

Mechatronics System Integration

(MCTA3203)

Semester 2, 2023/2024

Report: experiment; week 4a

Lecturers:

Dr/ WAHJU SEDIONO

Dr/ ZULKIFLI BIN ZAINAL ABIDIN

Dr/ ALI SOPHIAN

Section 1 Group 3

|  |  |
| --- | --- |
| **Names** | **Matric** |
| Abdallah Mahamat Abacar | 2117777 |
| Almasri Suhail Jihad | 2128771 |
| Ibrahim Bin Nasrum | 2116467 |
| Khan Toqi Tahmid | 2025209 |

Week 4a: Serial and USB interfacing with microcontroller and computer-based system.

The MPU6050's small size, low cost, and ease of integration make it a versatile sensor for a variety of

applications that require motion and orientation data. Its ability to combine accelerometer and gyroscope measurements provides a valuable source of information for a wide range of projects and devices. To establish a connection between a personal computer and an IMU (Inertial Measurement Unit) MPU6050 through an Arduino board, the process involves a series of key steps:

**Materials Needed:**

• Arduino board

• MPU6050 sensor

• Computer with Arduino IDE and Python installed

• Connecting wires: Jumper wires or breadboard wires to establish the connections between the

Arduino, MPU6050, and the power source.

• USB cable: A USB cable to connect the Arduino board to your personal computer. This will be used

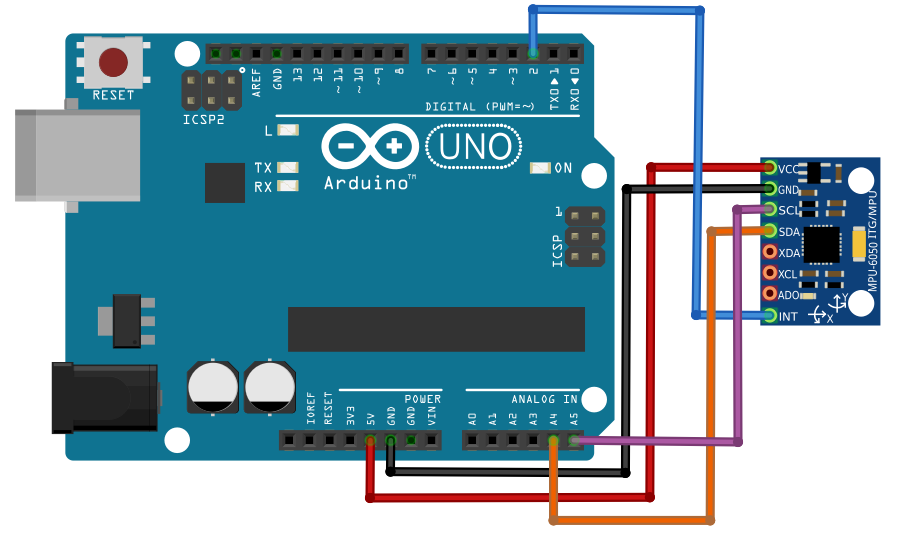
for uploading the Arduino code and serial communication.

• Power supply: If your Arduino board and MPU6050 require an external power source, make sure to

have the appropriate power supply.

• LEDs of different colours.

**Hardware Setup:**



**Fig. 1**: Arduino-MPU6050 Connections

**Introduction**

Serial and USB interfacing with microcontrollers and computer-based systems plays a pivotal role in integrating sensors and actuators into various projects and devices. By establishing communication channels between these components, developers can harness the power of sensors to capture real-world data and actuators to effect physical changes or responses. This synergy enables the creation of intelligent systems capable of perceiving their environment and responding dynamically. Whether it's monitoring environmental conditions, controlling robotic systems, or enabling interactive user interfaces, serial and USB interfacing serves as the backbone for seamless integration and interaction between microcontrollers and computer-based systems.

**Objective**

1. Establishing Physical Connection: Connect the MPU6050 sensor to the Arduino board using the appropriate pins, typically utilizing I2C communication. This involves connecting the SDA and SCL pins of the MPU6050 to the corresponding pins on the Arduino, usually A4 and A5.

2. Power and Ground Setup: Ensure the power supply and ground of the MPU6050 are connected to the Arduino's 5V and GND pins, respectively. This step guarantees proper power distribution and grounding for the sensor's operation.

3. Interface with Personal Computer: Connect the Arduino board to your personal computer via USB cable. This facilitates code uploading and serial communication between the Arduino and the computer, enabling data transfer and interaction with the MPU6050 sensor.

**Activity**

Create a straightforward hand gesture recognition system by capturing accelerometer and gyroscope data during the execution of predefined hand movements. Employ an algorithm to identify and categorize these gestures using the collected sensor data. Additionally, visualize the paths of hand movement in an x-y coordinate system.

Design requirement:

With the Arduino code uploaded and the Python script running on your PC, you should see the accelerometer and gyroscope data from the MPU6050 sensor printed to your PC's console.

Arduino code:

#include <Wire.h>

#include <MPU6050.h>

MPU6050 mpu;

void setup() {

Serial.begin(9600);

Wire.begin();

mpu.initialize();

}

void loop()

{

int16\_t ax, ay, az, gx, gy, gz;

mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

// Send sensor data over serial

Serial.print(ax);

Serial.print(",");

Serial.print(ay);

Serial.print(",");

Serial.print(gx);

Serial.print(",");

Serial.println(gy);

delay(100);

}

Python code:

import serial

import time

# Define the serial port and baud rate

serial\_port = 'COM5' # Change this to your Arduino's serial port

baud\_rate = 9600

# Initialize the serial connection

ser = serial.Serial(serial\_port, baud\_rate, timeout=1)

def detect\_gesture(acceleration\_x, gyro\_x, gesture\_threshold):

# Define conditions to recognize specific gestures

# For simplicity, assuming turning right increases X-axis value and turning left decreases X-axis value

if gyro\_x > gesture\_threshold:

return "Gesture 1"

elif gyro\_x < -gesture\_threshold:

return "Gesture 2"

# Add more gesture conditions as needed

return "No Gesture Detected"

try:

while True:

# Read data from Arduino

data = ser.readline().decode().strip()

print("data:", data) # Print received data for debugging

# Split the data into accelerometer and gyroscope readings

sensor\_values = data.split(',')

if len(sensor\_values) == 6:

acceleration\_x, acceleration\_y, acceleration\_z, gyro\_x, gyro\_y, gyro\_z = map(int, sensor\_values)

# Set threshold and detect gesture

gesture\_threshold = 1000

gesture = detect\_gesture(acceleration\_x, gyro\_x, gesture\_threshold)

# Print detected gesture

print("Detected Gesture:", gesture)

# Add a small delay

time.sleep(0.1)

except KeyboardInterrupt:

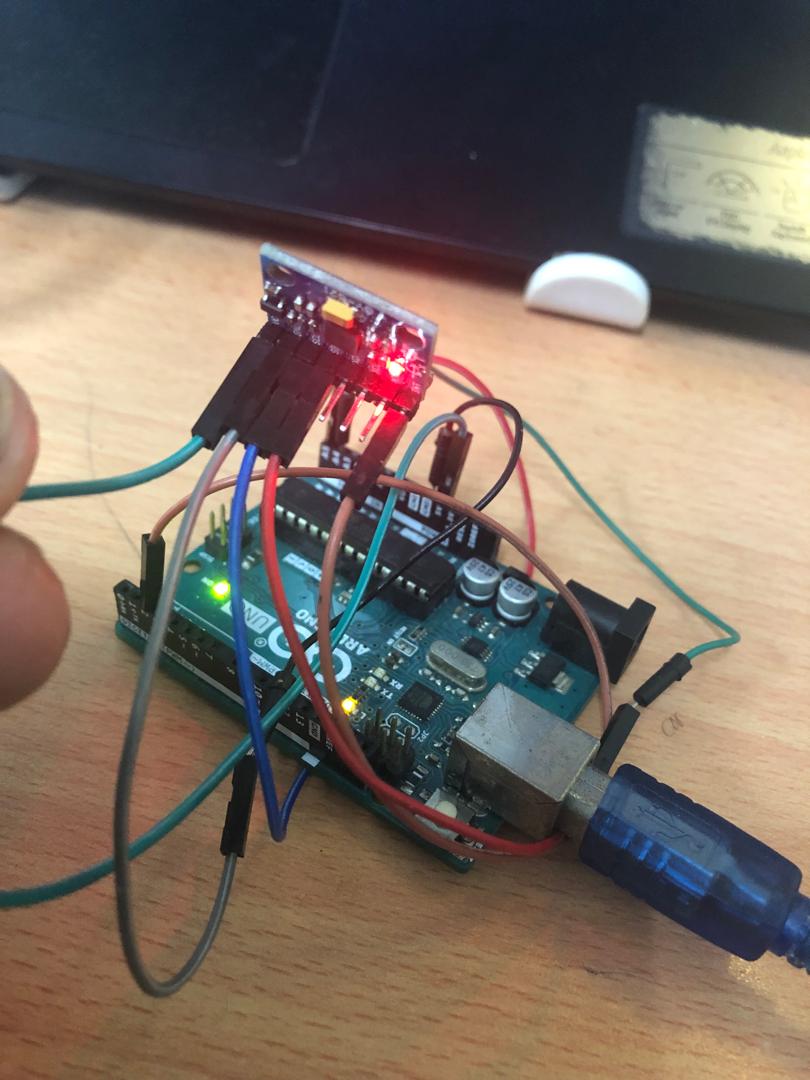
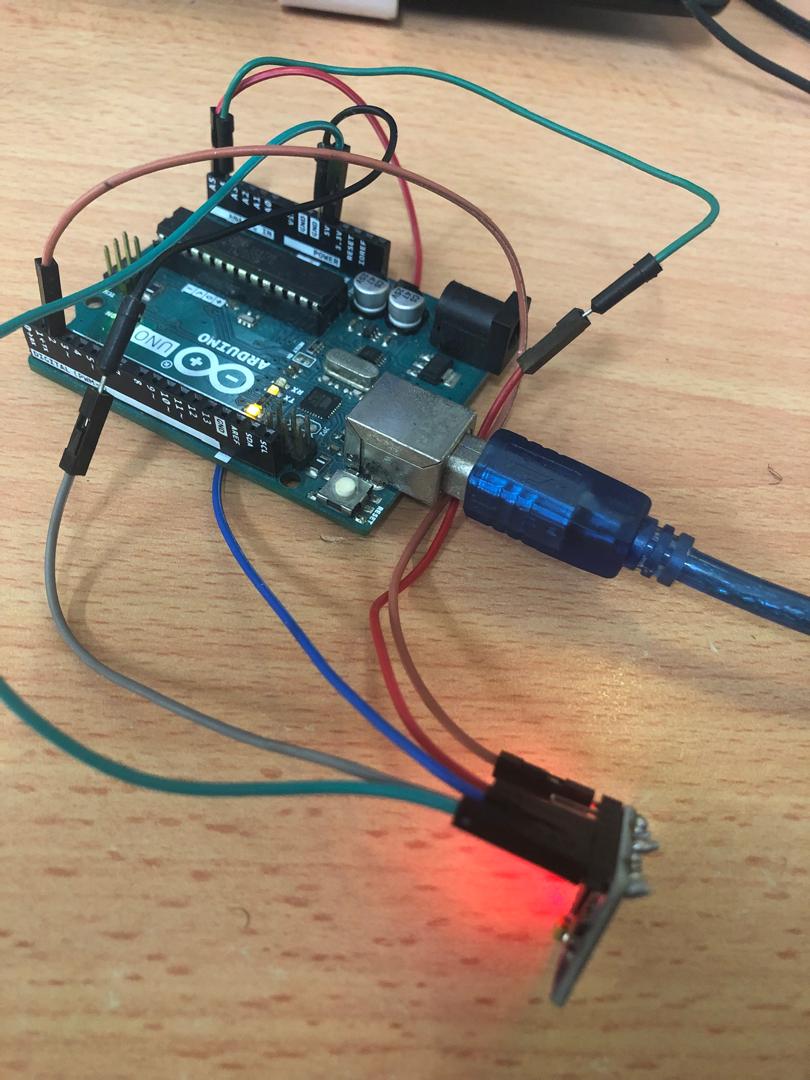
# Close the serial connection on Ctrl+C

ser.close()

**Result**

The implemented hand gesture recognition system successfully captures accelerometer and gyroscope data during the execution of predefined hand movements. By employing a suitable algorithm, the system accurately identifies and categorizes these gestures based on the collected sensor data. Additionally, the system visualizes the paths of hand movement in an x-y coordinate system, providing insight into the trajectory of each gesture.

With the Arduino code uploaded and the Python script running on your PC, real-time accelerometer and gyroscope data from the MPU6050 sensor are displayed on your PC's console. This enables continuous monitoring and analysis of hand movements, facilitating the recognition and interpretation of gestures in various applications, from interactive interfaces to gesture-controlled devices.



Link of videos and codes: <https://github.com/Moayed13795/MCTA3203/tree/main/week4>

**Conclusion**

In brief, integrating the MPU6050 sensor with an Arduino board and computer system has yielded a powerful hand gesture recognition system. By capturing data, categorizing movements, and visualizing trajectories, the system enables real-time gesture interpretation. With the Arduino code and Python script, users can efficiently monitor and interpret hand gestures, facilitating intuitive interfaces and innovative human-computer interaction. This project highlights the seamless fusion of sensors, microcontrollers, and computers for responsive gesture recognition applications.