

{يَرْفَعُ اللَّهُ الَّذِينَ آمَنُوا مِنْكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ}

Requirements of Graduation Project report



An-Najah National University

Faculty of Engineering and Information Technology

Graduation Project 1

Smart Glove for Sign Language

**Presented as a part of the prerequisites for earning
A bachelor's degree in (Electricals Engineering)**

Prepared by:

Ahmad Kamel Al-Sadeq (12027739)

Faiq Aref Zeidan (11926544)

Supervised by : Dr. Haneen Al-Autt

Table of Contents

Abstract:.....	4
Chapter 1: Introduction.....	5
1.1 General Introduction.....	5
1.2 Objectives of the project.....	5
1.3 Importance of the project.....	5
1.4 Organization of the report.....	6
Chapter 2: Theoretical Background and Previous Work.....	6
2.1 Overview of Sign Language Recognition Systems.....	6
2.2 Wearable Sensor Technology for Gesture Recognition.....	7
2.3 Embedded Systems and Microcontroller Architecture	7
2.4 Signal Processing and Data Preprocessing	8
2.5 Artificial Intelligence Techniques for Gesture Classification.....	8
2.6 Wireless Communication and User Interface	9
2.7 Previous Work and Related Research.....	9
Chapter 3: Methodology	10
3.1 System Architecture	10
3.2 Data Acquisition	10
3.3 Wireless Communication.....	10
3.4 Data Processing Pipeline	10
3.4.1 Local Processing on ESP32	11
3.4.2 Inter-Glove Data Synchronization.....	11
3.4.3 Data Transmission to Mobile Application.....	11
3.4.4 Mobile Application Data Processing	11
3.4.5 Output Generation	11
3.5 Standards and Specifications	12
3.6 Constraints	12
3.7 Data Collection and Preparation Methodology:.....	12
3.8 Components of the project:.....	14
3.9 Power Supply and Battery Management System.....	20
Chapter 4: Artificial Intelligence Methodology and System Implementation	21
4.1 Introduction	21
4.2 Problem Formulation	21
4.3 Sensor Data Acquisition and Feature Representation.....	21
4.3.1 Sensor Configuration.....	21
4.3.2 Feature Vector Construction.....	22
4.4 Dataset Collection and Labeling.....	22
4.5 Data Preprocessing	22

4.6 Machine Learning Model Design	22
4.7 Model Training and Evaluation	23
4.8 TinyML Deployment on ESP32	23
4.9 Communication and Application-Level Processing	23
4.10 Sentence Formation and Contextual Processing	23
Chapter 5: Results and Analysis	24
5.1 Circuit Diagram:	24
5.2 Schematic Diagram of the Circuit:	25
5.3 Simulation.....	25
5.4 Simulation Code	28
Chapter 6: Standards and Constraints	30
6.1 Introduction	30
6.2 Engineering Standards	30
6.3 System Constraints	31
Chapter 7: Discussion.....	32
7.1 Primary goal	32
7.2 Future Work.....	33
Chapter 8: Conclusions and Recommendation.....	34
References:	35

Abstract:

Sign language is a primary means of communication for individuals with hearing and speech impairments however communication barriers still exist between sign language users and the general public. This project presents the design and implementation of an AI-based smart glove for real time sign language recognition. The proposed project utilizes a wearable gloves equipped with flex sensors and inertial measurement units “IMU” to capture finger bending and hand motion data. Sensor readings are processed using an embedded ESP32 microcontroller, where a lightweight TinyML model performs gesture classification locally without the need for internet connectivity.

A supervised machine learning approach is adopted, in which sensors data is collected, preprocessed, and used to train a neural network capable of recognizing predefined sign language gestures. The trained model is converted to a TensorFlow Lite format and deployed on the ESP32 for efficient real time inference. Recognized gestures are transmitted wirelessly to a mobile application via Bluetooth Low Energy “BLE”, where they are displayed as text and converted into speech. An additional application level processing layer is introduced to arrange recognized words into meaningful sentences, enhancing communication effectiveness.

The proposed system demonstrates the feasibility of integrating artificial intelligence, embedded systems, and TinyML to develop an efficient assistive technology solution. The Smart Glove offers a low cost, portable, and privacy preserving approach for sign language interpretation, with potential applications in daily communication, education, and social inclusion for the speech impaired community.

Chapter 1: Introduction

1.1 General Introduction

Communication is a fundamental human need, millions of people with hearing or speech impairments face daily challenges when interacting with the rest of society.

Sign language is an effective and widely used means of communication for the deaf and mute community however, the lack of understanding of sign language among the general population creates a significant communication gap. This barrier affects many aspects of life, including education, employment, healthcare, and social interaction, highlighting the need for practical and accessible technological solutions.

1.2 Objectives of the project

The purpose of this graduation project is to design and develop a smart wearable glove that can recognize sign language gestures and translate them into readable text and audible speech in real time.

The proposed system aims to capture hand movements using sensors embedded in the glove, process these signals using a microcontroller and artificial intelligence techniques and then transmit the recognized gestures through a speaker. Through this approach the project seeks to provide an efficient, user-friendly, and portable communication tool that bridges the gap between sign language users and non-sign language users.

1.3 Importance of the project.

The importance of this project lies in its practical engineering application and its relevance to current market trends in wearable technology and assistive systems.

There is an increasing demand for low cost, portable, and real time gesture recognition devices driven by growth in smart wearables and AI enabled healthcare technologies. By combining sensor fusion, embedded system design, wireless communication and machine learning the proposed smart glove offers a scalable

and cost effective solution that can be further developed for commercial and medical applications. The project also provides a strong platform for applying theoretical electrical engineering concepts in signal processing, embedded programming, and system integration.

1.4 Organization of the report.

This report is organized as follows: the first chapter introduction of the project. The second chapter explains the theoretical background. The third chapter explain Methodology including System Architecture, hardware selection and software design. The fourth chapter discusses Artificial Intelligence Methodology and System Implementation. The fifth chapter review Results and Analysis for the simulation of the project. The sixth chapter summarizes the project and suggests potential improvements and future work. The final chapter for conclusion and references.

Chapter 2: Theoretical Background and Previous Work

2.1 Overview of Sign Language Recognition Systems

Sign language recognition systems aim to translate hand gestures and movements into meaningful linguistic representations such as text or speech. From an engineering perspective, these systems are classified into two main categories: vision-based systems and sensor-based systems. Vision-based systems rely on cameras and image processing techniques, which often suffer from environmental limitations such as lighting conditions, background complexity, and high computational requirements. In contrast, sensor-based systems utilize wearable devices equipped with sensors to directly measure hand motion and finger bending, offering higher robustness, portability, and real-time performance. The proposed smart glove adopts a sensor-based approach, making it more suitable for embedded applications where reliability, low latency, and power efficiency are critical design constraints.

2.2 Wearable Sensor Technology for Gesture Recognition

Wearable sensor technology plays a key role in capturing hand gestures accurately. In smart glove systems, multiple sensors are integrated to measure different aspects of hand movement:

1. **Flex Sensors:** These sensors change their resistance when bent and are commonly used to measure finger curvature. The resistance variation is converted into voltage signals using voltage divider circuits, allowing the microcontroller to read finger positions.
2. **Accelerometers:** Accelerometers measure linear acceleration along three axes (X, Y, Z) and are useful for detecting hand motion and orientation changes.
3. **Gyroscopes:** Gyroscopes measure angular velocity and provide information about rotational movements of the hand.

Inertial Measurement Units (IMUs): IMUs combine accelerometers and gyroscopes in a single module, improving motion tracking accuracy through sensor fusion techniques.

The combination of these sensors enables the system to capture both static gestures “finger positions” and dynamic gestures “hand movement and orientation”.

2.3 Embedded Systems and Microcontroller Architecture

An embedded system is the core component responsible for data acquisition, preprocessing, and communication. In this project, a microcontroller is used to:

1. Read analog and digital sensor signals.
2. Perform signal conditioning and noise filtering.
3. Normalize and package sensor data.
4. Transmit data wirelessly to an external processing unit.

Microcontroller selection is based on criteria such as processing speed, number of input/output pins, power consumption, and communication capabilities. Commonly used microcontrollers for wearable applications include Arduino-based platforms and ESP-series controllers, which support analog to digital conversion “ADC” and integrated wireless communication.

2.4 Signal Processing and Data Preprocessing

Raw sensor data obtained from wearable sensors often contains noise and fluctuations caused by hand tremors and environmental interference. Therefore, signal preprocessing is essential to improve system accuracy. Common preprocessing techniques include:

1. Low-pass filtering to remove high-frequency noise.
2. Signal normalization to reduce variations between different users.
3. Feature extraction to reduce data dimensionality while preserving gesture characteristics.

These processed features serve as inputs to the gesture classification algorithm, improving recognition reliability and reducing computational load.

2.5 Artificial Intelligence Techniques for Gesture Classification

Artificial intelligence and machine learning algorithms are widely used in sign language recognition systems to classify gestures based on extracted features. Popular techniques include:

1. Artificial Neural Networks (ANN)
2. Support Vector Machines (SVM)
3. Decision Trees
4. Deep Learning models for complex gesture patterns

In sensor-based systems, supervised learning is typically employed, where the model is trained using labeled gesture data.

The trained model can then classify new gestures in real time with high accuracy. AI-based classification significantly enhances system performance compared to rule based approaches, especially when dealing with variations in hand size and movement style.

2.6 Wireless Communication and User Interface

Wireless communication enables real time data transfer between the smart glove and external devices such as smartphones or computers.

Technologies such as Bluetooth and Wi-Fi are commonly used due to their low power consumption and ease of integration.

Once gestures are recognized, the output is displayed as text and converted into speech using text-to-speech (TTS) technology. This human-machine interface is critical for practical usability, allowing seamless communication between sign language users and non-sign language users.

2.7 Previous Work and Related Research

Several research studies have proposed smart glove systems for sign language translation. Early designs relied mainly on flex sensors and simple threshold-based classification, which limited accuracy. More recent works have integrated IMUs and machine learning algorithms, achieving higher recognition rates and real-time performance.

However, many existing systems suffer from high cost, complex hardware design, or limited portability. Some solutions also require external cameras or extensive calibration. These limitations motivate the development of a compact, cost-effective, and AI-enabled smart glove, as proposed in this project, which aims to balance accuracy, usability, and affordability.

Chapter 3: Methodology



3.1 System Architecture

The system consists of two smart gloves, each containing an ESP32 microcontroller, flex sensors, and an IMU.

One ESP32 acts as a master device receiving data wirelessly from the second glove.

3.2 Data Acquisition

Each ESP32 collects analog data from flex sensors and digital motion data from the IMU using the I2C protocol. Sampling is performed at a fixed rate to ensure consistency.

3.3 Wireless Communication

ESP-NOW is used for inter-glove communication due to its low latency and power efficiency. Bluetooth Low Energy is used for communication with the mobile application.

3.4 Data Processing Pipeline

Sensor data undergoes filtering, feature extraction, normalization, and classification using machine learning models.

3.4.1 Local Processing on ESP32

Each ESP32 performs initial data processing to reduce noise and minimize wireless bandwidth usage. The following steps are applied:

- Analog-to-digital conversion of flex sensor signals
- IMU data acquisition via I2C protocol
- Noise filtering using moving average filters
- Feature extraction (mean and variance over short time windows)
- Normalization of sensor values to a range of 0–1

Only the processed feature vectors are transmitted wirelessly, ensuring efficient communication.

3.4.2 Inter-Glove Data Synchronization

Wireless communication between the two gloves is achieved using the ESP-NOW protocol. Each data packet includes a timestamp to allow synchronization between left and right hand data. The master ESP32 aligns incoming packets and merges them into a single feature vector representing both hands.

3.4.3 Data Transmission to Mobile Application

The master ESP32 operates as a Bluetooth Low Energy (BLE) server. The mobile application connects as a BLE client and subscribes to a custom characteristic used for streaming gesture data. Data packets are transmitted at fixed intervals, providing real-time performance with low power consumption.

3.4.4 Mobile Application Data Processing

The mobile application receives BLE packets and processes them using the following pipeline:

1. Data buffering and windowing
2. Machine learning inference using a trained classification model
3. Post-processing to remove duplicate gestures
4. Natural language processing to form meaningful sentences

3.4.5 Output Generation

The final output is presented as readable text on the mobile interface. An optional text-to-speech module converts the text into audio output, enhancing accessibility.

3.5 Standards and Specifications

The project follows relevant standards including IEEE 802.11 “WI-FI” , Bluetooth Low Energy specifications, and I2C communication standards.

3.6 Constraints

Design constraints include cost limitations, power consumption, social and ethical considerations, safety, manufacturability, and sustainability.

3.7 Data Collection and Preparation Methodology:

Step 1: Definition of Gesture Classes

The first step in the data collection process is defining the gesture classes to be recognized by the system.

The project supports a Word-based gesture classification scenario such as: Hello, Thanks, Yes,

Each class is recorded multiple times to ensure dataset diversity. A sufficient dataset size is maintained, with at least 200–300 samples per gesture to improve model generalization and classification accuracy.

Step 2: Glove Preparation for Data Recording

The glove is prepared with the following hardware components:

1. 10 Flex Sensors

Provide analog readings corresponding to finger bending

2. 2 IMU Sensors

Each IMU provides:

Accelerometer (AccX, AccY, AccZ)

Gyroscope (GyroX, GyroY, GyroZ)

The total number of sensor values per sample is:

10 Flex + 12 IMU = 22 values per sample

Step 3: Arduino Code Development for Data Acquisition

An Arduino-based program is developed for the ESP32 to perform the following tasks:

1. Read all 22 sensor values at a fixed sampling rate

2. Apply basic filtering to reduce noise
3. Transmit the sensor readings via Serial communication
4. Format the output in a structured, comma-separated format (CSV-compatible)

Step 4: Data Recording on Computer

The transmitted serial data is captured on a computer using serial monitoring tools. The data is saved as CSV files, where:

1. Each row represents one sample
2. Each column corresponds to a specific sensor value
3. An additional column is used to store the gesture label

This format facilitates further processing and machine learning training.

Step 5: Repeated Data Collection

To improve dataset robustness and variability, data recording is repeated multiple times while:

1. Slightly changing hand posture
2. Adjusting glove tightness
3. Modifying environmental conditions such as lighting and surroundings

These variations help the model handle real-world usage conditions.

Step 6: Data Cleaning

The collected dataset is cleaned to remove inconsistent or invalid samples:

1. Sensor values are checked to ensure they fall within normal operating ranges
2. Corrupted or incomplete records are discarded
3. Outliers caused by sensor errors are filtered out

Step 7: Data Normalization

To ensure uniform scaling across all sensors, normalization is applied:

1. Minimum and maximum values are calculated for each sensor
2. All values are scaled to a range between **0 and 1** using min-max normalization

This step improves machine learning performance and training stability.

Step 8: Dataset Preparation for Training

The processed dataset is prepared for machine learning by:

1. Shuffling the data randomly
2. Splitting the dataset into training and testing subsets
3. Formatting the data according to the requirements of the selected ML framework

Step 9: Model Training and Conversion to TinyML

A machine learning model is trained using the prepared dataset. After achieving satisfactory performance:

1. The trained model is saved
2. The model is converted into a TinyML-compatible format
3. The optimized model is deployed either on the ESP32 or within the mobile application

This structured data collection and preparation pipeline ensures high-quality, well-labeled, and normalized data. The methodology supports reliable gesture recognition while maintaining compatibility with embedded and mobile-based TinyML deployment.

3.8 Components of the project:

Our project consists of two wearable gloves with a wireless connection between them. Using two gloves not only one enables us to get the most benefits from the project because using two gloves together enables us to get words and sentences from them not only letters while using one glove.

Components used in the gloves:

1- ESP 32

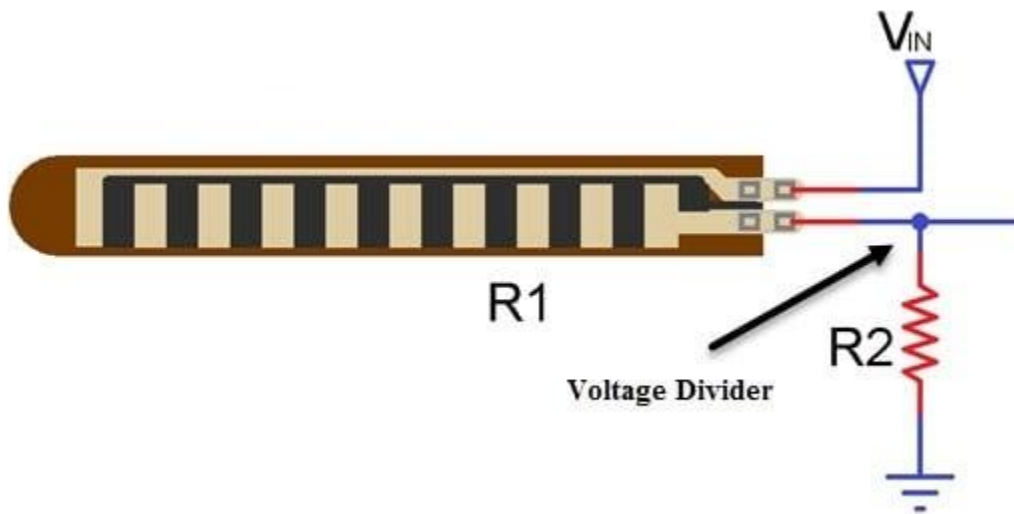


We chose ESP 32 as the core processing unit of the system to use it because it offers several advantages such as:

1. Integrated Wi-Fi and Bluetooth Low Energy “BLE” eliminating the need for external communication modules.
2. High processing capability with a dual core processor suitable for real time sensor data handling.
3. Low power consumption making it ideal for wearable applications.
4. Large number of GPIO pins supporting analog, digital, I2C, SPI, and UART interfaces.
5. Built in support for ESP-NOW enabling fast and reliable wireless communication between the two gloves.

All these features make the ESP32 an excellent choice for handling multi sensor data and wireless transmission efficiently.

2- Flex Sensors



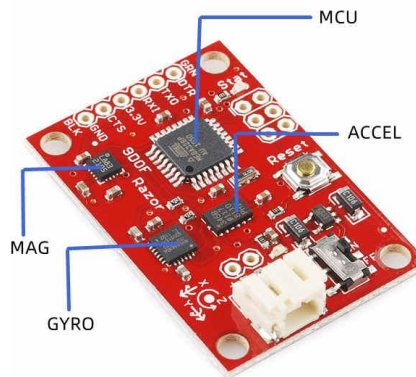
Flex sensors are used to detect finger bending and hand posture

Advantages of Flex Sensors:

1. Flexible and light weight makes it suitable for wearable gloves.
2. It provides continuous analog output allowing precise measurement of finger movement.
3. Easy to interface with microcontrollers using simple voltage divider circuits.
4. Low cost and low power consumption.

Overall Flex sensors enable accurate detection of finger gestures, which is essential for sign language recognition.

3- Inertial Measurement Unit “IMU”



IMU sensors consists of an accelerometer and a gyroscope they are used to capture both hands orientation and motion.

Advantages of IMU:

1. It provides 6 degrees of movements consists of 3 axis for accelerometer and 3 axis for gyroscope.
The accelerometer tracks linear acceleration such as gravity, while the gyroscope measures rotational movement.
2. High sensitivity and accuracy for motion tracking.
3. Communicate digitally via the I2C protocol, reducing noise compared to analog sensors.
4. Small size and low power consumption.
5. IMU data complements flex sensor data by capturing dynamic hand movements and rotations.

4- Lithium-Polymer (Li-Po) rechargeable battery



Each glove is powered by a 3.7V Lithium-Polymer (Li-Po) rechargeable battery, chosen for the following reasons:

1. High energy density relative to weight.
2. Compact size suitable for wearable devices.
3. Rechargeable and widely available.
4. Capable of supplying sufficient current for ESP32 operation.
5. Compatible with common charging ICs.

5- Spandex or Lycra Gloves



The best gloves for attaching the Flex sensors should be thin, flexible and breathable fabric, all these requirements are found in Spandex or Lycra.

6- Speaker



Used as an output for the project and it receives the sentences from the mobile application after processing and then broadcasting them.

7- Mobile Application

Receives the data collected from Flex sensors and IMU then processing it to generate words and sentences then transfer the final result to the speaker.

3.9 Power Supply and Battery Management System

The smart glove is powered by a 3.7V lithium-polymer rechargeable battery integrated into each glove. This type of battery was selected due to its high energy density, compact size, and suitability for wearable applications. A capacity range of 1000–2000 mAh is considered sufficient to provide several hours of continuous operation.

Because the battery voltage varies between 3.0V and 4.2V, a high-efficiency buck-boost DC-DC converter is employed to generate a regulated 3.3V supply for the ESP32 microcontroller and all connected sensors. Switching regulators were selected over linear regulators to minimize power losses and extend battery lifetime.

Battery charging is achieved using a TP4056-based charging module powered from a standard 5V USB input. The charger incorporates dedicated protection circuitry against over-charge, over-discharge, over-current, and short-circuit conditions to ensure safe operation.

The estimated system current consumption ranges from 230 to 320 mA during active use, resulting in an expected operating time of approximately 5–7 hours when using a 1500 mAh battery. Through the implementation of power-saving techniques such as deep-sleep modes, adaptive sampling, and wireless duty cycling, the operating time can be extended beyond 8 hours.

Safety measures include continuous battery voltage monitoring, thermal protection, electrical insulation, and secure mechanical placement of the battery inside the glove enclosure.

Chapter 4: Artificial Intelligence Methodology and System Implementation

4.1 Introduction

This chapter presents the complete methodology used to integrate Artificial Intelligence “AI” into our project. The integration of AI enables the system to automatically learn gesture patterns from sensors data and perform accurate real time classification.

The chapter covers data acquisition, preprocessing, model design, training, TinyML deployment, and system-level integration.

4.2 Problem Formulation

The Smart Glove system addresses a supervised learning classification problem. The objective is to map multidimensional sensor data obtained from the glove to a predefined set of sign language gestures.

Let:

1. $X \in \mathbb{R}^{22}$ be the input feature vector representing sensor readings.
2. $Y \in (1, 2, \dots, C)$ be the gesture class label.

The goal is to learn a function $f(X) = Y$ that minimizes classification error while maintaining low computational complexity suitable for embedded systems.

4.3 Sensor Data Acquisition and Feature Representation

4.3.1 Sensor Configuration

The glove integrates multiple sensors to capture hand motion and orientation consists of :

1. Ten flex sensors to measure finger bending angles.
2. Two Inertial Measurement Units (IMUs), each providing three-axis accelerometer and gyroscope data.

4.3.2 Feature Vector Construction

Each gesture instance is represented by a 22-dimensional feature vector composed of 10 flex sensor values and 12 IMU values. This representation preserves both static finger posture and dynamic hand movement information.

4.4 Dataset Collection and Labeling

Gesture data is collected in real time while the user performs predefined sign language gestures wearing the glove. For each gesture instance the ESP32 microcontroller samples all sensor values and transmits them to a host computer for storage.

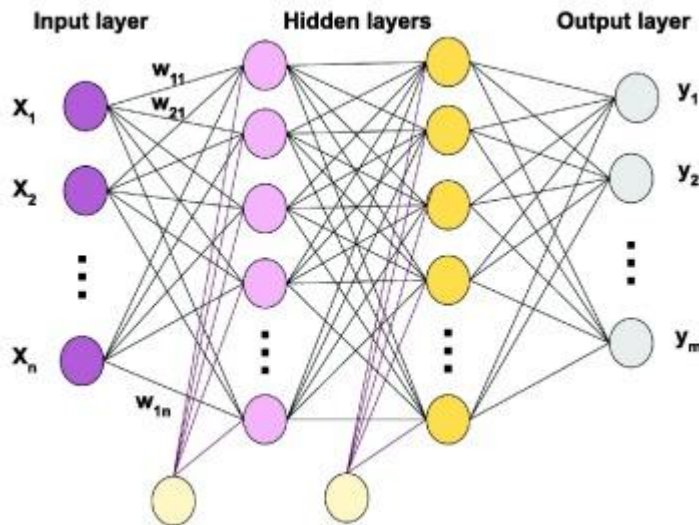
Each gesture class contains 200–300 samples stored in CSV format, ensuring sufficient variability and robustness.

4.5 Data Preprocessing

All sensor readings are normalized using Min–Max scaling to ensure uniform feature scaling. Gesture labels are encoded numerically, and the dataset is split into training (80%) and testing (20%) subsets.

4.6 Machine Learning Model Design

A lightweight Multi-Layer Perceptron “MLP” neural network is selected due to its suitability for embedded systems and tabular sensor data. The network consists of an input layer with 22 neurons, two hidden layers (32 and 16 neurons), and an output layer corresponding to the number of gesture classes.



4.7 Model Training and Evaluation

The model is trained offline using TensorFlow/Keras with categorical cross entropy loss and the Adam optimizer. Model performance is evaluated using classification accuracy on the testing dataset.

4.8 TinyML Deployment on ESP32

The trained model is converted to TensorFlow Lite format, quantized to INT8 precision, and embedded into the ESP32 firmware as a C array. During operation, the ESP32 performs sensor sampling, normalization, inference, and gesture classification in real time.

4.9 Communication and Application-Level Processing

The predicted gesture is transmitted to a mobile application via Bluetooth Low Energy “BLE”. The application displays the recognized word and send it back to the glove to converts it to speech.

4.10 Sentence Formation and Contextual Processing

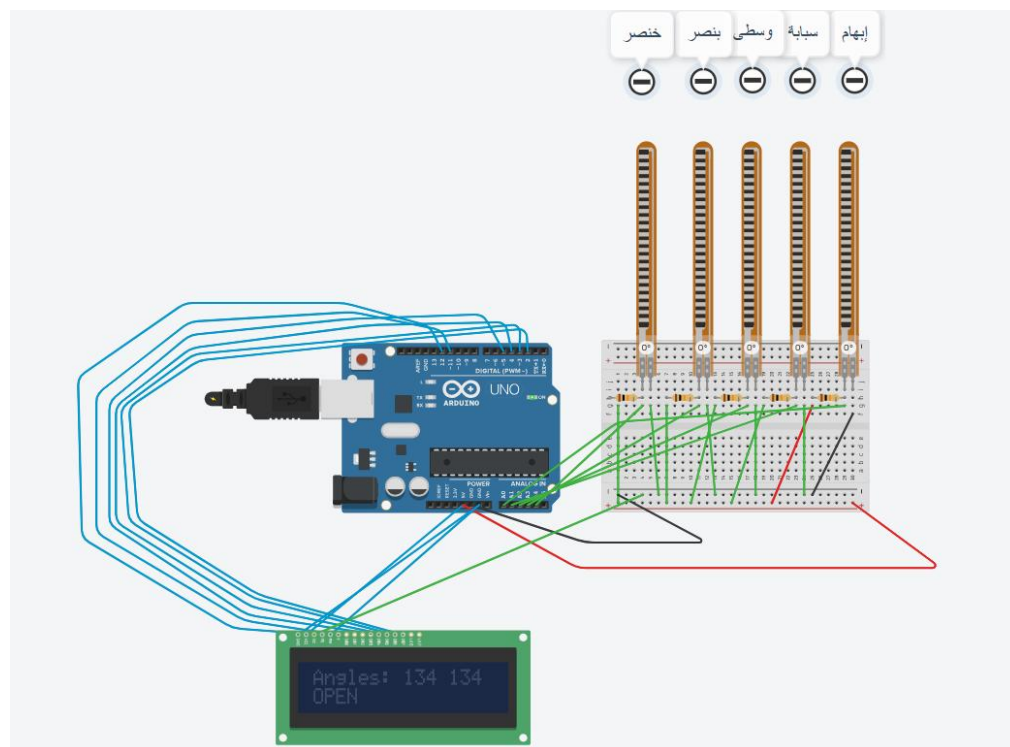
Recognized words are buffered and processed using rule-based natural language techniques to generate grammatically correct sentences, improving communication effectiveness.

Chapter 5: Results and Analysis

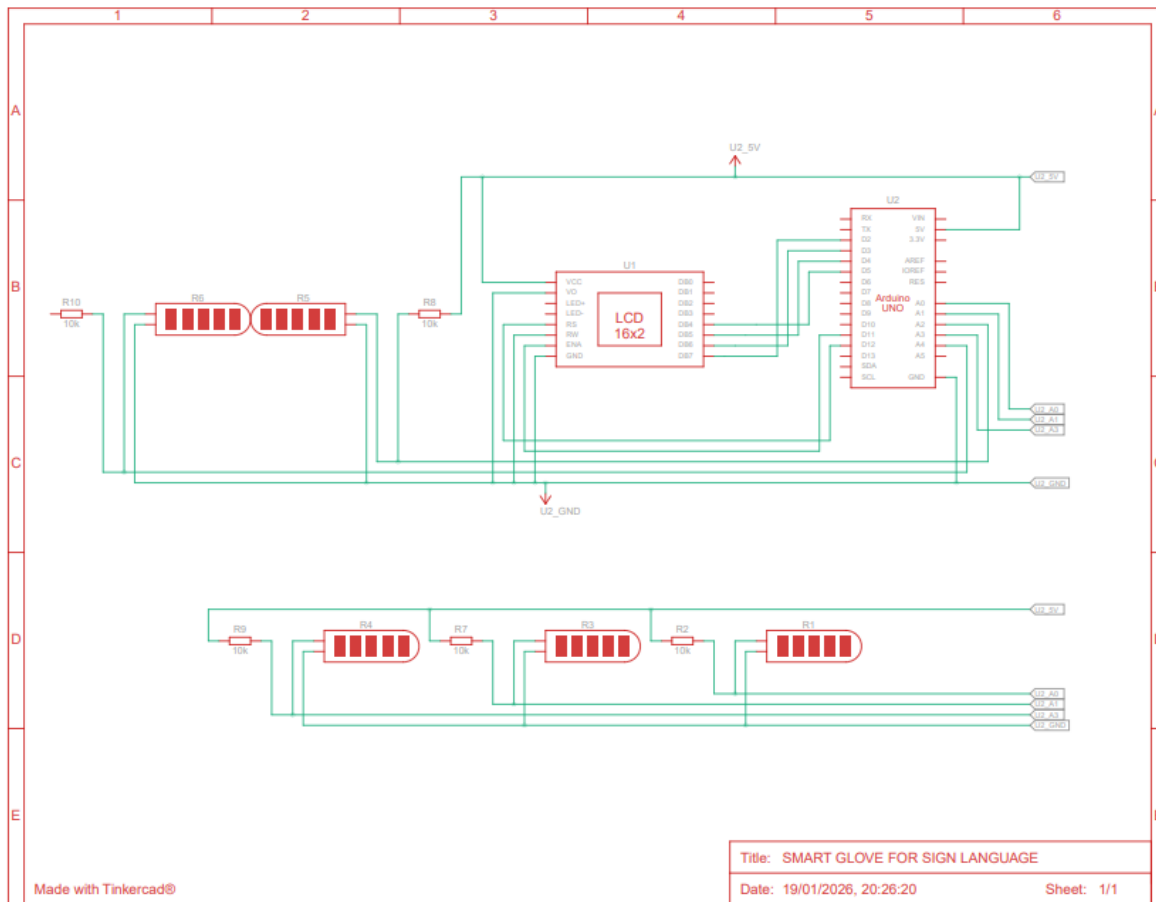
We faced a lot of difficulties while simulating our project including the lack of simulation programs and the existing programs are missing some components

First we used proteus but it didn't support the ESP 32 and the flex sensors, even after installing the libraries for ESP 32 and Flex sensors always there been a problem preventing the simulation so we tried to use Multisim and we faced the same problem so we used Tinkercad instead and used Arduino UNO as process unit instead of ESP 32 and we didn't find the IMU and speaker so we used LCD screen to display the results, overall we obtained promising results.

5.1 Circuit Diagram:



5.2 Schematic Diagram of the Circuit:



5.3 Simulation

The simulation circuit consists of the following components:

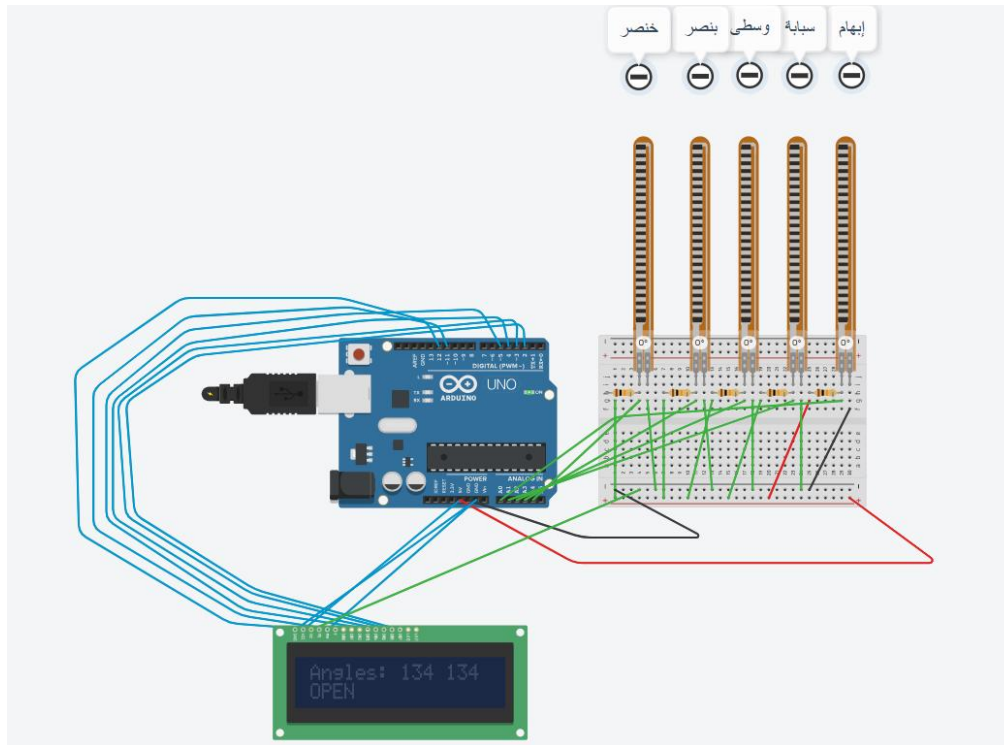
1. Arduino UNO as the process unit of the project.
2. Voltage divider circuit for each flex sensor.
3. LCD screen to show off the results.

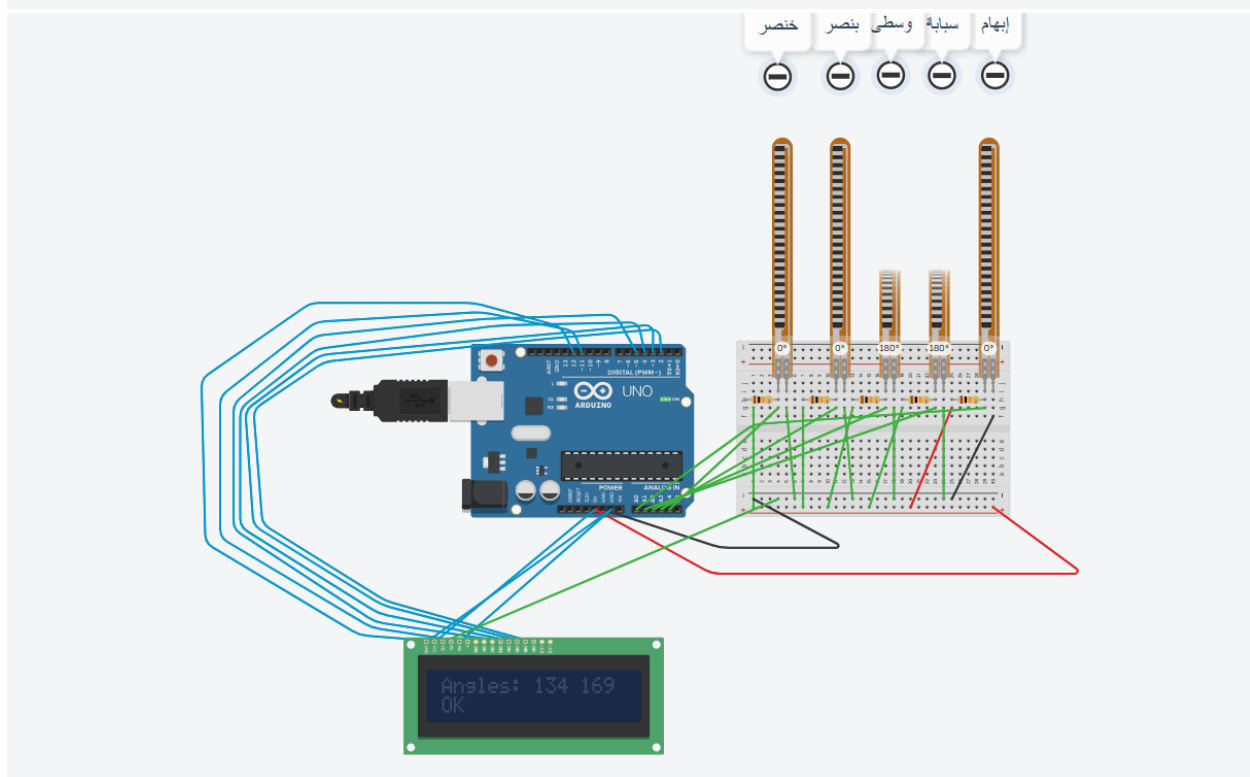
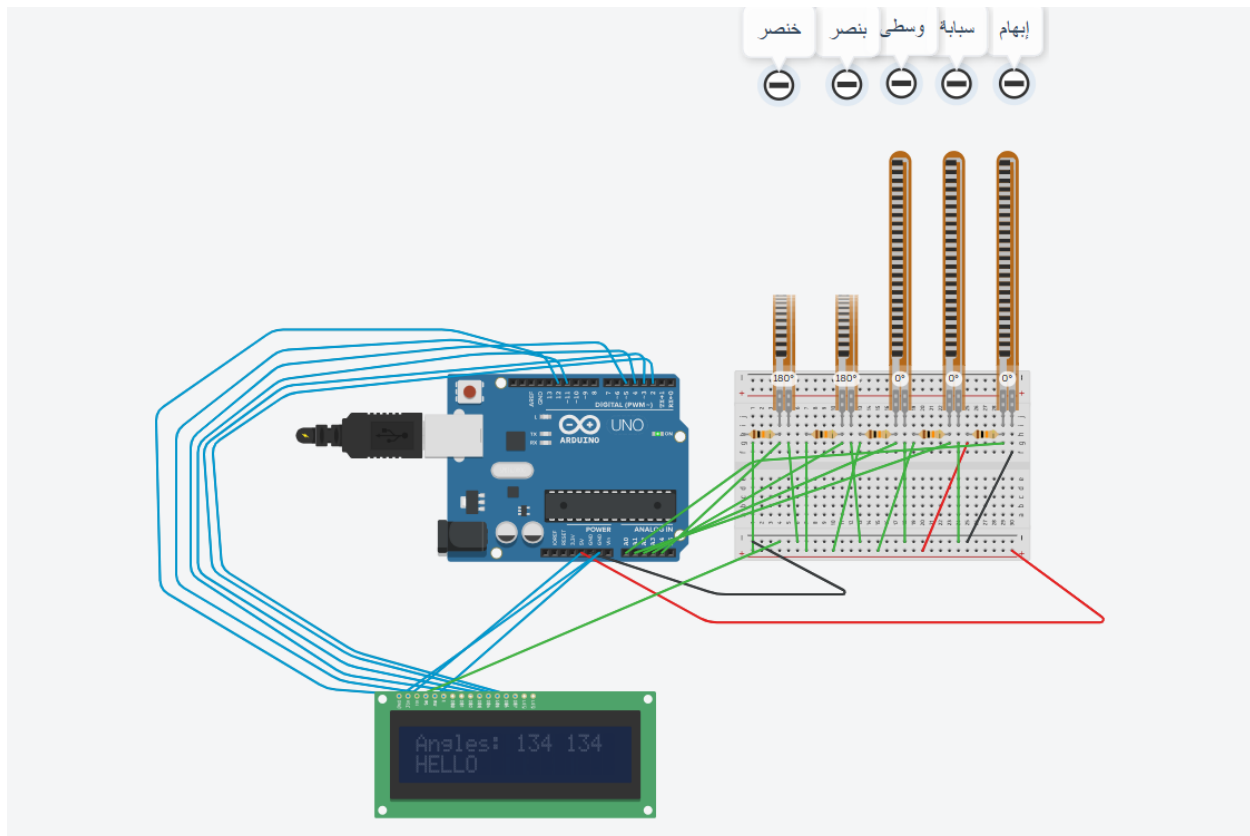
With this circuit we get a promising results using a trial code for this simulation that process the analog inputs data from the flex sensors and gives words results according to pre set data for words in the code.

The pre sets data are combined from the 5 sensors which mean every sensor should be in a different position according to the pre set value to make a word and every word has a unique set of values to achieve. This principle is the core principle of our project when

we will build the final project in Graduation Project 2. Every word has its unique pattern of values and the word can't be achieved without fulfill all the required values.

There are some results from the circuit we build:





We can see that we can achieve different word outputs according to the inputs of the circuit from the flex sensors.

5.4 Simulation Code

The code we used:

```
#include <LiquidCrystal.h>

// LCD Pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// Flex pins
const int flexPins[] = {A0, A1, A2, A3, A4};
int fVal[5];
int angles[5]; // 0-180 degree
int th[5] = {150, 150, 150, 150, 150}; // threshold for each finger

void setup() {
  lcd.begin(16, 2);
  lcd.print("Ready...");
  delay(500);
}

void loop() {
  // قراءة وتحويل كل سينسور إلى زاوية
  for (int i = 0; i < 5; i++) {
    fVal[i] = analogRead(flexPins[i]);
    angles[i] = map(fVal[i], 0, 1023, 0, 180); // تحويل ADC إلى زاوية
  }

  lcd.setCursor(0, 0);
  lcd.print("Angles: ");
  for(int i=0;i<5;i++){
    lcd.print(angles[i]);
    lcd.print(" ");
  }

  lcd.setCursor(0, 1);
```

```

// أمثلة لكلمات حسب الزوايا
if (angles[0]<th[0] && angles[1]<th[1] && angles[2]<th[2] && angles[3]<th[3] &&
angles[4]<th[4])
    lcd.print("OPEN ");

else if (angles[0]>th[0] && angles[1]>th[1] && angles[2]>th[2] && angles[3]>th[3] &&
angles[4]>th[4])
    lcd.print("FIST ");
else if (angles[0]<th[0] && angles[1]>th[1] && angles[2]>th[2] && angles[3]>th[3] &&
angles[4]>th[4])
    lcd.print("YES ");
else if (angles[0]<th[0] && angles[1]<th[1] && angles[2]>th[2] && angles[3]>th[3] &&
angles[4]<th[4])
    lcd.print("PEACE ");
else if (angles[0]<th[0] && angles[1]<th[1] && angles[2]<th[2] && angles[3]>th[3] &&
angles[4]<th[4])
    lcd.print("HELLO ");
else if (angles[0]>th[0] && angles[1]<th[1] && angles[2]<th[2] && angles[3]<th[3] &&
angles[4]>th[4])
    lcd.print("THANKS ");
else if (angles[0]<th[0] && angles[1]>th[1] && angles[2]<th[2] && angles[3]<th[3] &&
angles[4]>th[4])
    lcd.print("LOVE ");
else if (angles[0]>th[0] && angles[1]>th[1] && angles[2]<th[2] && angles[3]<th[3] &&
angles[4]<th[4])
    lcd.print("HI ");
else if (angles[0]<th[0] && angles[1]>th[1] && angles[2]>th[2] && angles[3]<th[3] &&
angles[4]<th[4])
    lcd.print("OK ");
else
    lcd.print("UNKNOWN ");

delay(200);
}

```

Chapter 6: Standards and Constraints

6.1 Introduction

This chapter outlines the engineering standards and practical constraints considered during the design and implementation of the Smart Glove for Sign Language Recognition. Adhering to relevant standards ensures system reliability, safety, and interoperability, while identifying constraints clarifies the design limitations and tradeoffs made throughout the project.

6.2 Engineering Standards

6.2.1 Embedded Systems Standards

The design of the AI Glove follows general embedded systems engineering principles, including modular design, efficient resource utilization, and real-time operation. The system architecture aligns with IEEE recommendations for embedded and real-time systems to ensure maintainability and scalability.

6.2.2 Software Quality Standards

Software development practices are guided by ISO/IEC 25010, which defines quality characteristics such as reliability, efficiency, maintainability, and usability. These standards ensure that both the embedded firmware and the mobile application meet acceptable software quality levels.

6.2.3 Wireless Communication Standards

Bluetooth Low Energy (BLE) communication follows Bluetooth SIG specifications (Bluetooth 5.0), ensuring low power consumption, secure data transmission, and compatibility with modern mobile devices.

6.2.4 Artificial Intelligence and Ethical Standards

The AI component is developed in accordance with ethical AI principles, emphasizing transparency, fairness, and user safety. The system is designed to operate offline, preserving user privacy by preventing unauthorized data sharing.

6.2.5 Assistive Technology Standards

The project aligns with international assistive technology guidelines, including World Health Organization (WHO) recommendations, ensuring accessibility, inclusiveness, and usability for individuals with hearing or speech impairments.

6.3 System Constraints

6.3.1 Hardware Constraints

The ESP32 microcontroller has limited processing power, memory, and storage capacity. These limitations restrict the complexity and size of the machine learning model, necessitating the use of lightweight TinyML architectures and quantized models.

6.3.2 Power Consumption Constraints

As the glove is a wearable device, power consumption is a critical constraint. All components, including sensors, microcontroller, and wireless communication modules, are selected to minimize energy usage and extend battery life.

6.3.3 Sensor Accuracy and Noise

Flex sensors and IMUs are subject to noise, drift, and calibration errors. These factors may affect data quality and classification accuracy, requiring preprocessing and normalization techniques to mitigate their impact.

6.3.4 Dataset Constraints

The performance of the AI model is directly influenced by the size and diversity of the collected dataset. Limited data samples or restricted user diversity may reduce generalization capability.

6.3.5 Real Time Processing Constraints

The system must perform gesture recognition in real time. This requirement limits computational latency and imposes strict timing constraints on data acquisition, preprocessing, and inference stages.

6.4 Design Trade Offs

To address the identified constraints, several trade-offs were made:

1. Model accuracy versus computational complexity

2. Power consumption versus processing speed
3. Number of recognized gestures versus system memory usage

These tradeoffs were carefully balanced to achieve optimal system performance within hardware and energy limitations.

Chapter 7: Discussion

7.1 Primary goal

Design a wearable system capable of recognizing sign language gestures and translating them into text and speech in real time.

Based on the experimental results presented in the previous chapter, this objective was largely achieved. The system successfully captured hand and finger movements using embedded sensors and correctly classified a predefined set of gestures with acceptable accuracy under controlled conditions. These outcomes indicate that the proposed solution effectively addresses the communication challenge targeted by the project.

In terms of contribution, this work presents a practical and low-cost sensor-based approach to sign language recognition that integrates wearable hardware, embedded processing, and machine learning techniques. Unlike vision-based systems reported in the literature, which are sensitive to lighting conditions and background complexity, the proposed glove operates independently of environmental visual constraints. The system also demonstrates reduced computational requirements, making it suitable for portable and embedded applications.

This combination of simplicity, portability, and real-time operation represents a meaningful contribution to assistive communication technologies.

When compared with previous studies, the obtained results are consistent with current findings that sensor based wearable systems can achieve reliable gesture recognition performance for vocabularies. However, similar to existing systems, the proposed design exhibits limitations related to user variability and gesture complexity. Recognition accuracy may be affected by differences in hand size, sensor placement, and gesture execution style. Additionally. Our goal is to reduce these challenges and make a better version of the glove to help make the world a better place for everyone.

The logical implications of these results suggest that smart glove based systems are a feasible solution for real time sign language recognition when designed with appropriate sensor selection, signal preprocessing, and classification algorithms. The findings reinforce the suitability of wearable sensor technology for assistive applications and highlight the tradeoff between system complexity and recognition performance.

7.2 Future Work

Future work includes expanding the gesture dataset, implementing advanced deep learning models for improved generalization, and incorporating adaptive calibration techniques for different users. Additional applications may extend to educational tools for learning sign language, healthcare communication aids, and human computer interaction systems where gesture based input is required.

Chapter 8: Conclusions and Recommendation

This project presented the design and implementation of a smart glove system for real-time sign language recognition and translation into text and speech. The obtained results demonstrate that a wearable, sensor-based approach can effectively address communication barriers between sign language users and non-sign language users.

The developed system successfully captured hand and finger movements using flex sensors and inertial measurement units. These signals were processed and classified using an artificial intelligence-based model, enabling accurate recognition of a predefined set of gestures. The results confirm that the project objectives were achieved, particularly in terms of real-time performance, portability, and cost-effectiveness.

The main conclusions drawn from this project are that wearable sensors can reliably represent sign language gestures, artificial intelligence enhances classification accuracy, and a low-cost hardware design can deliver practical assistive functionality. These findings support the feasibility of deploying smart glove systems for real-world communication assistance.

References:

Artificial Intelligence & Machine Learning

1. C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
2. I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, 2016.
3. K. P. Murphy, Machine Learning: A Probabilistic Perspective, MIT Press, 2012.
4. S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 4th ed., Pearson, 2021.

TinyML & Embedded AI

1. P. Warden and D. Situnayake, TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers, O'Reilly Media, 2019.
2. TensorFlow Team, “TensorFlow Lite for Microcontrollers,” 2023.
3. A. Banbury et al., “Benchmarking TinyML Systems: Challenges and Directions,” IEEE Micro, vol. 40, no. 5, pp. 65–75, 2020.

ESP32 & Embedded Systems

1. Espressif Systems, ESP32 Technical Reference Manual, Espressif Systems, 2023.
2. J. Ganssle, The Art of Designing Embedded Systems, 2nd ed., Newnes, 2008.
3. M. Barr and A. Massa, Programming Embedded Systems, O'Reilly Media, 2006.

Sensors (Flex & IMU)

1. R. S. Dahiya et al., “Tactile Sensing—From Humans to Humanoids,” IEEE Transactions on Robotics, vol. 26, no. 1, pp. 1–20, 2010.
2. InvenSense, MPU-6050 Six-Axis Gyroscope and Accelerometer Datasheet, 2019.
3. J. L. Crowder, “Flexible Sensors for Human Motion Detection,” Sensors Journal, vol. 18, no. 4, pp. 1–12, 2018.
- 4.

Gesture & Sign Language Recognition

1. S. Mitra and T. Acharya, “Gesture Recognition: A Survey,” IEEE Transactions on Systems, Man, and Cybernetics, vol. 37, no. 3, pp. 311–324, 2007.

2. A. Elons, M. Abull-Ela, and M. Tolba, “Hand Gesture Recognition Using Multilevel Neural Networks,” *IEEE Transactions on Neural Networks*, vol. 24, no. 4, pp. 624–635, 2013.
3. R. Sharma et al., “Vision-Based Hand Gesture Recognition for Human–Computer Interaction,” *Pattern Recognition*, vol. 38, pp. 159–172, 2005.

Bluetooth & Wireless Communication

1. Bluetooth SIG, Bluetooth Low Energy Specification, Version 5.0, 2022.
2. Espressif Systems, “ESP-NOW Wireless Communication Protocol,” 2023.

Assistive Technology & Social Impact

1. World Health Organization (WHO), World Report on Disability, WHO Press, 2011.
2. G. Vanderheiden, “Assistive Technology: Toward Universal Access,” *IEEE Computer*, vol. 33, no. 5, pp. 32–40, 2000.

Academic & Engineering Standards

1. IEEE, IEEE Code of Ethics, IEEE Standards Association, 2020.
2. ISO/IEC, ISO/IEC 25010: Systems and Software Quality Models, 2011.