MECHATRONICS SYSTEM INTEGRATION (MCTA 3203)

SEMESTER 1 2024/2025

WEEK 4A: SERIAL COMMUNICATION IMU

SECTION 2

GROUP 8

LECTURER: ZULKIFLI BIN ZAINAL ABIDIN & WAHJU SEDIONO

NO.	GROUP MEMBERS	MATRIC NO.
1.	AHMAD DARWISH BIN AHMAD TERMIZI	2212089
2.	SYABAB ALHAQQI BIN JAAFAR	2211117
3.	WAN NAIMULLAH BIN MOHD AZRUDDIN	2214837
4.	ZAIMUDIN ZAKWAN BIN NORZAMRI	2217513
5.	ZAMIR MUTTAQIN BIN MUHAMMAD BALYAN	2212985

Date of Experiment: Wednesday, 30 October 2024

Date of Submission: Wednesday, 6 November 2024

ABSTRACT

In this experiment, we created a simple hand gesture recognition system using an MPU6050 motion sensor and an Arduino. The MPU6050 captured accelerometer and gyroscope data, which was sent to a computer via serial communication. Using a Python script, we processed this data to identify specific hand gestures based on preset conditions. The detected gestures were also visualised on an x-y coordinate system to show hand movement paths. This setup demonstrates a basic way to recognize gestures using motion data, which could be useful for gesture-based controls in various applications.

TABLE OF CONTENTS

INTRODUCTION

This experiment aimed to create a basic hand gesture recognition system using an MPU6050 motion sensor, an Arduino, and a computer. The MPU6050, with its accelerometer and gyroscope, captured motion data to detect simple hand gestures, which were identified on the computer via a Python program. By transmitting real-time sensor data through serial communication, we could recognize specific gestures based on preset patterns. This setup demonstrates a practical approach to gesture-based control using motion sensors and microcontrollers.

MATERIALS AND EQUIPMENTS

1. ARDUINO MEGA 2560
2. MPU6050 SENSOR
3. JUMPER WIRES
4. SERVO MOTOR
5. USB CABLE

EXPERIMENTAL SETUP

1. Since the MPU6050 uses I2C communication, connect its SDA and SCL pins to the matching pins on the Arduino (typically A4 and A5 on most boards).
2. Link the power (VCC) and ground (GND) pins of the MPU6050 to the 5V and GND pins on the Arduino.
3. Use a USB cable to connect the Arduino board to your computer for data transfer and programming.

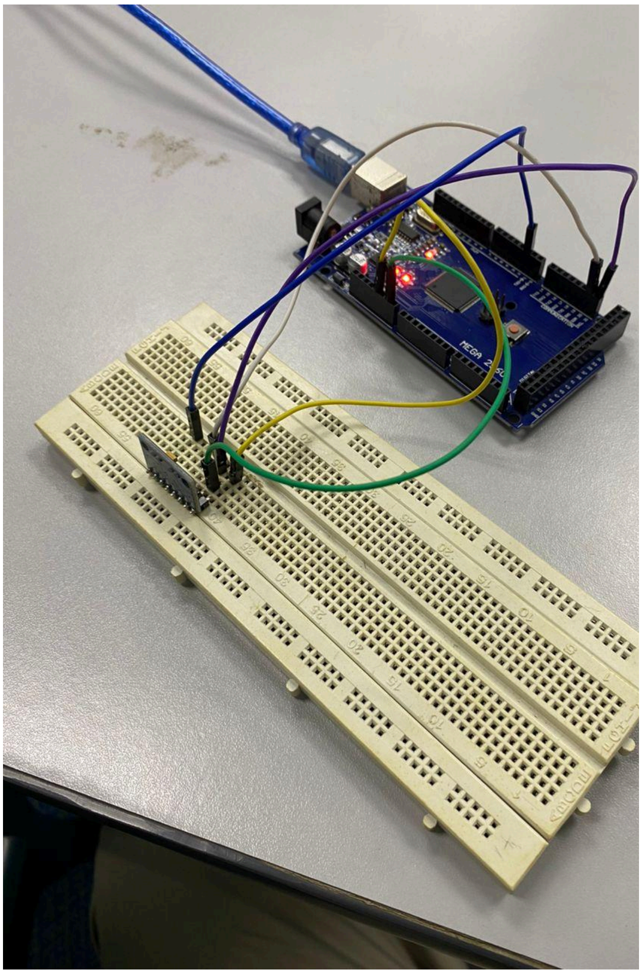

Figure 1: Hardware setup

METHODOLOGY

1. Setup the Arduino Mega 2560 with the MPU6050
2. Code implementation
3. Testing
4. Servo changes when pitch changes
5. Code snippet

Arduino Code

```
#include <Wire.h>
#include <MPU6050.h>
#include <Servo.h>
Servo servo;
MPU6050 mpu;

const int threshold = 122;
int previousGesture = -1;

void setup() {
Serial.begin(9600);
servo.attach(9);
pinMode(9, OUTPUT);
Wire.begin();
mpu.initialize();
}
void loop() {

int gesture = detectGesture();

  if (gesture != previousGesture) {

    if (gesture == 1) {
      Serial.println("Detected Gesture: Gesture 1");
      servo.write(45);
    // Perform an action for Gesture 1
    } else if (gesture == 2) {
      Serial.println("Detected Gesture: Gesture 2");
      servo.write(115);
    // Perform an action for Gesture 2
    }
  }
// Add more gesture cases as needed
previousGesture = gesture;
```

```
}

int detectGesture() {
    int16_t ax, ay, az;
    int16_t gx, gy, gz;
    int pitch;
    mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
    pitch = map(ay, -17000, 17000, 0, 255);
    // Perform gesture recognition here based on sensor data
    // Define conditions to recognize specific gestures
    if (pitch >= threshold) {
    return 1; // Gesture 1
    } else if (pitch < threshold) {
    return 2; // Gesture 2
    }
    // Add more gesture conditions as needed
    return 0; // No gesture detected
}
```

Python Code

```
import serial
ser = serial.Serial('COM7', 9600)


while True:
  try:
    data = ser.readline().decode('utf-8').strip()
    print(f"{data}")
    u[]
    if data == "Detected Gesture: Gesture 1":
      # Perform an action for Gesture 1
        print("'Servo Left'")
    elif data == "Detected Gesture: Gesture 2":
      # Perform an action for Gesture 2
        print("'Servo Right'")
      # Add more gesture actions as needed
  except Exception as e:
    print(f"Error reading data: {e}")
```

DATA COLLECTION

Pitch	Servo Angle (°)
0	45
122	45
123	115
255	115

Table 1

DATA ANALYSIS

The data in table 1 above indicates a relationship between pitch values and corresponding servo angles that suggest a control mechanism responsive to orientation changes. When the pitch is in the range of 0 and 122, the servo angle remains at 45°, but as the pitch increases slightly to 123, the servo angle jumps to 115°. This also applies through pitch of 123 to 255. This pattern suggests a threshold-based control system where the pitch is below 123 will make the servo angle set to 45° and when the pitch reaches 123 or higher, the servo angle shifts to 115°. This behaviour could serve to some applications that require automatic positional adjustments based on tilt like a stabilizing platform, adjusting a component's position based on orientation, or triggering responses at specific angles. The simplicity of the two-tiered response also implies that the system may be calibrated for straightforward and efficient pitch-based activation or orientation control.

RESULT

The experiment was successful in demonstrating basic hand gesture recognition using data from the MPU6050 sensor connected to an Arduino. By monitoring accelerometer and gyroscope data, specific hand movements were detected and categorised into predefined gestures. The sensor data, transmitted to the computer via serial communication, was processed in real time using a Python script, which identified gestures based on threshold conditions set in the Arduino code. Recognized gestures were displayed in the console, and the movements were visualised on an x-y coordinate system, showing clear distinctions between different gestures. This result confirms the MPU6050's capability to capture and differentiate hand gestures, providing a functional foundation for gesture-based interaction systems.

DISCUSSION

The experiment successfully utilised data from the MPU6050 sensor coupled to an Arduino to demonstrate basic hand gesture recognition. The MPU6050 combines accelerometer and gyroscope measurements, making it an ideal sensor for detecting motion and orientation changes. Connecting the MPU6050 to an Arduino via I2C allowed us to capture and send sensor data to a computer for further processing. By establishing serial communication, we use Python on the computer to read and display this data in real-time. Basic gesture recognition was implemented in the Arduino code, where conditions based on threshold values in the accelerometer data identified predefined hand movements. Gesture 1 was detected when the accelerometer's X-axis data exceeded a threshold and the Y-axis was below it while Gesture 2 was identified with opposite conditions on the X and Y axes.

While the experiment was successful in detecting simple gestures, there are several challenges and limitations like setting appropriate thresholds is challenging, as different users may perform gestures with varying intensities, and slight changes in the environment or sensor position may alter readings. This can lead to inconsistencies in gesture detection. Despite limitations, the setup demonstrated the practicality of using accelerometer and gyroscope data for simple gesture-based applications, laying the groundwork for more sophisticated gesture recognition systems.

CONCLUSION

In conclusion, this experiment successfully demonstrated the process of interfacing an MPU6050 sensor with an Arduino to collect accelerometer and gyroscope data and transmit it to a computer for gesture recognition. By implementing simple threshold-based logic, we were able to detect basic hand gestures, illustrating the potential for using IMU data in motion-tracking and gesture-based applications.

RECOMMENDATIONS

We recommend implementing data filtering techniques.The accelerometer and gyroscope data can be made more stable by using filters such as a low-pass or complementary filter, which can reduce noise. This can prevent false detections due to small, unintended movements.

REFERENCES

- https://howtomechatronics.com/tutorials/arduino/arduino-and-mpu6050-accelerometer-and-gyroscope-tutorial/
- https://lastminuteengineers.com/mpu6050-accel-gyro-arduino-tutorial/

ACKNOWLEDGEMENTS

Special thanks to ZULKIFLI BIN ZAINAL ABIDIN & WAHJU SEDIONO for their guidance and support during this experiment.

Certificate of Originality and Authenticity

This is to certify that we are **responsible** for the work submitted in this report, that **the original work** is our own except as specified in the references and acknowledgement, and that the original work contained herein have not been untaken or done by unspecified sources or persons. We hereby certify that this report has **not been done by only one individual** and **all of us have contributed to the report**. The length of contribution to the reports by each individual is noted within this certificate. We also hereby certify that we have **read** and **understand** the content of the total report and no further improvement on the reports is needed from any of the individual's contributors to the report. We therefore, agreed unanimously that this report shall be submitted for **marking** and this **final printed report** has been **verified by us.**

Signature: darwish Read [/]
Name: AHMAD DARWISH BIN AHMAD TERMIZI Understand [/]
Matric Number: 2212089 Agree [/]
Contribution: Introduction, Materials and Equipments, Experimental Setup

Signature: syabab Read [/]
Name: SYABAB ALHAQQI BIN JAAFAR Understand [/]
Matric Number: 2211117 Agree [/]
Contribution: Data Collection, Data Analysis

Signature: naim Read [/]
Name: WAN NAIMULLAH BIN MOHD AZRUDDIN Understand [/]
Matric Number: 2214837 Agree [/]
Contribution: Discussion, Conclusion, Recommendation

Signature: zaim Read [/]
Name: ZAIMUDIN ZAKWAN BIN NORZAMRI Understand [/]
Matric Number: 2217513 Agree [/]
Contribution: Methodology, Result

Signature: zamir Read [/]
Name: ZAMIR MUTTAQIN BIN MUHAMMAD BALYAN Understand [/]
Matric Number: 2212985 Agree [/]
Contribution: Abstract, References