The Bracelet: An American Sign Language (ASL) Interpreting Wearable Device

Sam Aba

Ahmadre Darrisaw

Thomas Leonard

Pei Lin

Executive Summary

The goal of this project is to design a wearable device that can interpret ASL signs in real time. The device will be designed around an impedance-to-frequency converter, designed by MLAB at the University of Tennessee - Knoxville, that will be able to measure and translate the impedance of electrodes on the user's skin into a digital signal. These signals will be wirelessly transmitted to an accompanying app where it can be classified using a machine learning (ML) algorithm. Visual output of the signals will be made through mobile devices that have a bluetooth connection. The major engineering characteristics of the project are as follows: the ML classifier accuracy, the battery life of the system, the weight/comfort of the final design, the system latency, i.e. the time it takes to classify a word, and the communication range.

For our final design, we chose to use an Arduino Nano BLE for the microcontroller, utilizing Bluetooth Low Energy for the communication protocol, for flexibility in the design. A support vector machine is chosen as the ML model due to its workability in high-dimensional feature spaces. The design will utilize a bracelet as the framework of the measurement system to remain unobtrusive to the user and ultimately be forwards-compatible with MLAB's existing impedance to frequency converter.

Table of Contents

Executive Summary

Table of Contents

List of Figures

List of Tables

Problem Definition & Background

Requirements Specification

Technical Approach

Design Concepts, Evaluation & Selection

- Chassis Design Concept
- Microcontroller Design Concept
- ML Design Concept
- Conclusion

Embodiment Design

Test Plan

Project Deliverables

Project Management

Budget

References

List of Figures

Figure 1: Technical Approach Flowchart

Figure 2: Potential Chassis Designs -- Wristband (left) v.s. Glove (right)

Figure 3: Potential Microcontrollers

Figure 4: Machine Learning Models Accuracies

Figure 5: Impedance-to-Frequency Converter (Schematic View)

Figure 6: Prototype Bracelet

List of Tables

Table 1: Customer Requirements

Table 2: Engineering Characteristics

Table 3: Weighted Decision Matrix -- Chassis

Table 4: Weighted Decision Matrix -- Microcontroller

Table 5: Machine Learning Design Concept Weighted Matrix

Table 6: Project Deliverables

Table 7: Project Budget

Problem Definition & Background

Our project sponsor, Dr. McFarlane, has an on-chip impedance to frequency converter originally used to gather biomedical data, which could be used in cellular-level detection and diagnosis. Due to the chip's ultra-low power consumption and miniscule footprint, we believe that it has applications to fields outside of biomedicine. The problem that we have identified is that, at the moment, there exists a lack of communication between those that are hearing-impaired or mute and those who are not.

One solution to such a problem would be that everyone that interacts with the mute or hearing-impaired learn ASL; however, this is not feasible because learning a new language is difficult and not a priority for some. Instead, we are designing a wearable device that can be placed on the hands or wrist of the hearing-impaired. Such devices already exist, but they have many problems associated with them. For example, those based on image processing require good lighting and are sensitive to motion [2]. Others employ sensors with limited sampling speeds and bandwidth, which limits the amount of data that can be used to predict the hand's position. Most importantly, most systems have long, complex signal paths, which opens them up to problems of noise, power consumption, and size. Some tools on the market already can unintentionally be used to help our customers, like Google Translate, which allows for the mute or hearing impaired to type their thoughts and have them read aloud. Unfortunately, this solution does not facilitate real-time conversations.

Before we begin developing such a device, we must first address the background information required to address the problem. First, we need an understanding of sign language, specifically ASL. We need to be able to install a good variety of signs and prioritize the ones that are most often used in the language. Second would be the information related to hardware. The project is based on a chip used for impedance-to-frequency conversion, so an understanding of low noise, low power systems would be useful in accurately obtaining the sensor data, especially in a system that won't be significantly bandwidth limited. Finally, we need experience in app development to support our platform and in machine learning for our predictions.

The device will work by attaching electrodes to the user's forearm. The impedance between the electrodes will be altered when muscles move. This movement change will be translated by the impedance-to-frequency converter, converted to a digital signal, and then communicated to the our backend, wherein the inputs will go through preprocessing, feature extraction, and then be fed into a linear classifier. The algorithm will interpret the signed input and produce its corresponding text/audio.

The bracelet will be used in either a casual (daily conversation) or business (i.e. taking the times in a court procession) setting. In a casual setting, the idea is to have the bracelet read off words in real time when they are signed; input will need to be properly paced for this to be possible. A business setting's priority is recording input so the bracelet would store data and be able to take input at faster speeds. It would be carried by the user so the goal is to make it fairly pocketable and have an 8 hour battery life minimum for continuous usage on a single charge.

Disclaimer: The Bracelet does not aim to discourage people from learning ASL, but rather it is built to ease day to day communication. ASL, like any other language, has a deep history. Communicating with people who are verbally impaired in ASL can be exceptionally meaningful, something that technology simply cannot convey.

Requirements Specification

The requirement for the device is that at the end of the spring semester we have a working prototype to show off. Although hand gesture recognition can be applied to languages other than ASL and even other disciplines entirely, we have been asked to focus on ASL because of our project sponsor's proximity to the language -- one of her research assistants. Because we have not been given the MLAB chip directly (cost and quantity limitations), we must instead implement an impedance-to-frequency converter that functions identically to that chip. In this way, Dr. McFarlane will be able to simply swap her chip out for our circuitry without any complex modifications. To summarize, our implementation of an impedance-to-frequency converter must be a drag-and-drop replacement for Dr. McFarlane's own hardware.

Because we need to implement the impedance-to-frequency converter using discrete components, we will have an exponentially larger footprint than what would be the case if we had the MLAB chip. We have therefore not been given a size requirement for this prototype; it just would not make sense to have one given the sheer difference in component count after Dr. McFarlane swaps in her chip. On the other hand, we recognize that the prototype should still demonstrate direct applicability to the market if possible, and so we have set a soft, subjective limit on the size of the board ourselves: it should be small enough to sit strapped on the forearm or hand and not present excessive discomfort in the user.

Dr. McFarlane's lab focuses on ultra-low power integrated circuits, especially those for biosensing where losing power could mean the difference between life and death. It stands to reason that she would set a strict limit on power consumption, and she has. The PCB should be able to run for 8 hours, uninterrupted. There are two immediate approaches to this problem. The simplest would be to simply buy a larger battery. A larger battery with more capacity could easily support all of the circuitry on the board for hours at time without much effort. Unfortunately, larger battery capacity is directly correlated to the size of the battery, and although we have not been given a stringent size requirement, when thinking of user experience, it makes sense to use a smaller battery and attempt to minimize power consumption instead to achieve the required 8 hour battery life.

For this type of device to be practical, it needs to be able to communicate wirelessly with software on our backend. We have not been given a preferred communication protocol, so at the moment we are planning to use a microcontroller and bluetooth module. To accommodate the battery life and size requirements, it may make sense to swap to a different type of communication that does not utilize a microcontroller.

The final bracelet will need to be portable and comfortable to wear on a daily basis. The device is intended for long-term use and should therefore be lightweight and easy to carry. It should not pose any discomfort even under extreme weather conditions. Likewise, the device should be able to withstand a large variety of temperature ranges. Standard commercial range is 0 to 70 degrees Celsius, although for this prototype it is sufficient that the device works from 20 to 30 degrees Celsius.

All such requirements are summarized in Table 1 below.

Table 1: Customer Requirements

Priority	Requirement	Description
1	ML Algorithm	A classifier must be developed to predict ASL signs.
2	Functioning PCB Design	The MCU and all peripherals should be integrated on a single PCB.
3	MLAB Chip Compatibility	The design should be forwards-compatible (drag/drop replace) for Dr. McFarlane's chip solution
4	Battery Life	Battery life should exceed 8 hours
5	Wireless Communication	The MCU should communicate wirelessly to backend for input classification

The engineering characteristics were developed as a result of the customer requirements, and in a sense reflect them. The classifier accuracy is a measure of how well our system performs. If the classifier was randomly guessing at, for example, letters of the alphabet, it would have an accuracy of 1/26= 3.85%. 80% felt like a high enough score to prove the merit of our design. The battery life was stripped directly from our customer requirements and is again a minimum of 8 hours. System latency and communication range were chosen to be comparable to existing wearable ASL devices, while the weight was chosen as a good measure of the comfort of the design for the user. As a whole, the design targets are fairly lax. This is on purpose given the amount of steps this specific project involves and the team's limited manpower/timeline. The Engineering Characteristics for the summarized are listed in Table 2 below.

Table 2: Engineering Characteristics

Engineering Characteristics	Units	Design Target	Variable/Constraints
ML Classifier Accuracy	%	≥80%	Variable
Battery Life	hrs	≥ 8 hrs	Constraint
Weight	lbs.	≤ 1 lb.	Constraint
System Latency	sec	≤ 1 sec	Variable
Communication Range	meters	≥ 0.3 m	Variable

Technical Approach

We split the project into three main subsections: research, hardware, and software. First we research the scope of the sign language letters we will be implementing. We would like to include the whole alphabet but have agreed that it would be best to start small with a few letters then work onwards. Full words are goals for future ambitions as well but since they require tracking of the hand position, we are unsure of that as of now. We will get a set of sign letters we can all agree is a fair amount and are far enough apart in terms of form to not confuse our machine learning algorithm. We scarcely know any American sign language so we will have to practice over and over until we can comfortably and accurately pull them off in quick succession. Eventually, we would like to onboard someone with signing experience to give us the most accurate data for our sets.

For the hardware portion, to ensure forwards-compatibility with MLAB's impedance-to-frequency converter, the design takes great inspiration from theirs. Discrete components were chosen to mimic the operation of the chip and subsequently ordered from Digikey and Amazon. Using the components a prototype was constructed to be able to convert arbitrary resistances and capacitances to pulse-width modulated outputs. This was integrated with the microcontroller which needs to be able to control multiplexed inputs to the system, interpret pulse width modulated outputs, and transmit the data to a computer wirelessly. With all of this completed the PCB can be ordered and the electronics portion of the design is considered complete. The final part of the hardware section involves connecting actual electrodes to the multiplexer of the board so that hand signals specifically can be interpreted and mounting the entire contraption on a wristband. The hardware design plays a major role in the communication range and battery life (EC 2 & 5) of our design. The former is dependent on the wireless communication protocol chosen and the latter by the quiescent draw of the chosen components. It will also affect the weight (EC 3) by virtue of how many components are needed and the microcontroller chosen for the design.

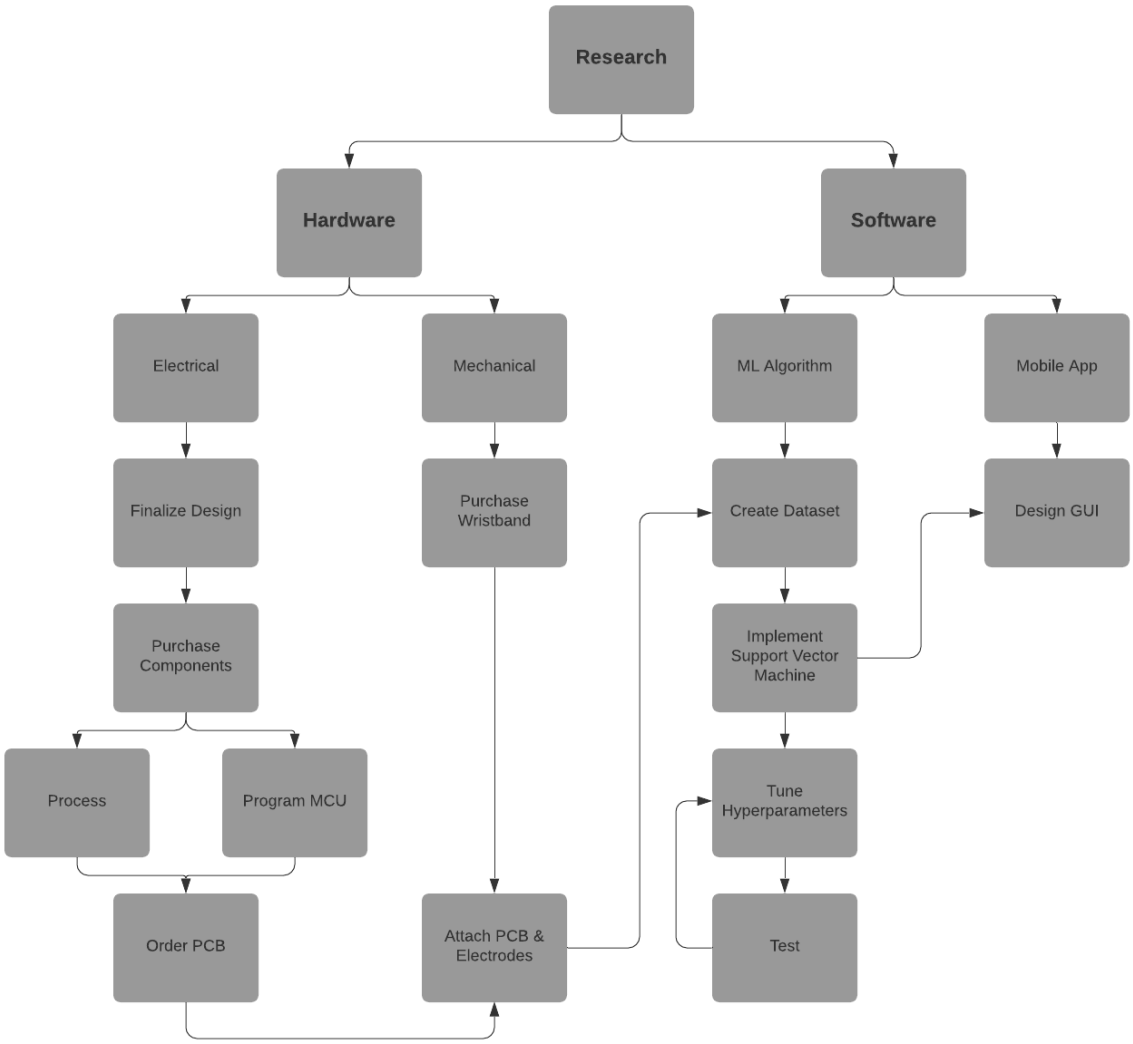
While the hardware is being built we will be working on an app for user interface and the machine learning algorithm that will be classifying our data. The app will be the main point of interaction for the user, and it will connect the algorithm to the hardware. It will be built in Android Studio, which has an emulator function for ease of testing. As of now, the plan is to only have the app available for Android users. Future work can be done to create a version for macOS. There are many voice libraries available online that we will choose from. Our main goal is ease of use, so the design will be simple. The app is where communication range comes into play. It will use bluetooth to interact with the bracelet and then send the data to a remote terminal to be classified by the algorithm then returned. We don't know the max range of our bluetooth connection yet but we would like it to be comfortable to use.

The remote algorithm in question will first take in the original dataset obtained through the wristband in csv format then normalize it to increase accuracy and efficiency. The data will then be trained and split into two different X and Y datasets, one for predictions of accuracy and another for use in our classifications. Before we input our data into the support vector machine we will run a grid search on it to find the best possible parameters for our data set and maximize accuracy. Once we have the best parameters we will initialize the model and obtain our classifications. The hyperplanes will be saved and once a new data point is recorded from the app it will see on which sides of which hyperplanes it falls then return its classification back to the app. This format should minimize latency but we will have to

have a system in place to figure out when to begin recording, when to end recording, and an average time in between letters/words for continuous recording.

A diagram outlining the technical approach is given below.

Figure 1: Technical Approach Flowchart



Design Concepts, Evaluation & Selection

Chassis Design Concept:

The chassis was a major concern for us because it almost solely accounted for EC #3, weight. We needed something that would be lightweight yet robust enough to withstand everyday usage. After researching existing solutions, we narrowed our choice of chassis down to either a glove or a bracelet. In either case we would use electrodes attached to the user's skin to measure the change in impedance when a different hand sign is performed. One benefit of the glove is that it's placement is relatively constant among users, i.e. no matter who is using or how large their arm is, the electrodes would almost always be located in a similar place. This allows collect consistent measurements across the broad range user's, which in turn boost classification accuracy. Moreover, the glove is more impervious to the user's movements, since it is much more easily fixed to the hand. The wristband, however, has many more degrees of freedom in terms of instantaneous position. Excessive movement could cause it to shift up or down the arm. It even has the potential to rotate a full 360 degrees around the forearm. These additional variables make the wristband much more difficult to design for and have the potential to reduce EC #1, classification accuracy. The final benefit of the glove is that it allows for a larger dynamic range in the measurements. The difference in impedance between your fingers is likely to be much larger than the difference between the muscles in your forearm for the simple reason that they move much farther. The increased range of potential measurements could help to separate each sign within the n-dimensional feature space that our ML classifier is operating in. This increased separation can make it easier for our classifier to make decisions and thus improve accuracy.

On the other hand, the wristband is also much less intrusive to the user. Because it lies on the forearm, it would be far less likely to interfere with everyday tasks. This in and of itself may seem small compared to the large number improvements the glove provides in terms of accuracy, but it must be noted that our customer requirements were that we had a working classifier, not that it would perform perfectly. In terms of importance to our sponsor, user experience was noted to be more important than an arbitrarily high accuracy. Because of this we chose to use the wristband as our chassis.

Example images of the aforementioned glove and wristband are shown in Figure 2, while Table 3 is a weighted decision matrix used to make the final choice. The weights are on a scale from 0 to 3 based on customer requirements.

Figure 2: Potential Chassis Designs -- Wristband (left) v.s. Glove (right)



Table 3: Weighted Decision Matrix -- Chassis

Criteria	Weighting	Glove Score	Bracelet Score
Comfort	2	0	2
Measurement Consistency	1	1	0
Dynamic Range	0.5	0.5	0
Total		1.5	2

Microcontroller Design Concept:

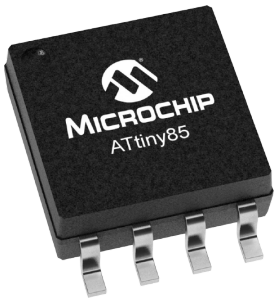
We had a lot of freedom with our choice of microcontroller. The microcontroller is the brain of our system. It controls when to sample data from each pair of electrodes, it converts the pulse-width modulated output of the impedance-to-frequency converter to an interpretable floating point number, and it communicates with a bluetooth module to send the information to our backend. Its importance demonstrates that it has a hand in every engineering characteristic. Its clock speed and processing power directly affect the system latency and power consumption, its size affects the weight and structure of the chassis, its measurement accuracy/consistency (in our case the speed of its response time of its timing interrupts and jitter of its clock) affects the ML classifier's accuracy.

For this extremely important design decision, the two options that we settled on were the Arduino Nano BLE and an ATtiny85 with an HC05 module for Bluetooth connectivity. The ATtiny85 was the original choice for our design. It was small and had an ultra-low power consumption. It also had an operating voltage down to 1.8 V! This meant that we would be able to power it off a single coin cell battery, which would further reduce its effective PCB footprint. The ATtiny85 was also remarkably cheap -- less than $3.00 at the time of writing. This was important to us because of the current chip shortage. The ATtiny is essentially a legacy part with an enormous backlog on numerous sites which has kept the price low, unlike more recent and popular MCUs and chips that are increasingly bought out by the automotive industry and other large corporations. The ATtiny is relatively slow, but the latency that it

would introduce was estimated to be on the order of tens of microseconds and thus imperceptible to the human ear. Overall, when paired with the HC05 serial-to-bluetooth module, the ATtiny85 would have made a great choice for our design.

The strength of the Arduino Nano BLE (Nano), however, was two-fold. First, the Nano is a development board that has several additional peripherals included, e.g. accelerometer, gyrometer, etc. This was appealing because at the early stages of the design, we are currently unsure if the impedance measurements alone are enough to establish accuracy within our ML classifier. Using these additional onboard devices, we would be able to add and subtract additional features to our measurements without being forced to redo the design from scratch. The Nano also had the benefit of an IDE and an expansive network of open-source code, which would allow us to quickly program the device and focus more on the high-level software aspects that are off-board. In terms of engineering characteristics, the Nano is objectively faster than the ATtiny and would improve classifier consistency because of its quick timing interrupts, but the additional peripherals and its need for two coin cell batteries to power it make it much larger and unwieldy, which could reduce comfort. Its increased power consumption would also reduce the battery life. In the end we chose the Nano because of its increased performance and chose to sacrifice the power consumption and size. This tradeoff is further explained in Table 4, where the weights are between 0-3 based on the importance to the group. Figure 3 below shows images of both the Nano and the ATtiny85.

Figure 3: Potential Microcontrollers -- ATtiny85 (left) v.s. Arduino Nano BLE (right)



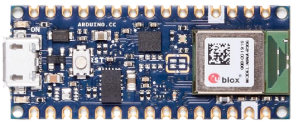


Table 4: Weighted Decision Matrix -- Microcontroller

Criteria	Weighting	ATtiny85	Arduino Nano BLE
Power Consumption	1	1	0
Board Footprint	1	1	0
Peripheral Integration	3	0	3
Total		2	3

ML Design Concept:

The machine learning design concept we went with was support vector machines. These draw hyperplanes to classify the data they're given. It excels at linear classification but can also perform efficient non-linear classification using the kernel trick (a class algorithm for pattern analysis). Two other concepts we considered were K-Nearest Neighbors and K-nearest means. K-nearest neighbors finds the distance between the current node and some number 'K' of its nearest neighbors then assigns it the most common class amongst its neighbors. K-nearest means, at least the version we were going to use is a modification on the nearest neighbor's algorithm. The idea was that it would still classify the data like before but it would find the averages of distances across multiple runs and take the mean of these as the value of the classification. Any new data would be assigned the class it was closest to. Since KNM (K-Nearest Means) stores the classes it is less memory intensive then KNN (K-Nearest Neighbor) but becomes more computationally intensive with more inputs. Computation eats up time so the most efficient one will be the quickest. This is the one of the main factors we chose SVMs (support vector machines). It is computationally more intensive then the other two at first to draw the hyperplanes then becomes really quick and efficient in classification. Since it doesn't have to remember each point's classification it is also good with memory.

To test accuracy we wrote an algorithm that used the general sk.learn definitions of the functions defined above. Since KNM is a modification on KNN it normally has to be manually implemented. We used the next closest algorithm as a stand in, K-means clustering. It does the later half of the algorithm by finding clusters of data, averaging them, then taking the mean of them as the classifications. The problem

with this algorithm is that it isn't really accurate unless you have the right of clusters. All of this to say that the K-means accuracy we got is just a place holder and would usually be slightly higher than KNN using the modified KNM version. Our code called each algorithm without parameters and operated on the same dataset for fairety and returned a 83 percent accuracy for KNN, 72 percent accuracy for K-means, and a 85 percent accuracy. Since we could not obtain an accuracy for KNM at this time we will substitute it out for K-means clustering, which has similar memory and computation cost, in our weighted matrix. Below is our terminal's output of the accuracy followed by the weighted matrix.

Figure 4: Machine Learning Models Accuracies

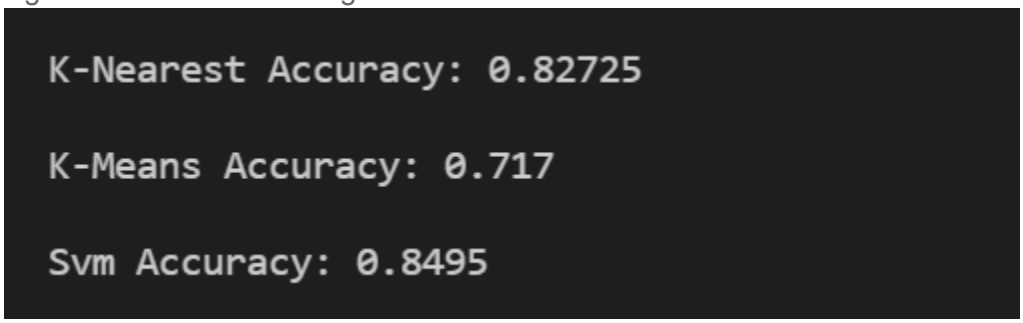


Table 5: Machine Learning Design Concept Weighted Matrix

Criteria	Weighting	K-Nearest Neighbors Score	Total	K-Means Clustering Score	Total	Support Vector Machines Score	Total
Memory	1	1	1	1	1	3	3
Computation	2	1	2	2	4	3	6
Accuracy	3	2	6	1	3	3	9
Total			9		8		18

As seen in our weighted matrix, support vector machines far out performs K-nearest neighbors and K-means clustering. K-nearest means would be slightly more accurate than KNN but since we can't prove that, it is ignored in our weighted matrix.

Conclusion:

After weighting customer requirements, engineering characteristics, and our own preferences we were able to make a final decision for all major aspects of our design. We chose to go with the Arduino

Nano BLE for a microcontroller for increased performance, ease of programming, and the flexibility to add additional types of measurements later down the line to improve classification accuracy. A wristband was chosen as the chassis so that the device would be unobtrusive to the user. Finally, support vector machines were chosen as the ML classifier because of their efficiency in memory usage/computation for large datasets and because of demonstrated increased performance on a test dataset similar to our own.

Embodiment Design

The Bracelet uses an impedance to frequency converter to measure the impedance of the user's arm as different gestures are being made. The impedance to frequency conversion circuitry consists of electrodes that apply a constant-amplitude alternating signal to the user's arm. The current that flows is converted to a voltage using a transimpedance amplifier. The amplifier's output is fed to a comparator which converts an input sinusoid to a pulsed waveform whose duty cycle is proportional to the magnitude of the input current. The phase of the input impedance is measured by measuring the phase shift between the input voltage and current which are also converted pulsed waveforms and fed into a digital type I phase detector.

To repeat, the signal chain is thus: Sinusoidal generator → Human Wrist → Transimpedance Amplifier → Comparator → MCU.

Figure 5: Impedance-to-Frequency Converter (Schematic View)

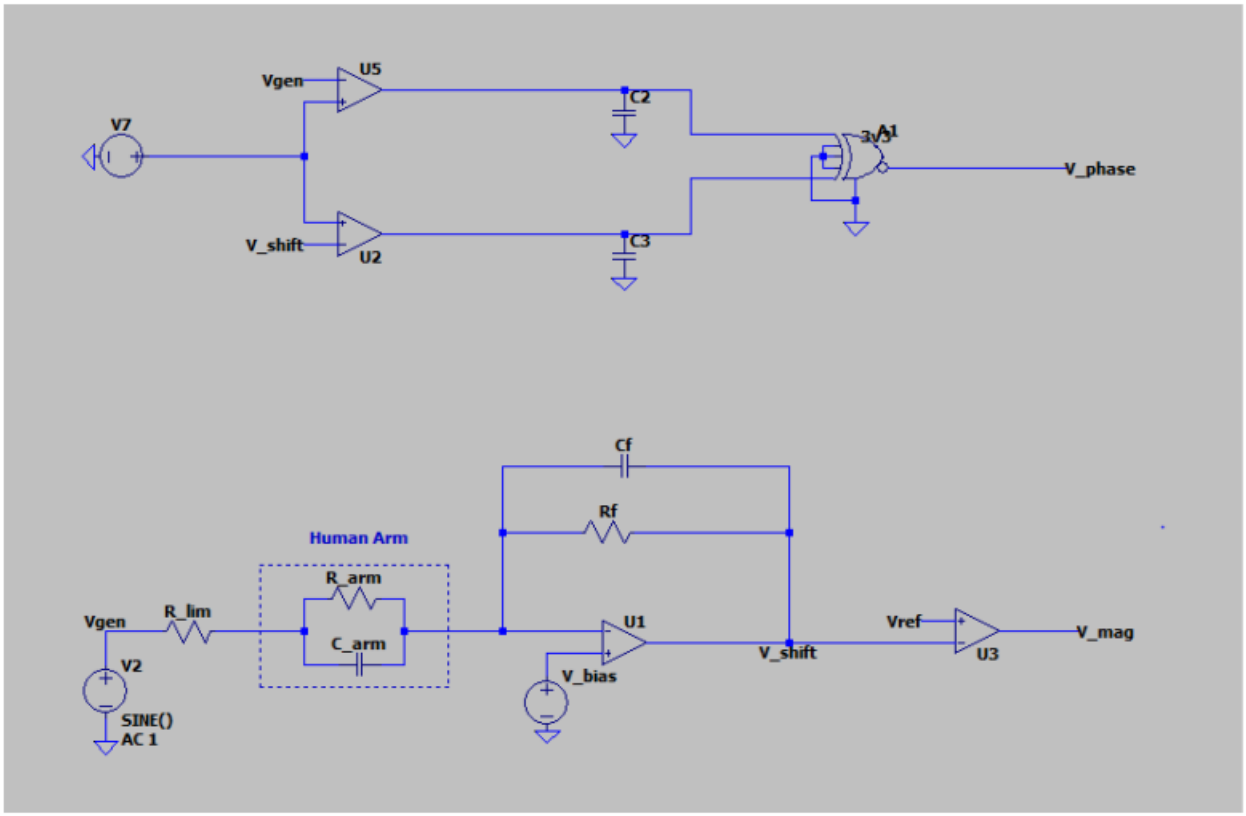
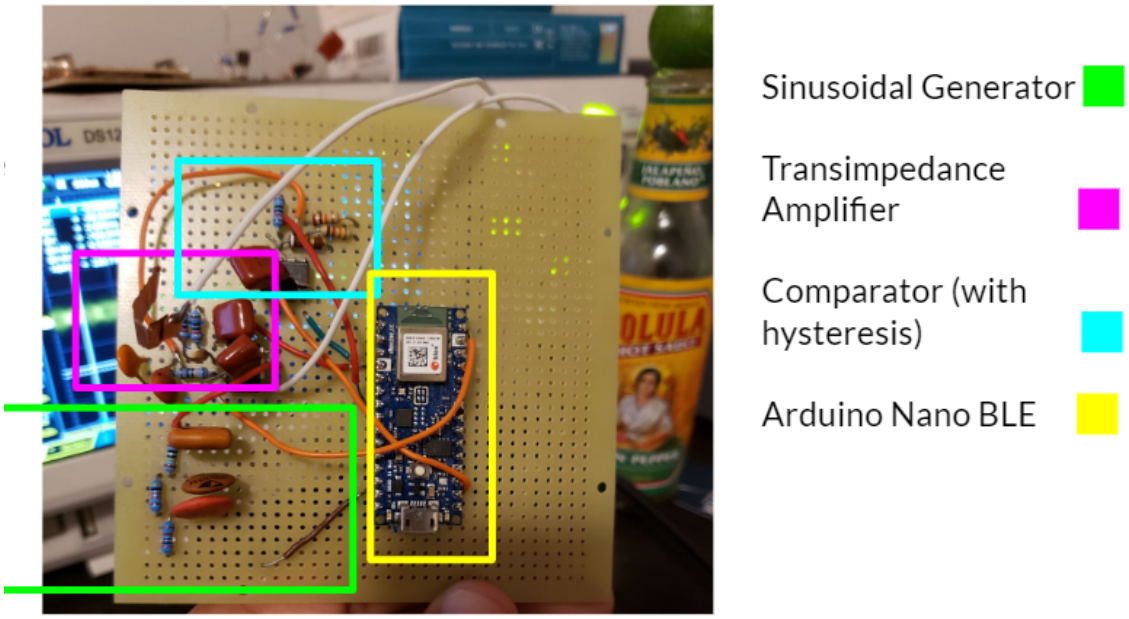


Figure 6: Prototype Bracelet



Again, the signal to be read are pulsed waveforms with varying duty cycles, the MCU reads the data by measuring these pulse widths, ideally using hardware interrupts. The downside to such a method is that interrupts may not trigger immediately, and so we are at the mercy of whatever process is currently being run on the MCU and the clock frequency. Luckily, for the Arduino Nano BLE, the latter is quite high, which allows for good resolution for these measurements.

The passive components chosen for the design were relatively inexpensive. 1% tolerance resistors were used throughout the design to minimize error due to variation in components. 1 uF bypass capacitors were added to the supply pins of all ICs to provide transient currents and assist in noise rejection. We chose ceramics for the other capacitors on the board for their low ESR. Unfortunately, we could not use NP0 or C0G type capacitors due to the values we needed being too large, so we settled for X7R. Just to note that the first two types have the best insensitivity to DC voltage and temperature.

The values for the passive components were chosen for varying reasons. The cascaded RC network shown in Figure 5 had its passives sized such that each successive resistor was ~20x larger than the previous. This helped to keep the low frequency response close to an ideal 3rd order Butterworth filter. The value of the first resistor, 2kΩ, was chosen to set the peak current through the network to be no more than 1.5 mA to minimize heat and power loss. The feedback resistor for the transimpedance amplifier was chosen in an attempt to accommodate the lowest input current signals, while not saturating on the large input amplitudes. A value of 100 kΩ was chosen as a good compromise. The feedback capacitor was then chosen to set the bandwidth of the circuit and attenuate noise using the formula f_cutoff = 1/(2*pi*Rf*Cf).

We chose to use an MC34072 operational amplifier for the transimpedance stage and to buffer the input sinusoide because of its acceptable gain-bandwidth and small cost. It was also a dual package (it contained two amplifiers within the package) which helped to reduce the board footprint. The best option for such an amplifier would likely be a FET input current feedback amplifier, for low input bias current and less bandwidth variation with source impedance, both of which result in smaller errors in current measurements. We chose not to go with such an amplifier due to cost.

The comparator used in the prototype was an LM393N, again used as a cost-cutting measure. The minimum qualifications for the comparator were that its rise time had to be significantly shorter than the input pulse train's fundamental frequency. For our prototype, we chose a signal frequency of 1kHz, which meant the LM393N could function quite well here.

This data can be paired with gyroscopic measurements recorded by The Bracelet. All of this is sent to the user-friendly application via a wireless communication protocol (e.g. Bluetooth Low Energy). The method that the Arduino uses to do so is by updating a Bluetooth Characteristic associated with its measurements every time a sample is taken. The characteristic can then be read by any paired devices as if the information were posted onto a billboard.

The app will work as a multi-operational portal from the machine learning algorithm to the hardware. In its ideal state, the app will initialize a user of the device and ask them for a gesture to be the start and stop symbol for reading in ASL. Once this gesture is set, the user will be asked to put the app in verbalization mode, where ASL is read aloud, or recording mode, where ASL strings are stored in a database for future access. If the app is set to verbalization mode and the user makes the start gesture, until the user makes this gesture again, their signing data will be sent to the machine learning algorithm. The machine learning algorithm will compare this data to the dataset it is trained on to return a classification. This classification is then sent back to the app to be read aloud. There will be a delay in data to be read aloud to ensure no chaining words are present that would change the meaning of the word given prior. If the user is set to recording mode, it will follow the same path, but when sent back to the app, it will be stored in notes with proper punctuation. At first the punctuation will be added in using a self coded method but eventually we'd like to make use of existing robust grammar correction libraries. There are two ways for the user app to interact with the machine learning algorithm. The first way would be to export a trained copy of it to the app and use it as a black box where data goes in, and classifications come out. The second way would be to make use of google docs. For this method, every time a new user is initialized, they would be assigned a row in a document. The algorithm would constantly be running on a server using threads, and when data is written to that user's row, it will be read, interpreted, and reported back for the app to take and act upon accordingly.

Test Plan

The machine learning algorithm and hardware need to be rigorously tested for accuracy and efficiency. Criteria for passing is fast response time and high accuracy. To ensure all measures are accounted for it must be tested on data taken from multiple people. On one amatuer signer data should be taken. For best accuracy at least 25 signs of each letter should be made. This is then imputted into the algortihm to be trained. The amatuer will then sign the letters out again and compare the output with the expected. Tweaks in the parameters of the algorithm will be made accordingly to increase the accuracy of what is signed. Once sufficient results are obtained the algorithm will then be tested on varying amatuer signers to

see how it holds up. The algorithm will be updated to take into consideration outliers and deadspace in signing. Unique and common ways to sign any letter will be added to the data set to be trained in. A professional signer will record their data to add to the training set and this process will be repeated to ensure accuracy on both ends of the signing spectrum. By default, the support vector machines response times are fast. Things that might slow them down are how they communicate with the hardware. At present, it is suggested to read from a live google docs. Google limits how many reads can be called in a second. To ensure quickest time, how many allowed reads in a second must be recorded and the code must be updated. To test this we will have the code run with a counter for the number of reads allowed until google disconnects our access. We will keep updating the number of reads performed until the code runs smoothly and access is never revoked.

Weight is an important factor to take into consideration when designing a bracelet. A lot of movement is required to sign words and if the device is too heavy it can cause the user to become fatigued quick. To test weight we suggest wearing the device for prolonged time on one wrist and see how comfortable the user feels at the end. If one arm is drastically more fatigued than the other, the weight of the device needs to be re-assessed.

Communication range is vital to knowing how the device will react with different unknown variables added in. If the device is disconnected due to a maximum range being reached, will it smoothly reconnect or cause hassle? To test this the user should try signing a set of symbols at different ranges from the bluetooth source and reviewing the accuracy and speed of data transfer. The communication protocol should be adjusted to meet our goal. Objects and weather also pose a threat to connectivity and communication range. The device should be tested in varying environments and weather conditions to see how well the devices connect over Bluetooth and react to these changes. The device should also be tested with different materials placed between it and the bluetooth source. To ensure consistency these tests should be combined after fully explored alone.

The battery life can be tested by recording how much energy the bracelet uses, how much energy the battery provides, and calculating the expected life time. Let the bracelet run until dead to see how accurately it reaches the expected time. If the expected time is low, modify the energy consumption or energy output during testing to approach expectations.

Project Deliverables

The project will result in a functional prototype at the end of the Spring 2022 semester. The discussion of the prototype will be divided in two. First, the hardware that will be presented at the end of the course should all be mounted upon a custom PCB. The sections of the board will consist of an impedance-to-frequency converter based on the chip designed in Dr. McFarlane's lab, a sine wave generator, a power supply (most likely a button-type battery), and circuitry to enable wireless communication. At the moment, it is planned to use a microcontroller and a bluetooth model to enable the wireless communication functionality; however, because adding an MCU to the board would be incredibly expensive in terms of price, footprint, and power consumption, we are currently looking into other methods of wireless communication.

The PCB should be treated as an evaluation board rather than a finished product, and should therefore contain test points to easily measure voltages and debug the system in case of malfunction or even total failure. For example, there will be vias or header placed at the outputs of the impedance-to-frequency converter, so that their outputs can be verified independently of the wireless communication circuitry.

The software deliverables will consist of code that is able to preprocess and interpret the data. Some of the functions that will need to be implemented are the Fast Fourier Transform and/or averaging measurements in the time domain. There may be some basic digital signal processing (low-pass or high-pass filtering) applied to the incoming data as well to save the cost and PCB space associated with using analog circuitry. This preprocessing code should be able to extract features and export them to the machine learning algorithm of our choice.

The machine learning algorithm used will be an example of supervised learning, meaning the training data fed to it will have class labels tied to them. The algorithm will then be able to read in the feature sets of incoming data and produce a prediction of the sign that was made. There will be no reinforcement learning aspect to our implementation. We can also provide the dataset used to train the machine learning model, which will be generated by our own hand movements.

As a part of the usability of the bracelet, we plan to design an application that gives audio output of the signs inputted. The bracelet will be connected to the app through Bluetooth. The user interface simply aims to be easy to use and functional. Users can decide on a masculine or feminine voice to voice their signs. Our main focus will be the bracelet and the machine learning components for the accuracy of detection of signs. This is definitely an area that can be improved in the future. Besides audio output, the main function we would like to design is a self-training interface where users can add words that are part of their personal vocabulary. This can become an extension of the word library that we already have. An open source library could be a great option to add word banks of different specialized topics, where other users, especially new users, can download for ease of use. This idea is derived from fingerprint recognition security of phones, where the fingerprint must be detected in multiple positions in order to get a scope of the fingerprint. With this feature, the app can adapt with ASL's changes.

Most importantly, our final product will come with a large amount of documentation so that the work can be understood by our project sponsor and potentially given to others to build upon. This documentation should consist of Gerber files, SPICE files, and hand calculations from the hardware team.

The software team will produce well-commented code as well as documentation describing the software architecture and the functionality of individual Python functions.

Finally, we will produce a final report that will detail our design process in a much more explicit manner.

A summary of the deliverables is provided in the chart below.

Table 6: Project Deliverables

Printed Circuit Board	● Take measurements from electrodes ● Transmit data wirelessly to the backend
Pre-processing Code	● Basic signal processing (e.g. filtering) ● Data cleaning ● Feature Extraction
Dataset	● Cleaned data used to train the model ● Generated using our hand symbols
Machine Learning Algorithm	● Pretrained on our own hand signs
Technical Documentation	● Spice and Gerber files ● Hand calculations ● Well-commented code

Project Management

Discord is used to communicate online. There is a Google Document Folder within Google Drive with all powerpoints and papers relevant to the project. Notion is used sparingly for project management and external resources. Inside Notion, there is a calendar function that details our scheduled project timeline. Communication with our project sponsor, Dr. McFarlane, is done over email or over Zoom meetings.

Software will be in our project's GitHub repository. This way, updates will be clear and different categories of the project can be allotted to different branches before merging with the master branch. Additionally, team members will be able to check functionalities before merging. We are considering using the Milestone feature on GitHub to manage our progress on the software development side of the project.

The following consists of the timeline of our objects:

Fall Semester 2021

November 1, 2021	**Outline Hardware Outputs and Software Outputs**
	Clarifications of what all the project entails. Run by project sponsor for suggestions and advice
November 12, 2021	**Decide on machine learning model**
	Having multiple models in mind to use as backup or testing is a possibility.
November 22, 2021	**Create a prototype of the circuit**
	The prototype will be mapped out in Altium for PCB design. Spice will be able to simulate the prototype prior to working on the physical hardware.
December 10, 2021	**Order PCB**
	Gather all hardware components necessary for the project.

Spring Semester 2022

February 11, 2022	**Acquire Data**
	We will reach out to UT Education of Deaf and Hard of Hearing to collect more reliable data of how people who communicate with ASL actually move. We want to minimize inaccurate data that can be the result of inexperienced people of ASL. Each data input that will be trained and used as a control will be taken multiple times to give us a larger sample size. Depending on how big the numbers we obtain are we might also have to normalize input around 1to make it easier to

understand and use. We were able to get volunteers but never got to the stage where our project was approved for testing on unaffiliated parties.

February 25, 2022 **Train and Test Model**

Run data through our chosen machine learning algorithm to get a trained and optimal set of comparables. Stress test comparables to make sure expected output is given on input. If the output is incorrect, recode the algorithm to account for found shortcomings. We also want to ensure that both settings of this device work properly. One, it should be able to output on consecutive input with a symbol recognition of under one second. The other setting will need to work is being able to feed a bunch of input at once and have the program wait to output the input until the user is completely done. This should also store the input so it is saved for future use. The model will also need to be tested and trained for inputs and outputs that a specific person may want to use that is not already included.

March 18, 2022 **Gather Results from Real People**

One of the main goals of this project is to have a vocabulary building function. We hope to have people try this function out. Additionally, we would like people who communicate with ASL to test out the bracelet and give us feedback for possible improvements and what functionalities would actually ensure that the bracelet will be useful on a day to day basis.

April 25, 2022 **Meet with Project Sponsor and Make Final Changes**

After going through the testing phases and gathering results from people, we will reevaluate and share the results and feedback with Dr. McFarlane. With input from Dr. McFarlane, results from testing, and feedback from people who know ASL, this will allow us to make final touches to the bracelet to ensure the bracelet is ready to be presented and used for everyday use.

May 6, 2022 **Tickle College of Engineering: Senior Design Showcase**

We will present our final design at the senior design showcase.

Budget

Since we will not have direct access to Dr. McFarlane's chip, our evaluation board will need to emulate its functionality using discrete components. Because of this, we will need to include a sine wave generator, a few voltage comparators, a transimpedance amplifier, and potentially a few logic gates if we wish to modulate the output to higher frequencies. Beyond the chip, we will also need to purchase electrodes to measure muscle movements. For wireless communication, we are currently planning to use a microcontroller in conjunction with a bluetooth module. At the moment, we have decided on an Arduino Nano BLE, which has native bluetooth functionality. The PCB itself should be relatively small and ideally only 2 layers, thereby making it inexpensive.

The PCB will be designed using Altium, which is provided for free by the university. All hardware simulations will be completed using LTspice, which is also free to use. Hardware testing is projected to have no cost because Sam owns equipment at home to do so, e.g., bench power supplies and oscilloscope. Moreover, we have access to measurement equipment in two different research labs.

There are no planned expenses for our software. We will be using open source Python libraries to implement our algorithms, like Scikit-learn, NumPy, etc.

Because we are students of the University, we have free and unlimited access to IEEE Xplore, which will constitute the majority of our reference literature.

Our expenses are totaled below excluding the cost of shipping.

Table 7: Project Budget

Arduino Nano BLE	$22.50
8:1 Analog MUX	$0.62
Transimpedance Amplifier	$1.20
Phase Detector (Comparators/XOR Gates)	$1.60
Magnitude Detector (Comparator)	$0.30
PCB	$5.00
Bracelet	$20.00
Electrodes	$20
Total	$71.22

References

Wen, F., Zhang, Z., He, T. et al. AI enabled sign language recognition and VR space bidirectional communication using triboelectric smart glove. Nat Commun 12, 5378 (2021). https://doi.org/10.1038/s41467-021-25637-w

Zhou, Z., Chen, K., Li, X. et al. Sign-to-speech translation using machine-learning-assisted stretchable sensor arrays. Nat Electron 3, 571–578 (2020). https://doi.org/10.1038/s41928-020-0428-6

System Latency and Accuracy Test Matrix	
Test	Experiment Results
Test on amateurs.	Successful
Test on professionals.	Not yet conducted
Test response time.	Needs Work

Weight Test Matrix	
Test	Experiment Results
Test wearability fatigue.	Not yet conducted

Comunication Range Test Matrix	
Test	Experiment Results
Test at varying distances.	Needs Work
Test in varying environments.	Needs Work
Test in varying weather conditions.	Not yet conducted
Test in different combinations of all.	Not yet conducted

Battery Life Test Matrix	
Test	Experiment Results
Test battery life.	Not yet conducted

The Bracelet: ASL Interpretting Wearable Device

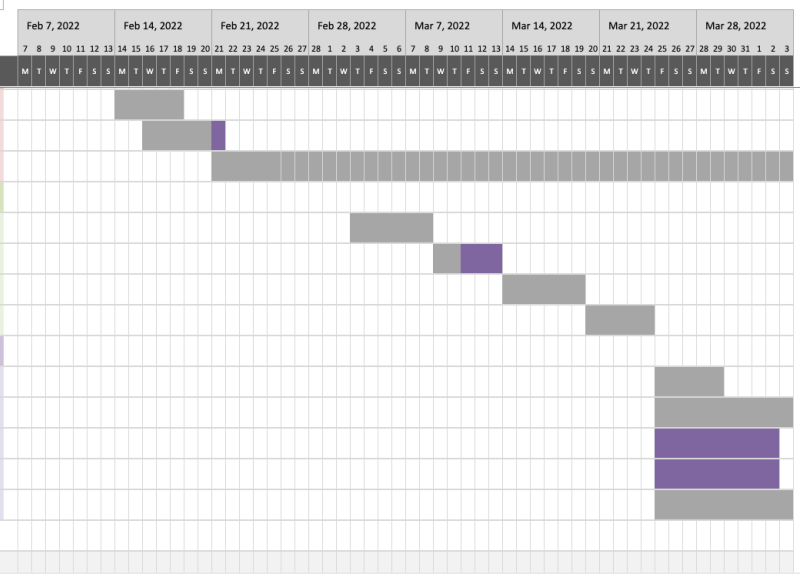
SIMPLE GANTT CHART by Vertex42.com
https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

Project Lead: Sam

Project Start: Wed, 2/9/2022

Display Week: 1

TASK	ASSIGNED TO	PROGRESS	START	END
Choose comparator,	Sam	100%	2/14/22	2/18/22
App: front end desig	Pei & Thomas	99%	2/16/22	2/21/22
Test SVM model	Ahmadre	10000%	2/21/22	2/24/22
Phase 3 Title		100%		
Finish soldering pro	Sam	100%	3/3/22	3/8/22
Test Bluetooth conn	Pei	50%	3/9/22	3/13/22
Start recording data	Ahmadre	100%	3/14/22	3/19/22
Reach out to ASL us	Thomas	100%	3/20/22	3/24/22
Phase 4 Title				
Finish PCB design	Sam	100%	3/25/22	3/29/22
Gather data from ea	All	100%	3/25/2022	4/25/22
Implement Bluetoot	Pei	50%	3/25/2022	date
Implement Databas	Thomas	100%	3/25/22	date
Run grid search on I	Ahmadre	100%	3/25/22	4/13/22
Insert new rows ABOVE this one				



Appendices (Business Matrix Canvas)

