Boston University
Electrical & Computer Engineering

EC463 Senior Design Project

First Semester Report: Vobot - Team 14

Submitted to:

Andrey Vyshedsky
ImagiRation
(617) 433-8577
imagiration@gmail.com

Team Members:

Laura Salinas: laums@bu.edu
Priya Kapadia: priyak@bu.edu
Steven Graham: smgraham@bu.edu
Shivani Bhatia: scbhatia@bu.edu
Arley Trujillo: arley78@bu.edu

Submitted: ______________

Table of Contents

Executive Summary

Vobot First Semester Report
Team 14– Vobot

One in sixty-eight U.S children has Autism Spectrum Disorder (ASD). Of these children, 30-40% remain nonverbal for the rest of their lives and are placed into institutions as they cannot take care of their own needs. Children with ASD only have a short window of time for language acquisition – the first five years. That being said, it is imperative to maximize their exposure to language from a young age, thus improving their communication and speech skills. Maximizing the amount and quality of exposure is not a simple undertaking, and is a responsibility that usually falls on the child's parents. Parents often opt to enlist the talents of trained professionals for treatment, but doing so is very costly. While such assistance is helpful and can provide results, the costs mentioned may cut sessions short or deter a family from even considering the option. To further compound these issues, there is a personnel shortage of these professionals. For those that are already in the field, their talents may be spread thin across many different students in schools resulting in fragmented progress.

Our group is determined to deliver a voice controlled language therapy robot that will help assist the child's language acquisition. Our proposed technical approach is to modify the WowWee MiP and CHiP Robot, which come with an API to control its movements, as well as bluetooth capabilities. In order to communicate with the child, we will be including a small microphone-speaker bluetooth module. The bluetooth module will interface with the Android application to determine what the child has said and run the recording through an algorithm that will return a similarity score of how close the child's articulation is to the word that is being practiced. The robot will also be paired with the Android application, which will contain information about the performance of the child. Upon a successful attempt, the robot will reward the child by performing one of its skills like dancing. This will allow parents or therapists to track the results of the child as they interact with and learn from the robot. The robot and application are meant to be a cost efficient supplement to speech therapy, being a tool for doctors and parents to use on the days when speech therapy through a professional is not available (at home sessions).

While there are devices in the market that aim to do the similar language acquisition training, neither of these products offer an individualized learning curriculum, nor do they communicate effectively with the child. We hope to change this by providing a fun, friendly, adaptable, and customizable language therapy robot that will interact with the child, while also motivating them to learn different words.

1.0 Introduction

ImagiRation's problem stems from the need to help children with ASD in developing their abilities to talk before the language acquisition windows closes around the age of five. Speech therapy exists for children with this disability, but remains very costly for parents and suffers from a lack of trained personnel. The motivation behind our effort is born out of the realization that there exist several compounding factors contributing to the problem, and that these factors may have their impact reduced cohesively and simultaneously with our project.

The purpose of this project is to motivate children affected by this disorder to start speaking, with the intention of giving them a better chance at succeeding in life and integrating into society. The technique we are employing for children to learn is called "progressive shaping of successive approximations". Shaping is a behavioral term that refers to the gradual molding or training of someone to perform a specific response through positive reinforcement. In the case of our project, we are going to be shaping the child's vocalizations through a lesson plan created by the child's speech language pathologist (SLP). In order to accomplish this, our team's approach is to modify an existing robot and leverage it to listen to a child's vocalizations. In order for the robot to encourage the child to learn, it will need to reward the child when he or she has done a satisfactory job speaking the word. In doing so, the robot will return a reward through onboard skills such as dancing, singing, and doing yoga positions(CHiP). Level of difficulty for when reward is given should increase as child's progression increases. This is to say on a first attempt the robot will reward child on any vocalization, next attempt will only reward with vocalization slightly more similar to model word, otherwise after child has five attempts to vocalize, the level will be dropped until child can successfully complete vocalizations that sound increasingly more like the model word. This strict scoring is based on client requirements that prevent child progressing by trial-and-error without mastering skills. Once the child reaches the similarity score of 90 out of 100, a new word is introduced.

Our project uses a speech recognition algorithm that returns a similarity score for how close the child's vocalizations are to a model word. Products in the market currently only work on training a child's ability to understand receptive language, which correlates to understanding what is said to you versus the ability to use vocabulary and to put words together into sentences to express yourself. By creating a cost effective, portable solution to aid or replace speech language therapists, users will be able to practice their verbal communication skills in the comfort of their own home.

2.0 Concept Development

After first meeting with our client, our team left with the understanding that our client wanted to transform the WowWee MiP Robot into a language-therapy robot for children with Autism Spectrum Disorder. The minimum requirements for our system are to be able to process a child's vocalization, produce a similarity score of the vocalization to a "model word", provide an appropriate response through the robot, and create an interface for the child's parents and SLP to monitor the child's progress. Based on our customer's needs, our team ultimately broke down the development of our project into four main components: an application, a bluetooth microphone-speaker module, a database, and a speech recognition scoring algorithm.

Our team decided that it was best to create an application so that we can centrally do all of our computations, while also providing an interface for the child's parents and SLP to use. Our three options were to create a web application, an iPhone application, or an Android application. After looking at the compatibilities of the MiP Robot and the speech processing API's we were hoping to use for our product, we decided that building an Android application would be best since it is compatible with both components and our team members have more experience with building Android applications.

Once we decided on an Android application, our team's next step was to decide on a speech recognition and scoring algorithm. Since we are scoring the child's accuracy compared to the model word, our client specified that the child should attempt the word five times, each producing its own similarity score to the model word. After recording the word five times, the system would average all of the similarity scores for better accuracy. Since there are many open source speech recognition API's, our team began to look into different API's that we could integrate into our application. One of the main API's our team looked into was SnowBoy, which was recommended to us by our client. Our team also looked into the possibility of using Sphinx, one of the oldest and most extensive speech recognition API's to exist. The problem with both of these API's is that they were difficult to use and implement. SnowBoy would not integrate properly with Android Studio, making it impossible to test, and Sphinx was unable to recognize unintelligible speech, such as "ta" or "ble" when trying to say the word "table". Our team then came across SpeechAce, a speech recognition API that also provided a similarity score of a vocalization compared to a model word. After testing, our team decided to use SpeechAce as both our speech recognition software and scoring algorithm. We decided to use SpeechAce because it not only acted as a speech recognition software, but also provided us a similarity score percentage that highlights portions of the model word that were mispronounced, which we otherwise would have had to write by ourselves. Latency is also a key factor in our system. Since we are creating a robot that is interacting with children, it must be able to respond to the child within 2 seconds of the child's vocalization, if a response is elicited. SpeechAce works with a low-latency, on average, of 10ms after the first run.

Our client specified that he would like for us to use a bluetooth microphone-speaker module that plays "Say the word ______", processes the child's vocalization, and sends it via bluetooth to the Android application. The two options we have been considering are the Pred Smart Cube and an Arduino based bluetooth module. The Pred SmartCube is an existing bluetooth microphone-speaker module that was given to us by our client. It is one of the smallest bluetooth modules in market, measuring approximately 1" on all sides and costing only $15.00. The Pred Smart Cube already has all of the technologies we would need for our application to successfully communicate with the child, making it a much more cost effective and easier option to implement. On the other hand, we have the option to create our own bluetooth communication system via the Arduino. This system would consist of a microphone, bluetooth module, speaker, and an Arduino. Realistically, instead of an Arduino, we would use a version of the Sparkfun MiP Pro-Mini Board, which has all of the functionalities and capabilities of the Arduino, but fits nicely inside the robot's casing. This option, though more costly, allows for more flexibility in terms of the information we would like to send to the application, as well as what we can include in this system. We have yet to choose our microphone-speaker module, as both present possible communication issues when switching between receiving microphone input and producing output from the speaker. Our initial latency tests on both modules have given us comparable latency speeds, measuring an average of 1 second for the Arduino based module, and 2.5 seconds for the Pred Smart Cube.

Since our client wanted to create an interface where parents can track and monitor their child's progress, we have chosen to develop a database to store the child's information and progress for up to four years. As the child completes his or her session, the application will send information to the database for storage. The database will store the parent's phone number, child's name, model words, the child's vocalizations, and the child's lesson plan. Since we would have to retrieve information from the database and display it on the application for parents to see, we want to pick a database that is fast, but also easy to use. Our team is currently considering Oracle 2C and MySQL, but we would also like to explore other options before making a decision.

3.0 System Description

Our deliverables will include an Android application and language therapy robot. We intend to have an Android application that allows for the SLP or child's parent to record various sounds corresponding to levels of a word into the application, with the goal word being the last level. The Android application will also store each week's curriculum into a database, so that the child's parents can monitor their progress. The database must be able to hold 3 years of data. Thus, the application must be lightweight to download (<10MB) for users since they would be keeping the application for more than a year. It will also interface with the robot to start the learning process. The language therapy robot is responsible for communicating with the child. As opposed to creating a new robot, we have chosen to modify the child-proof WowWee CHiP Robot, which already comes with an API to control its movement, as well as bluetooth capabilities. We will also attach a bluetooth module capable of communicating with the Android application. The Android app will use the bluetooth mic/speaker to prompt the child to say a word, record the child's attempt, and make the robot provide a positive reinforcement if the child made progress from the last time. Some examples of positive feedback include dancing, laying down, or saying encouraging words. Additionally, the application should allow for the parent to customize the type of positive reinforcement the child receives when incrementing a level.

To determine how close the model word is to the word that the child has spoken, we will be using signal processing in which the Android application will receive the child's vocalization and run it through a cross-correlation algorithm. The cross-correlation algorithm will compare the child's vocalization to the recorded sounds in the curriculum and return a similarity score comparable to the child's accuracy. Depending on the similarity score, the application will determine whether the child should increment, decrement, or stay at the same level in the curriculum. We intend to use an API from SpeechAce to accomplish this task. The cross correlation algorithm will return similarity score between 1-100. Levels of difficulty will increment with ≥ 90% similarity score, and decrement with ≤ 60% similarity score.

Database Model

The unique identifier (key) for every child is the parent's phone number. Our client did not want the user to login, and so the parent would save all the information under their phone number without having a password. Each child has multiple words that they are learning, and each word will have multiple attempts and reports. We need a separate database for all model words. Since these are specific to the child due to the parent's accent, we can store them under the same account. The model is as follows:

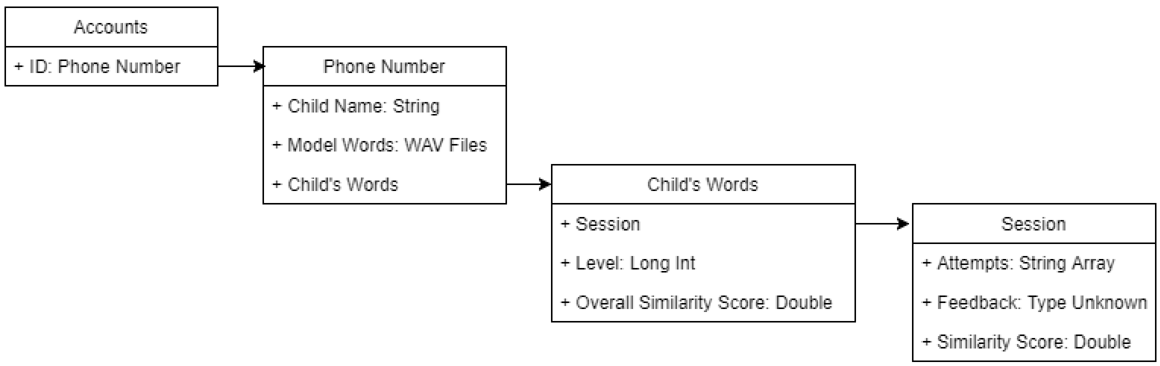


Figure 1: Database UML

Mockup of the Android Application and Pseudocode

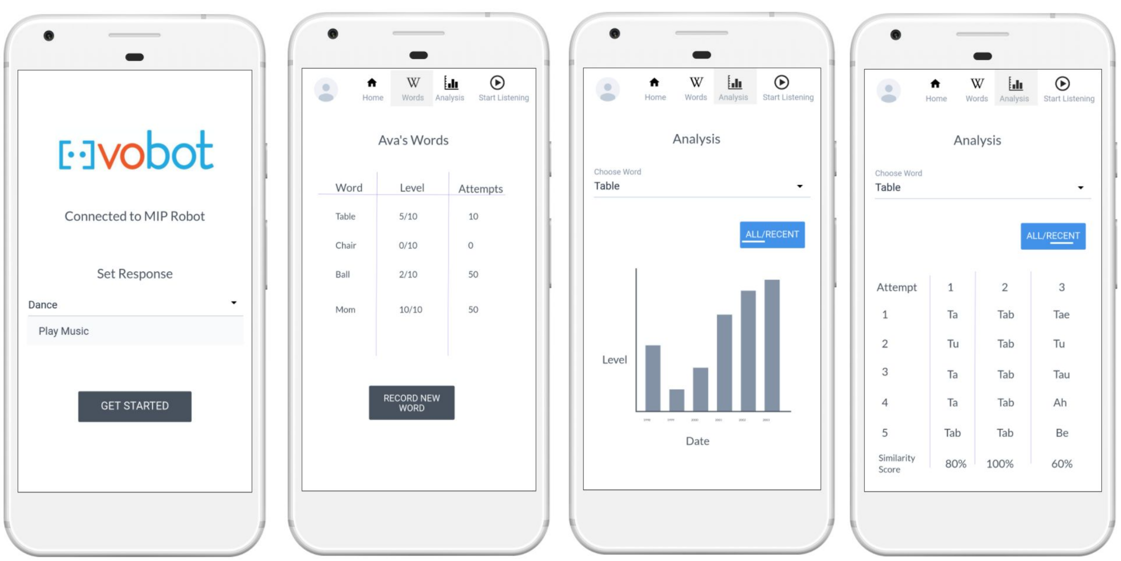


Figure 2: Android application UI mockups

When the app is first installed, it will ask the user for the phone number that the account should be associated with, and the child's name. Once received, the account is set. This is important for storing all the child's information in the database with a unique key.

The leftmost figure in *Figure 2* displays the home page of the app. It shows the user if the MiP robot is connected or not. The user can also set a response that will be the positive feedback for the child.

Pseudocode:

1. Instantiate an instance of a robot object, and connect the instance to the robot using bluetooth.
2. Set a global variable for the positive feedback chosen.

The second figure (from left to right) displays a list of available words that the child can practice. It will also show a summary of the child's progress for every word, and the number of attempts they have made. The parent can determine which word they should make the child practice. Last, it would provide the parent an option to record another new word.

Pseudocode:

1. Access the database and display the statistics required per word.
2. Enable access to the user's phone microphone to record a new word and store it.

The third and fourth figures show the parent specific analytics on individual words. They can toggle between a graph view of the levels progressed over time, and a table view of what the child said for every attempt. The analysis page may additionally reveal a summary of phonemes that the child needs to focus on developing. The specifics will be determined by our client as he is currently reviewing the mock-up with his partners. The graph view would be motivating for the parent, and the table view would give more insight and allow the parent to focus on specific phonemes.

Pseudocode:

1. Create both the tabular and graphical views template.
2. Access the database and display the statistics required dynamically.

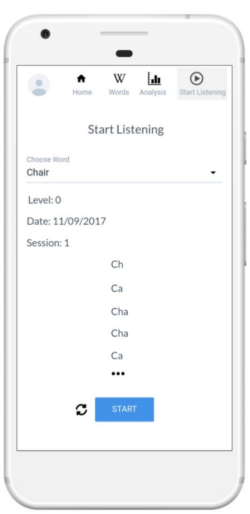


Figure 3 depicts the learning page. The parent can select the word they want the child to start learning from the list of recorded model words. When 'Start Listening' is clicked on, the app will prompt the child to say that particular word. As per our client's requirements, the app will record 5 attempts and store the session feedback. The parent can also refresh and start a new session.

Figure 3: Application page for vocalizations heard

Pseudocode

1. Provide a menu option for the user to select the word being practiced.
2. Upon clicking the ‘Start’ button, output an audio file to the microphone to prompt the child. Then, record the word spoken, and output the results on the screen. Compute the similarity score and provide a response through the robot.
3. Save the new session into the database.

System Block Diagram

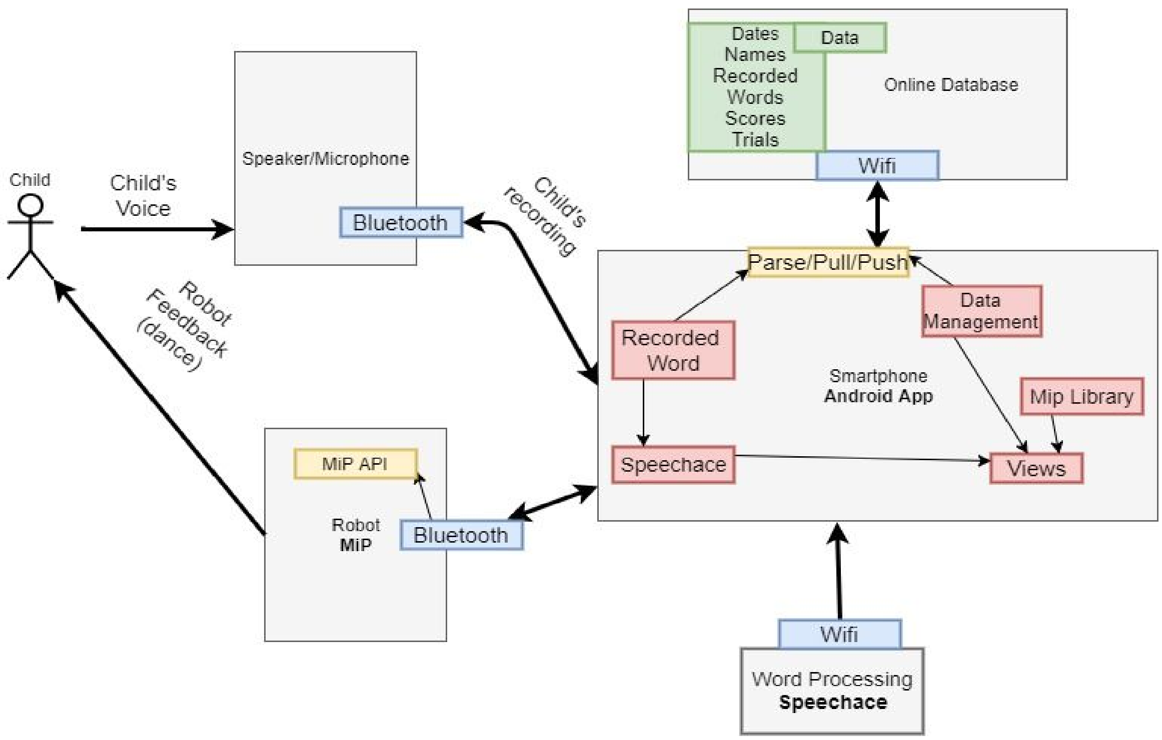


Figure 4: Level 2 system block diagram

As seen in the system block diagram above, the major subsystems include the application, robot, bluetooth mic/speaker, database, and SpeechAce API. The modes of communication are in blue. As a result, the child only interfaces with the robot and mic/speaker.

4.0 First Semester Progress

The primary focus this semester was on developing an user interface to control the robot, as well as creating and testing a simple pipeline that would take in user recordings and correlate that to an action performed by the robot. In doing so we would have the foundation for the rest of the project complete and would only have to work on integrating additional functionality and securing external components to the main body of the robot.

Hardware

This semester on the hardware side, we tested the PRED Smart Cube and its ability to capture and reproduce sound of high enough fidelity to meet our functional needs. We were also able to acquire and work with the MiP robot model from the manufacturer our client has partnered with for the duration of this project (WowWee). By having the two biggest physical components out of the way we could proceed to develop the software that would integrate them into a functioning system for speech therapy.

During initial testing of integration with the Smart Cube and our Android app, we noticed that the bluetooth module was opening two way communication at the same time, meaning that it would listen to and then hear itself at the same time. For this reason, we chose to interrupt the module's ability to listen to sound and instead take that input from the phone via the Android application we had developed. At the time, this decision was made in order to meet first deliverable needs and as a way to test whether initial speech recognition efforts were functional. While we are still working to resolve the two-way functionality, we also felt that it was necessary to work on another solution in parallel in case we were not successful with the Smart Cube.

Out of this technical roadblock, we conceived the idea to build our own microphone-speaker module using an Arduino based platform. The goal of this wireless communication system is to be able to transmit information from a microphone via bluetooth to an application. The current Arduino based bluetooth LE communication circuit consists of the Adafruit Electret Microphone-Amplifier, Adafruit Bluefruit SPI Friend Bluetooth LE module, and an Arduino Uno. 5 digital pins from the Arduino are connected to the pins of the Bluefruit SPI friend, and 1 analog pin is connected to the Electret Microphone. Both the Bluefruit SPI Friend and the Electret Amplifier are connected to the Arduino's 3.3V pin and GND. The circuit layout for this system can be seen on *Figure A3-1.* The wireless communication system receives vocalizations made into the microphone and processes them on the Arduino. The Arduino then transmits this information via bluetooth to an application built for this test, which then displays text on the screen. The application then sends back an appropriate message based on the information received from the Arduino. To test that the microphone was receiving valid inputs, we qualified this information by integrating speech-to-text conversion using the uSpeech library written for the Arduino. Integrating this library made it possible for our team to test that this wireless communication set-up works because what we vocalize into the microphone would appear on the Serial monitor of the Arduino, and the screen of the phone application. In order to test the latency of this module and ensure that is was optimal to use in our product, we included a timing circuit within the Arduino code. Latency was measured from the time an input was received in the microphone to the time an LED lit up based on the response received from the phone application. The total latency for this module, on average of 10 trials, was 1 second, versus the Pred Smart Cube, which had a latency of 2.5 seconds. A graph of these latency measurements is shown in *Figure A3-2.* With the Arduino based bluetooth module, our team has successfully created a preliminary module capable of two-way bluetooth communication that can be integrated into our project.

Software

On the software side we were able to accomplish a majority of the requirements we set out to complete for this year and laid out the groundwork for what the user interface will both look and function like. The work consisted of developing an Android application from the ground up that would integrate both the robot's API as well as speech recognition API to test whether the bluetooth module was functional for our project goals.

The application was developed using Java in the Android Studio environment and uses Gradle, which is an open source build automation system. The application uses Java for the functionality of the application, and XML for the view or display. There are two pages made up of the home page, and the robot driving page. The app can be built using Gradle, and uploaded to local Android phones for demoing purposes. The home page uses the MiP library to automatically detect any MiP robots available in the bluetooth connectivity range. It uses the MiP library authentication to connect to the phone without requiring the user to select the robot and input a PIN number. To prompt the robot to carry out functionality, the library sends a command to the sensors inside MiP. On the homepage, there is a button called "Start Listening" that begins the entire listening pipeline. First an audio file is played to the speaker that prompts the child to say a particular word. For instance, this could be "Say the word table." Following the prompt, the application starts listening for the child's attempt. If the child correctly says the word table, it sends a message to the robot to celebrate. If the child does not say the correct word, the robot does not provide any feedback. The application uses speech to text API to listen to the user, and also changes the text on the screen to display what the user has said. If the API does not detect a recognizable word, it simply prompts the user to try again. The application employs speech recognition through the Google Speech Recognition component built into the Android phone. The second page, which is the drive page, allows the user to move the robot by clicking on the screen, and dragging their finger in the direction of movement.

In addition to the application development, we made significant progress in finding an algorithm that would take the vocalizations of children and return a similarity score that would be used to regulate the rewarding structure from the robot. We found and contacted a language learning company that developed speech recognition software with the ability to score speech and pinpoint individual syllable and phoneme level mistakes in real time. By using this API in standalone tests we were able to confirm that it can understand both intelligible and unintelligible speech and provide an in depth analysis identifying where mistakes have been made in pronunciation. We performed multiple latency tests to ensure that this method of attaining a similarity score would meet our needs for fast, near real-time processing on a speech recognition level. Over the course of ten trials the response rate for analysis done through the terminal was a consistent 10ms. Leveraging this tool within the android application will be the next step for software development.

Deliverable Testing

During deliverable testing this semester, we were able to successfully demo the full communication pipeline between the robot, the application and the bluetooth module (with one way communication). In addition we were able to demo the functioning Arduino circuit that transmits recognized speech to an application and would be used to replace the bluetooth module if two way communication cannot be accomplished.

The communication pipeline using the PRED Smart Cube consisted of launching the Android application, which automatically connected to the MiP robot in range, followed by a user pressing the “Start Listening” button which would send a signal to the PRED Smart Cube to play an audio file consisting of “Say the word ___”. After this recording was played, the Google speech recognition window would pop up on the phone and wait for user vocalizations. Once the user spoke whatever was heard would be compared to the model word and the robot would either perform a little spin or say “I love you”. The action was only performed if the vocalization matched the model word, otherwise no action was performed.

The communication pipeline for the Arduino that was demonstrated during our first deliverables testing was one-way bluetooth communication. In the test, we were able to speak into the microphone, process the input, qualify the microphone input by converting it into a phoneme (i.e. if ‘e’ was said into the microphone, ‘e’ would display on the Serial monitor when it was sent), and send this phoneme via bluetooth to the phone application. The phone application then displayed the received information from the Arduino. This test was successful because we were able to measure that our microphone was capable of receiving appropriate inputs and the system was capable of processing these inputs and sending them via bluetooth to an application. Data from the phonemes sent to the Arduino vs. phonemes received on the application was collected. After analyzing the data collected, 100% of the phonemes sent from the Arduino were received by the application. By proving that this pipeline worked we were able create a proof-of-concept design that the child would be able to speak into the microphone and the application would receive it.

5.0 Technical Plan

Task 1 – Integrate SpeechAce with Application

We must migrate SpeechAce's service from testing on our own machines to being implemented and utilized in our Android application. Considerations include exactly which method of calling the service through our application – the API offers language support for Node.js, CURL requests, jQuery, PHP, etc.

Lead/Co-leads: Laura Salinas, Assisting: Steven Graham

Task 2 – Implement Sound File Conversions on Arduino Module

Since SpeechAce takes in sound files as inputs, the Arduino module must be able to either create sound files to send to the Android application, or send information in such a way that the application can create a temporary sound file that can be used as input for SpeechAce. The design will be built upon the existing Arduino module and will be tested by playing the received or created sound file on the application.

Lead/Co-leads: Shivani Bhatia, Assisting: Steven Graham

Task 3 – Get SpeechAce Feedback Report

Similar to testing on our own machines, the detailed breakdown that the SpeechAce API returns must be manipulated by our Android application. This is simply the start to the manipulation, and entails successfully acquiring key elements from the breakdown (quality score, phoneme/syllable mistakes).

Lead/Co-leads: Steven Graham, Assisting: Laura Salinas

Task 4 – Integrate Database to Store Results

Upon acquiring key elements from SpeechAce's API breakdown, we shall leverage a database service that will store these key elements.

Lead/Co-leads: Shivani Bhatia, Assisting: Arley Trujillo

Task 5 – Create Model for Data Storage

The model for data storage that shall be created requires considerations of its specific instance. The data model will either be logical – in which we detail semantics as represented by our manipulation of the key elements of SpeechAce's API breakdown in the form of descriptions of tables, columns, as well as object oriented classes – or physical, wherein we would consider the physical means by which data is stored (partitions, CPUs, tablespaces).

Lead/Co-leads: Arley Trujillo and Priya Kapadia

Task 6 – Create an Interface from the Database

We will abstract the database layers, forming an interface that bridges the gap between our Android application and the database itself. This task is entirely dependent upon our investigation we conduct in Task 3 – wherein we incorporate an API and its libraries to unify SpeechAce's breakdowns associated with a child's vocalizations to ultimately be provided for monitoring.

Lead/Co-leads: Arley Trujillo and Priya Kapadia

Task 7 – Implement Data Management for Child, Word, Level Schema

Once the interface is established, we will then implement management and treatment of the data being processed in real-time in order to captivate the child during language therapy sessions. Levelling and scoring schema are to drive varying interactions as a function of key elements found in SpeechAce's breakdown.

Lead/Co-leads: Priya Kapadia and Arley Trujillo

Task 8 – Provide Parent Tabular Metrics of Progress from Data

We will measure and provide performance metrics in tabular form that will provide both overall and lower-level, specific insights (for parents and professionals) regarding the child's progress in sessions.

Lead/Co-leads: Priya Kapadia and Arley Trujillo

Task 9 – Output Graph from Data Metrics

Similar to Task 7, we shall visualize data for parents and professionals. Data visualization in the form of candlestick charts, histograms, and other various mediums like pie charts are up for consideration such that we may provide an even-bigger picture of progress from many sessions.

Lead/Co-leads: Priya Kapadia, Arley Trujillo, and Steven Graham

Task 10 – Polish Android Application for Deployment

We shall both debug our Android application and optimize its user-friendliness/intuitiveness for parents and professionals alike. We will run multiple tests on our Android application to make sure we tackle corner cases and remove any potential bugs.

Lead/Co-leads: Priya Kapadia and Arley Trujillo

Task 11 – Create Mechanical Structure to Mount Mic/Speaker

We shall secure the microphone/speaker module to the language therapy robot. Considerations for this task include child safety (childproofing, designed to prevent tampering or opening), as well as location optimization with respect to the robot's center of gravity as well as high quality audio input and output.

Lead/Co-leads: Laura Salinas, Assisting: Steven Graham and Shivani Bhatia

Task 12 – Complete Testing of Application and Robot Package

It is in this task in which we begin convening with Dr. Helen Tager-Flusberg and her children at BU's Center for Autism Research Excellence to test the application and robot. We will gain insights on our performance here, especially latency.

Lead/Co-leads: Entire team

6.0 Budget Estimate

Part	Description	Quantity	Cost
PRED Smart Cube	Bluetooth microphone and speaker smart cube	1	$14.99
MiP Robot	Toy robot	1	$56.00
CHiP Robot	Robot dog	1	$159.99
Arduino Uno	Microcontroller Board	1	$21.99
MiP Pro-Mini Board	Would replace Arduino as microcontroller board	1	$39.99
Adafruit Bluefruit SPI Friend	Bluetooth device to transmit data	1	$17.50
Adafruit Electret Microphone Amplifier	Takes vocalizations as input	1	$6.95

It should be noted that the Pred Smart Cube, MiP Robot, and CHiP Robot have all been paid for by our client. The main items in our budget are the robots. Our project revolves around the MiP and CHiP robot, which will interact with the child. It should also be noted that only either the Pred Smart Cube or the Arduino Uno, MiP Pro-Mini Board, Adafruit Bluefruit SPI Friend, and Adafruit Electret Microphone Amplifier will be included in our final budget, depending on the bluetooth module that we choose to use. The cost for the MiP Pro-Mini Board is currently a quote from Advanced Circuits, as we would have to custom order the PCB since SparkFun has discontinued production of the board. Additionally, Arduino Uno, Adafruit Bluefruit SPI Friend, and Adafruit Electret Microphone Amplifier are being donated temporarily to our project by our group members and do not need to be accounted for as a true expense.

7.0 Attachments

7.1 Appendix 1 – Engineering Requirements

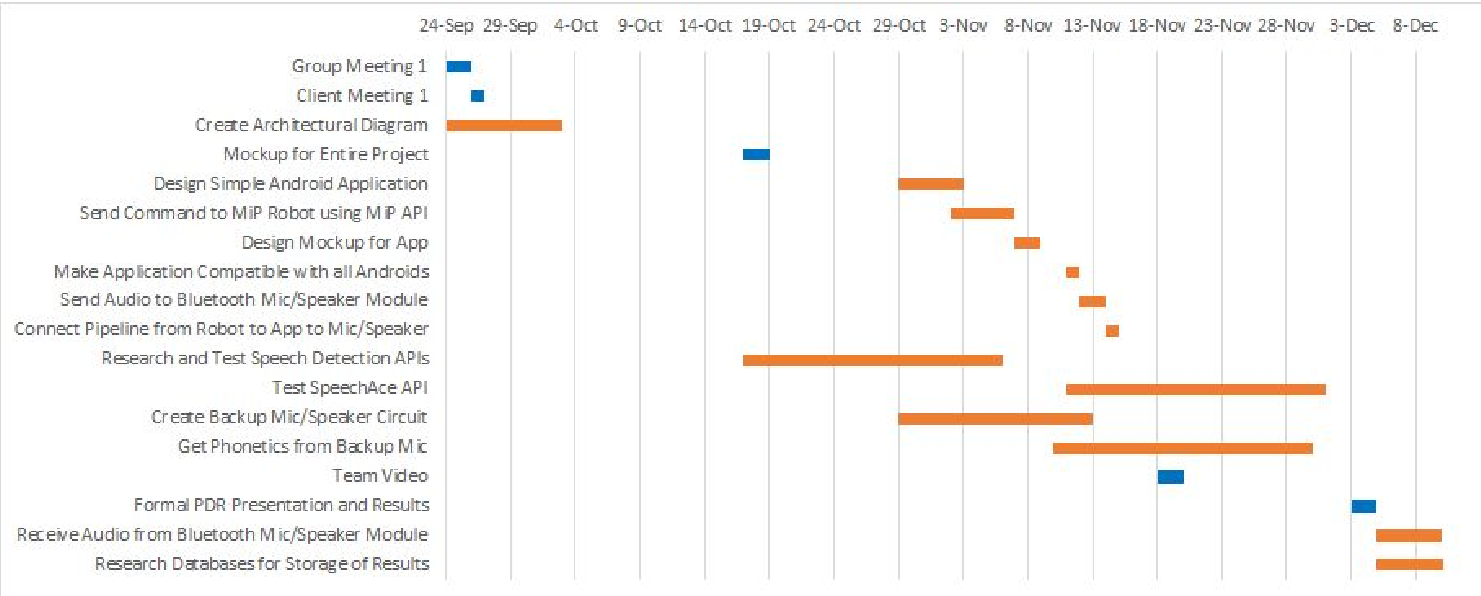
Team 14 Team Name: Vobot

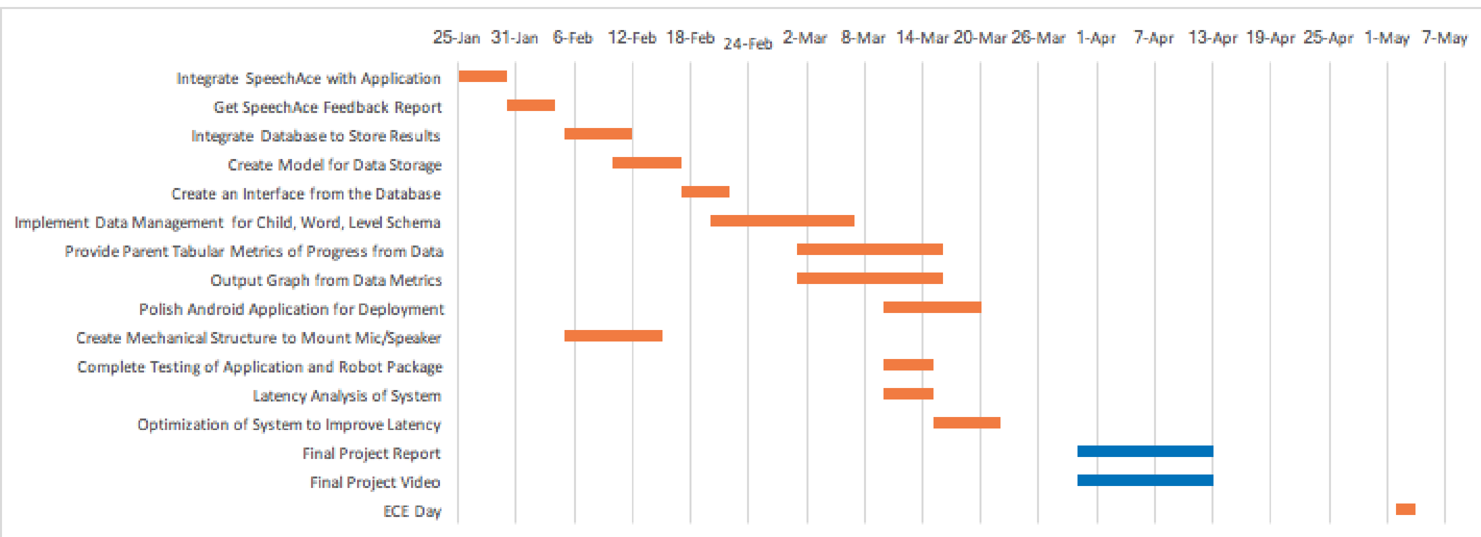
Project Name: Vobot

The following requirements have been specified by our client:

Requirement	**Value, range, tolerance, units**
Weight	< 3 lbs
Networking/Connectivity	Wifi on 802.11ac, BLE
Recording Frequency	16KHz
Recording Duration	< 20 sec
Recording Format	.wav, .mp3
Response Time (speech recognition)	< 1 sec
Response Time (child reward)	< 3 sec
Similarity Score	Measured between 0-100%

7.2 Appendix 2 – Gantt Chart





7.3 Appendix 3 – Figures

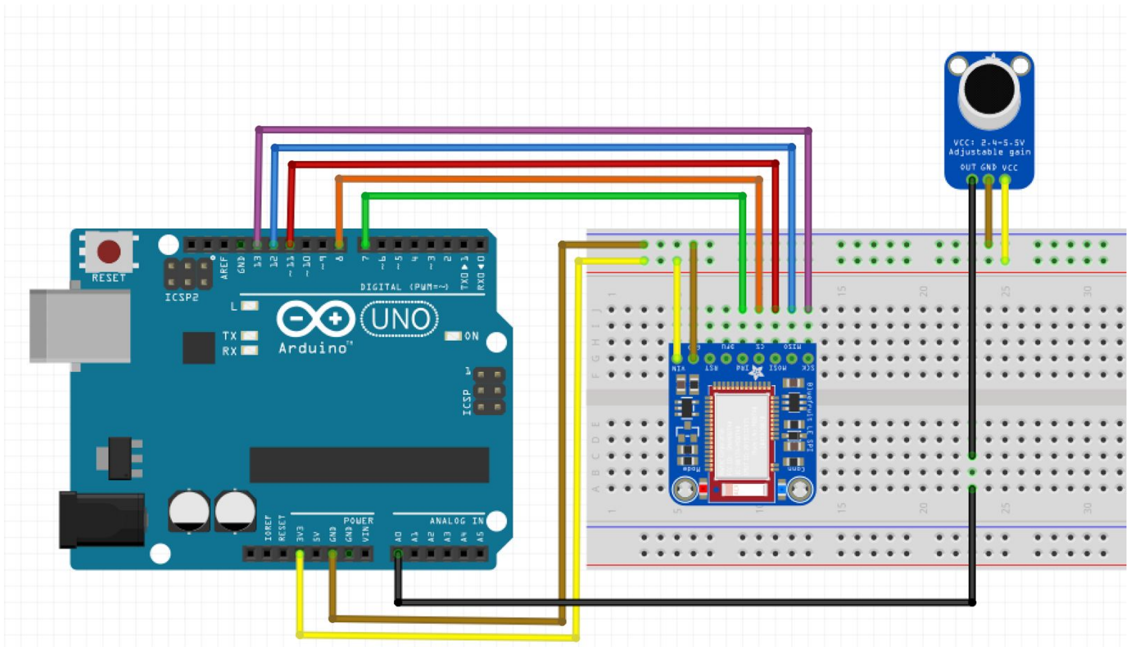


Figure A3-1: Arduino-Based Bluetooth Communication Module Circuitry

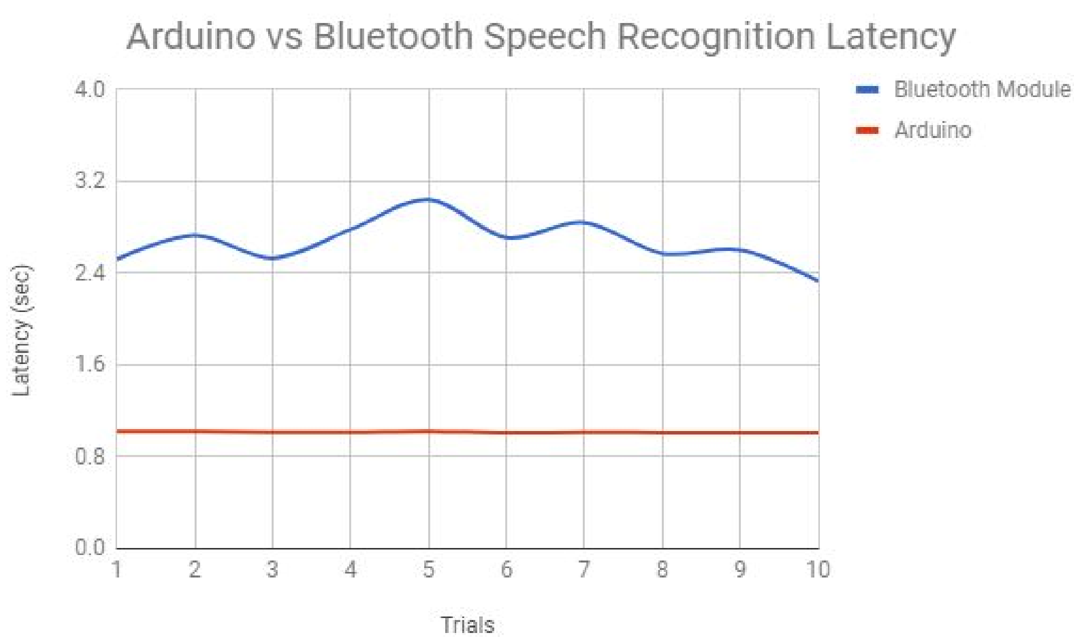


Figure A3-2: Latency Comparison between Pred Smart Cube and Arduino

7.4 Appendix 4 - Team Information Sheet

Name	Phone Number	Email Address	Role	Description
Laura Salinas	(305) 904-2582	laums@bu.edu	Similarity Score Algorithm Co-Lead, 3D Printed Enclosure Lead	Computer Engineering, '18
Shivani Bhatia	(267) 885-8880	scbhatia@bu.edu	Arduino Bluetooth Lead, Database Lead	Computer Engineering, '18
Steven Graham	(860) 387-8813	smgraham@bu.edu	Similarity Score Algorithm Co-Lead, Data Visualization Co-Lead, Enclosure Co-Lead	Electrical Engineering, '18
Priya Kapadia	(551) 999-1648	priyak@bu.edu	Android App Co-Lead, integrating with Similarity Score Algorithm and Databasing	Computer Engineering, '18
Arley Trujillo	(786) 715-1355	arley78@bu.edu	Android App Development Co-Lead	Computer Engineering, '18

7.5 Appendix 5 - Code Repository

Our repository for the Android Application can be found here:
https://github.com/priyakapadia/Vobot