INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA

Garden of Knowledge and Virtue

SEM 1 25/26

SECTION 1

COURSE CODE: MCTA 3202

LECTURER NAME:
ZULKIFLI BIN ZAINAL ABIDIN

WEEK 4: SERIAL INTERFACING WITH MICROCONTROLLER: SENSORS AND ACTUATORS (ver 3.0)

PREPARED BY:
GROUP 1

NAME	MATRIC NUMBER
AMIR MUIZZUDDIN BIN NORAIDIL RAZMAN	2129579
MUHAMMAD HAIKAL HANIF BIN ABDUL RAZAK	2213297
AHMAD ILHAM BIN ABDUL AZIZ	2112109

TASK 1

INTRODUCTION

The objective of this project is to connect an Arduino microcontroller to an MPU6050 motion sensor in order to track hand motion in real time and display it using Python. Both linear acceleration and rotational velocity can be measured with the MPU6050 sensor since it includes a 3-axis accelerometer and a 3-axis gyroscope. Real-time accelerometer data is sent from the Arduino to the computer via serial communication. The matplotlib library in Python is then used to process and plot the data dynamically.

Through this assignment, students learn about motion visualisation, real-time sensor data processing, and serial interface between an Arduino and a computer. Additionally, it develops a basic understanding of motion detection algorithms, namely the ability to identify basic motion patterns like circular movements, which form the foundation for later experiments with more complex control systems.

EXPERIMENTAL SETUP

The MPU6050 sensor is connected to the Arduino Uno via I²C communication as part of the experimental setup. The analogue pins A4 and A5 of the Arduino are linked to the SDA and SCL pins of the sensor, respectively. The Arduino's 5V and GND pins are linked to the VCC and GND pins.

The Arduino and computer are connected via a USB cable for serial communication and power supply. A software on the Arduino continuously reads data from the accelerometer and gyroscope and transmits it to the serial monitor. This data is read via the serial port by a Python application, which then uses matplotlib to plot the X-Y movements in real time.

Motion pattern identification and visualisation are made possible by this configuration. For instance, the plotted data will follow a circle route on the graph when the user moves the sensor in a circular motion, indicating that motion tracking and serial data connection between the Arduino and the PC are successful.

MATERIALS

- Arduino board
- MPU6050 sensor
- Jumper wires
- USB Cable

- Computer with Arduino IDE and Python installed

DISCUSSION

In Task 1, the MPU6050 sensor was successfully interfaced with the Arduino to capture real-time hand-motion data, which was displayed dynamically in Python using *matplotlib*. The plotted motion accurately reflected changes in acceleration, confirming that serial communication worked properly. However, noise and data fluctuation were observed due to sensor sensitivity and hand instability. Applying smoothing or filtering techniques could improve accuracy and motion detection. Circular motion was detected by analyzing X-Y coordinate patterns, although inconsistent hand speed sometimes affected recognition. Overall, the task demonstrated effective real-time motion tracking suitable for integration with other systems.

CONCLUSION

In Task 1, the MPU6050 sensor was effectively used to capture and visualize real-time motion data through Arduino–Python serial communication, successfully demonstrating basic motion tracking and circular gesture detection. Although some noise and delay were present, the system proved reliable for recognizing motion patterns and provided a solid foundation for further integration.

TASK 2

INTRODUCTION

The second task focuses on integrating multiple components—RFID reader (MFRC522), MPU6050 motion sensor, servo motor, and indicator LEDs—into a single smart access control system. The system uses a two-step verification process: (1) user identification through RFID authentication and (2) motion validation via a specific gesture detected using the MPU6050 sensor.

When a valid RFID tag is detected, the system prompts the user to perform a circular hand motion. If the motion pattern is correctly identified, the system grants access by activating the servo motor to unlock and lighting up the green LED. Otherwise, access is denied, and the red LED is turned on.

This experiment demonstrates the integration of sensors and actuators using serial communication between Arduino and Python, combining both hardware control and real-time data analysis to simulate a secure access mechanism.

EXPERIMENTAL SETUP

The hardware components for Task 2 include an Arduino Uno, MFRC522 RFID reader, MPU6050 sensor, servo motor, two LEDs (red and green), and a breadboard with jumper wires.

- The RFID reader is connected to the Arduino using SPI communication: SS to D10, RST to D8, and other SPI pins as required (MOSI, MISO, SCK).

- The servo motor's signal pin is connected to D9, while its power lines are connected to 5V and GND.

- The green LED is connected to D4 and the red LED to D3, each in series with a resistor.

- The MPU6050 sensor remains connected to A4 (SDA) and A5 (SCL) from Task 1.

- All components share a common 5V and GND line.

The system operates as follows: the Arduino reads the RFID tag through the MFRC522 module. The UID is then sent via serial communication to the Python program, which checks if the UID is authorized. If authorized, the user is prompted to perform a circular motion. Python analyzes the MPU6050 data stream, determines if the correct motion pattern was performed, and sends a command ('A' or 'D') back to the Arduino to control the servo and LEDs accordingly.

This integrated setup showcases the synchronization between hardware (Arduino) and software (Python) for secure, motion-based access control.

MATERIALS

- Arduino board
- RFID reader (MFRC522) + RFID tags/cards
- Servo motor
- Red & Green LEDs + resistors
- MPU6050 sensor (from Task 1)
- Jumper wires and breadboard

DISCUSSION

In Task 2, the RFID reader, MPU6050, and servo motor were integrated into a smart access system requiring both valid RFID authentication and a correct circular-motion gesture. The Arduino controlled the servo and LEDs based on Python's verification result through serial commands. The system successfully distinguished between authorized and unauthorized users, granting access only when both checks were met. Minor delays occurred due to serial communication and processing time, but overall performance was stable. This task proved that combining RFID and motion detection enhances security and demonstrates successful sensor-actuator integration in a mechatronic system.

CONCLUSION

In Task 2, the RFID module, MPU6050 sensor, and servo motor were successfully combined to create a dual-authentication smart access system. The setup accurately verified user identity and motion before granting access, showing how multiple sensors and actuators can be integrated to form an intelligent mechatronic control system. Overall, both tasks achieved their objectives and highlighted the importance of serial communication in linking sensing and actuation for smart automation applications.