Day 3 – Hardware Simulation & Cloud Integration

Objective

The primary objective of Day 3 was to transition from a virtual simulation environment to a hardware-level prototype, involving sensor interfacing with the Raspberry Pi Pico W, establishing stable Wi-Fi communication, and integrating with the ThingSpeak IoT cloud platform to stream and log real-time sensor data. This phase also aimed to validate the system's responsiveness in detecting obstacles and human presence, and its ability to share this information remotely via the cloud.

Detailed Activities Performed

🔌 1. Hardware Setup & Sensor Interfacing

- **Components Used:**
  - **Raspberry Pi Pico W** – Wi-Fi-enabled microcontroller for control and communication.
  - **Ultrasonic Sensor (HC-SR04)** – Measures distance to detect obstacles.
  - **IR Sensor** – Detects surface variations or line tracking (e.g., white/black surface).
  - **PIR Sensor** – Detects motion or human presence in the AGV's path.
  - Breadboard, jumper wires, USB cable (for power and programming).
- **Wiring & Connections:**
  - VCC and GND of each sensor were connected to Pico W's 3.3V and GND.
  - Digital and analog pins were assigned to GPIO pins on the Pico W.
  - Care was taken to ensure pin compatibility and avoid overvoltage.
- **MicroPython Code Execution:**
  - A script was written and uploaded using **Thonny IDE**.
  - Each sensor was initialized with proper delay and I/O configuration.
  - Code modules:
    - ultrasonic.py for pulse and echo timing.
    - pir_ir.py for digital read of motion and surface detection.
    - wifi_thingspeak.py for handling HTTP requests.

---

2. Wi-Fi Configuration and Cloud Communication

- **Wi-Fi Setup:**
    - SSID and password were coded into the MicroPython script.
    - Connection to local Wi-Fi was established using the network module.
    - A status check and IP address print confirmed successful connectivity.
- **ThingSpeak Integration:**
    - A **ThingSpeak channel** was created with 3 fields:
        - **Field 1:** Ultrasonic Sensor Distance
        - **Field 2:** IR Sensor Output (0/1)
        - **Field 3:** PIR Sensor Output (0/1)
    - Data was sent as an HTTP GET request using the urequests library.
    - Channel API Key and URL parameters were securely embedded in the script.

---

3. Rate-Limit Handling and Data Stream Verification

- **Rate Limiting:**
    - To comply with ThingSpeak's **15-second minimum update interval**, a time.sleep(15) was added after each successful upload.
    - This ensured no data packets were rejected or lost due to over-frequency.
- **Dashboard Monitoring:**
    - Live plots were observed on the ThingSpeak web interface.
    - Field values changed dynamically based on sensor inputs, confirming successful data streaming.

---

4. Testing and Validation

- **Sensor Trigger Testing:**
    - An object was moved near the ultrasonic sensor → change in distance observed.
    - A hand was passed near the PIR sensor → motion detected and updated in cloud.
    - Surface was changed under the IR sensor → digital change recorded.
- **Real-Time Data Capture:**

    - Timestamped values were updated and could be visualized as time-series graphs.
    - Events such as "Obstacle Detected" or "Motion Detected" were confirmed.
- **CSV Export:**
    - ThingSpeak's export function was used to download the data log in CSV format.
    - This log can be used to build machine learning models later (e.g., failure detection, predictive control).

Outcomes & Deliverables

- Physical integration of Ultrasonic, IR, and PIR sensors with Raspberry Pi Pico W.
- Real-time sensor data successfully transmitted to ThingSpeak IoT cloud.
- Confirmation of cloud-based logging, graphing, and CSV export functionalities.
- 15-second delay ensured continuous and uninterrupted data flow to the server.
- System responded appropriately to environmental changes (motion, distance, surface type).