**K-Nearest Neighbor Algorithm-based Filipino Sign Language (FSL) to Text Converter through a Portable Flex Sensor Integrated Gloves and Mobile Application using ATmega328P Microcontroller**

**by**

**John Vincent G. Calites**

**Jean Tristan L. Reyes**

**Glenn Christian D. Sioson**

A Design Report Submitted to the School of Electrical Engineering, Electronics Engineering, and Computer Engineering in Partial Fulfilment of the Requirements for the Degree

**Bachelor of Science in Computer Engineering**

**Mapúa University**

March 2019

# Chapter 1

**BACKGROUND AND INTRODUCTION**

Hearing impairment is the decreased ability to hear sound which can occur at any age. Hearing disability can be acquired due to some genetic factors which can make a newly born child deaf, or people at any age can become deaf due to complications, disease, nerve damage and injury.

Communication plays a vital role in society, it employs the transfer of information that is essential in developing relations. People who are deaf and has other forms of hearing impairment hinders them to properly communicate. Although other forms of communication for them exists, such as sign language, it is not understood by many. Thus, the communication for those who have little to no knowledge of sign language and people with hearing disabilities must be addressed.

On October 30, 2018, the Philippines has recognized Filipino Sign Language as the national sign language of the Filipino deaf and the official sign language of the government (RA 11106), this involves all the transactions involving the deaf, its usage in schools, broadcast media and workplaces. Since most of the schools use American Sign Language (ASL), various government offices such as the DepEd, CHED and TESDA and all national and local government authorities involving the deaf are tasked to use the Filipino Sign Language (FSL) as the medium of instruction for education.

## Customer

The customer for this design project is Kaisahan ng Magulang at Anak na may Kapansanan Inc. (KAISAKA), a Community Based Rehabilitation (CBR) program for children and young adults with disability and their families based in the urban poor community of Manila. Started as a parish outreach program in 1989, KAISAKA, Inc. was registered in the Securities and Exchange Commission (SEC) as a People’s Organization (PO). On August 6, 1998, with mothers and people with disabilities themselves as the frontrunner of the program.

## Needs

The officers of the said organization which directly deals with the PWD’s have low proficiency regarding Filipino sign language, thus they encounter difficulty when communicating with deaf people. Since they are non-profit organization which, they do not have sufficient funds to be able to hire translators, it will also take time to fully learn and study Filipino Sign Language. As one of the officers said, they barely understand Filipino sign language, and most of the time they have a hard time understanding it. The need of the client is a device which can aid the officers in understanding the deaf people. The device must also be easy to use, making it usable even by children. It must also be portable in such a way that it can be used in everyday situations.

## Solution

As response to the need of the client, the group intends to create a device that would help the client in a way that the communication Filipino Sign Language Speaker be understood with having little to no knowledge regarding sign language.

The proposed solution is a glove that will translate Filipino Sign Language (FSL) to text, which can be viewed on an android phone. The device is portable in a sense that it will only need an android phone in viewing the translated gesture. The gloves, while worn by the user, will do a real time language translation through the means of capturing hand movements and gestures. The movement and gesture of the hands will serve as the input data, these data could be gathered by providing data to the microcontroller coming from the flex sensors, gyroscope, and accelerometer. The flex sensors will be responsible for taking the bending motion of the fingers. The gyroscope and accelerometer will take the position and speed of the hand in motion. Combining these three data inputs, the interpretation of a specific gesture could be possible.

The data coming from the sensors will then be sent by the microcontroller to the connected android phone using a Bluetooth module. In the android phone, the hand movement and gesture are interpreted through the K-nearest neighbor algorithm that predicts the output using the created mobile application. The device will be equipped with force sensitive resistors to identify whether the glove is equipped on the hand of the user. The device will be powered by a rechargeable battery with a charging module.

In terms of how the device will be made, the group plans to design it as small as possible for the user to be able to move his hand and make gestures comfortably. The flex sensors will be placed along the fingers. The gyro sensor is placed on the back of the hand while the force resistive resistor will be placed at the palm. The microcontroller, Bluetooth module and the battery along with the charging module will be placed on top of the wrist. A small and compact PCB will be designed to connect all these components.

## Objectives

In this design project, the group intends to develop a real-time language translation of Filipino Sign Language (FSL) to text. Specifically, the group wants to achieve the following:

1. To use a flex sensor, gyroscope and accelerometer to capture hand movements and gestures.
2. To create a portable device utilizing an Android application capable of translating Filipino Sign language (FSL) to text.
3. To use the K-Nearest Neighbor Algorithm for predicting gestures.

## Scopes and Delimitation

The device is limited on one-way communication from signer to a non-signer, the android device only translates Filipino Sign Language (FSL) to text in real time. The output text will be based only from the data gathered from the gloves. Facial expressions, head movement, body positions, and hand space are not translatable by the device. The microcontroller will only send the data coming from the sensor to the android phone, all processing will be done by the android application. Only one android phone can be paired to the glove to view the translated words. The android phone will be used to process the data and convert the signals to text rather than the ATmega328p microcontroller, since KNN uses large amounts of resources.

## Differentiation

A thesis published from Mapua Institute of Technology Manila: MIT School of Graduate Studies, “A heuristic decision tree algorithm for Filipino Sign Language (FSL) translator implemented using accelerometers with Bluetooth connectivity to a host PC” (Piscasio,2014) was found to have a similar problem to solve, to bridge the communication gap between signers and non-signers. There are however some differences in the implementation of the solution to the problem. The design to be implemented aims to make the device portable for it to be usable in an everyday situation as compared to the work of Piscasio that uses a PC as the output and processing device. Also, the device will not be limited to finger spelling unlike the similar device, the design aims to recognize movements and gestures of the hand that can be translated into words. There are also differences on the components, method and technology used as seen in Table 1.1. Our group will use the ATmega328p microcontroller and translate Filipino Sign Language (FSL) to text and will have a rechargeable battery. Lastly, the machine learning algorithm that the group will use is K-Nearest Neighbor algorithm unlike the similar device which uses a Heuristic Decision Tree.

Table 1.1: Differentiation from Previous Work

|  |  |  |
| --- | --- | --- |
|  | **Design Solution** | **Nearest Similarity** |
| **Technology** | **Microcontroller:** ATmega382p  **Sensors:**  Flex Sensors  Force Sensitive Resistors  MPU6050 Sensor  **Algorithm:**  K-Nearest Neighbor | **Microcontroller:**  ATmega2560  **Sensors:**  Accelerometer  **Algorithm**  Heuristic Decision Tree |
| **Functionality** | Converts Filipino Sign Language to Text | Converts Filipino Sign Language to Text |
| **Features** | Portable  Rechargeable  Uses Android phone | Uses PC |

## Benefits

The product is made specifically to benefit the officers of the organization who have little knowledge about Filipino Sign Language (FSL) and has a hard time in understanding it. This product will also benefit the deaf people of whom the organization help to, since they can now be able to communicate to other people, without changing how they used to interact with each other. The product being designed and built as a compact and portable device, makes it easier to be carried around, thus it can be used anywhere when needed.

# Chapter 2

**REVIEW OF RELATED LITERATURE AND STUDIES**

This chapter involves related studies and developments of the design of the Filipino Sign Language (FSL) to text converter. The researchers will give an overview to its readers how this study’s design works and develop a new concept.

## Technology for Capturing Gestures

Language translation can be done through the use of electromagnetic sensors that have a direct contact with the skin along with accelerometers, gyroscopes and vision through means of a supervised machine learning algorithm requiring the user up to 30 minsf training per gesturutes of training per gesture. The algorithm consists of different type of machine learning algorithm which includes Naïve Bayes, Nearest Neighbor, decision trees and vector machines which enables the comparison for accuracy of 40 common gestures. An important consideration in creating a communication device is by following the safety standards. To bring a device as close as a commercial product, it must follow the safety standard like prevention of shorting wires that might cause injuries. Also, electrocution must be prevented through means of having a total voltage lower that 25V which was referred to as the lowest lethal voltage by a research paper completed at the City University of New York.

## Flex Sensor

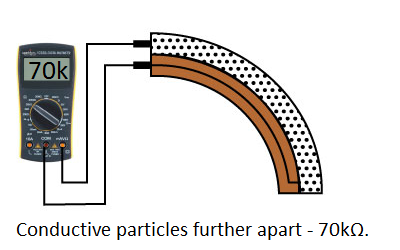
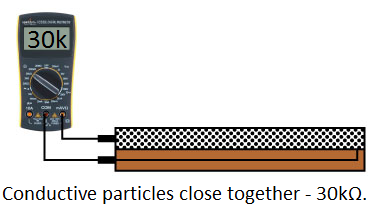
A flex sensor is a variable resistor, its resistance increases as the component bends. One side of the Flex Sensor is made out of polymer ink which has conductive particles. When the flex sensor is straight, it gives the polymer ink a resistance of approximately 30kΩ, when bent, it gives around 50kΩ~70kΩ depending on the degrees it is bend.

Figure 2.1: Comparison of Flex Sensor resistance

Voltage divider will be required in incorporating the sensor to a project, it must be connected in the Arduino’s A0 port and the other end at 5V. When developing the program, the straight and bend resistance must be declared. And then calculate for the resistance using various formulas.

## Mechanism for Filipino Sign Language (FLS)

Sign Language is not an international language, but those who use sign language may understand fellow sign language users quickly because there are universal features in sign languages as stated by the World Federation of the Deaf. Filipino Sign Language is used by the deaf people in the Philippines, this is distinct in comparison to the American Sign language (ASL) because Filipino sign language poses unique structural features in terms of its phonology, morphology, syntax and discourse.

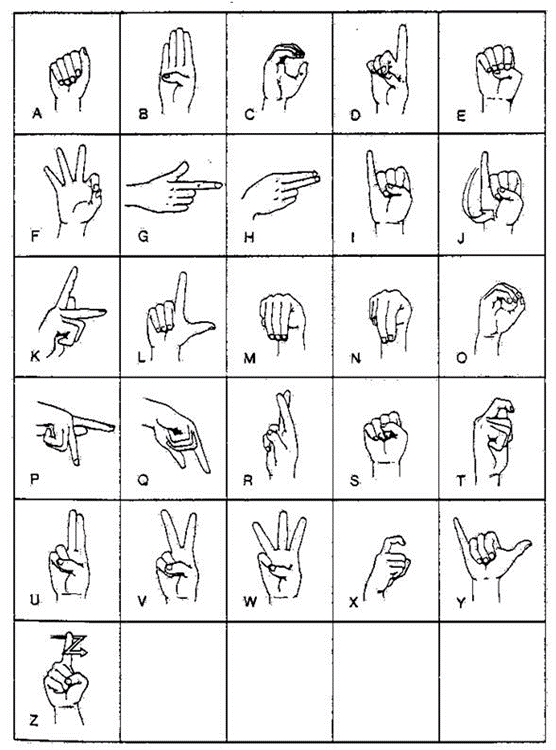


Figure 2.2: Filipino Sign language alphabet

## MPU6050 Gyro Sensor

The MPU6050 is an 8-pin with 6 axis chip which combines the accelerometer and a gyro. To use the Gyro Sensor, pin 18 (GND) must be connected to the ground, 3.3V must be connected to pin 13 (VDD). Pin 23 (SCL) and Pin 24 (SDA) of the MPU6050 must be connected to the A4 pin and A5 pin of the microcontroller respectively. Data pin is directly connected to the digital pin numbers 10-13. In order to program using the MPU6050 Sensor, a library must be imported from GitHub and install it in the IDE, this is the MPU6050 Library. This library supports reading and displaying gyro and accelerometer readings. The Gyro sensor uses the earth’s gravity to determine the x-axis, y-axis and z-axis position of the sensor while the accelerometer detects based on the movement.

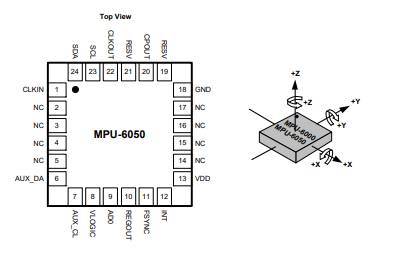


Figure 2.3: MPU6050 Pin diagram and orientation

## Force Sensitive Resistors

FSRs are simple and low-cost sensors that allows detection of pressure and weight. It is made of 2 layers which is the flexible substrate with printed semiconductor and a flexible substrate with printed interdigitating electrodes which are separated by a spacer. The resistance of the FSR changes as pressure is applied. When no pressure is applied, the FSR acts as a resistor, as pressure is applied the resistance decreases. One end must be connected to a 5V supply while the other end is connected to the analog input.

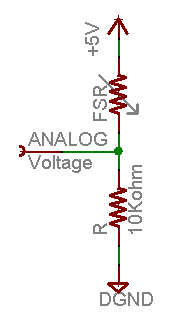


Figure 2.4: FSR Connection

The voltage of the Force Sensitive Resistor can be computed as , whereas Vo is the output voltage, Vcc being equal to 5V, FSR being the resistance of the force sensitive resistor when 0 pressure is applied and lastly R being the value of the pull-up resistor.

## Interfacing an Android Phone

Interfacing an Android phone with Arduino via Bluetooth is possible through the use of a Bluetooth module. Connect first the Pin 0 (RX) to TX, TX(Pin 1) to RX, VCC and ground accordingly. Turning on the microcontroller also powers up the HC 05/06 Bluetooth module, the android phone needs to be paired via Bluetooth by entering the default password which can either be 1234 or 0000. After a successful connection, data transfers via Bluetooth can now be used.

## K-Nearest Neighbor Algorithm

It is a supervised machine learning algorithm that is dependent on labeled data inputs in order to learn and will produce an output based on the new given input unlabeled data. This is commonly used to solve classification and regression problems. This algorithm assumes that similar things exist in close proximity.

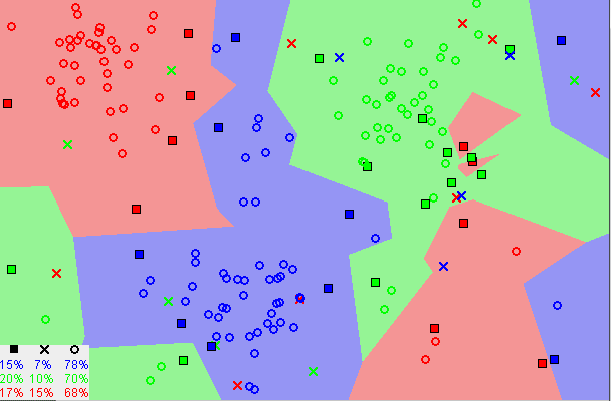


Figure 2.5: Similar datapoints existing close to each other

Figure 2.5 shows that most of the time, similar data points are located close enough from each other. The KNN algorithm bases on this assumption being true, it captures the idea of similarity. Several steps are required in using the K-Nearest Neighbor algorithm, these are as follows;

1. Load the data
2. Initialize the value of K to the number of chosen neighbors
3. For each sample data, compare the distance between the query example and current example then add the distance along with its index to an ordered collection
4. Sort the ordered collection of distances and indices in ascending order
5. Pick the first K entry
6. Get the labels of the selected entry
7. Return the mean or the mode of the K labels if it is regression (mean) or classification (mode)

## K-Nearest Neighbor Data Pre-Processing

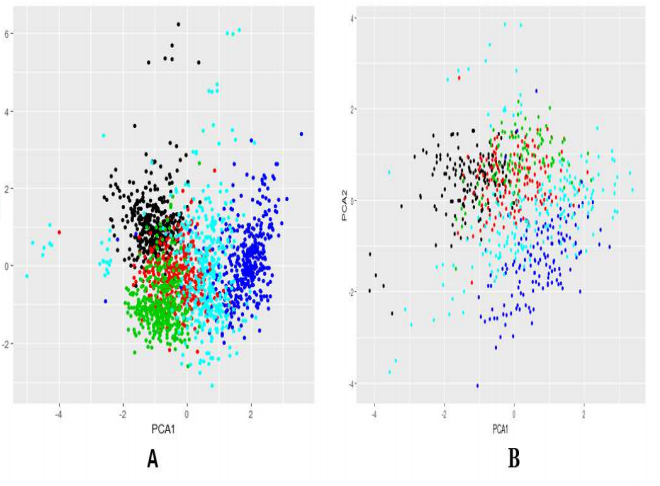
When implementing a complex prototype selection algorithm with regards to a large scale of data, the execution time needed to accomplish the task is long due to the repeated iterations in the training set. The gloves acquire the data from the flexion of the sensors when performing different sign languages, thus a large number of data are stored, which needs to be pre-processed. The data which are like the noise to the database are then removed. For data acquisition, the gesture must be done for several minutes for one minute in each position. These data are stored in the new matrix **T**, of **m** x **n** order wherein the value of m is the number of samples and the variable n is the number of attributes that represents the data.

Figure 2.6 Sign Language database

Figure 2.6 illustrates in part A the matrix T and in part B the matrix U, colors: black (number 1), red (number 2), green (number 3), blue (number 4) and cyan (number 5).

## TensorFlow Lite

TensorFlow Lite is TensorFlow’s lightweight solution for mobile and embedded device. It lets you run machine learning models on mobile devices with low latency quickly so you can take advantage of them to do classification, regression, or anything else that you might want without necessarily incurring a roundtrip to the server. It’s presently supported on Android devices via a C++ API, as well as a Java wrapper for android developers. On android devices that supports it, the interpreter will also use Android Neural Networks API for hardware acceleration.

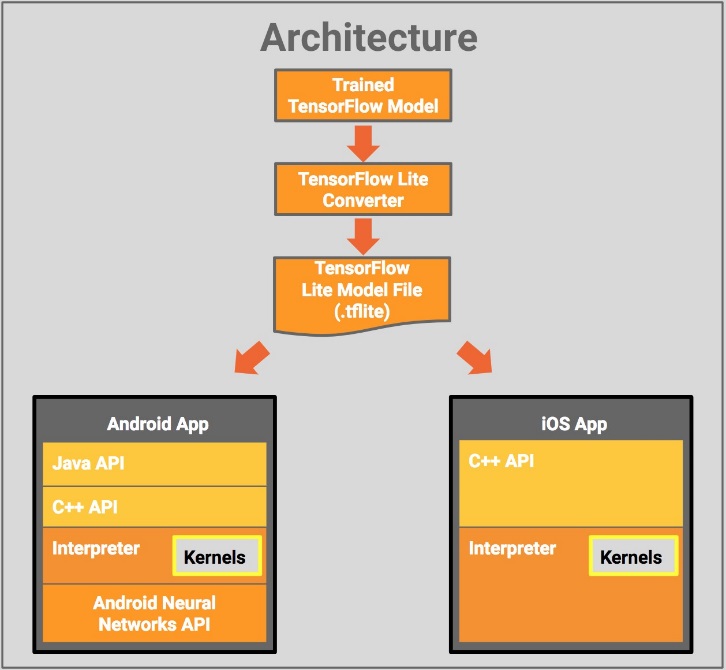


Figure 2.7 Architectural Design of TensorFlow Lite

In Figure 2.7, starting with a trained TensorFlow model on disk, the model will be converted to TensorFlow Lite file format (.tflite) using TensorFlow Lite Converter. Then the converted file can now be use in the developed mobile application.

# Chapter 3

## Design Procedures

In compliance to the client’s needs, the researchers have designed the Filipino Sign Language (FSL) to text converter. In this chapter, the stages of designing the system that will comply to the needs of the client is discussed.

## A. Hardware Development

*Step-by-Step Procedure of Hardware Development*

1. Identify the needed components to be used, the major materials are:
   1. ATmega328p – The microcontroller that will be used.
   2. HC05 Bluetooth Module – Used in transmitting converted FSL to an android phone.
   3. Flex Sensor – Used to measure the flexion of the fingers.
   4. FSR 406 – Used to identify whether the hand is in contact.
   5. MPU6050 – Used to measure the orientation of the hand.
   6. TP4056 Module – Used to charge the batteries.
   7. Op-amp – Used to amplify the voltage input.
   8. Gloves – Used to hold the components.
   9. 18650 Li-Ion Battery – Used to power the device.
2. Construct a block diagram that will show the hardware design flow.
3. Create a schematic diagram for the FSL Translator Circuit.

### Design Process Flow

Prototype Development

Testing of Prototype

Does Prototype meet the  
Objectives?

Modify the Design to meet the Requirements

Data Gathering

Interpretation of Results

Formulation of Conclusion

End

A

No

Yes

Start

Client Interview

Create objectives based on the needs of the Client

Review of Related Literature

Planning of Preliminary Design

No

Modify the Design to meet the Requirements

Is the preliminary design approved by the client?

Yes

Documentation

Design of the System Hardware in accordance to the Objectives

Design of the System Software in accordance to the Objectives

A

Figure 3.1 Design Process Flow

Figure 3.1 illustrates the flow of the design development. First, the client was interviewed in order to know their needs with regards to the existing problems of persons with physical disabilities. Then the objectives were established in accordance to the needs of the client. The researchers then read documents, journals and previous studies to have an idea on how the problem will be solved. A preliminary design will then be presented to the client, it will be modified depending on the needs of the client. Once it has been approved by the client, the next step is to design the hardware part of the design which includes the listing of components needed and creating the schematic diagram. Next is to design the software part of the system, this will describe the logical flow of the system by means of programming. Then the hardware design will be implemented to form the actual hardware as well as the software to be uploaded into the microcontroller to build the prototype of the system. Testing will be conducted to see if the prototype functions in accordance to the objectives. If it meets the objectives, data gathering, and formulation of conclusion is done, if not then the current prototype is modified until the objectives are met.

### Conceptual Framework

Converted Text

Gesture Sensing

Hand Gestures

Process

Output

Input

Read and Transfer Data

Gesture Classification

Figure 3.2: Conceptual Framework

Figure 3.2 shows the conceptual framework of the system. The input will come from the hand gestures of the user that corresponds to the Filipino Sign Language (FSL). There will be sensors providing the data from the hand gestures. The data will be read by the microcontroller and will be sent to the android phone via Bluetooth connection. In the android phone, the data will be processed using the Machine Learning Model that is trained using K-Nearest Neighbor Algorithm which finds the most frequent label for a given value of K on the trained data. Lastly, the label’s equivalent text will be displayed on the android phone.

### Block Diagram

Flex Sensor

Android Phone

HC05 BT Module

MPU6050

ATmega328p

FSR 406

TP4056

18650 Li-Ion Battery

Figure 3.3 Block Diagram for the Design Project

Figure 3.3 illustrates the interaction among the components of the device. The microcontroller ATmega328p will serve as the controller for the system in which it controls and reads data from the different components. The ATmega328p reads the resistance coming from the flex sensor, the acceleration and position from the MPU6050 and if the device is in contact to the hand by means of the resistive value coming from the FSR. The microcontroller then sends it to the android phone through means of Bluetooth connection. Then, the smartphone uses the data gathered by the sensors to interpret the gesture through a given set of algorithms and translates it accordingly. The translated text will be displayed on the smartphone. The power will come from the battery that is rechargeable through means of the TP4056.

### Schematic DiagramC:\Users\calites\Desktop\54799613_428022191103763_7232707531846451200_n.png

Figure 3.4: Schematic Diagram for the Design Project

The schematic diagram is illustrated in figure 3.4. The microcontroller is powered by 2 pieces of 3.7V Li-ion battery which uses the voltage divider to have a Vout = 5V, this is done using , with the value of R5 being set to 1020 Ω and having an input voltage V in = 7.4V, the value of R4 can be computed which results to 487 Ω. The 5V output is connected at the on/off switch. The switch is connected to the FSR406 which lowers its resistance when force is applied, thus when in contact a voltage lesser than 5V will be generated as the output which is then connected to the non-inverting terminal of the operational amplifier for the gain to be positive. The Vout produced by the FSR will be amplified for it to be able to reach approximately 5V, this is because the resistance produced by the FSR will not be exact, it will depend on the force the hand will apply on the FSR when the gloves is worn. The resistors R6 and R7 are set to 1kΩ and 3kΩ, respectively, to get an amplification of 1.333 by the equation , that will amplify the input voltage coming from the FSR up to approximately 5V when the resistance is set to 100KΩ. The pin 7 of the microcontroller which is the VCC which is connected to the output voltage of the operational amplifier while 8 and 22 of the microcontroller is connected to the ground. The analog pins 0-4 are used for the connection of the flex sensors. The RX and TX pins of the HC05 Bluetooth module are connected in pins 2 and 3 of the ATMega328p which are the RXD and the TXD pins respectively, this is used for communication between the ATmega328p board to communicate with other devices serially. The RX pin of the Bluetooth module only requires 3.4V as input, thus a voltage divider was used to lower the voltage of the incoming 5V input, this is done using and the value for R10 is equal to 2490 Ω, thus making the R9 be equal to 1180 Ω . Pins 9 and 10 of the ATmega328P microcontroller is also connected to a 16Mhz Crystal Oscillator with both ends being connected to a 22pF capacitor. The data signal pin of the pin 24 of the MPU6050 is the SDA, this is connected to the analog input 5 of the microcontroller which permits the data to be transferred from the gyro sensor to the microcontroller. The microcontroller is short of analog input pins thus an extender is used which is the ADS1119, the pin 23 of the MPU6050 which is the SCL or the clock is connected in the analog input 1 which is the pin 2 of the analog input extenders, while the SDA is connected to the analog input 0 or the pin 1 of the ADS1119. A TP4056 is connected to the battery which enables the lithium-ion battery to charge. Although, in the case of inserting the battery incorrectly, it will cause reverse polarity, and will end up damaging the circuit. To prevent this, a circuit was created for reverse polarity protection. A MOSFET was placed between the battery and charging module. When off, the MOSFET does not let any current through, effectively opening the circuit like a reverse-polarized diode. When ON, it behaves more like a resistor with a very low resistance, causing a voltage drop that decreases as the cell becomes full and the current slows down. A resistor was placed from the source to the gate to make sure the MOSFET is off by default and a transistor is added to override the resistor and flip the voltage on the gate when the battery is correctly positioned. The values for the limiting resistors are computed using the formula, whereas a two 3.7V batteries in series have a current of I=0.0011A and a required voltage of V=3.4V, the value for R can be computed. This resulted in the formula wherein VT is the total input voltage which is equal to 7.4V and Vo is the output voltage that is equal to 3.4V, by substituting the values will result to an approximate value of 3650 Ω. The fuse rating that will be used for the circuit is 0.65A, from the formula Fuse Rating = Wattage / Voltage x 1.25, where the values for the power and voltage are based from the datasheet of 18650 Li-ion battery.

**Isometric Model**



Figure 3.5: Isometric Model of the Prototype

Figure 3.5 illustrates the isometric model of the design. The design will be compact for the user to freely move the hand and make gestures with ease. The components will be placed at the top of the wrist for simple wiring, this includes the ATmega328p, TP4056 as well as the battery that will be contained in a compact case that is placed along the top of the wrist. The flex sensors will be placed on top of the fingers. The force sensitive resistor is placed on the palm while the gyro sensor is placed on the back of the hand. To provide safety for the user, the case that will contain the MPU6050, and the case for the ATmega328P together with the other components will both be properly insulated.

## B. Software Development

The device requires a program that will take the input data coming from the flex sensors, contact sensor, and gyroscope, and output the equivalent text in the android application. For this to happen, the group needs to work on the program for the Microcontroller, the Android Application, and the Machine Learning Model.

### ATmega328p Microcontroller Programming

*Step-by-Step Procedure of Software Development for the Microcontroller*

1. Identify the needs of the program that must be satisfied.
2. Plan the program’s logic and create the system flowchart that contains the steps of the program.
3. Prepare the software to be used in building the program such as IDE and programming language to be used.
4. Construct the code of the program.
5. Test the program for syntax and logical errors.
6. Deploy the program to the device.

#### Needs of the Program that must be satisfied

The program must be able to let the ATmega328p microcontroller to receive data from the flex sensors, accelerometer, gyroscope, and force resistive resistor and send it to the Android phone via Bluetooth.

#### System Flowchart

Start

Setup()

Initialize all sensor Variables to 0

Setup()

Loop()

Define Pin Modes

return

End

Figure 3.6 Atmega328P Program Flowchart

Figure 3.6 shows the flow of the program being implemented on the microcontroller. The Setup() function is called as the program starts. This function will only run once, after each powerup or reset. In this function, the variables which will receive the flex sensor and the Gyro sensor values that will be used in the program will be initialized to zero as its initial values. Then the pin modes will be defined, wherein the used pins are assigned whether as an input or an output. The function will end and return to the main program. Then the Loop() function will be called, which can be seen on figure 3.7. This function will loop consecutively, which allows the program to change and respond. Inside the function, the FSR Sensor value will be read. If it is not triggered, the function will end and return to the main program, else the program continues. As the program continues, the values from the Flex sensors and MPU6050 will then be read, and eventually be sent to the android phone via Bluetooth.

Send SensorValues to the Android Application

Loop()

Read FSR value

FSR Value=>200?

Yes

return

No

Read FSR value

CompileSensorValues()

Figure 3.7.1 Loop Function Flowchart

Figure 3.7.2 illustrates the process of creating a single string containing all of the sensor values which are separated by a delimiter “ - “, this string will be sent to the android device which will extract all the values accordingly.

int i=0  
string SensorValues=””  
int TotalFlexSensor = 5

CompileSensorValues()

declare variables

Increment the value of i by 1

Is FlexSensor[i]=>TotalFlexSensor

Append Flex Sensor[i] Value to SensorValues

Read Flex Sensor[i] Value

Append Delimiter to SensorValues

Read Gyro X Value

Append Gyro X to SensorValues

Append Delimiter to SensorValues

Read Gyro Y Value

Append Delimiter to SensorValues

Append Gyro Y to SensorValues

Append Delimiter to SensorValues

Read Gyro Z Value

Append Gyro Z to SensorValues

Return SensorValues

Figure 3.7.2 Creating a string containing all Sensor Values

#### Preparing the Software to be used

In writing and uploading the code to the microcontroller, the open-source Arduino Software (IDE) will be used. It is an IDE especially made for Arduino and it uses C/C++ as its programming language.

### Android Development

*Step-by-Step Procedure of Software Development for the Android phone*

1. Identify the needs of the program that must be satisfied.
2. Plan the program’s logic and create the system flowchart that contains the steps of the program.
3. Prepare the software to be used in building the program such as IDE and programming language to be used.
4. Construct the code of the program.
5. Test the program for syntax and logical errors.
6. Deploy the program to the device.

#### Needs of the Program that must be satisfied

The main purpose of developing an Android application is to display the equivalent text of a corresponding gesture. To implement this, the application should be able to let the Android phone to connect and receive sensor data from the microcontroller, classify the input data using K-Nearest Neighbor Algorithm Model, and display the output word as text on the screen of the android phone. It must also provide a simple and friendly user interface, for easier use.

#### System Flowchart

Start

Initialize the application

Is Bluetooth  
Connected?

Display ConnectivityUI

Is Bluetooth toggled on?

No

Ask to turn on Bluetooth

Yes

Show List of Available Devices

No

Pair with the Bluetooth device (Glove)

Yes

Display MainUI

A

Figure 3.8.1 Android Development Flowchart

Clear TextBox

End

No

No

Yes

Yes

A

Yes

Exit App?

No

Does the user tap the Translate Button?

Is clear

button tapped?

Receive SensorValues from the Gloves

ExtractValues()

ClassifyGesture()

Display the corresponding Gesture on TextView of the Android Application

Figure 3.8.2 Android Development Flowchart

Figure 3.8.1, 3.8.2 and 3.8.3 shows the flow of the program of the android application. As the application starts, the application will initialize and show the user interface for connectivity. A toggle switch is available for activating the phone’s Bluetooth. Once activated, the application will show a list of available devices to connect. The user must choose the gloves as the device to connect to. Once connected, the main user interface will appear. The main user interface contains a toggle switch for translate function, a textbox for viewing of text, and a button for clearing text on the textbox.

While the toggle switch is on, the input data from the microcontroller is read and will be used for classifying the hand gestures’ equivalent word. The words will be displayed as text and can be viewed in the textbox, one word per line. The clear button clears the text inside the textbox and will only activate when the translate toggle switch is off. The program will end when the user exits from the application.

Figure 3.9 illustrates the function to classify gesture. Variables to be used within the function are declared at the start of the function. The content of the totalTrainedData variable will be initialize by the total number of trained data of our data set. Then get the computed distance from our testInput to each trainedInput in the data set and store it in the list of distance, this is computed using equation 3.1.

(3.1)

After getting all the distances, sort the list of distance in increasing order. In this way, we can get the K-nearest distances from the list of distance and get all their trainedLabels. All the trainedLabels will be stored in the list of targets. Now that we have the list of trainedLabels within the range of k, we can get the most common trainedLabel in the list and set it as the outputLabel of the testInput.

ExtractValues()

Int counter = 0  
string [] parts  
double [] values  
char delimiter = ‘-‘

declare variables

Get SensorValues string

Split SensorValues using delimiter

Insert split values to string [] parts

Get number of elements of parts using length() function

Increment counter by 1

Insert value to values [] array

No



Parse string to double using parseDouble method

Is counter  
<   
length of parts?

Yes

Return values

Figure 3.8.3 Extracting Sensor Values from a single string

To extract the compiled sensor values from a string with a delimiter, Figure 3.8.3 will be implemented. The string will be split using a delimiter of ‘-‘, the split parts will then be placed to an array named string [] parts that will contain all of the sensor values but with a string data type, to be able to process this data it needs to be converted to a double data type. The length of the string [] parts array, each element of this array will be parsed to double and will be placed in a new array named double [] values array.

K=7

list distance = new ArrayList

list targets = new ArrayList

num trainedDataIndex = 0

num kIndex = 0

num totalTrainedData = 0

string outputLabel = “”

ClassifyGesture()

declare variables

get totalTrainedData and trainedDataIndex form the Gesture Data Set

increment trainedDataIndex

trainedData

Index <= totalTrained Data

compute distance between 2 data points, trainedInput and testInput using Equation 3.1

Yes

add the computed distance to the list of distance

No

sort the list of computed distance

increment

kIndex

Yes

kIndex

<= K

get the corresponding trainedLabel of the current index from the array list of distance

add the trainedLabel to the list of targets

No

set outputLabel as the most common element in the list of targets

Return outputLabel

Figure 3.9 ClassifyGesture() Flowchart

#### Preparing the Software to be used

In writing the program and creating the installer for the android application, the Android Studio will be used. This is the official IDE for android development and it uses Java as its main programming language.

### Machine Learning Model

In creating the machine learning model, TensorFlow will be installed and setup. This enables on-device machine learning inference with low latency and it also supports hardware acceleration thus making machine learning viable on android devices. By training the Machine learning model with the training data, gesture translation can be possible.

There are several steps required in Training a Machine Learning Model, these are as follows;

1. Preparing for Collection of Data
2. Create the algorithm for the model
3. Training the model
4. Evaluation

The first step is to create a data set which will be used in training the machine learning model, these data are based on the sensor output values. The data set format is in a Comma-Separated Values (CSV) format which can be easily parse in the code to get the relevant details which is viewed on Table 3.1. In this step, is where we can prepare the data for use, these data are to be randomized to make a determination of a gesture independent of its order from the data set, the data is then visualized to prevent data imbalance which can cause the prediction to have a variable biasing.

Table 3.1: Filipino Sign Language to Text Data Set

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Flex  1 | Flex  2 | Flex 3 | Flex 4 | Flex 5 | Gyro  X | Gyro Y | Gyro Z | Accel  X | Accel  Y | Accel  Z | Label |
| 1 |  |  |  |  |  |  |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |  |  |  |  |  |  |
| … |  |  |  |  |  |  |  |  |  |  |  |  |
| n |  |  |  |  |  |  |  |  |  |  |  |  |

Next, the algorithm for the model that will be used is the K-Nearest Neighbor Algorithm. Information about the algorithm for the model is available on Chapter 2 and its program flowchart is shown in Figure 3.9. Next is to train the model, the data will be split by 80-20 ratio 80 being the training data set and the remaining will be used for evaluating predictions. Then train the model using the training data. After the model has been trained, the next step is to evaluate the model whether it predicts correctly, if it does not, then tune the parameters to have a higher accuracy in predicting a gesture. As soon as the Machine Learning Model is ready, it can now be converted to TensorFlow Lite file and can now be added to the Android Application.

## C. Prototype Development

## D. Multiple Design Constraints

# Chapter 4

**Testing, Presentation, and Interpretation of Data**

# Chapter 5

**Conclusion and Recommendation**

**References**

Cloutier, P., Helderman, A., & Mehilli, R. (2016). Prototyping a Portable, Affordable Sign Language Glove.

Lom, J. B. (2016). Flex Sensor Hookup Guide. Retrieved March 3, 2019, from https://learn.sparkfun.com/tutorials/flex-sensor-hookup-guide/all

Esposa, J. I., Jr. (2017, August 05). Difference between ASL and FSL. Retrieved February 27, 2019, from https://deafphilippines.wordpress.com/2017/08/03/difference-between-asl-and-fsl

Inquirer, P. D. (2016, November 6). In The Know: Sign languages. Retrieved February 27, 2019, from https://newsinfo.inquirer.net/302184/in-the-know-sign-languages

Sadam (2018, March 9). MPU6050 Gyro Sensor Interfacing with Arduino. Retrieved March 2, 2018, from https://circuitdigest.com/microcontroller-projects/mpu6050-gyro-sensor-interfacing-with-arduino

Ada, L. (2015, November 19). Force Sensitive Resistor (FSR). Retrieved March 1, 2019, from https://learn.adafruit.com/force-sensitive-resistor-fsr/using-an-fsr

Girish, M. (2018, March 23). Bluetooth Basics: How to Control an LED Using a Android phone and Arduino. Retrieved March 1, 2019, from https://maker.pro/arduino/tutorial/bluetooth-basics-how-to-control-led-using-android phone-arduino

Harrison, O. (2018, September 10). Machine Learning Basics with the K-Nearest Neighbors Algorithm. Retrieved March 5, 2019, from https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

Rosero-Montalvo, P. D., Godoy-Trujillo, P., Flores-Bosmediano, E., Carrascal-Garcia, J., Otero-Potosi, S., Benitez-Pereira, H., & Peluffo-Ordonez, D. H. (2018). Sign Language Recognition Based on Intelligent Glove Using Machine Learning Techniques. 2018 IEEE Third Ecuador Technical Chapters Meeting (ETCM).

TensorFlow Lite guide - TensorFlow Lite (n.d.). Retrieved March 3, 2019, from https://www.tensorflow.org/lite/guide