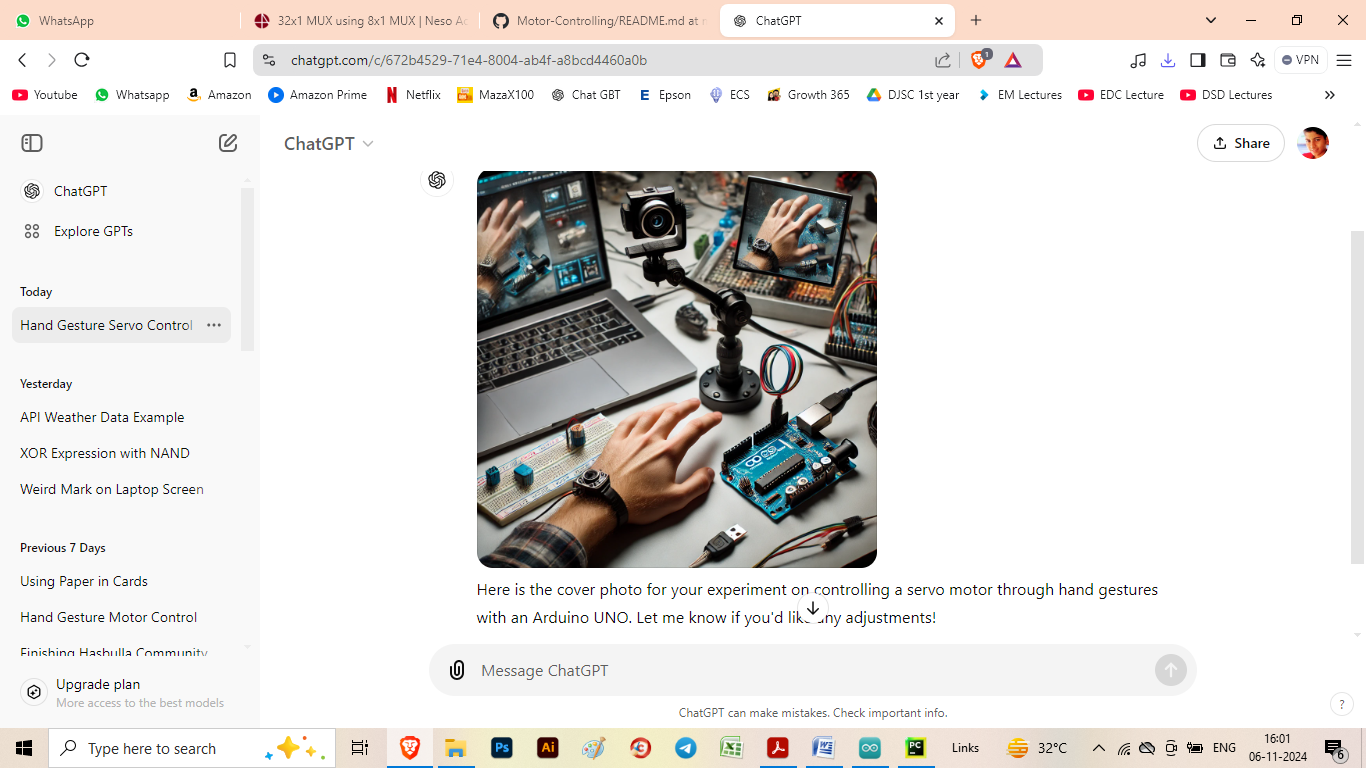
**Project Report**

**On**

**Motor Controlling using Hand Gestures**

**Title:-** Controlling Servo Motor through Hand Gestures with Arduino

UNO.

****

**Done and Executed by Chaitra Milind Rane**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Title** | **Page Number** |
| 1. | Introduction | 3 |
| 2. | Proposed Methodology | 4 |
| 3. | Algorithm | 6 |
| 4. | Output | 8 |
| 5. | Conclusion | 11 |

1. **Introduction**

My project is all about controlling motor speed using hand gestures. The main objective of this experiment is to demonstrate effective communication between Python code, Arduino, and a servo motor. We’ve utilized a range of powerful libraries for this task, including pyserial (for serial communication), opencv-python (for image processing), cvzone (to bridge Python with Arduino), mediapipe (for hand tracking), and numpy (for numerical operations).

This project employs a webcam to capture hand gestures, detects hand landmarks using Mediapipe, and calculates the distance between specific points (the thumb and index finger) to map and send motor control signals to the Arduino. The result is a responsive system that adjusts motor speed based on hand gestures.

1. **Proposed Methodology**
2. **Hardware Setup:**

* Connect the Arduino to the computer and ensure it is properly interfaced with the servo motor.
* Attach the servo motor's signal wire to the 9th digital pin on the Arduino.
* Connect the power wires: positive to the 5V pin and negative to GND on the Arduino.

1. **Software Environment:**

* Use Python to write and execute the primary code for gesture detection and data transmission.
* Ensure pyserial, opencv-python, cvzone, mediapipe, and numpy are installed for seamless operation.

1. **Hand Gesture Detection:**

* Capture the live video feed using OpenCV.
* Convert video frames to RGB and use Mediapipe to detect and track hand landmarks.
* Extract coordinates for the thumb and index finger tips.

1. **Distance Calculation and Mapping:**

 Calculate the Euclidean distance between the detected points of the thumb and index finger.

 Map this distance to motor control values corresponding to different speeds and directions:

* **Inward and Fast:** Higher speed and clockwise rotation.
* **Outward and Fast:** Higher speed and anticlockwise rotation.
* **Inward and Slow:** Lower speed and clockwise rotation.
* **Outward and Slow:** Lower speed and anticlockwise rotation.

1. **Data Transmission to Arduino:**

* Send the calculated control value from the Python script to the Arduino using the cvzone.SerialModule or pyserial library.
* Ensure the format is appropriate for the Arduino to interpret the data.

1. **Motor Control Logic on Arduino:**

* Receive the speed and direction data from the Python script.
* Translate the received values to control the servo motor’s position and speed using PWM signals on pin 9.

1. **Feedback Loop and Visualization:**

* Visualize the detected hand landmarks and the line between the thumb and index finger in real-time using Open CV.
* Display debug information such as the mapped speed and direction values to aid in troubleshooting and fine-tuning.

1. **Testing and Optimization:**

* Perform initial tests to verify gesture detection and servo response.
* Refine the mapping function for smoother motor control and response accuracy.

**3. Algorithm**

1. **Algorithm:**
2. **Start**
3. **Initialize System:**

* Set up the webcam.
* Initialize the servo motor on pin 9.

1. **Run the Arduino IDE code.**
2. **Start Video Capture:**

* Begin capturing video from the webcam.

1. **Process Each Frame:**

* For each frame captured from the webcam:
  1. Convert the frame to grayscale.
  2. Apply gesture detection algorithm (e.g., using OpenCV for contour detection).

1. **Identify Hand Gestures:**

* If a hand gesture is detected:
  1. **Check Gesture Type**:
     1. **Inward and Fast**: Move servo to position for inward fast.
     2. **Outward and Fast**: Move servo to position for outward fast.
     3. **Inward and Slow**: Move servo to position for inward slow.
     4. **Outward and Slow**: Move servo to position for outward slow.
  2. If no gesture is detected, keep the servo at the last position.

1. **Control Servo Motor:**

* Based on the identified gesture, send the corresponding command to the servo motor.
* Adjust the position of the servo motor based on the gesture type.

1. **Repeat Process:**

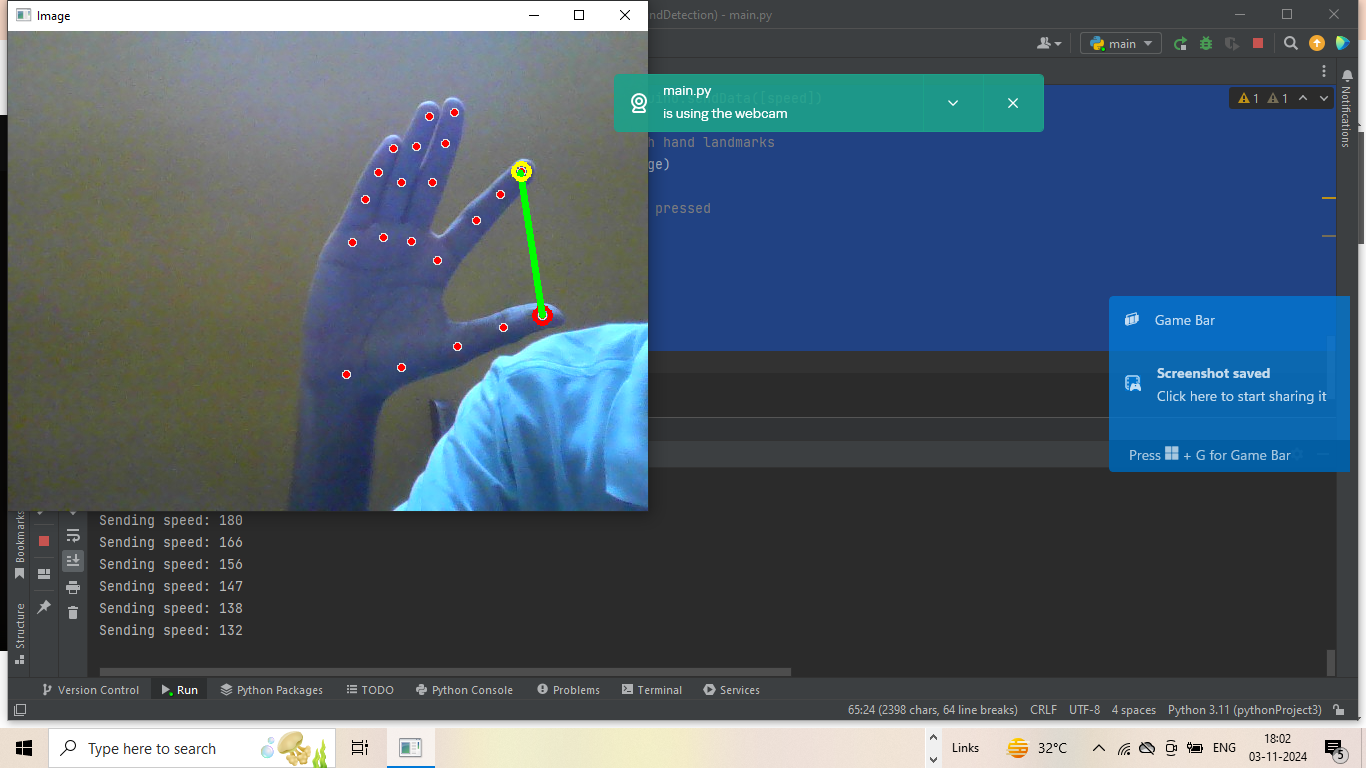
* Repeat steps 3-5 until the program is terminated.

1. **Cleanup:**

* Release the webcam and any resources used when exiting.

1. **End**

**4. Output :**

****

**Serial Monitor in pycharm:**

C:\Users\Admin\PycharmProjects\pythonProject3\venv\Scripts\python.exe C:\Users\Admin\PycharmProjects\pythonProject3\main.py

Attempt 1 of 5 to connect...

Serial Device Connected

INFO: Created TensorFlow Lite XNNPACK delegate for CPU.

WARNING: All log messages before absl::InitializeLog() is called are written to STDERR

W0000 00:00:1730637333.271257 15256 inference\_feedback\_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling support for feedback tensors.

W0000 00:00:1730637333.319877 15256 inference\_feedback\_manager.cc:114] Feedback manager requires a model with a single signature inference. Disabling support for feedback tensors.

C:\Users\Admin\PycharmProjects\pythonProject3\venv\Lib\site-packages\google\protobuf\symbol\_database.py:55: UserWarning: SymbolDatabase.GetPrototype() is deprecated. Please use message\_factory.GetMessageClass() instead. SymbolDatabase.GetPrototype() will be removed soon.

warnings.warn('SymbolDatabase.GetPrototype() is deprecated. Please '

Sending speed: 67

Sending speed: 69

Sending speed: 69

Sending speed: 74

Sending speed: 105

Sending speed: 127

Sending speed: 127

Sending speed: 139

Sending speed: 141

Sending speed: 137

Sending speed: 143

Sending speed: 145

Sending speed: 141

Sending speed: 148

Sending speed: 151

Sending speed: 153

Sending speed: 144

Sending speed: 149

Sending speed: 152

Sending speed: 140

Sending speed: 147

Sending speed: 159

Sending speed: 147

Sending speed: 160

Sending speed: 160

Sending speed: 148

Sending speed: 162

Sending speed: 159

Sending speed: 145

Sending speed: 156

Sending speed: 165

Sending speed: 167

Sending speed: 138

Sending speed: 114

Sending speed: 79

Sending speed: 40

Sending speed: 37

Sending speed: 26

Sending speed: 29

Sending speed: 27

Sending speed: 47

Sending speed: 60

* + - 1. **Conclusion**

The hand gesture motor control experiment successfully demonstrated the integration of computer vision and robotics to create an intuitive and responsive system for controlling a servo motor using hand gestures. By utilizing a webcam for gesture detection and an Arduino for motor control, the project showcased how modern technology can simplify user interactions with machines.

Key outcomes from the experiment include:

* **Real-time Gesture Recognition**: The implementation of a gesture detection algorithm allowed for effective recognition of specific hand movements, translating them into corresponding motor actions. This highlights the potential of computer vision applications in robotics.
* **Intuitive Control Interface**: By leveraging natural hand gestures, users can interact with the servo motor in a more intuitive manner compared to traditional controls. This can enhance user experience in various applications, from robotics to assistive technology.
* **Challenges and Learning**: The project encountered challenges, such as ensuring accurate gesture detection and minimizing latency. These challenges provided valuable insights into the complexities of real-time systems and the importance of calibration and optimization.
* **Future Applications**: The principles established in this experiment can be expanded to more complex systems involving multiple motors and advanced gesture recognition techniques. Potential applications include robotic arms, remote control systems, and assistive devices for individuals with mobility impairments.

Overall, this experiment not only fulfilled its objective of controlling a servo motor with hand gestures but also laid the groundwork for future explorations in gesture-based control systems. Continued refinement and expansion of the system can lead to innovative solutions across various fields, demonstrating the exciting possibilities at the intersection of robotics and human-computer interaction.