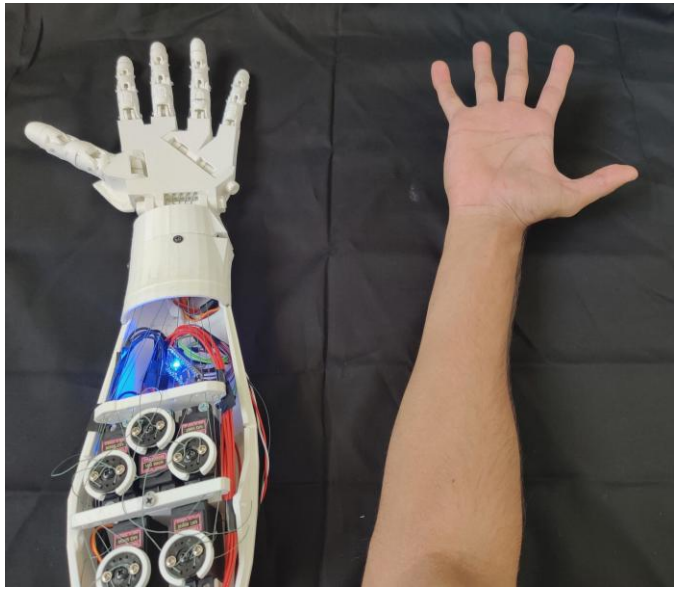
AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

Smart Prosthetic hand

INTRODUCTION TO NN, CNN AND GNN
ANALOG SYSTEM DESIGN

Guru Jaya Surya Yadav	CB.AI.U4AIM24101.
J. Tej Krishna Sai	CB.AI.U4AIM24117.
P. Teja Prakash Royal	CB.AI.U4AIM24136.
S. Ankith	CB.AI.U4AIM24147.



Guided by
Dr. Snigdhatanu Acharya
Dr. Amrutha Veluppal

Abstract

- Developed a cost-effective smart prosthetic hand
- Uses EMG sensors to capture muscle signals
- Classifies gestures using a trained neural network
- Real-time control via Arduino and Raspberry Pi

Introduction

Limb loss severely impacts functional independence and quality of life.

Conventional prosthetic devices often lack affordability, dexterity, and natural control.

EMG V3 module(EMGv3) signals provide a non-invasive interface for intuitive prosthetic control.

Combining EMG with deep learning enables real-time, accurate gesture recognition.

Problem Statement

- Many prosthetic hands are expensive and hard to use.
- They cannot move like real hands or understand signals from muscles.
- People with limb loss struggle to do daily tasks with current prosthetics.
- There is a need for a smart, low-cost hand that responds to muscle signals easily.

Objectives

- Design and build a cost-effective EMG-controlled prosthetic hand system.
- Capture and preprocess EMG signals using V3 electrodes and Arduino Uno.
- Train a lightweight neural network model to classify 'open' and 'close' gestures.
- Deploy the model on Raspberry Pi 5 via TensorFlow Lite for real-time inference.
- Integrate classification output to control six servo motors in a 3D-printed hand.

Smart Prosthetic Hand – Signal Flow:

- EMG signals are captured and processed to control the hand.
- Real-time communication between Arduino and Raspberry Pi enables smooth movement.

Methodology Overview

EMG Signal Acquisition → Preprocessing → Feature Extraction→Servo actuation→Hand movement.

Neural Network Model Training and Validation on Google Colab.

Conversion to TensorFlow Lite for embedded deployment.

Real-time data flow: EMG → Arduino → Raspberry Pi → Model → Servo→Hand Movement.

Hardware Used

- V3 EMG Electrodes for surface muscle signal acquisition.
- Arduino Uno for analog signal conditioning and serial data transmission.
- Raspberry Pi 5 for running the TensorFlow Lite model.
- Six SG90 Servo Motors to drive the prosthetic fingers.
- 3D-Printed Prosthetic Hand Frame.
- Power Supply, Breadboard, and Jumper Wires.

Arduino Uno

- Microcontroller: ATmega328P
- Operating Voltage: 5V
- Input Voltage: 7–12V
- Digital I/O Pins: 14 (6 PWM)
- Analog Input Pins: 6
- Flash Memory: 32 KB
- Clock Speed: 16 MHz

V3 EMG Sensor

- Operating Voltage: +3.3V to +5V
- Output Signal: Analog (0–5V)
- Bandwidth: 10 – 500 Hz
- Electrodes: Snap-type disposable gel electrodes
- Dimensions: Compact module with onboard amplifier and filters

Raspberry Pi 5

- CPU: Quad-core Cortex-A76 @ 2.4GHz
- RAM: 4GB / 8GB LPDDR4X
- GPIO: 40-pin header
- USB: 2 × USB 3.0, 2 × USB 2.0
- Connectivity: Gigabit Ethernet, Wi-Fi 5, Bluetooth 5.0
- Video: Dual 4K display support (via micro-HDMI)

Servo Motors (e.g., MG995)

- Operating Voltage: 4.8V – 7.2V
- Stall Torque: ~9.4 kg·cm @ 4.8V, ~11 kg·cm @ 6V
- Speed: 0.17 sec/60° @ 4.8V
- Rotation Angle: 0°–180°
- Weight: ~55g
- Gear Type: Metal gears

Software & Tools

- Arduino IDE for firmware development on Arduino Uno.
- Python 3 with TensorFlow/Keras for neural network training.
- TensorFlow Lite for efficient model inference on Raspberry Pi.
- PySerial library for serial communication.
- Google Colab for model development and testing.

Neural Network Architecture

- Input: Preprocessed EMG feature vectors (e.g., MAV, RMS).
- Hidden Layers: Two Dense layers with ReLU activation.
- Output Layer: Softmax activation for binary classification ('open', 'close').
- Optimizer: Adam with binary cross-entropy loss.
- Validation Accuracy: ~97% on held-out dataset.

Circuit Diagram

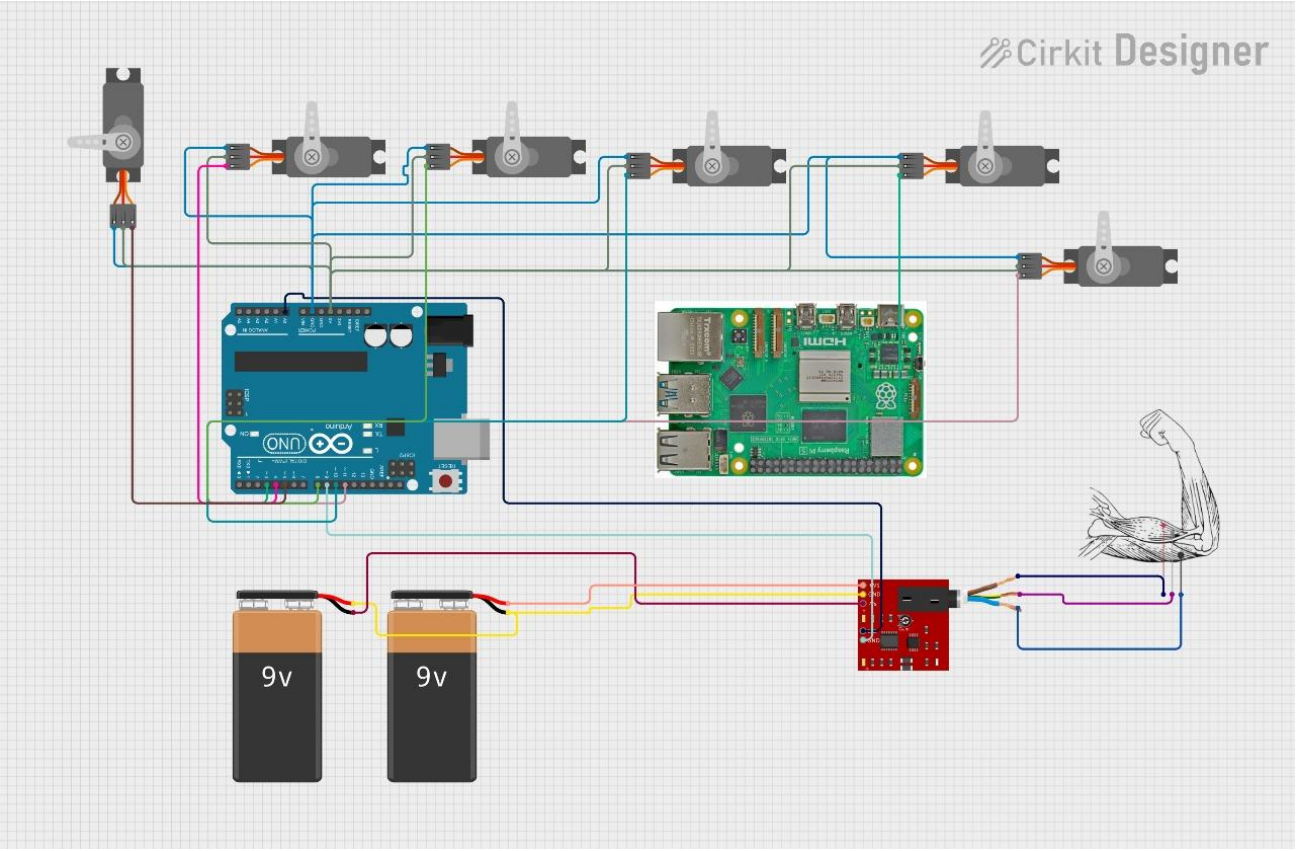
EMG electrodes connected to Arduino analog inputs.

Arduino USB → Serial → Raspberry Pi for data transfer.

Servo motor PWM outputs driven by Arduino digital pins.

Power distribution for sensors, microcontrollers, and servos.

Insert detailed circuit schematic here.



Testing & Results

- Classification Accuracy: 97% on real-time validation.
- Total System Latency: ~200 ms (acquisition to actuation).
- Actuation Delay: ~50 ms from command to servo movement.
- Robustness: <7% drop in accuracy with electrode repositioning..

Future Scope

- Add more gestures (Grip, Point)
- Gesture adaptation to multiple users
- Feedback mechanism (touch, vibration)
- IoT integration for health monitoring

Conclusion

Developed a low-cost, real-time EMG-controlled prosthetic hand.

Achieved high accuracy and low latency with lightweight NN model.

Demonstrated feasibility of cost-effective deep-learning prosthetics.

Platform ready for expansion to multi-gesture control and miniaturization.

References

[1] B. Hudgins, P. Parker, and R. N. Scott, “A new strategy for multifunction myoelectric control,” IEEE Trans. Biomed. Eng., vol. 40, no. 1, pp. 82–94, 1993.

[2] M. Oskoei and H. Hu, “Myoelectric control systems—A survey,” Biomed. Signal Process. Control, vol. 2, no. 4, pp. 275–294, 2007.

[3] L. Atzori et al., “Electromyography data for non-invasive naturally-controlled robotic hand prostheses,” Sci. Data, vol. 1, 140053, 2014.

[4] A. Phinyomark, C. Phukpattaranont, and P. Limsakul, “Feature extraction and classification for myoelectric control: A review,” IEEE Trans. Neural Syst. Rehabil. Eng., vol. 20, no. 3, pp. 297–311, 2012.

[5] R. N. Khushaba et al., “A review on electromyography-based gesture recognition: Methods, techniques, and applications,” IEEE Signal Process. Mag., vol. 33, no. 4, pp. 110–128, 2016.