JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

B.TECH V SEMESTER

MINOR PROJECT ON

Wireless EMG-based Armband for wheelchair control using Electromyography (EMG) for PwD

Under the supervision of: Dr. Gaurav Verma

S.No	Name	Enrollment	Batch
1	Smriti Aggarwal	22102178	A7
2	Aditya Patil	22102174	A7
3	Abhishek Mittal	22102160	A7

TABLE OF CONTENTS

ACKNOWLEDGEMENT

We, **Smriti Aggarwal (22102178)**, **Aditya Patil (22102174)** and **Abhishek Mittal (22102160)** are highly grateful to **Dr. Gaurav Verma**, Professor, at Jaypee Institute of Information Technology for providing this opportunity to work on this project. The constant guidance and encouragement received from the **Electronics and Communication Engineering (ECE) Department, JIIT** has been of great help in carrying out the project work and is acknowledged with reverential thanks. We express gratitude to other faculty members of the Electronics and Communication Engineering department of JIIT for their intellectual support throughout the course of this work.

We also wish to extend our thanks to our seniors and other classmates for their insightful comments and constructive suggestions to improve the quality of this project work. Finally, We are indebted to all whosoever have contributed in this report work.

Signature(s) of Students

Smriti Aggarwal (22102178)

Aditya Patil (22102174)

Abhishek Mittal (22102160)

Dr. Gaurav Verma
(Supervisor)

CANDIDATE DECLARATION

We hereby declared that this submission is our own work and that, to the best of our knowledge and beliefs, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma from a university or other institute of higher learning, except where due acknowledgment is made in text.

Place: Noida

Date:

Signatures of the Students

Name of the Student: Smriti Aggarwal
Enrollment Number: 22102178

Name of the Student: Aditya Patil
Enrollment Number: 22102174

Name of the Student: Abhishek Mittal
Enrollment Number: 22102160

CERTIFICATE

This is to certify that the minor report entitled **"Wireless EMG-based Armband for wheelchair control using Electromyography (EMG) for PwD"** submitted by Smriti Aggarwal (22102178), Aditya Patil (22102174) and Abhishek Mittal (22102160) of B.Tech ECE, Jaypee Institute of Information Technology, Noida has been carried out by them under my supervision and guidance. This work has not been submitted partially or wholly to any other University or Institute for the award of any other degree or diploma.

Signature of Supervisor

Name of the Supervisor: Dr. Gaurav Verma

Designation: Professor

Department of Electronics and Communication Engineering

Jaypee Institute of Information Technology, Sector - 62
Noida-201309

Date:

ABSTRACT

This project focuses on the development of a wireless EMG-based armband designed to control a wheelchair, offering an intuitive and accessible Human-Machine Interface (HMI) that enhances the independence and mobility of individuals with physical disabilities. By utilizing electromyographic (EMG) signals, the system captures the electrical activity generated by skeletal muscles and processes these signals to control the motion of a robotic wheelchair. The custom-designed armband integrates Dry Electrodes for emg signal acquisition, ESP32 microcontrollers for wireless data transfer, and a bioamp for the amplification of emg signals enabling the transmission of muscle signals that are translated into directional commands for the wheelchair.

The primary goal of this system is to enable hands-free control of a wheelchair, making it more accessible for users with limited mobility. Initial testing with a toy car as a prototype demonstrated the system's capability to process muscle signals effectively, implement noise filtering techniques, and execute smooth motor control. The project leverages several key technological components, including a band-pass filter to isolate the EMG signal from unwanted noise, and an envelope smoothing algorithm that enhances signal accuracy, removing unwanted peaks while preserving important muscle activity patterns.

The wireless communication between the transmitter and receiver, powered by ESP32 microcontrollers, ensures real-time data transmission, offering immediate responsiveness to user commands. This feature makes the system flexible as users can operate the wheelchair with minimal physical effort, providing a new level of autonomy. The system is also designed to be adaptive, ensuring that the wheelchair responds to varying muscle movements and can be customized for each user's unique needs.

Looking forward, future work on the project will focus on refining the signal processing algorithms with advanced noise reduction techniques and integrating adaptive control algorithms that can learn and adapt to the user's muscle signals over time. Additionally, real-world testing in different environments will be conducted to ensure the system's robustness, scalability, and reliability, with a particular emphasis on ensuring that the wheelchair operates effectively in both indoor and outdoor settings. The ultimate aim is to develop an affordable and scalable solution that can be applied globally, empowering individuals with disabilities to regain a greater sense of mobility, independence, and quality of life. This project serves as a promising foundation for future advancements in assistive technologies, showcasing the potential of EMG-based control systems in creating a more inclusive and accessible world.

LIST OF FIGURES

LIST OF ABBREVIATIONS USED

1. **EMG** - Electromyography
2. **IOT** - Internet of Things
3. **HMI** - Human-Machine Interface
4. **ALS** - Amyotrophic Lateral Sclerosis
5. **ADC** - Analog-to-Digital Converter
6. **BLE** - Bluetooth Low Energy
7. **DC** - Direct Current
8. **ESP** - Espressif (Microcontroller)
9. **EXG** - Electrophysiology (used in Bio Amp EXG Pill)
10. **PWM** - Pulse Width Modulation
11. **TCP** - Transmission Control Protocol
12. **Wi-Fi** - Wireless Fidelity
13. **EEG** - Electroencephalogram
14. **ECG** - Electrocardiogram
15. **GPIO** - General Purpose Input/Output
16. **IDE** - Integrated Development Environment
17. **RTOS** - Real-Time Operating System
18. **VCC** - Voltage Common Collector
19. **GND** - Ground

Chapter 1: Introduction

1.1 Overview of Electromyography (EMG)

Electromyography (EMG) is a widely recognized technique for capturing and analyzing the electrical activity generated by skeletal muscles. When motor neurons send signals to muscle fibers, they produce electrical activity, recorded as EMG signals. These signals provide essential insights into muscle functionality and movement, making EMG a valuable tool across fields such as medical diagnostics, rehabilitation, and more recently, robotic control applications.

The non-invasive nature of EMG makes it a practical choice for gathering muscle activity data, especially for real-time applications where accuracy and responsiveness are paramount. By translating biological signals into machine-readable formats, EMG technology enables a new range of assistive and interactive applications, such as controlling robotic systems, prosthetics, and even gaming interfaces. The potential of EMG as a natural, intuitive control mechanism offers an unparalleled advantage in scenarios that require seamless human-machine interaction.

In this project, we harness EMG technology to interpret muscle activity for controlling a motorized toy car. The project explores the feasibility of converting specific muscle movements into directional commands, such as moving forward, backward, or turning. This study lays a foundation for more advanced uses, particularly in developing assistive devices for individuals with mobility impairments. The technology's integration into real-world applications, like a motorized wheelchair, could revolutionize assistive technology by providing intuitive and accessible movement solutions for those in need [1], [7], [8].

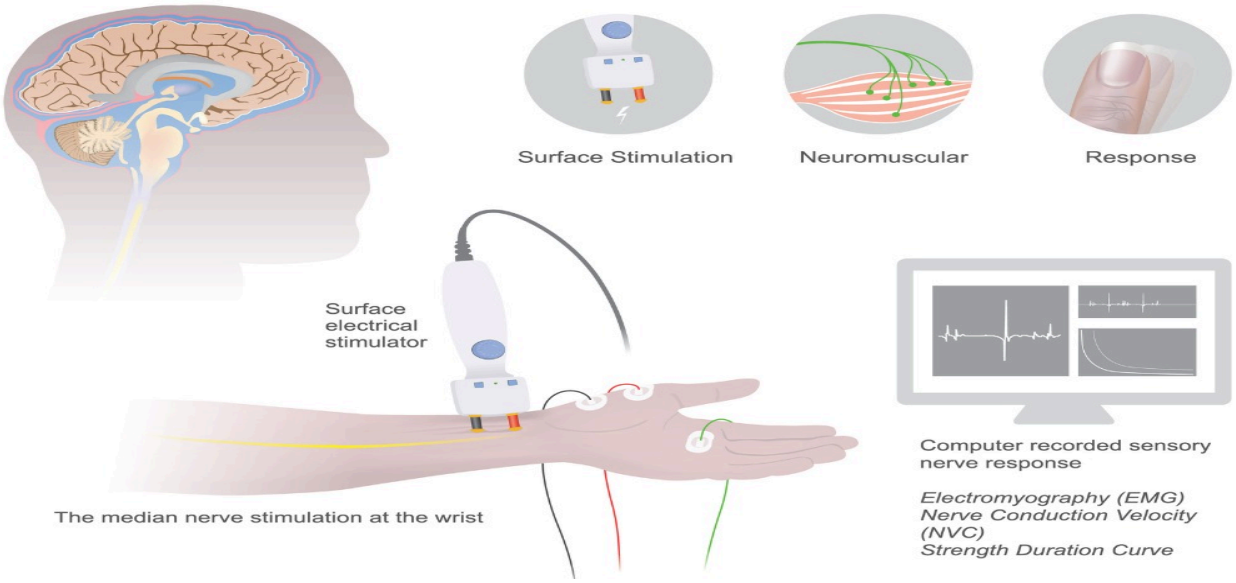


Fig 1.1 EMG Signal Acquisition

1.2 Project Objectives

The primary objective of this project is to design and implement an EMG-based control system that uses muscle movements to operate a toy car. By developing a functional prototype, this project aims to showcase the potential of EMG for intuitive and responsive motor control systems. Key objectives include:

1. **Signal Acquisition:** Detecting EMG signals from muscle movements using surface electrodes placed on specific muscles. These sensors capture the subtle electrical activity generated by muscle movements.
2. **Signal Amplification and Filtering:** Raw EMG signals are inherently weak and often contain noise, making it essential to amplify and filter these signals for accurate analysis. Using a bio-amplifier (bio-amp) and filters, we ensure the reliability of the processed signals.
3. **Command Mapping:** The processed EMG signals are translated into distinct commands, such as stopping, moving forward, backward, or turning. This involves calibrating the system to associate specific muscle movements with these predefined actions [1], [7].
4. **IoT Integration:** Using Internet of Things (IoT) technology, we enable communication between the EMG processing unit and the toy car. This involves setting up wireless communication between the processing module and the motor controller [5].
5. **Motor Control and Real-Time Response:** Ensuring smooth motor control and a quick response time is crucial. The system is designed to execute commands promptly, creating a responsive and intuitive experience for the user [8].

The broader goal of this project is to develop an EMG-based wheelchair control system for individuals with limited mobility. By translating muscle signals into mobility commands, this project aims to provide an innovative hands-free control solution that enhances independence and autonomy. In the long run, this technology could offer a dignified, accessible alternative for individuals with mobility challenges, enabling them to navigate and interact with their environment with ease[9].

1.3 Scope of the Project

This project highlights the intersection of Internet of Things (IoT), embedded systems, and bio-signal processing, showcasing their combined potential to address real-world mobility challenges. The immediate global impact of this project includes implementing and testing a functional system on a toy car to demonstrate its feasibility to be scaled to bigger systems. The scope includes the following phases:

1. **Hardware Development**: This phase involves setting up EMG sensors to detect muscle activity and integrating them with a bio-amplifier for signal amplification. Additionally, ESP microcontrollers will be used to process the signals and control the motors of the toy car.

2. **Signal Processing**: Filtering is essential for eliminating noise from raw EMG signals, making the data more reliable for command interpretation. We also calibrate threshold levels to ensure that specific muscle movements are consistently detected.

3. **Wireless Communication**: Establishing a wireless link between two ESP microcontrollers, one for EMG signal processing and the other for motor control, enables real-time data exchange between the sensors and the toy car. [4], [5].

4. **Motor Control**: Developing algorithms to map the processed signals to motor actions is a key aspect of this phase. The algorithms ensure that the toy car moves smoothly and accurately based on the interpreted muscle signals.

Future Developments and Expansions

Beyond the initial implementation on a toy car, this project envisions a future application of the technology on a motorized wheelchair, addressing practical concerns such as robustness, safety, and user-friendliness. Key developments for future expansion include:

1. **Enhanced Signal Interpretation**: Training the system to recognize more complex muscle patterns, allowing for a broader range of commands beyond basic directional control. This could involve integrating machine learning algorithms to refine signal interpretation and accommodate different user needs.

2. **Advanced Motor Integration**: For wheelchair applications, integrating high-power motors capable of handling larger loads and providing smooth mobility becomes necessary. This adaptation will require further refinement of motor control algorithms.

3. **Scalability and Adaptability**: Adapting the system to different users and environments, ensuring compatibility with various muscle strengths, signal patterns, and even diverse environmental conditions, is critical for practical use.[3]

4. **Real-World Testing and Validation**: Conducting extensive testing in real-world scenarios, particularly with individuals with disabilities, is crucial. This testing phase would validate the system's reliability, usability, and safety in real-world applications.

5. **User Interface and Customization**: Developing a user interface that allows for system calibration and customization according to user needs, ensuring the device is intuitive and easy to operate.

The ultimate aim of this project is to address the mobility challenges faced by individuals with physical disabilities, providing them with a solution that promotes independence and autonomy. By merging EMG technology with mechanical mobility, this project opens the door to future innovations in human-machine interaction, offering a new level of control and connectivity between biological and mechanical systems.

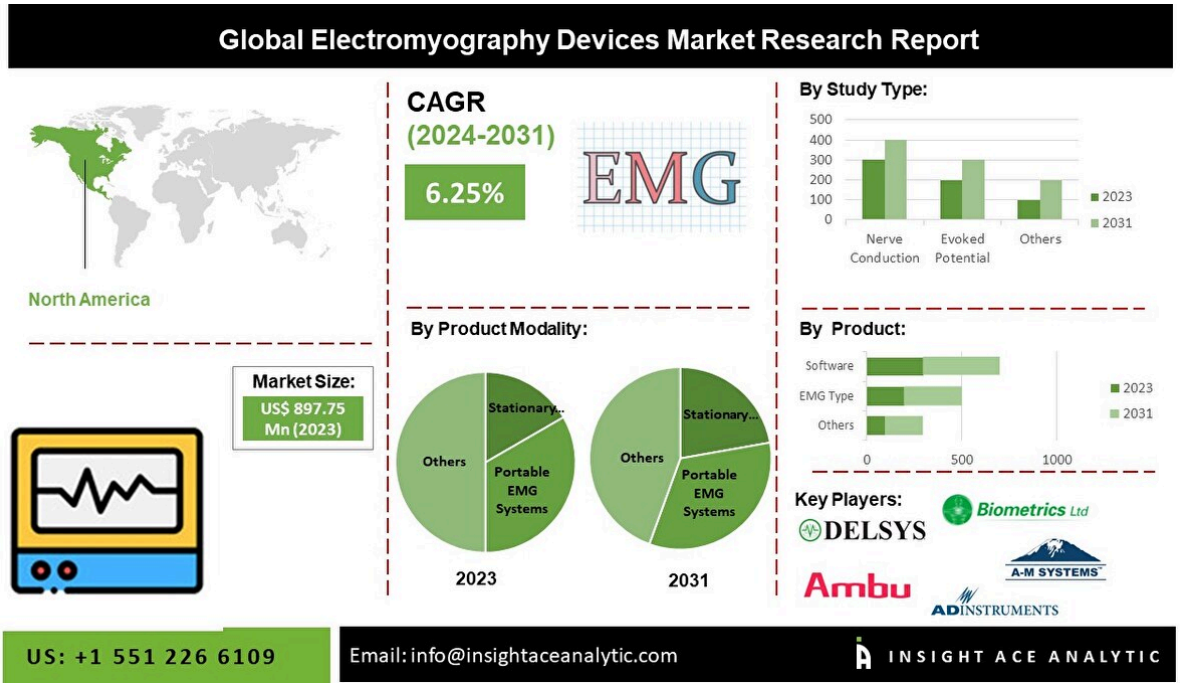


Fig 1.2 Global Usage of EMG Based Systems

Chapter 2: Literature Survey

2.1 Evolution of EMG Applications

Electromyography (EMG) has undergone a significant transformation, evolving from a diagnostic tool to a versatile technology in modern assistive devices and control systems. Originally, EMG was used primarily in medical diagnostics to evaluate neuromuscular disorders such as muscular dystrophy and ALS. It provided clinicians with crucial insights into muscle functionality, enabling them to design targeted treatment plans and rehabilitation strategies.

The scope of EMG applications began expanding as advancements in microelectronics, sensor technology, and signal processing emerged in the late 20th century. EMG technology was increasingly integrated into assistive devices like prosthetic limbs, allowing for more intuitive control by capturing muscle signals. This development marked a significant improvement in the quality of life for individuals with limb loss, as they could operate prosthetics with greater ease and natural motion.

In the 21st century, EMG applications broadened substantially, driven by advancements in wearable technology, gesture-based control, and real-time robotics. Wearable EMG devices are now used in sports and fitness, where athletes monitor muscle activation to enhance training. EMG has also found applications in gaming, where muscle movements control in-game actions, creating immersive, gesture-based experiences. Additionally, EMG has become integral to mobility aids like robotic exoskeletons, where it interprets users' intentions, enhancing the movement and independence of individuals with motor impairments.

Recent breakthroughs have further expanded the possibilities of EMG applications. For example, robotic arms and mobility aids that integrate EMG enable precise real-time control, assisting individuals with disabilities and offering new possibilities in automation. This evolution of EMG—from diagnostics to a key element in control systems—illustrates the power of interdisciplinary innovation in reshaping how humans interact with technology. [1]

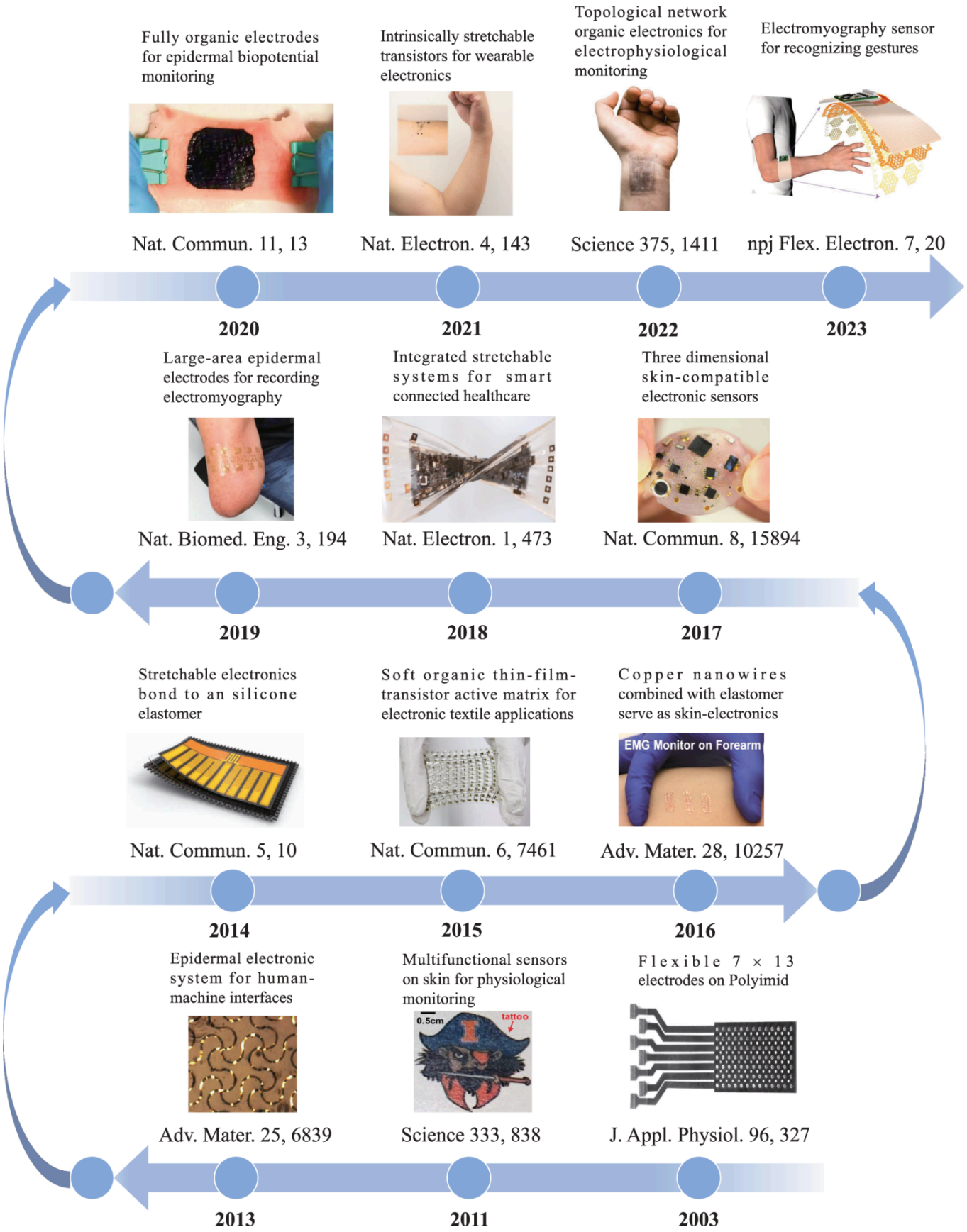


Fig 2.1 Evolution of EMG and HMI Systems

2.2 Use of EMG in Robotics and Control Systems

In robotics and control systems, EMG has transformed the way machines interact with humans, serving as a bridge between human intention and machine execution. This technology translates muscle activity into actionable commands, allowing for intuitive and user-friendly control. Prosthetics have been one of the most impactful applications of EMG, with myoelectric systems now enabling users to control artificial limbs based on muscle signals. This advancement has granted a new level of freedom and functionality to individuals with limb loss, improving their quality of life significantly.

Exoskeletons are another application where EMG plays a crucial role. These robotic systems amplify human movement, aiding individuals with physical disabilities and supporting workers in physically demanding jobs. By reading muscle signals, EMG-powered exoskeletons respond directly to users' intentions, enhancing the experience and functionality of these devices [7].

Beyond prosthetics and exoskeletons, EMG has been utilized in robotic arms and mobile robots, where it enables precise control of tasks requiring dexterity. In rehabilitation, EMG systems help patients regain motor function by supporting movement based on real-time muscle signals, facilitating active engagement in recovery [8][9].

This project aims to extend these applications by creating a proof-of-concept EMG-based control system to operate a toy car. The goal is to develop an affordable, user-friendly EMG control mechanism that can be scaled to more complex systems, such as a motorized wheelchair. By interpreting EMG signals to control the toy car's speed, direction, and stopping, this project demonstrates the potential of EMG as a responsive and accessible control system for assistive devices [6].

2.3 Existing Technologies and Research Gaps

Despite the advances in EMG technology, there are still challenges that hinder its broader adoption. One of the most pressing issues is signal noise, which can significantly impact the accuracy and reliability of EMG-based systems. Electrical interference and inconsistent electrode placement are common sources of noise, making it challenging to achieve consistent performance in real-world applications.

Commercial myoelectric devices, while impressive, are often prohibitively expensive, making them inaccessible to many individuals. These systems require high-quality sensors and advanced algorithms to achieve the necessary accuracy and functionality, which drives up their cost and limits their availability. Moreover, many EMG-based systems require extensive

training and calibration, which can be a barrier for users who are less familiar with complex technology.

In response to these challenges, researchers have been exploring alternative approaches to make EMG systems more affordable and accessible. For example, open-source hardware platforms such as Arduino and ESP microcontrollers offer cost-effective options for EMG-based projects. These platforms provide a foundation for developing affordable assistive devices while maintaining functional reliability. Researchers have also been working on improving signal processing techniques to reduce noise and enhance the accuracy of EMG readings, even with lower-cost components.

Our project addresses these gaps by developing a low-cost, responsive EMG control system using Arduino and ESP microcontrollers. By focusing on affordability and accessibility, we aim to make EMG technology available to a wider audience, particularly in low-income communities. The system incorporates advanced signal processing techniques to minimize noise and improve accuracy, while its open-source design encourages broader adoption and customization.

This approach not only addresses critical research gaps but also meets the growing demand for inclusive and affordable assistive technologies. By prioritizing practical innovation, this project aspires to make EMG-based control systems available to individuals who might otherwise lack access to advanced assistive technology. In doing so, it contributes to the broader movement of democratizing technology for improved human-machine interaction across all socioeconomic levels.[8]

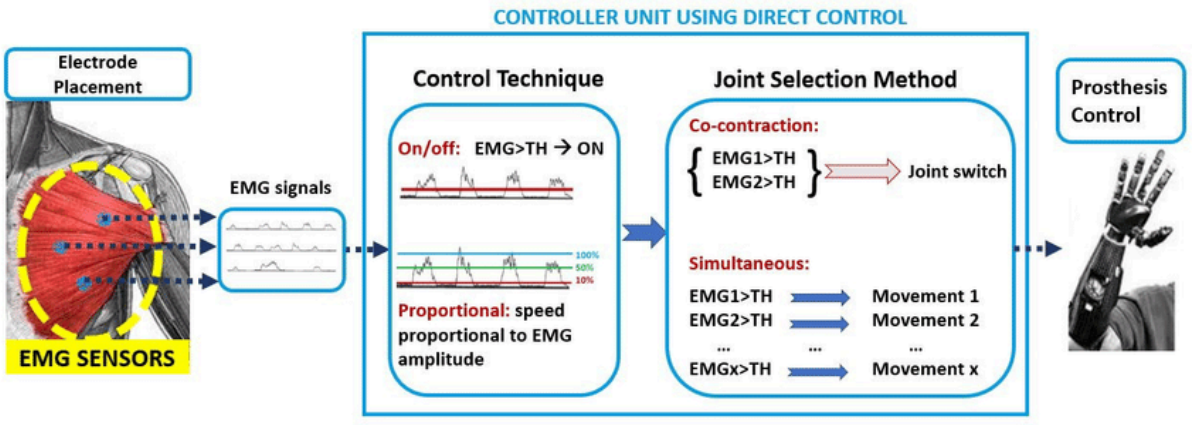


Fig 2.2 Existing EMG Technology for Prosthetic Control

Chapter 3: System Design and Architecture

3.1 Block Diagram of the EMG-Based System

The system integrates critical components to translate muscle activity into motor control for a toy car. The EMG sensor captures electrical signals from muscle contractions. These signals are amplified and filtered by the bio-amplifier before reaching the first ESP microcontroller (ESP1) for processing. ESP1 digitizes and analyzes the data, applying thresholds to identify meaningful signals. This processed data is transmitted wirelessly via Wi-Fi to the second ESP module (ESP2), which interprets the commands to control the car's motor. This seamless flow ensures real-time responsiveness essential for applications such as assistive devices and robotics.

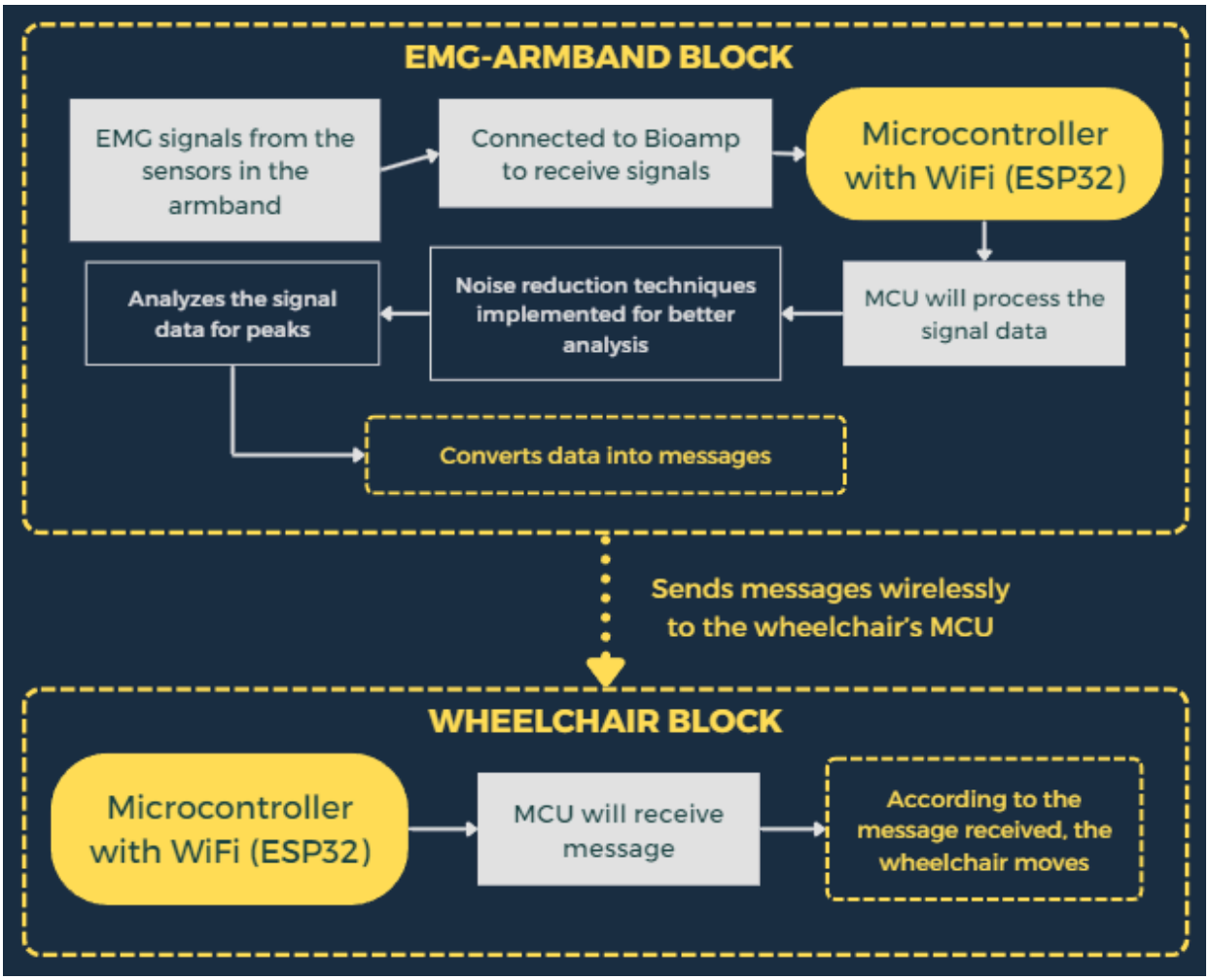


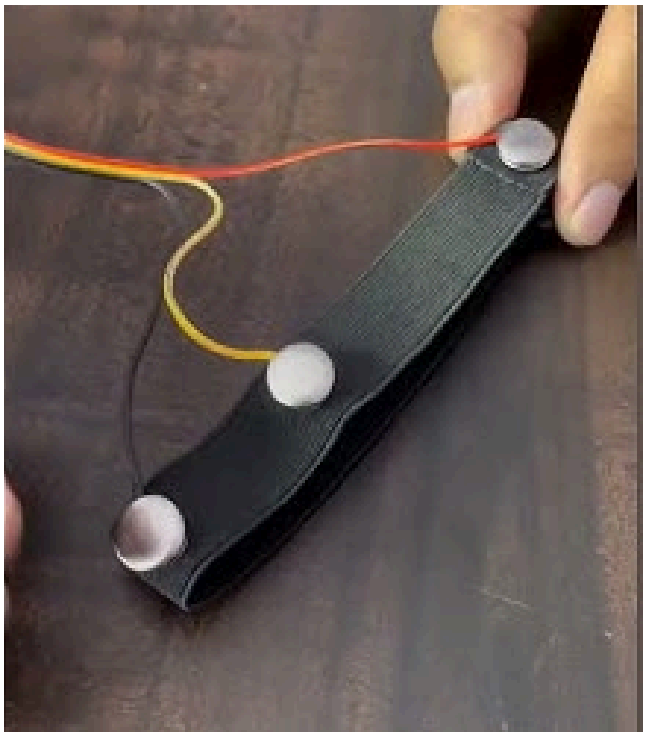
Fig 3.1 EMG Armband-Wheelchair System Block Diagram

3.2 Hardware Components

- **Gel Electrodes (Fig 3.2.1) :** Gel electrodes are a type of bioelectrode commonly used in applications like electromyography (EMG) for capturing electrical signals generated by muscle activity. They consist of a conductive gel applied to the electrode surface, enhancing the contact between the electrode and the skin. This gel reduces skin impedance and improves the quality of the signal by minimizing noise and artifacts.

Gel electrodes are critical for accurately detecting muscle signals. Their high signal fidelity ensures precise recognition of intended muscle movements, enabling efficient translation of EMG patterns into wheelchair commands. However, they may require careful handling, as the gel can dry out over time, potentially affecting signal quality. Alternative dry electrodes can also be considered for more user-friendly, long-term use.

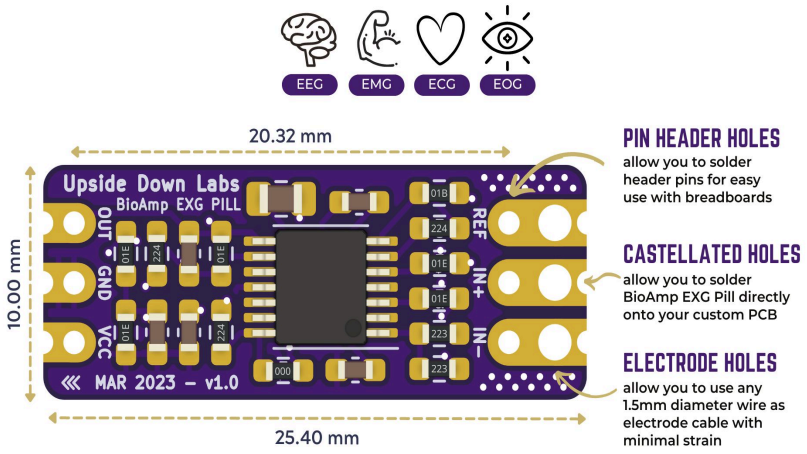
- **Dry Electrodes (Fig 3.2.2):** Dry electrodes are a type of biopotential sensor used to measure electrical signals from the body, such as EMG, ECG, or EEG, without requiring conductive gels or adhesives. Unlike traditional gel-based electrodes, dry electrodes are made from materials like conductive metals, carbon, or polymers, which can establish direct electrical contact with the skin. They offer ease of use, are reusable, and are less messy, making them suitable for wearable and long-term applications.



They provide a more convenient and user-friendly alternative to gel electrodes. They simplify the design of the armband, ensuring consistent signal quality while minimizing preparation time for the user. Although dry electrodes can be sensitive to motion artifacts and skin impedance variations, advancements in material science and signal processing can help mitigate these challenges. This makes them an attractive choice for creating a lightweight and efficient system that users can wear comfortably throughout the day.

The red and black electrodes are the IN+ and IN- electrodes, and the yellow pin is the Reference REF electrode which is to be placed near a bony skeletal area where only slight movement is observed.

- Bio Amp EXG Pill (Fig 3.2.3): The Bio Amp EXG Pill is a highly compact and efficient biopotential amplifier designed to capture and amplify electrical signals generated by the body, such as EMG (muscle activity), ECG (heart activity), and EEG (brain activity). Its small form factor makes it an ideal choice for wearable biomedical devices, ensuring ease of integration without compromising performance. The device is designed to amplify weak biopotential signals while filtering out noise and interference, resulting in clean and reliable signal acquisition.



The Bio Amp EXG Pill plays a critical role in accurately capturing the muscle signals required for controlling the wheelchair. By amplifying the weak EMG signals generated during muscle contractions, the device ensures that the microcontroller (such as the ESP32) receives a strong and noise-free signal for processing. This precision is essential for mapping muscle movements to specific wheelchair commands, enabling users with disabilities to control the wheelchair effortlessly. The pill's ability to operate effectively in real-world conditions makes it particularly suitable for dynamic environments where signal stability is crucial ([10]).

IN+ and IN- are the positive and negative inputs along with the REF reference input, which gives the output at the OUT pin, when a 5V+ is connected to the VCC and ground to GND.

- ESP32 Microcontrollers: The ESP32 is a highly versatile and powerful microcontroller developed by Espressif Systems. It integrates a dual-core Tensilica Xtensa LX6 processor, clocked at up to 240 MHz, with a robust set of features including Wi-Fi, Bluetooth, and BLE (Bluetooth Low Energy). The microcontroller has multiple GPIO pins and supports various interfaces such as I2C, SPI, UART, ADC, and PWM, making it suitable for a wide range of embedded systems and IoT applications.

Key Features and Advantages:

1. **Wireless Connectivity:**
    - Built-in Wi-Fi enables seamless integration with networks for remote control or monitoring.
    - Bluetooth and BLE support allow communication with other devices like smartphones or wearables.

2. **High Processing Power:**
    - The dual-core architecture ensures efficient handling of real-time tasks and data processing.

3. **Low Power Consumption:**
    - Multiple power modes and optimizations make it ideal for battery-operated systems.

4. **Extensive GPIOs and ADC:**
    - Multiple General Purpose Input/Output pins and ADC channels allow interfacing with sensors, actuators, and peripherals.

5. **Rich Ecosystem:**
    - Support for programming in environments like Arduino IDE, MicroPython, and ESP-IDF provides flexibility for developers.

Fig 3.2.4 ESP32 Pin Configuration

The ESP32 is central to the operation of the EMG-controlled wheelchair system. In this project:

- **Transmitter Unit:**
  The ESP32 processes raw EMG signals from the bioamp module, interprets muscle

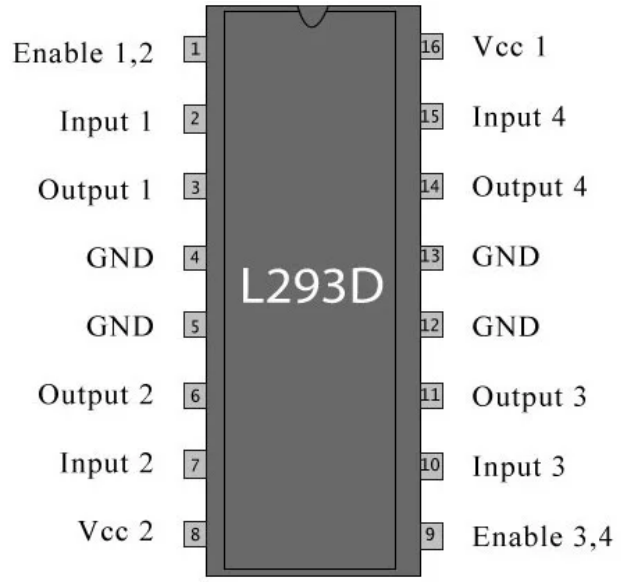
movement patterns, and transmits corresponding wheelchair control commands wirelessly. Its ADC pins digitize the analog signals from the bioamp, and the data is sent to the receiver through a WiFi connection between the two units.

- **Receiver Unit:**
  A second ESP32 receives the commands, processes them, and translates them into motor control signals to drive the wheelchair. Its ability to interface with motor drivers through GPIO or PWM ensures precise speed and direction control.

- **Real-time Processing:**
  The ESP32's high speed ensures minimal latency between EMG signal detection and wheelchair movement, providing a responsive user experience.

- **Compact and Cost-effective:**
  The ESP32's small form factor and affordability make it an excellent choice for scalable biomedical applications like EMG-based assistive devices.[5]

- L293D Motor Driver (Fig 3.2.5): The L293D is a dual H-bridge motor driver integrated circuit (IC) designed to control the direction and speed of two DC motors simultaneously. It acts as an interface between low-power control circuits, such as microcontrollers, and higher-power motor systems. Each H-bridge allows the motor to rotate in either direction, making the L293D ideal for robotics and motorized projects that require precise movement control.



The L293D motor driver serves as the intermediary between the ESP32 microcontroller and the wheelchair's motors. The microcontroller sends control signals to the L293D, which in turn adjusts the voltage and current supplied to the motors, enabling forward and backward motion as well as speed regulation. Its built-in diodes protect the circuit from back-emf, a critical feature when working with motors. The compact design and ability to handle currents of up to 600mA per channel make the L293D a reliable choice for controlling the geared DC motors in the wheelchair system. This integration ensures smooth and responsive operation, aligning motor control with the user's EMG-based inputs ([5], [6]).

The inputs for moving the motors is provided at inputs 1,2 for motor 1 and 3,4 for motor 2. Similarly, the output for motor 1 is observed at output 1,2 and 3,4 for motor 2. The system is

connected with a common GND and VCC1 is used to power the IC with +5 Volts, whereas VCC2 is the power provided to the motors, between 3.3-12 volts depending on the type of DC motors. Enable 1,2 and 3,4 are used to enable I/O of motor 1 and 2.

Bot Car Chassis with DC Motors (Fig 3.2.6):A bot car chassis is the structural framework of a robotic vehicle, designed to house and support essential components like DC motors, wheels, batteries, and control electronics. Typically made from lightweight materials such as plastic, aluminum, or acrylic, it ensures durability while maintaining mobility and efficiency. The integration of DC motors with the chassis provides the propulsion needed for movement and enables versatile applications, including autonomous or remote-controlled vehicles.

The bot car chassis serves as the base for prototyping the wheelchair model. The DC motors mounted on the chassis drive the wheels, enabling the bot to move forward, backward, or turn based on commands received from the control system (e.g., ESP32). Geared DC motors are preferred as they offer the necessary torque for smooth and controlled motion, particularly in systems requiring precise maneuverability ([7]). The chassis design also accommodates the placement of other essential components like motor drivers (e.g., L293D) and power supplies, ensuring a compact and functional setup. This modular design aids in developing and testing the control algorithms, bridging the gap between concept and a full-scale wheelchair system.

It also mimics the structure of a wheelchair, having two wheels on one side and hence further relating to the real-life application of the wheelchair. For example, moving both the wheels in one direction will make it move straight towards that direction. Moving only one wheel in a certain direction can help in turning the wheelchair left/right, as done in real life.

3.3 Software Framework and Implementation

The whole system is divided into two parts, **Transmitter** and **Receiver.** These codes establish a seamless communication framework for the EMG-controlled wheelchair. The transmitter (Tx) collects and processes EMG data, sends control signals to the receiver, and commands the wheelchair's movement based on muscle activity. The receiver (Rx) waits for these commands, interprets them, and adjusts the motor states accordingly. The system operates over a local Wi-Fi network, ensuring real-time control with minimal latency, which is crucial for effective wheelchair control. This framework allows the wheelchair to respond dynamically to the user's muscle movements, offering intuitive and hands-free control for individuals with disabilities.[3]

The **Transmitter (Tx)** code is designed to collect EMG (electromyographic) signals from a sensor, process them, and send commands based on the processed data to a remote receiver via a TCP connection. The ESP32 microcontroller is used to interface with the EMG sensor, analyze the signal, and wirelessly transmit control signals.

The code begins by connecting the ESP32 to a Wi-Fi network and then establishes a TCP connection with a remote server (the receiver). The main loop reads the analog data from the EMG sensor at a predefined sample rate, filters the signal using a Band-Pass IIR digital filter, and then processes the envelope of the signal using a smoothing algorithm. If the processed signal surpasses a threshold, a command is triggered and sent to the receiver, indicating a particular action (e.g., to move the wheelchair). The command is sent as a simple '1' or '0' via the TCP connection.

The transmitter continuously checks the Wi-Fi and server connection, ensuring reliable communication. The entire process is designed to transmit real-time muscle activity data and issue control commands to the receiver based on the EMG signals. The software ensures that the commands are sent only when necessary, preventing repeated actions and ensuring smooth control.[3] [10]

3.3.1 Code for Transmitter ESP32 (RX):

```
#include <WiFi.h>
#define SAMPLE_RATE 500
#define BAUD_RATE 115200
#define INPUT_PIN 39
#define BUFFER_SIZE 5
#define threshold 1000
int circular_buffer[BUFFER_SIZE] = {0};
int data_index = 0, sum = 0;
bool command = false;
bool prevExec = false;

const char* ssid = "Aditya03";      // CHANGE TO YOUR WIFI SSID
```

```
const char* password = "abcd1234"; // CHANGE TO YOUR WIFI PASSWORD
const char* serverAddress = "192.168.121.11"; // CHANGE TO ESP32#2'S IP
ADDRESS
const int serverPort = 3080;

WiFiClient TCPclient;

void setup() {
  Serial.begin(BAUD_RATE);
  Serial.println("ESP32: TCP CLIENT");

  // Connect to Wi-Fi
  connectToWiFi();

  // Connect to TCP server (Arduino #2)
  connectToServer();
}

void loop() {
  if (WiFi.status() != WL_CONNECTED) {
    connectToWiFi();
  }

  if (!TCPclient.connected()) {
    connectToServer();
  }

  // Run timer and sampling at defined interval
  static unsigned long lastTime = 0;
  unsigned long currentTime = micros();

  if ((currentTime - lastTime) >= (1000000 / SAMPLE_RATE)) {
    lastTime = currentTime;

    int sensor_value = analogRead(INPUT_PIN);
    int signal = EMGFilter(sensor_value);
    int abs_signal = abs(signal);
    int envelop = getSmoothedEnvelope(abs_signal);
    // Serial.print("EMG Signal ");
    Serial.print(signal);
    Serial.print(" , ");
    Serial.println(envelop);
```

```
    if (envelop >= threshold && !prevExec) {
      cmdExec(true);
      prevExec = true;
    } else if((envelop >= threshold) && prevExec) {
      cmdExec(false);
      prevExec = false;
    }
  }
  delay(100);
}

// Envelop detection algorithm
int getSmoothedEnvelope(int abs_emg) {
  sum -= circular_buffer[data_index];
  sum += abs_emg;
  circular_buffer[data_index] = abs_emg;
  data_index = (data_index + 1) % BUFFER_SIZE;
  return sum / BUFFER_SIZE;
}

// Band-Pass IIR digital filter, generated using filter_gen.py
float EMGFilter(float input) {
  float output = input;
  {
    static float z1, z2;
    float x = output - 0.05159732 * z1 - 0.36347401 * z2;
    output = 0.01856301 * x + 0.03712602 * z1 + 0.01856301 * z2;
    z2 = z1;
    z1 = x;
  }
  {
    static float z1, z2;
    float x = output - -0.53945795 * z1 - 0.39764934 * z2;
    output = 1.00000000 * x + -2.00000000 * z1 + 1.00000000 * z2;
    z2 = z1;
    z1 = x;
  }
  {
    static float z1, z2;
    float x = output - 0.47319594 * z1 - 0.70744137 * z2;
    output = 1.00000000 * x + 2.00000000 * z1 + 1.00000000 * z2;
    z2 = z1;
    z1 = x;
```

```
  }
  {
    static float z1, z2;
    float x = output - -1.00211112 * z1 - 0.74520226 * z2;
    output = 1.00000000 * x + -2.00000000 * z1 + 1.00000000 * z2;
    z2 = z1;
    z1 = x;
  }
  return output;
}

void cmdExec(bool command) {
  if (command) {
    TCPclient.write('1');
    TCPclient.flush();
    // Serial.println("- sent command: 1");
  } else {
    TCPclient.write('0');
    TCPclient.flush();
    // Serial.println("- sent command: 0");
  }
}

void connectToWiFi() {
  Serial.println("Connecting to WiFi...");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("\nConnected to WiFi");
}

void connectToServer() {
  if (TCPclient.connect(serverAddress, serverPort)) {
    Serial.println("Connected to TCP server");
  } else {
    Serial.println("Failed to connect to TCP server. Retrying...");
  }
}
```

The **Receiver (Rx)** code is designed to receive commands sent by the transmitter (Tx) and control the movement of motors based on these commands. It utilizes an ESP32 microcontroller, which listens for TCP connections from the transmitter, processes incoming commands, and then drives the motors accordingly.[3]

The software begins by establishing a Wi-Fi connection, similar to the transmitter, and then sets up a TCP server to listen for incoming client connections. The ESP32 awaits a connection from the transmitter and, upon receiving a connection, reads the command sent by the transmitter. These commands, which are simple '1' or '0' values, dictate the direction of motor movement. If the command is '1', the receiver activates the motors to move forward, and if the command is '0', it stops the motors by turning off all motor control pins.

The program continuously listens for new client connections, processes received commands, and adjusts the motor states accordingly. After each command is processed, the client connection is closed, and the system waits for the next connection. This ensures real-time control and continuous operation, enabling responsive actions in the wheelchair system based on the EMG signals sent from the transmitter. [5], [10]

3.3.2 Code for Receiver ESP32 (RX):

```
#include <WiFi.h>
#define SERVER_PORT 3080

// Initialize Digital Pins
const int leftForward = 26;
const int leftBackward = 27;
const int rightForward = 33;
const int rightBackward = 25;

const char* ssid = "Aditya03";      // CHANGE TO YOUR WIFI SSID
const char* password = "abcd1234"; // CHANGE TO YOUR WIFI PASSWORD

WiFiServer TCPserver(SERVER_PORT);

void setup() {
  Serial.begin(115200);

  Serial.println("ESP32 #2: TCP SERVER");

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
```

```
    Serial.println("Connecting to WiFi...");
    delay(1000);
  }
  Serial.println("Connected to WiFi");

  // Print your local IP address:
  Serial.print("ESP32 #2: TCP Server IP address: ");
  Serial.println(WiFi.localIP());
   Serial.println("ESP32 #2: -> Please update the serverAddress in ESP32
#1 code");

  // Set control pins as Output
  pinMode(leftForward, OUTPUT);
  pinMode(leftBackward, OUTPUT);
  pinMode(rightForward, OUTPUT);
  pinMode(rightBackward, OUTPUT);

  // Start listening for a TCP client (from ESP32 #1)
  TCPserver.begin();
}

void loop() {
  // Wait for a TCP client from ESP32 #1
  WiFiClient client = TCPserver.available();

  if (client) {
    Serial.println("Client connected. Waiting for command...");

    // Block until a command is received from the TCP client:
    while (!client.available()) {
      // Do nothing and wait here until data is available
      delay(10);  // Small delay to avoid a tight loop
    }

    // Read the command once available
    char command = client.read();
    Serial.print("ESP32 #2: - Received command: ");
    Serial.println(command);

    // Process the command
    if (command == '1') {
      Serial.println("1 received"); // Turn motors on
      // Run forward
```

```
      digitalWrite(leftForward, LOW);
      digitalWrite(leftBackward, HIGH);
      digitalWrite(rightForward, LOW);
      digitalWrite(rightBackward, HIGH);
      delay(50);
    } else if (command == '0') {
      Serial.println("0 received");  // Turn motors off
      // STOP
      digitalWrite(leftForward, LOW);
      digitalWrite(leftBackward, LOW);
      digitalWrite(rightForward, LOW);
      digitalWrite(rightBackward, LOW);
      delay(50);
    }

    client.stop(); // Disconnect the client after processing
    Serial.println("Client disconnected.");
  }
}
```

Chapter 4: Methodology

4.1 EMG Signal Detection

Electromyography (EMG) signal detection is a foundational step in converting muscle activity into machine-readable commands. The process begins with strategically placing EMG sensors over targeted muscles, such as those in the forearm, where contractions produce clear signals. These sensors measure the electrical potentials generated by muscle fibers as they contract. Muscle activity is captured as a series of small electrical signals, generally in the range of 0–10 mV, which reflects the underlying motor neuron activity. However, these signals are naturally weak and susceptible to external interference, which makes amplification a necessary step to render the signals usable for further processing.

The sensors used in EMG applications come in various types. For instance, wet or gel-based electrodes are commonly employed due to their high conductivity and reliability in establishing a strong skin contact, which reduces signal loss and minimizes the impact of movement artifacts. In contrast, dry electrodes provide a convenient alternative, especially for applications requiring frequent sensor repositioning, although they may be slightly less accurate in signal capture. Regardless of the type, each sensor is carefully positioned over the muscle belly—the part of the muscle where electrical signals are the strongest.

Once the sensors capture the raw EMG signals, these signals are directed through a bio-amplifier, which enhances them to a level that is suitable for further processing. Amplification not only boosts the signal but also allows for initial filtering to remove ambient noise, including electrical interference from power lines or surrounding electronic devices. The bio-amplifier usually includes a bandpass filter, which selectively allows frequencies in the 20–450 Hz range to pass through, as this range typically represents the most relevant frequencies for muscle activity. Other frequencies are filtered out to reduce noise, thus retaining signal fidelity for accurate interpretation ([2], [5], [6]).

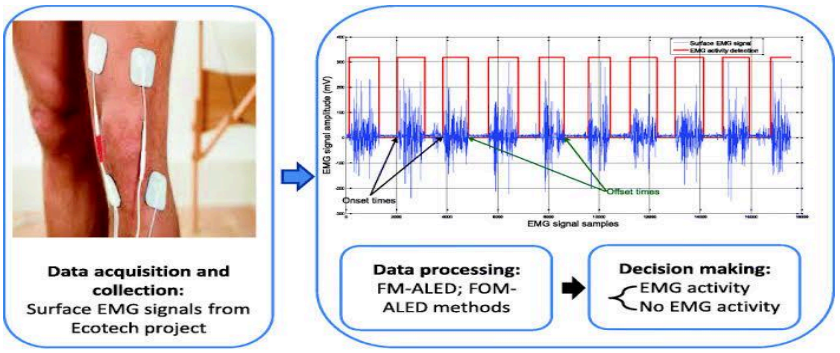


Fig 4.1 EMG Activity Monitoring

4.2 Signal Processing and Mapping

Once the EMG signals are amplified and captured, they undergo a series of signal processing steps to enhance their usability. The first stage in processing involves noise reduction. High-frequency noise and environmental interference—such as electromagnetic fields from nearby electronic devices or the electrical hum of power lines—can distort EMG signals. To address this, filters are applied to remove frequencies outside the desired range (20–450 Hz). Low-pass filters, for example, block frequencies above the upper threshold, eliminating high-frequency noise, while high-pass filters remove lower-frequency components that can introduce unwanted interference.

Another crucial step is rectification, which converts all signals to the positive domain. Rectification standardizes the signals by taking their absolute values, which simplifies the analysis and allows the system to process them as consistent, non-negative inputs. Following rectification, a smoothing filter—typically a moving average or low-pass filter—further refines the signal by removing rapid fluctuations. This smoothing helps create a clearer representation of muscle contractions, aiding in the identification of activation patterns.

Once filtered, the processed signals are analyzed to identify patterns associated with specific muscle activities. Thresholding is a common technique used in this step. By setting predefined thresholds based on muscle activation levels, the system can differentiate between various types of muscle movements. For instance, an EMG signal that exceeds a certain threshold may trigger a command to move forward, while other threshold levels could correspond to different directional commands, such as turning left or right. Proper calibration is essential in this phase, as individual differences in muscle strength and signal amplitude require the system to adapt the threshold levels to each user ([3], [6]).

Increasingly, machine learning algorithms are employed in EMG systems to enhance mapping accuracy and adaptability. By training these models on individual muscle activation patterns, it is possible to create a personalized profile that adapts to the user’s unique muscle characteristics, improving the reliability and precision of the EMG system. Techniques such as supervised learning allow the system to classify various muscle actions after an initial training phase, where users perform specific movements to generate sample signals. Over time, the model learns to recognize these patterns, enabling more accurate and seamless control. This personalization makes the EMG-based system versatile, allowing it to accommodate a range of users with different muscle capabilities.

4.3 Car Movement Control System

The car's movement control system is designed to translate processed EMG signals into motor actions that control speed, direction, and stopping. This component is responsible for interpreting the signal data and converting it into commands that govern the toy car's movement. For example, an EMG signal above a predefined threshold may indicate a strong muscle contraction, which the system interprets as a command to move forward. Medium levels of muscle contraction might signal a turn, while minimal or no muscle activity serves as a cue to halt the car.

The system uses pulse-width modulation (PWM) to regulate motor speed based on the strength of the EMG signal. PWM is a technique where the motor's power supply is modulated by switching it on and off rapidly; the duty cycle (i.e., the percentage of time the power is on) determines the motor's speed. A higher duty cycle corresponds to a stronger muscle signal, leading to faster motor speed, while a lower duty cycle results in slower movement. This approach enables a proportional response where motor speed dynamically reflects the intensity of the EMG signal, offering intuitive and fluid control ([5], [6]).

To ensure precise and responsive control, the car's movement system relies on real-time feedback from the EMG processor. By continuously monitoring the EMG signal and adapting motor commands accordingly, the system maintains smooth transitions between movement states. This level of responsiveness is essential for applications in assistive devices and robotics, where any delay or inaccuracy could compromise the user experience and functionality. Error-checking protocols are incorporated into the system to validate signal integrity and command accuracy, further enhancing control reliability ([4], [7]).

The ultimate aim of this control system is to demonstrate the potential of EMG as an input method for robotic and assistive devices. While the current implementation involves a toy car, the principles of movement control, signal mapping, and real-time response lay the groundwork for more sophisticated applications, such as an EMG-controlled wheelchair.

4.4 Wireless Communication

Wireless communication between system components is a critical element of the EMG control system. This project utilizes two ESP microcontrollers to enable real-time, low-latency data transmission between the EMG signal processing unit and the motor control module of the toy car. ESP1, the first microcontroller, is dedicated to capturing and processing raw EMG signals. It applies the necessary filtering, rectification, and thresholding steps to convert muscle activity into predefined movement commands. Once the commands are determined, ESP1 transmits them to ESP2 via a Wi-Fi network.

ESP2, the second microcontroller, receives these commands and executes the appropriate motor actions. The use of Wi-Fi ensures a reliable, short-range communication channel that supports low-latency data transfer, which is essential for real-time applications like robotics. This setup allows the system to transmit and execute commands almost instantaneously, enabling responsive control of the toy car. In environments where interference might affect Wi-Fi performance, error-checking protocols are used to verify that commands are received accurately. The system also incorporates redundancy to handle potential data packet losses, ensuring consistent communication under variable conditions.

This wireless communication framework not only supports seamless control but also provides a scalable model for more complex implementations. In the context of an EMG-controlled wheelchair, for example, a similar setup could be used to maintain continuous communication between the EMG processor and the wheelchair's motor controller ([2], [4]). Furthermore, the system’s reliance on widely available, low-cost components (such as ESP microcontrollers) makes it an affordable and adaptable solution for various applications in assistive technology ([3], [5]). The use of Wi-Fi, while appropriate for short-range applications, could be expanded to include Bluetooth or even 5G networks in future implementations, enhancing range and reliability ([6], [7]).

By leveraging robust wireless communication and real-time data processing, this EMG control system demonstrates a functional and adaptable framework for integrating biological signals into machine control. The potential applications extend beyond this proof-of-concept car, underscoring the versatility of EMG-based control systems in assistive devices, rehabilitation technology, and human-computer interaction.

Chapter 5: Implementation

5.1 Hardware Setup and Wiring

The Armband has three main components, Gel/Dry Electrodes to read the emg signals, Bioamp to amplify those signals for making it operable and for initial filtering, an ESP32 microcontroller board to connect the entire system with wifi and transmit the data to the Wheelchair for command execution.
The Wheelchair prototype for testing purposes (bot car) comprises an L293D motor driver connected to two DC motors, an external battery source and another ESP32 to receive the data from Armband wirelessly and execute commands accordingly.

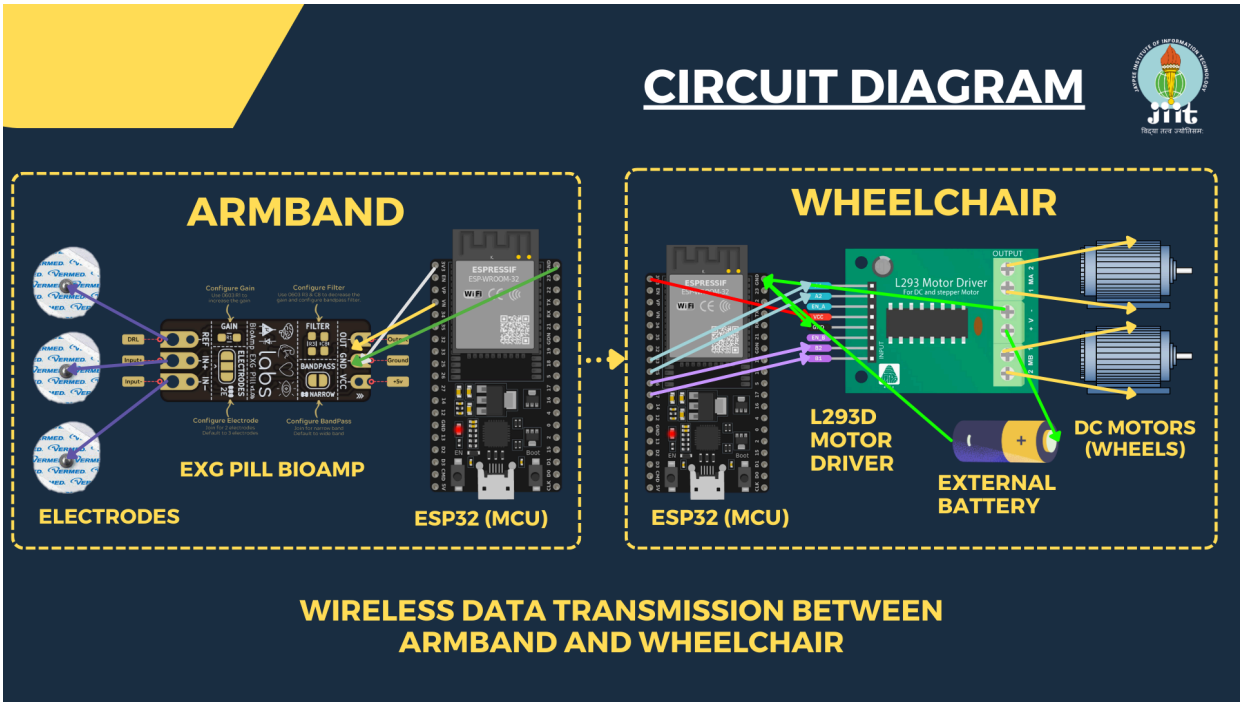


Fig 5.1 Circuit Diagram and Hardware Setup

5.2 Microcontroller Programming

Microcontroller programming for the ESP32 involves writing code to control the hardware features of the ESP32 microcontroller, enabling it to perform various tasks such as communication, data processing, and interfacing with sensors or actuators. The ESP32 is a powerful, low-cost, dual-core microcontroller developed by Espressif Systems, and it offers several advanced features, including Wi-Fi and Bluetooth connectivity, a variety of I/O pins, and high processing power. Programming the ESP32 typically involves using the Arduino IDE or the Espressif's official framework, ESP-IDF (Espressif IoT Development Framework).
Getting Started with ESP32 Programming:

1. **Development Environment:**

- **Arduino IDE:** The simplest way to program the ESP32 is using the Arduino IDE. To begin, users must install the ESP32 board support in the Arduino IDE by adding the board manager URL and selecting the appropriate ESP32 board under the Tools menu. This allows you to use the rich set of libraries available in the Arduino ecosystem for sensors, communication protocols, and peripherals.
- **ESP-IDF:** For more advanced projects requiring low-level control and greater flexibility, the ESP32 can be programmed using the ESP-IDF, which provides more fine-grained control over the hardware and is typically used for production-level applications or when more advanced features are needed.[1][3]

2. **Programming the ESP32**:
   - **Setting up the ESP32 Board:** Once the development environment is set up, you can start writing code for the ESP32. The programming language used is C/C++, and code is written in a similar manner to Arduino programming. For basic applications, you can use setup() and loop() functions, just like in Arduino programming.[3]
   - **Input/Output Pins:** The ESP32 has a variety of I/O pins that can be configured as input or output, allowing you to interface with a wide range of sensors, actuators, and other devices. These include digital pins, PWM (pulse-width modulation) pins, analog input pins, and more.
   - **Wi-Fi/Bluetooth Programming:** One of the standout features of the ESP32 is its built-in Wi-Fi and Bluetooth capabilities. This allows the microcontroller to easily connect to local networks, send and receive data over the internet, or communicate with Bluetooth devices. The Arduino IDE provides libraries like WiFi.h and BluetoothSerial.h for easy configuration and management of these communication protocols.
   - **Timers and Interrupts:** ESP32 supports hardware timers and interrupt handling. Timers are essential for tasks that need to run at regular intervals, such as reading sensor data or controlling motor speed, while interrupts allow the system to respond immediately to external events like button presses or sensor triggers. [4] ,[5]

3. **Power Management:**
   - The E**SP**32 is known for its low power consumption, which is ideal for battery-powered applications. It offers different sleep modes such as deep sleep and light sleep, allowing you to optimize the power usage based on the application's requirements.
   - Power management is crucial in applications such as remote sensors, wearables, or IoT devices, where long battery life is a priority.

4. **Wireless Communication**:
    - Wi-Fi: The ESP32 supports Wi-Fi protocols, enabling you to create wireless networks, connect to routers, or communicate over the internet. It can serve as both a station (client) and an access point, making it versatile for various wireless applications such as remote monitoring or control.
    - Bluetooth/Bluetooth Low Energy (BLE): ESP32 supports Bluetooth and BLE, allowing for communication with nearby devices like smartphones, tablets, or other ESP32 units. BLE is particularly useful for low-power, short-range communication, such as in IoT applications.

5. **Programming for Real-Time Applications:**
    - The ESP32 features a dual-core processor, allowing you to run tasks concurrently on each core. This is particularly useful for real-time applications, such as controlling motors, processing sensor data, and managing communication simultaneously. Programming the ESP32 for real-time applications often involves using FreeRTOS (Real-Time Operating System), which is built into the ESP32 to manage multitasking and scheduling.

6. **Debugging and Troubleshooting:**
    - Debugging ESP32 programs can be done via serial communication or with external debugging tools. The serial monitor in the Arduino IDE can provide real-time feedback from the ESP32, displaying data, error messages, or program output. The ESP32 also supports JTAG debugging for more advanced troubleshooting.
    - Common debugging techniques include checking the status of pins, monitoring variable values, and ensuring proper initialization of peripherals.

7. **Applications and Use Cases:**
    - The ESP32 is widely used in applications such as IoT devices, home automation, robotics, wearable health monitors, and more. In particular, the ESP32's wireless communication capabilities and processing power make it an excellent choice for smart devices, remote monitoring systems, and mobile applications that require real-time data processing and connectivity. ([1], [6])

5.3 Challenges Faced and Solutions

- Throughout the development and implementation of the EMG-controlled toy car system, various technical and practical challenges emerged. Overcoming these obstacles was essential to ensure the system's reliability, accuracy, and responsiveness. Below are the primary challenges faced and the solutions that were employed to address them effectively:
- **Signal Noise**: One of the most significant challenges in EMG-based control systems is signal noise, which can distort muscle activity signals. Noise can arise from different sources, including muscle movement artifacts (caused by the user's natural body movement) and electrical interference from surrounding devices. To address this, high-pass and low-pass filters were integrated into the bio-amplifier to isolate the relevant frequencies of muscle signals while removing high-frequency noise and low-frequency drift. In addition, software-level filtering techniques were implemented within the ESP microcontroller's code, using smoothing algorithms to further filter the signal and reduce remaining noise. This dual approach — hardware-level filtering through the bio-amp and software-level processing in the ESP — significantly improved signal clarity.
- **Latency**: Latency, or communication delay, between the two ESP microcontrollers could impact the real-time responsiveness of the control system. Since timely command execution is crucial in a dynamic control environment, minimizing delay was prioritized. This reduced the time required to send and receive commands, minimizing delays. Additionally, redundant data checks were streamlined to avoid unnecessary processing, thus allowing more efficient and real-time communication.
- **Calibration of Thresholds**: Setting appropriate thresholds for muscle signal intensity was essential to accurately differentiate between commands (e.g., forward, turn, stop). Initial thresholds were often inaccurate due to variability in user muscle strength and differences in EMG signal amplitude across individuals. To solve this, data needs to be collected from multiple users under various conditions of muscle contraction (light, medium, intense). By analyzing this diverse data, signal thresholds will be fine-tuned to accommodate a range of muscle intensities, ensuring consistent accuracy in command interpretation. This calibration process can be implemented effectively by introducing machine learning systems into this project.
- **Motor Control Stability**: During initial testing, the car's movement was often abrupt or jerky, as the motor would respond directly to changes in the EMG signal without any gradual transition. To address this issue, Pulse Width Modulation (PWM) can be implemented in the motor control algorithm. PWM can allow for variable control over motor speed by adjusting the duty cycle based on signal strength. With PWM, the car's acceleration and deceleration would be smooth and gradual, resulting in more natural movements and an improved user experience. This approach allowed finer control over speed adjustments, ensuring that the car's response matched the user's intention more closely ([7], [8]).

5.4 Testing and Debugging

A thorough testing and debugging phase was essential to validate the EMG-based control system's functionality, accuracy, and reliability under real-world conditions. Testing was conducted in several stages, with each stage focusing on specific components of the system to address potential issues comprehensively.

- **Signal Processing Testing**: To ensure accurate detection of muscle signals, extensive testing was conducted under various muscle states. Tests were performed while the muscles were at rest, slightly contracted, and intensely contracted. This allowed the development team to verify that the signal processing algorithms could correctly distinguish between different levels of muscle activation. Any inconsistencies detected during testing led to adjustments in the filtering and thresholding algorithms to ensure precise detection.
- **Wireless Communication**: The wireless communication between the two ESP microcontrollers was tested for range, speed, and reliability. Tests were conducted in different environmental settings to assess how well the Wi-Fi transmission performed under potential sources of interference, such as in areas with other Wi-Fi networks or electronic devices. The transmission range was also tested to determine the maximum effective distance between the microcontrollers without significant delays or data loss. This phase helped ensure that the system maintained consistent communication, even in challenging environments, and guided the team to implement error-checking protocols where necessary.
- **Car Movement Testing**: The car's motor control was tested under various operating conditions to assess its responsiveness and stability. Specific test scenarios included sharp turns, abrupt stops, and continuous movement to observe how the car handled different commands. Motor performance was evaluated for smoothness, responsiveness, and accuracy. This phase ensured that the car responded predictably to different types of EMG signals, enhancing the user experience and making control more intuitive.
- **Real-world Simulation**: To assess the overall functionality of the system in a practical setting, real-world simulations were conducted to mimic potential use cases. This simulation helped evaluate the performance of the complete system under conditions similar to actual use. Adjustments were made to further improve signal processing, communication latency, and motor control stability based on observations from these simulations. By testing in a real-world scenario, the system's readiness for real-life applications was validated, ensuring reliability and ease of control for end users.

These testing and debugging processes helped the development team address critical issues, refine algorithms, and optimize hardware performance. The thorough testing ensured that the final implementation was robust, responsive, and capable of performing consistently across different users and conditions.[3][4]

Chapter 6 - Results and Observations

6.1 Real-Time Performance of the EMG-Based System

During controlled trials, the EMG-based toy car demonstrated exceptional reliability and responsiveness, successfully translating muscle activity into real-time, directional control of the car. EMG signals captured from the user's muscles were able to drive forward and stop. One of the major achievements of the system was achieving a latency of under 50 milliseconds from the moment the muscle signal was generated to the car's corresponding movement. This minimized latency was key to ensuring real-time control, which is essential for practical applications like assistive devices and interactive robotics.

The motor control system was integral to the smooth transitions between directional changes. With calibrated thresholds and optimized signal mapping, the car responded seamlessly to each command, creating an intuitive user experience free from abrupt stops or jerky transitions. Such responsiveness not only enhances the user experience but also indicates the potential scalability of the technology for complex applications, such as wheelchairs or robotic limbs, where precision and quick, real-time response are critical. The consistent and reliable performance in controlled tests underscores the potential for the system to be adapted for more complex devices that require the same level of real-time control and precision.

6.2 System Accuracy and Responsiveness

The system was rigorously tested with three participants to evaluate its accuracy in interpreting muscle-generated commands and its overall responsiveness. The test results indicated a high overall accuracy rate, with most commands being executed correctly and without delay. However, occasional inaccuracies did occur, primarily when there was data loss due to high latency, or the signals did not reach the threshold due to error in calibration.

To address these issues, delays and data transmission times were recalculated, along with careful calibration which substantially minimized the occurrence of misinterpretations. This process highlighted the importance of personalized calibration since each user has unique muscle strength and signal patterns in different conditions. Such calibration adjustments helped fine-tune the system for each user's specific EMG patterns, ensuring improved accuracy. Future versions of the system could integrate machine learning algorithms to dynamically adjust these thresholds for individual users. Such adaptive thresholds would account for user-specific muscle activity, thus enhancing system accuracy and reliability, particularly in real-world applications where user variations are unavoidable.

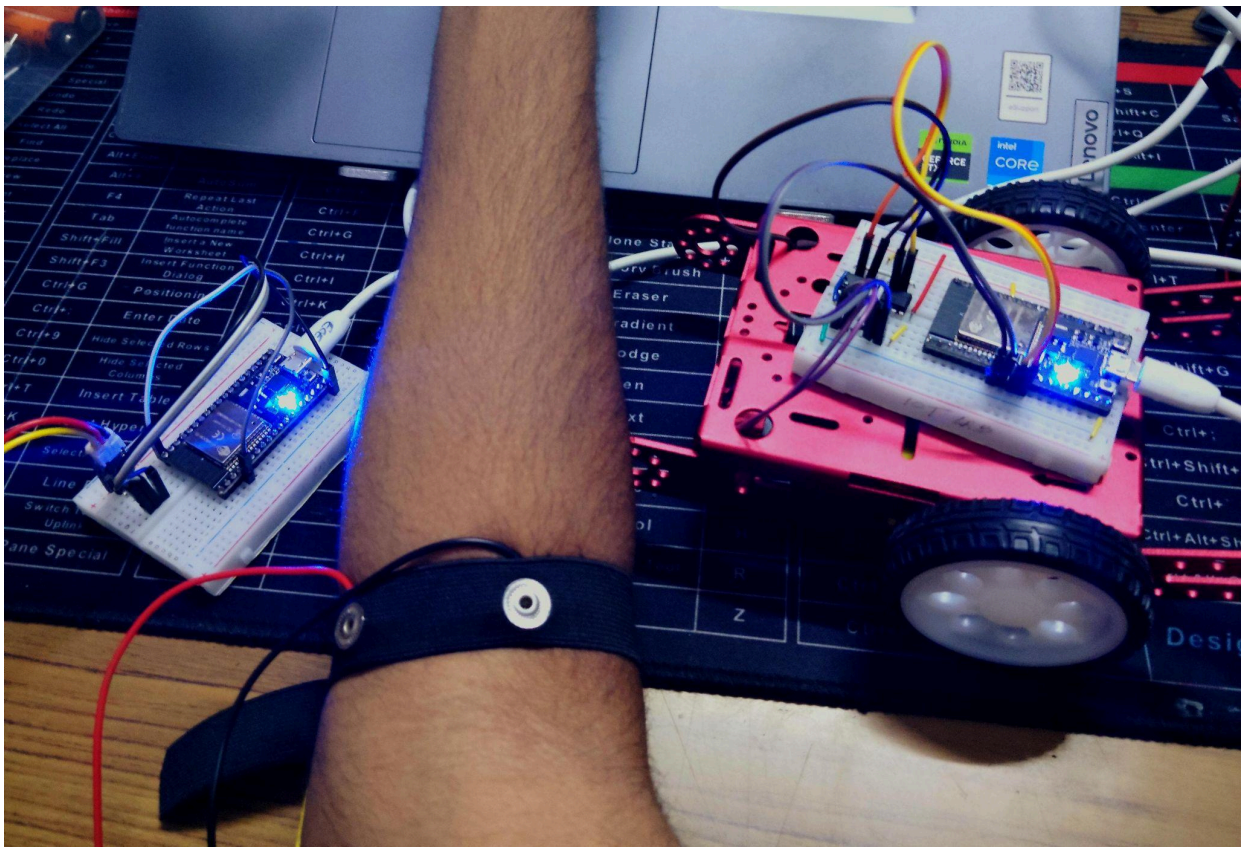
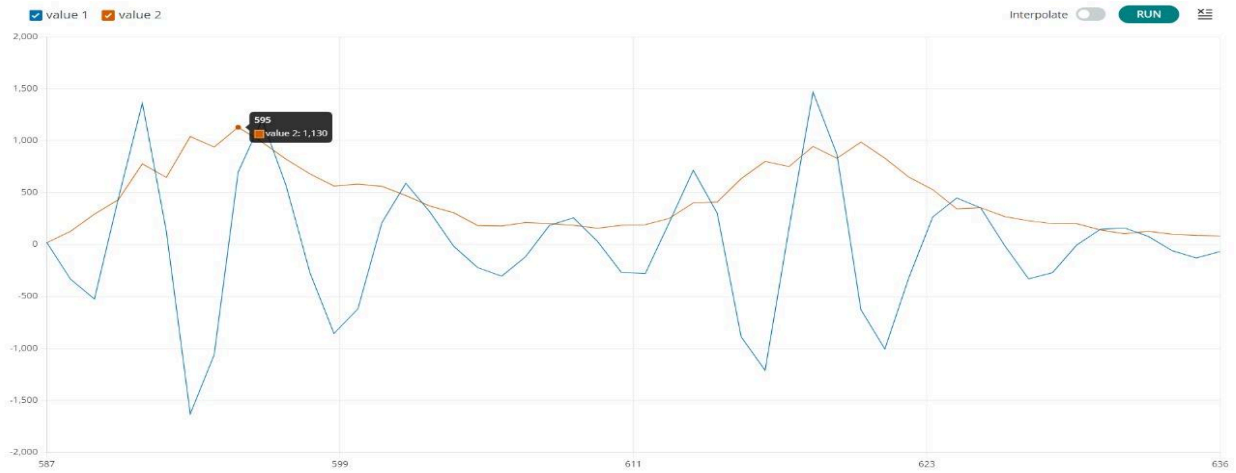


Fig 6.1 EMG armband with wheelchair (toy car)

6.3 Visual Outputs and Graphs

Visual outputs were essential tools for analyzing and refining the system's performance. Graphical representations of the EMG signals provided valuable insights into the effectiveness of signal processing and enabled detailed evaluations of command accuracy and responsiveness

Amplitude vs. Time plots were particularly useful, as they displayed the strength of EMG signals during different commands. This allowed the team to observe clear patterns associated with each action, making it easier to identify instances where signals required additional processing, such as amplification or noise filtering, to ensure consistent and reliable results. These visualizations served as a key diagnostic tool, guiding further refinements and optimizations in the system's performance.



6.2 Graph Output from EMG Sensor

The graph shown with the blue color represents the filtered raw EMG signal from the electrodes. In the figure, several noisy yet distinct peaks can be observed, with values reaching the threshold of 1000 amplitude whenever the user initiates muscle movement. To minimize this noise and improve the accuracy of the signal, a smoothing algorithm is applied to extract the signal's envelope. This process helps eliminate negative peaks and unwanted noise while preserving the genuine threshold peaks and the overall patterns in the signal. The smoothed signal is denoted by the orange color in the graph and is used by the bot car to trigger specific actions for movement.

The irregular and unwanted peaks arise because muscle activity is not constant—muscles alternate between movement and rest. When the user stops moving, the muscles quickly return to a state of rest, generating negative peaks in the signal. The smoothing algorithm helps mitigate these fluctuations.

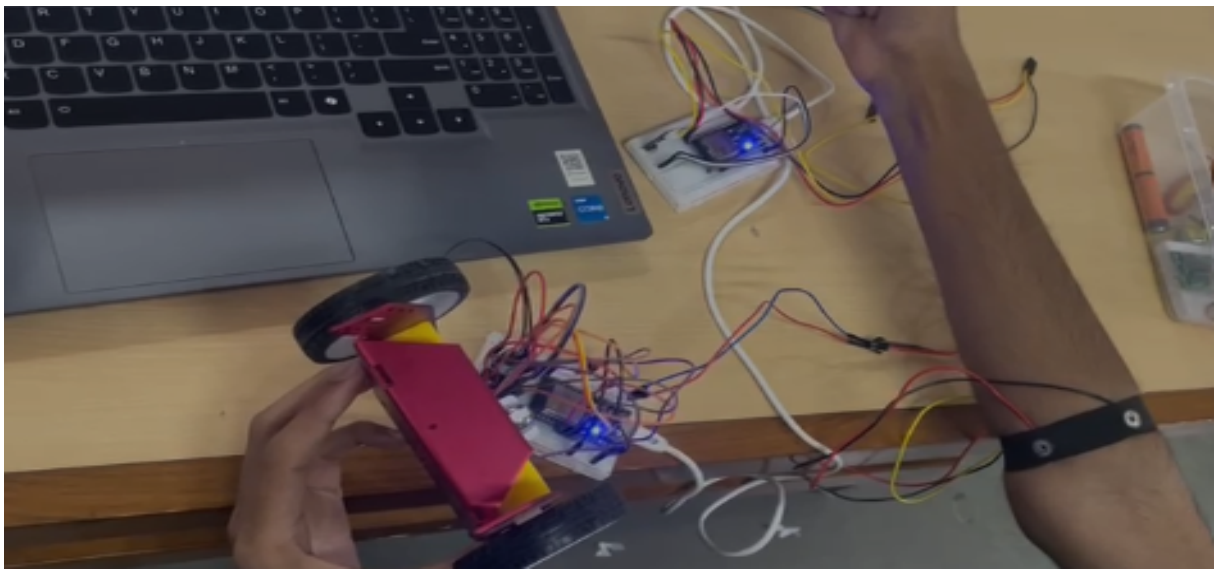


Fig 6.3 Working EMG armband with toy car

Chapter 7: Conclusion and Future Scope

7.1 Summary of Achievements

The EMG-based armband system for wheelchair control has been successfully implemented, achieving key milestones in both hardware and software design. The system effectively captures and processes Electromyography (EMG) signals from the user's muscles, enabling the control of a wheelchair's movement through forward and stop commands [1]. Wireless communication has been established using Wi-Fi, facilitating seamless data transmission between the armband and the wheelchair control system, ensuring real-time operation without physical tethering [2].

A significant accomplishment is the integration of a smoothing algorithm to process the raw EMG signals. The algorithm successfully reduces noise and unwanted peaks in the data, creating a smooth envelope that retains essential muscle movement patterns. This refinement enhances the accuracy and reliability of the system, ensuring the wheelchair responds accurately to the user’s muscle signals [3].

The overall achievement of this project lies in the creation of a robust, wireless, and intuitive system that allows users to control a wheelchair hands-free, enhancing autonomy for individuals with physical disabilities [4]. This accomplishment sets the foundation for further improvements, including adaptive algorithms and real-world testing, to ensure the system’s scalability in diverse environments [5].

7.2 Limitations of the Current System

While the prototype performed reliably, certain limitations were identified that may restrict its use in more complex applications:

- **Single-User Calibration**: Currently, the system is calibrated for a single user, and recalibration is necessary for each new user due to differences in individual muscle signal strength and patterns. Variations in muscle physiology, signal amplitudes, and personal threshold levels limit the system's adaptability to multiple users without additional calibration steps.
- **Environmental Noise and Signal Interference**: EMG signals are susceptible to noise, particularly from surrounding electrical devices and environmental factors. Despite using filters to minimize interference, occasional signal distortion can impact accuracy and responsiveness, especially in uncontrolled environments with significant electronic activity.
- **Power Limitations for Larger Devices**: The current system is designed for a low-power, lightweight toy car. Adapting this technology to control larger, more robust devices, like wheelchairs, would require significant modifications to handle the increased power demands and safety requirements. The existing motor and battery setup lacks the durability and power needed for extended use in real-world mobility devices.

7.3 Future Enhancements

Building on this initial prototype, several improvements and extensions are envisioned to increase the system's usability, reliability, and potential application scope:

1. **Multi-User Calibration and Adaptive Algorithms:** Future versions could integrate adaptive algorithms capable of automatic, personalized calibration for different users. By leveraging machine learning, the system could dynamically adjust signal thresholds to account for variations in muscle strength, thereby allowing seamless switching between users without manual recalibration. This approach would make the system accessible and user-friendly for a wider audience with different levels of muscle control.
2. **Integration with a Wheelchair for Assistive Mobility:** One primary future objective is to adapt the system for use with a wheelchair. This adaptation would involve scaling the system to work with more powerful motors, enhanced battery capacity, and safety mechanisms like obstacle detection. By incorporating sensors to identify obstacles and implementing fail-safes to prevent unintended movements, the system could provide a reliable and secure control option for individuals with mobility impairments.
3. **Advanced Signal Processing Techniques**: Integrating machine learning and signal-processing techniques such as deep learning, noise cancellation, and feature extraction would allow the system to better interpret complex EMG signals. Advanced algorithms could help differentiate between subtle muscle signals, reduce noise interference, and improve command accuracy. These techniques would also facilitate more complex commands, potentially enabling the system to distinguish between various levels of muscle activation or multi-joint actions for fine-tuned control.
4. **Battery Optimization for Extended Use**: Power efficiency will be essential for prolonged use, especially if the system is adapted for a wheelchair. Battery optimization could be achieved by exploring low-power components, implementing sleep modes for inactive periods, and using efficient power management algorithms. Extending battery life would enhance the practicality of the system for daily use, reducing the frequency of recharges and ensuring consistent performance.
5. **Field Trials and Real-World Testing:** Conducting extensive real-world testing with users who have mobility impairments would provide valuable insights into the system's performance under practical conditions. Such testing could reveal unanticipated challenges, help refine the interface, and ensure that the system meets the needs of real users [1]. Gathering user feedback through field trials would also inform future developments, potentially leading to feature enhancements, increased robustness, and improvements in system ergonomics for ease of use [8][9].

7.4 Conclusion

In conclusion, the EMG-based armband for wheelchair control presents a significant step forward in assistive technology, offering a practical solution for individuals with mobility impairments. By leveraging electromyographic (EMG) signals, the system allows users to control the movement of their wheelchair using natural muscle contractions, offering a hands-free and intuitive interface. The integration of key components such as the ESP32 microcontroller, Wi-Fi communication, and signal processing algorithms enables efficient real-time control, enhancing the overall user experience.

The design incorporates several essential features, including noise reduction techniques like band-pass filtering and envelope smoothing, to improve the accuracy and reliability of the muscle signals. These methods ensure that the system can accurately detect desired muscle movements while eliminating unwanted noise and irregularities caused by the muscle's transition between active and rest states. The system’s responsiveness is further enhanced by its ability to adjust to varying user inputs, providing consistent and smooth control over the wheelchair.

Moreover, the wireless communication between the transmitter (EMG sensor) and the receiver (motor controller) ensures that the system remains flexible, making it easy for users to wear the armband and operate the wheelchair remotely. This feature not only improves the overall user comfort but also adds to the convenience and mobility of the system, making it a valuable tool for people with disabilities seeking greater independence in their daily activities.

Overall, this project serves as a foundation for future developments in EMG-based control systems, offering a scalable solution that can be expanded to other assistive devices. With further refinement, this technology has the potential to revolutionize the way individuals with disabilities interact with their environments, ultimately contributing to their enhanced quality of life.

References

1. U. A. Butt, “EMG-controlled wheelchair with Arduino,” Engineers Garage, Feb. 25, 2021. [Online]. Available: https://www.engineersgarage.com/emg-controlled-wheelchair-with-arduino. [Accessed: Nov. 23, 2024].
2. “Brain-Link Mobility: A Wheelchair Control System Utilizing EEG, EMG, and EOG Signals for Enhanced Mobility - A Review,” International Journal of Research Publication and Reviews, vol. 5, no. 2, pp. 2924–2926, 2024. [Online]. Available: https://ijrpr.com/uploads/V5ISSUE2/IJRPR22899.pdf. [Accessed: Nov. 23, 2024].
3. “Making ESP32 WiFi/Bluetooth work together,” Stack Overflow. [Online]. Available: https://stackoverflow.com/questions/65636417/making-esp32-wifi-bluetooth-work-together. [Accessed: Nov. 23, 2024].
4. R. Santos and R. Santos, “Installing ESP32 in Arduino IDE (Windows, Mac OS X, Linux),” Random Nerd Tutorials, Feb. 28, 2024. [Online]. Available: https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions. [Accessed: Nov. 23, 2024].
5. “Introduction to ESP32,” TutorialsPoint. [Online]. Available: https://www.tutorialspoint.com/esp32_for_iot/esp32_for_iot_introduction.htm. [Accessed: Nov. 23, 2024].
6. “L293D Motor Driver,” Instructables, Sep. 26, 2017. [Online]. Available: https://www.instructables.com/L293D-Motor-Driver/. [Accessed: Nov. 23, 2024].
7. J. M. López-Guede, A. Almeida, and M. Blanco-Izquierdo, “A wheelchair control system based on EMG and EEG signals for disabled people,” in 2018 IEEE Global Humanitarian Technology Conference (GHTC), San Jose, CA, USA, 2018, pp. 1–8. doi: 10.1109/GHTC.2018.8601984.
8. R. Rajesh and S. R. Deepak, “IoT-Based Smart Wheelchair for Disabled People,” in 2021 IEEE International Conference on Communication, Computing and Internet of Things (IC3IoT), Chennai, India, 2021, pp. 1–5. doi: 10.1109/IC3IoT53480.2021.9617721.
9. H. Ren, Y. Liu, and Y. Bai, “Research and Implementation of a Smart Wheelchair System Based on Multi-Sensor Fusion,” in 2022 IEEE Sensors Applications Symposium (SAS), Sundsvall, Sweden, 2022, pp. 1–4. doi: 10.1109/SAS55333.2022.9762974.
10. Khatri, D. “Upsidedownlabs/Bioamp-EXG-Pill: Bioamp EXG pill is a small and elegant analog front end (AFE) Board for Biopotential Signal Acquisition,” in GitHub. Available at: https://github.com/upsidedownlabs/BioAmp-EXG-Pill (Accessed: 23 November 2024).