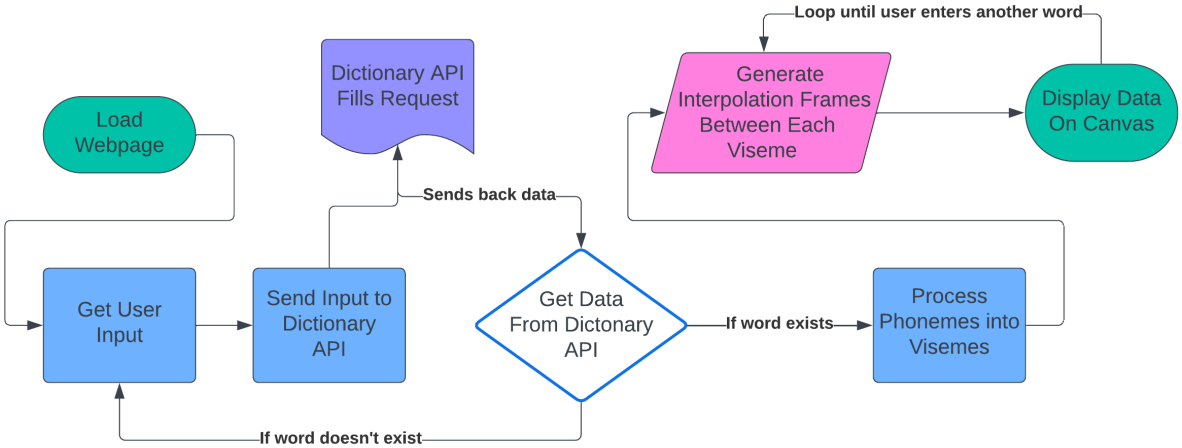

Figure 5: Diagram Showing the Execution Flow of the Program

lengths. This feature included adjusting the number of interpolation frames between each viseme based on how long a phoneme typically takes to speak. A dictionary was made with this information, all estimates. For a phoneme like /f/, the estimate was shorter compared to a phoneme like /oʊ/ (as in "slow"). After feedback from Korey Stading, MS, CCC-SLP, the ability for certain diphthongs (a sound composed by the combination of two vowels in a single syllable, e.g., /aʊ/ and /ɔɪ/) to map to multiple visemes was added. At this point, the implementation was completed. An execution flow diagram of the program can be seen in figure 5.

5 Results

The results of this program were generally decent. The animation looks more like the face is enunciating each sound of the word rather than natural speech, but this isn't necessarily a bad thing with regard to speech therapy. It's also the case that some words look better than others.

It's important to note that this application was not able to be tested on patients. This is due to various legal and ethical concerns, such as things like confidentiality and HIPPA;

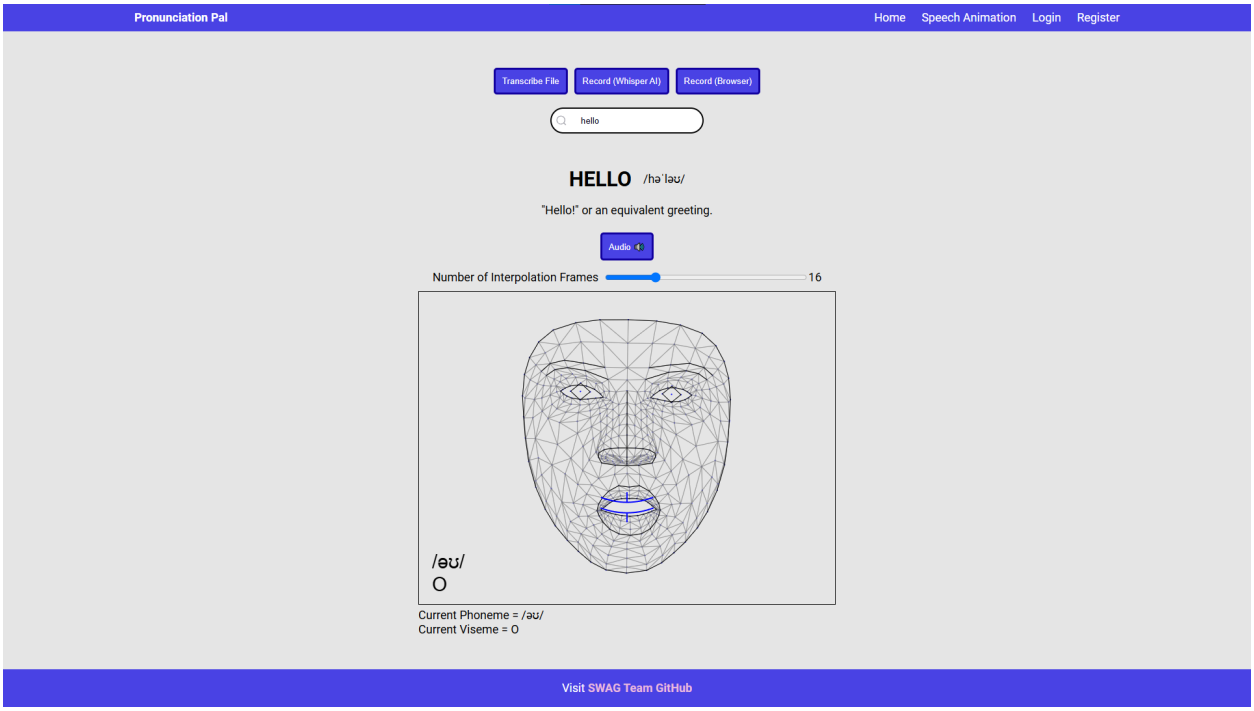

Figure 6: The Speech Animation Web Page on Pronunciation Pal

all outside my area of expertise. We tested our application ourselves and with some people we know, such as friends and parents.

The extension was a success with regard to usability. The user interface is simple, and when tested on friends and family, was quickly understood. Users were able to input words and interact with the results, agreeing that they could see the searched word being spoken in the animation. As mentioned before, because this doesn't use any machine learning approach, it easily runs entirely on the client side with minimal resource utilization. When tests were conducted, no bugs were found. A screenshot of the animation page with the program animating the word "hello" can be seen in figure 6. Another screenshot with the user logged in can be seen in figure 7.

To test the implementation yourself, you can run the Pronunciation Pal web application. This can be found at `https://github.com/cttomcak/Team-12-SWAG-Pronunciation-Pal`. The instructions for how to install the dependencies and run the application can be found in the README.md.

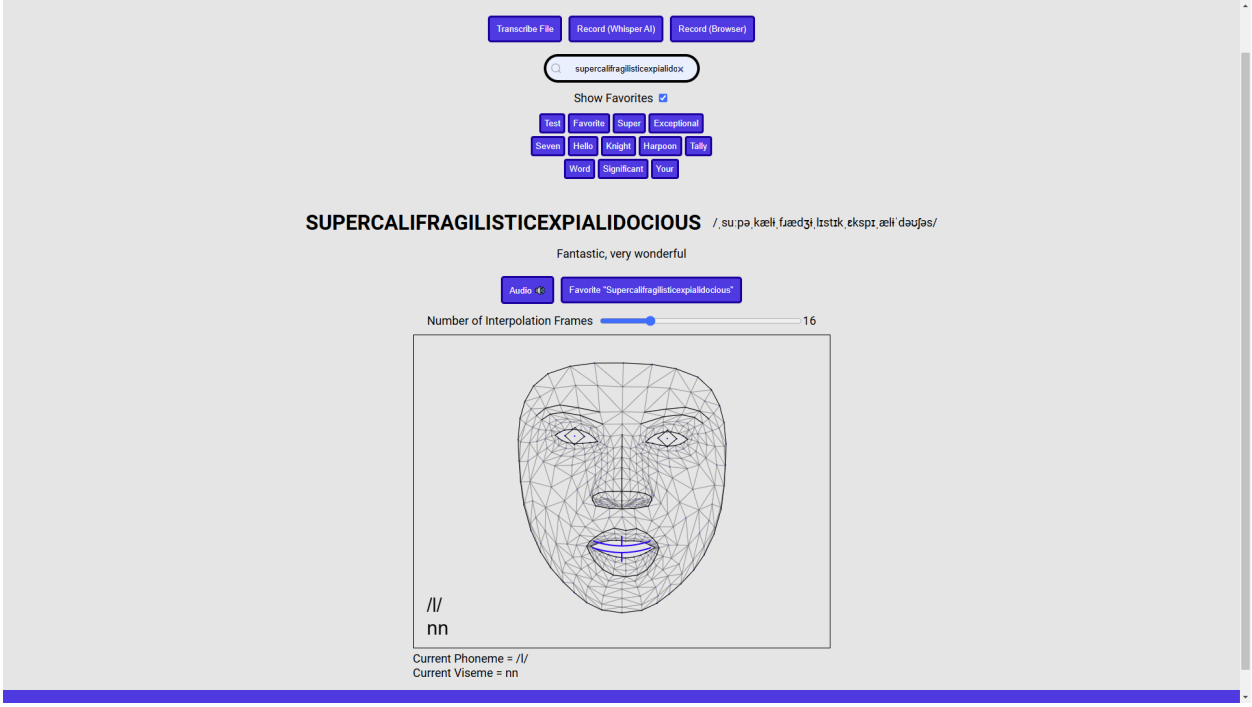

Figure 7: Additional Animation Screenshot (Logged In)

6 Challenges

The largest challenge in implementing this project was putting together an accurate phoneme to viseme dictionary. One of the reasons this presented so much difficulty was due to the fact that the dictionary API used often returned differing Unicode variations of phonemes, resulting in the program not recognizing them. This was mitigated through testing; when an unmapped phoneme character was found, it was immediately added to the dictionary with the proper viseme mapping. Even towards the end of development, the odd unmapped character was still being found, but as of the completion of the project, it's very rare.

There also wasn't as much information on phoneme and viseme mappings available online as was initially thought when starting the project. This presented further difficulty in generating accurate mappings, requiring significant manual effort to accomplish the task.

7 Future Improvements

There are many potential improvements to be made to this program. The first I would implement is adding more visemes and re-recording the ones currently existing. While many phonemes truly map to the same viseme, some that currently map to the same viseme in the program could be separated. One example is /m/, which maps to PP. This is questionably accurate, as both /m/ and /p/ each contain the closed mouth position, but it could be made more accurate by adding another viseme for /m/. The rerecording is also just for accuracy.

Another improvement is better mappings between phonemes and visemes. As discussed in challenges, the API can return various Unicode characters for each phoneme, making mapping all of them difficult. Occasionally one of these characters will be mapped incorrectly or cause the program to only detect one character out of a two-character phoneme (like /ɔɪ/), but this can be mitigated through more testing.

Next are feature additions to the face mesh and animation. Firstly, a proper face mesh could be added as an option; a more realistic mesh could be created in blender, with bones for each facial landmark. This would result in a realistic looking face animated in the same way the tessellated face is. Custom colors could also be added for facial features like eyes and lips. Besides this, perhaps the most vital addition is animation of the inner mouth, adding tongue movement. The appearance of the face and lips is only half of the story, so to speak. Internal articulators like the tongue and pharynx are equally important, if not more so. Due to lack of tools, resources, and time, details within the inner mouth (besides the prediction of teeth position) were not able to be implemented. However, this is something that could greatly enhance the usefulness of the program in various ways for its users. The ability to see an x-ray side profile with full animation of the tongue and potentially even airways could be exceptionally helpful.

The final improvement would be a redesign and remake of the program using machine learning. The most advanced text-to-speech animation would likely use a machine learning model trained on a colossal dataset. Rather than using the phonetic spelling of words to

determine facial positions, it would likely be audio driven, receiving audio input from an advanced text-to-speech model if it needed to do text-to-speech animation. It would have a much more detailed face mesh, and because it'd take audio input, it would be capable of animating most sounds the human mouth could make, not just basic speech. However, this takes a vast number of resources and a large, annotated dataset. This has actually been done by several organizations (including the researchers in some of the papers which were reviewed), but not to the extent where it truly looks like real speech.

8 Conclusion

In conclusion, this extension to the existing Pronunciation application is as a promising tool in aiding individuals, particularly those who are deaf or hard of hearing, in enhancing their English pronunciation skills. The project benefitted from the expertise and guidance of Korey Stading MS, a certified speech-language pathologist with over two decades of experience, ensuring its real-world feasibility and alignment with the needs of speech and language therapy. Deployed as a web-based application using TypeScript and the SvelteKit framework, Pronunciation Pal offers users the flexibility to access it from anywhere, with a user-friendly interface and database functionality allowing for personalized learning experiences. In the making of this extension, there was emphasis on accuracy, usability, and adherence to design principles.

In the making of this project, I learned a lot about speech and language. This includes information about phonemes, visemes, and English in a broader sense. I learned that speech is a very complex field, and contains an incredible number of small details that change the way words are spoken.

References

Bear, H. L. and Harvey, R. (2017). Phoneme-to-viseme mappings: the good, the bad, and the ugly. *Speech Communication*, 95:40–67.

EnglishClub.com (2024). Phonemic Chart | Learn English. `https://www.englishclub.com/pronunciation/phonemic-chart.php`. Last accessed: 2024-05-13.

International Phonetic Association (2024). Links to Phonetics Resources | International Phonetic Association. `https://www.internationalphoneticassociation.org/content/links-phonetics-resources`. Last accessed: 2024-05-12.

meetDeveloper (2024). Free Dictionary API. `https://github.com/meetDeveloper/freeDictionaryAPI`. original-date: 2018-05-12.

Meta (2024). Viseme Reference: Unity | Oculus Developers. `https://developer.oculus.com/documentation/unity/audio-ovrlipsync-viseme-reference/`. Last accessed: 2024-05-12.

Mozaffari, M. H., Guan, S., Wen, S., Wang, N., and Lee, W.-S. (2018). Guided Learning of Pronunciation by Visualizing Tongue Articulation in Ultrasound Image Sequences. In *2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pages 1–5. ISSN: 2377-9322.

Taylor, S., Kim, T., Yue, Y., Mahler, M., Krahe, J., Rodriguez, A. G., Hodgins, J., and Matthews, I. (2017). A deep learning approach for generalized speech animation. *ACM Transactions on Graphics*, 36(4):1–11.

Yu, J. and Wang, Z. (2016). A Realistic and Reliable 3D Pronunciation Visualization Instruction System for Computer-Assisted Language Learning. In *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 786–789.