Real-Time Gesture Control System for Precision Control

Problem definition

The traditional interface tools, such as keyboards, mice, and joysticks, lack the precision and accessibility necessary for people with movement disabilities. Additionally, current systems are limited in enabling real-time, continuous communication with both hardware and software applications.
This project aims to extend these capabilities, providing a more adaptable, precise, and inclusive solution for users who require enhanced control and seamless interaction.

Proposed solution

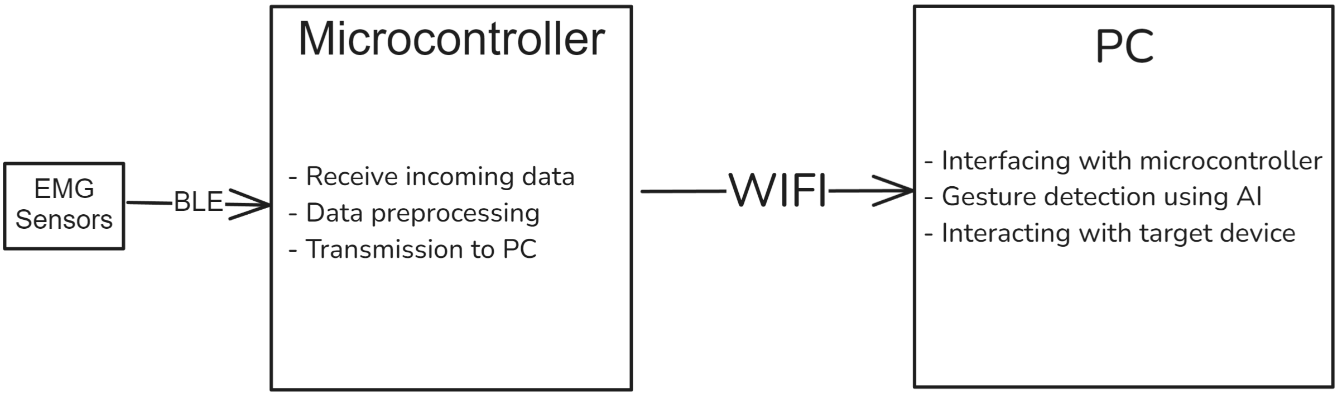
The solution will be a combined system consisting of a local data acquisition device and an external data processing hub, utilizing the computing power of a PC. The local device will be able to detect incoming signals, preprocess them, and send them for further processing and inference to the hub device. The hub device will identify a predefined set of gestures and send the corresponding control commands to a drone. We will demonstrate the functionality of the system using our own drone.

The proposed functionality includes:

- Data acquisition by combining EMG (electromyography) signals and IMU measurements
- Detecting a predefined set of movements
- Controlling applications (e.g., drones) via movements

Similar solutions have already been presented by MIT, but they are based on the deprecated Myo Thalmic Labs armband. Our solution aims to further explore data processing based on other research, such as "Wavelet transform theory and its application in EMG signal processing". Additionally, there are demos available for our sensor of choice (uMyo EMG) demonstrating control of a robot arm, for example.

System diagram:



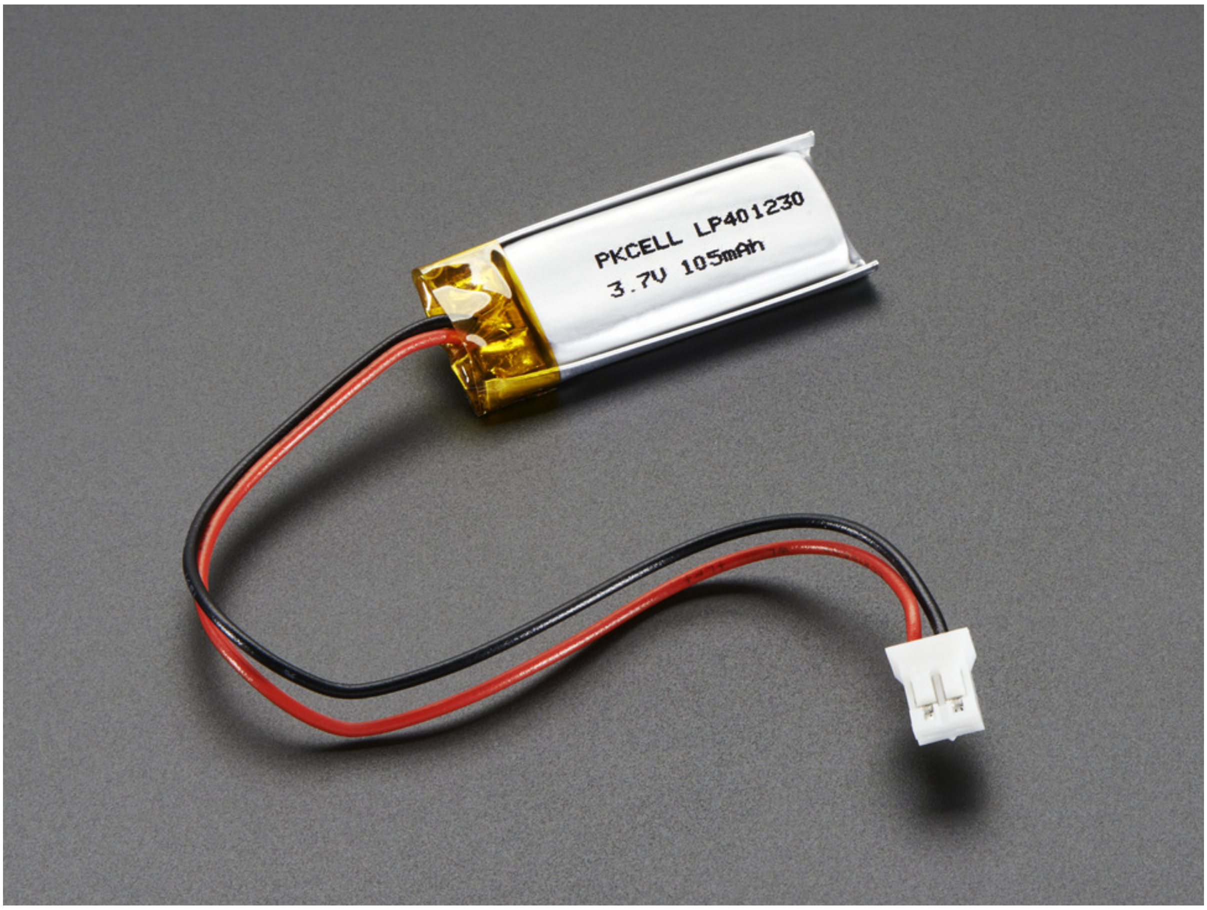
Hardware list

- **5x** uMyo EMG sensor (3x uMyo kit + 2x uMyo)
    - Small bracelet 25x290 mm
- **2x** 1s (~100mAh-200mAh) LiPo/Li-Ion battery with JST (PH2.0) connector

- **1x** ESP32 Microcontroller
- **1x** 3.3V Voltage Regulator for ESP32 (LDO would suffice)
- **1x** 1s (~500 mAh 1C) LiPo battery for ESP32 (ideally with protection circuit)
- **1x** TP4056 battery charger for ESP32
- **1x** IMU (6 or 9 DoF)
  - LSM6DSR
  - BNO055

More on battery

The battery connector has to have + and - on right sides. Here's an example of a suitable battery for uMyo sensors:



Optional (can be useful for experiments):

- USB receiver for uMyo

Alternatives:

- Myo Thalmic Labs armband (discontinued)
- MYOWARE 2.0
- BioAmp EXG Pill

Reasoning behind uMyo being the best choice:

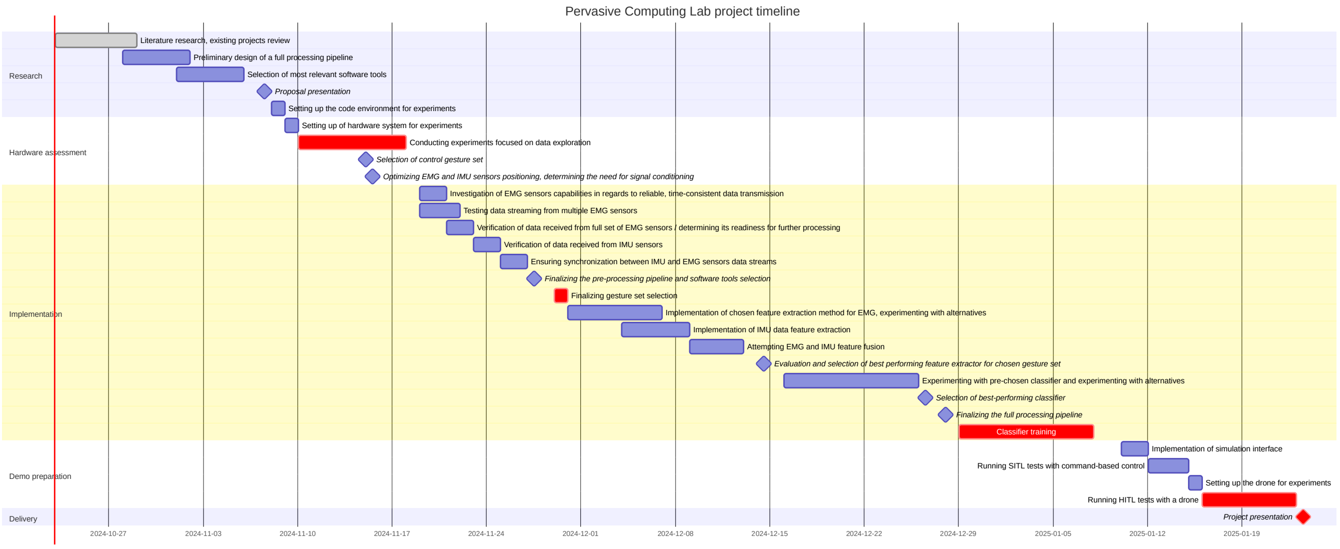
- Affordable price
- Open source hardware and software
- Fast delivery

- Enables reliable usage of dry electrode which will reduce parts waste and costs during project implementation
- Active support for both hardware and software
- Suitable for both prototyping and integration into end project

Potential software

- ESP IDF
  - Bluetooth API
  - Wi-Fi
- ESP-DSP
  - If signal conditioning is needed for better SNR
- uMyo provided libraries
- Scipy
- Eigen
- ROS2
- Machine Learning
  - PyTorch
    - If a neural network is needed for gesture recognition
  - Scikit-Learn / MLPACK
    - For clustering in Python or C++

Project timeline (Grantt chart)



Members - Group 1

- Nadezhda Varzonova, 12244356
- Gilbert Tanner, 12210784

Reference material

- Plug-and-Play Gesture Control Using Muscle and Motion Sensors
- Electromyography Gesture Identification Using CNN-RNN Neural Network for Controlling Quadcopters
- Electromyogram-Based Classification of Hand and Finger Gestures Using Artificial Neural Networks