Mini Project Report

Entitled

Controlling the speed of the motor through Hand Gesture using LinearRegression, Mediapipe and OpenCV

Submitted to the Department of Electronics Engineering in Partial Fulfilment for the Requirements for the Degree of

Bachelor of Technology

(Electronics and Communication)

: Presented & Submitted By :

Hemil Prajapati, Krish Vasoya, Harshal Patel

Roll No. (U21EC032, U21EC062, U21EC064)

B. TECH. VI (EC), 6th Semester

: Guided By :

Dr. Kishor Upla

Assistant Professor, SVNIT

(Year: 2022-23)

DEPARTMENT OF ELECTRONICS ENGINEERING

SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECHNOLOGY

Surat-395007, Gujarat, INDIA.

Sardar Vallabhbhai National Institute Of Technology

Surat - 395 007, Gujarat, India

DEPARTMENT OF ELECTRONICS ENGINEERING

CERTIFICATE

This is to certify that the Mini-Project Report entitled “**Controlling the speed of the motor through Hand Gesture using LinearRegression, Mediapipe and OpenCV** ” is presented & submitted by Hemil Prajapati, Krish Vasoya, Harshal Patel, bearing Roll No. U21EC032, U21EC062, U21EC064, of B.Tech. VI, 6^{th} Semester in the partial fulfillment of the requirement for the award of B.Tech. Degree in Electronics & Communication Engineering for academic year 2022-23.

They have successfully and satisfactorily completed their **Mini-Project** in all respects. We, certify that the work is comprehensive, complete and fit for evaluation.

Dr. Kishor Upla
Assistant Professor & Project Guide

Abstract

In this project, we present a novel approach to control the speed of a motor through hand gestures employing Linear Regression, Mediapipe, and OpenCV technologies. The traditional methods of motor control often rely on physical interfaces or predefined commands, limiting flexibility and ease of use. Leveraging advancements in computer vision and machine learning, our proposed system offers an intuitive and hands-free solution to control motor speed.

The core of our system lies in real-time hand gesture recognition achieved through the integration of Mediapipe and OpenCV libraries. Mediapipe provides robust hand tracking capabilities, enabling precise localization of hand movements in video streams. OpenCV facilitates image processing tasks, allowing for the extraction of relevant features from hand gestures.

To establish a relationship between detected hand gestures and motor speed, we employ Linear Regression, a supervised learning algorithm. Through a process of feature extraction and training, our model learns to predict motor speed based on the spatial characteristics of hand gestures captured by the camera. This enables seamless translation of hand movements into corresponding motor speed adjustments.

We evaluate the performance of our system through a series of experiments, measuring accuracy, responsiveness, and robustness across various hand gestures and environmental conditions. Results demonstrate the efficacy of our approach in accurately controlling motor speed in real-time, with minimal latency and high precision.

Our proposed system offers numerous practical applications across industries, including robotics, automation, smart-home automation and human-machine interaction. By providing an intuitive and non-intrusive means of motor control, it has the potential to enhance user experience and productivity in diverse settings.

Table of Contents

List of Figures

Chapter 1
Hardware and Software Used

1.1 Hardware Used:-

This section discusses about various hardware tools required for implementation.

1.1.1 Arduino UNO

Arduino Uno is a microcontroller board based on the ATmega328P chip. It is one of the most popular and widely used boards in the Arduino family due to its simplicity, versatility, and affordability.

Figure 1.1: Arduino UNO

Key features of the Arduino UNO include:
1. Microcontroller: ATmega328P running at 16MHz
2. Flash Memory: 32KB flash memory
3. SRAM: 2KB
4. EEPROM: 1KB
5. Digital I/O Pins: 14 (of which 6 provide PWM output)

6. Analog Input Pins: 6
7. Voltage Regulator: USB connection or an external power supply (7-20V DC)
8. Reset Button: To reset the microcontroller
9. Operating Voltage: 5V

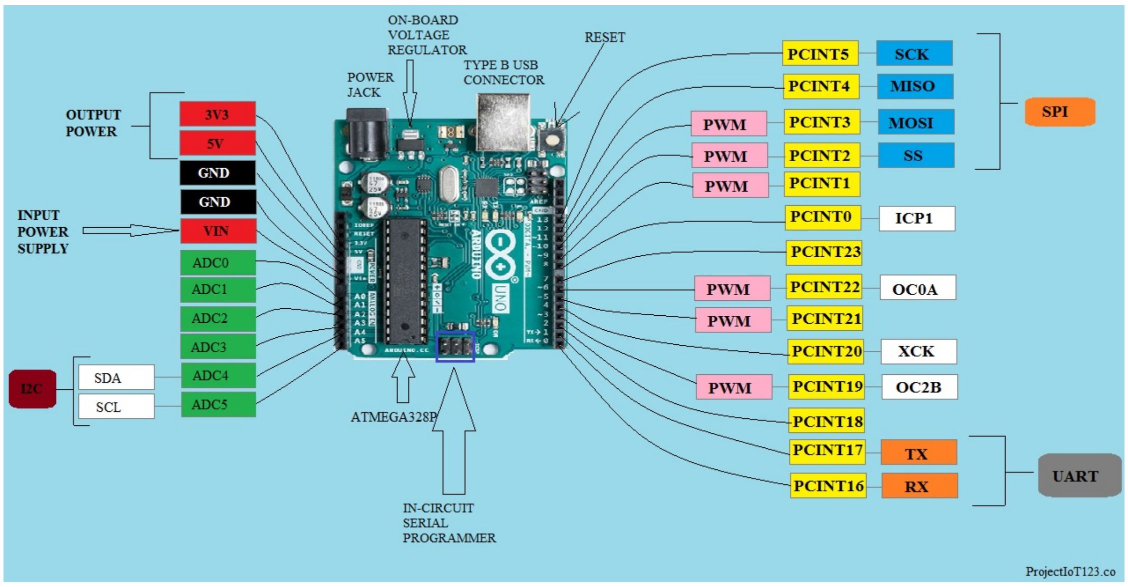

Figure 1.2: Arduino UNO pin diagram

Arduino Uno is widely used in various projects including robotics, automation, Internet of Things (IoT), and prototyping due to its ease of use, extensive community support, and a vast ecosystem of shields, modules, and libraries available for expansion and customization [1]. In our project, we have used Arduino UNO as a microcontroller unit for controlling the speed of the motors and providing it PWM.

1.1.2 Motor

A DC (direct current) motor converts electrical energy into mechanical motion. It consists of a stationary part (stator) containing magnets or electromagnets and a rotating part (rotor) with a coil of wire (armature). When current flows through the armature, it interacts with the magnetic field, causing the rotor to turn. A commutator and brushes reverse the current direction, ensuring continuous rotation. DC motors are used in various applications due to their simplicity and controllability, but they require maintenance and can be susceptible to electromagnetic interference.

1.1.3 MD10c Motor Driver

The MD10C motor driver is a dual-channel H-Bridge module designed for controlling two brushed DC motors independently [2]. It incorporates input control logic to interpret signals for speed and direction control, alongside features like braking and coasting modes. Additionally, built-in protection circuitry guards against overcurrent, overvoltage, and reverse polarity, ensuring the safety of both the driver and connected motors. Power management components such as voltage regulators maintain stable operation. The driver interfaces with external control signals through connectors or terminals, providing a versatile solution for a wide array of projects requiring efficient and reliable motor control. It include diagnostic features such as fault detection and feedback signals. These features help in monitoring the status of the motor driver and detecting any potential issues or faults during operation.

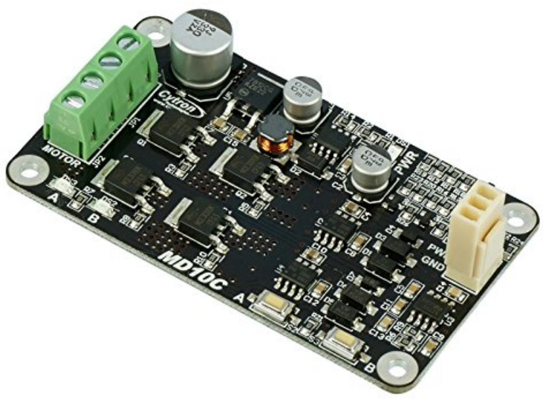

Figure 1.3: MD10c Motor Driver

Main features of it are:
1. Operating Voltage: 7-30V
2. Maximum Current: 10A per Channel
3. Internal Architecture: H-Bridge Configuration
4. Motor Channels: Can control 2DC motors simultaneously
5. Interface: Can be intefaced with Arduino, Raspberry PI

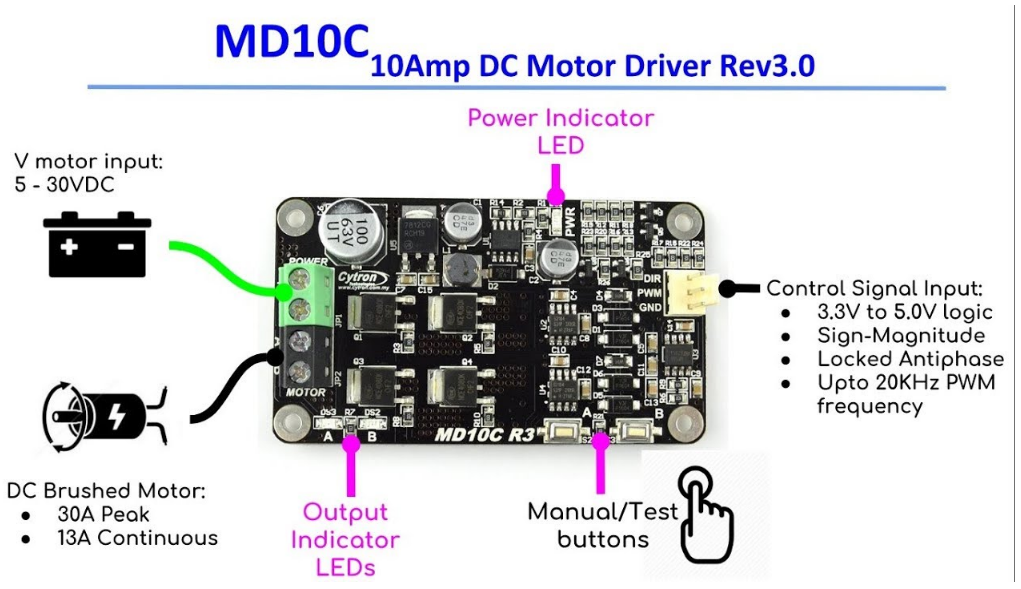

Figure 1.4: MD10c pinout

The MD10C motor driver offers precise control of DC motors in a variety of applications, including robotics, RC vehicles, automation projects, industrial machinery, electronic toys, and hobby projects. Its bi-directional control, PWM capabilities, and high current capacity make it ideal for tasks requiring accurate speed and direction control in diverse settings.

1.2 Software Used:-

Here we discuss about the software we have used for implementation.

1.2.1 VS Code

Visual Studio Code (VSCode) is a popular open-source code editor developed by Microsoft. It offers a lightweight yet powerful environment for coding in various programming languages. With features like IntelliSense for smart code completion, debugging support, Git integration, and a vast library of extensions, VSCode provides developers with a highly customizable and efficient tool for software development across different platforms. For our project, we have written our code in VS Code with .py extension.

1.2.2 ROS

ROS (Robot Operating System) is an open-source framework widely used for developing robotic systems. It provides a flexible and modular architecture that simplifies the process of creating complex robot applications. ROS offers tools and libraries for tasks such as hardware abstraction, communication between processes, visualization, and simulation, making it a popular choice among researchers and robotics enthusiasts for prototyping, testing, and deploying robotic solutions.

1.2.3 Arduino IDE

The Arduino Integrated Development Environment (IDE) is a user-friendly platform for programming Arduino microcontroller boards. It features a simple interface for writing and uploading code, making it accessible to beginners and experienced users alike. With a vast library of pre-written code (sketches), extensive community support, and compatibility with various Arduino boards, the IDE facilitates rapid prototyping and development of electronic projects and IoT devices. For programming of Arduino UNO, we have used Arduino IDE and saved our sketch with .ino extension.

Chapter 2
Libraries and Dataset Used

2.0.1 Rospy

The rospy library is a Python client library for the Robot Operating System (ROS), enabling developers to create ROS nodes for robotic applications [3]. It supports the publish-subscribe messaging model, service calls, and interaction with the ROS parameter server. Developers use rospy for tasks such as sensor data processing, robot control, and navigation due to Python's simplicity. Integration with other ROS tools in different languages is seamless. Extensive documentation and community support ensure accessibility and effectiveness. Overall, rospy is a vital component in the ROS ecosystem, empowering Python developers to build complex robotic systems efficiently.

2.0.2 Mediapipe

Mediapipe is an open-source library by Google offering pre-trained models and tools for real-time perceptual tasks such as hand tracking, pose estimation, face detection, and object tracking [4]. With a simple API and cross-platform support, it's accessible for developers on desktop and mobile platforms. Its versatility allows integration into applications written in Python, C++, and Java. The library's active community and comprehensive documentation facilitate ease of use and development. Overall, Mediapipe simplifies the creation of real-time computer vision applications by providing robust, efficient tools and models, making it valuable for developers in various industries [4].

2.0.3 OpenCV

OpenCV (Open Source Computer Vision Library) is a versatile open-source software library used for computer vision and machine learning tasks [5]. It offers a wide range of functionalities including image and video processing, object detection, facial recognition, and machine learning algorithms. With cross-platform support and bindings for popular languages like Python and Java, developers can use OpenCV on various operating systems and platforms. Its extensive documentation, tutorials, and active community support make it accessible for both beginners and experienced developers. OpenCV finds applications in industries such as robotics, augmented reality, medical imaging, and surveillance, making it an indispensable tool in the field of computer vision

2.0.4 Pandas

Pandas is a Python library for data manipulation and analysis, providing data structures and functions for working with structured data. Its main structures are Series and DataFrame, allowing easy handling of tabular data. Pandas supports various data formats, data exploration, and analysis tasks such as filtering, sorting, and aggregation. It integrates well with other Python libraries and is optimized for performance. With extensive documentation and a large community, Pandas is widely used across industries for tasks including data preprocessing, analysis, and modeling, making it an indispensable tool for data scientists, analysts, and researchers in the Python ecosystem

2.0.5 Numpy

NumPy is a foundational library for numerical computing in Python, offering support for multi-dimensional arrays and mathematical functions. Its ndarray data structure facilitates efficient manipulation and computation on homogeneous data. NumPy's extensive collection of mathematical functions includes arithmetic, trigonometric, and statistical operations, optimized for performance. It integrates seamlessly with other Python libraries like SciPy and Matplotlib, forming the core of the scientific computing ecosystem. With comprehensive documentation and an active community, NumPy is widely used in scientific research, engineering, data analysis, and machine learning applications, providing essential tools for numerical computation and data manipulation in Python.

2.0.6 Matplotlib

Matplotlib is a versatile plotting library for Python, offering extensive capabilities for creating various types of plots and customizing their appearance. It supports multiple interfaces including a MATLAB-like scripting interface, object-oriented interface, and procedural interface, accommodating different programming styles. Matplotlib can generate plots in various formats and integrates seamlessly with other Python libraries like NumPy and Pandas. With comprehensive documentation and a large community, it's widely used in scientific research, data analysis, and engineering for creating publication-quality plots. Its flexibility, customization options, and ease of use make it an indispensable tool for data visualization in Python.

2.0.7 Scikit-Learn

Scikit-learn is a comprehensive machine learning library for Python, offering a wide array of algorithms for classification, regression, clustering, and more. It provides tools for data preprocessing, model evaluation, and validation, facilitating efficient model development. Integrating seamlessly with NumPy and pandas, scikit-learn supports both supervised and unsupervised learning tasks, making it versatile for various applications. With extensive documentation and a vibrant community, it's accessible to users of all levels, from beginners to experts. Widely used in academia and industry, scikit-learn simplifies the implementation of machine learning algorithms in Python, serving as an essential tool for data analysis and predictive modeling.

2.1 Dataset

Given below is the dataset for this project Table

Distance_IndexFinger (Percentage)	Motor_SpeedPWM
0	0
3	5
5	13
8	22
13	31
17	35
22	56
31	77
35	88
40	102
48	128
53	133
60	167
71	185
77	195
81	212
87	220
93	239
97	249
100	255

Chapter 3
Algorithm

3.1 Algorithm

1) We captured our image with the help of OpenCV library.
2) Now we converted the image in co-ordinate system with the help of mediapipe library.
3) We took the thumb as reference (i.e. origin) and measured the distance of index figure from reference.
4) Now, we converted the distance into percentage which is the standard used in training dataset as well as at the time of testing.
5) Now based on percentage values, we applied the linear regression model to predict the speed of motor.
6) There are possibility of human error in positioning the index finger so we have taken 20 readings for dataset and applied linear regression model on the same to also compensate for this human error.
7) The predicted values received from the Linear Regression model are mapped into 0 to 255 and given as PWM to the motor.

Conclusion

The project successfully demonstrates the implementation of hand gesture recognition using Mediapipe and Linear Regression to control the speed of a motor. It shows that Hand Gestures applications can be implemented using Supervised Learning algorithms with efficiency that is comparable to neural networks. Here's a summary of the key points:

1.Hand Gesture Recognition: The system utilizes the Mediapipe library to detect and track hand landmarks in real-time video streams captured by a webcam. Specifically, it focuses on landmarks such as wrist position, which serves as the basis for gesture recognition.

2.Data Collection: Hand gesture data is collected by extracting relevant features from the detected landmarks. These features are then used to train a Linear Regression model. The model learns to map the hand gestures to corresponding motor speed values.

3.Model Training: A Linear Regression model is trained using the collected hand gesture data. This model learns the relationship between hand gestures and motor speeds.

4.Motor Speed Prediction: After training, the model is used to predict the motor speed based on the detected hand gestures in real-time. The predicted speed values are then passed to the motor control system.

5.Motor Control: The predicted motor speed values are utilized to control the motor's operation. Depending on the specific hardware setup, appropriate methods for motor control are implemented.

6.Potential Extensions: The project can be extended in various ways, such as incorporating more complex gesture recognition algorithms, integrating additional sensors for enhanced control, or optimizing the model for better accuracy and robustness.

Overall, the project demonstrates the feasibility of using hand gestures as an intuitive interface for controlling motorized systems using Supervised Learning Algorithm. With further refinement and customization, it can be adapted for a wide range of applications, including robotics, home automation, and interactive systems.

References

[1] Components101. Arduino Uno Microcontroller. [Online]. Available: https://components101.com/microcontrollers/arduino-uno

[2] Cytron Technologies. Motor Control Tutorial. [Online]. Available: https://www.cytron.io/tutorial/control

[3] ROS Wiki Contributors. rospy. [Online]. Available: http://wiki.ros.org/rospy

[4] Mediapipe Contributors. Mediapipe Documentation. [Online]. Available: https://chuoling.github.io/mediapipe/

[5] OpenCV Contributors. OpenCV Documentation. [Online]. Available: https://opencv.org/