物聯網智造基地

IOT SERVICE HUB

Ideas Hatch

Institute for Information Industry

HUB PAA3905 OMT

簡易距離計算器

Contents

- PAA3905E1-Q 簡介
- HUB PAA3905 OMT 介紹
- HUB 5168+ 介紹
- 案例說明
- 成果展示

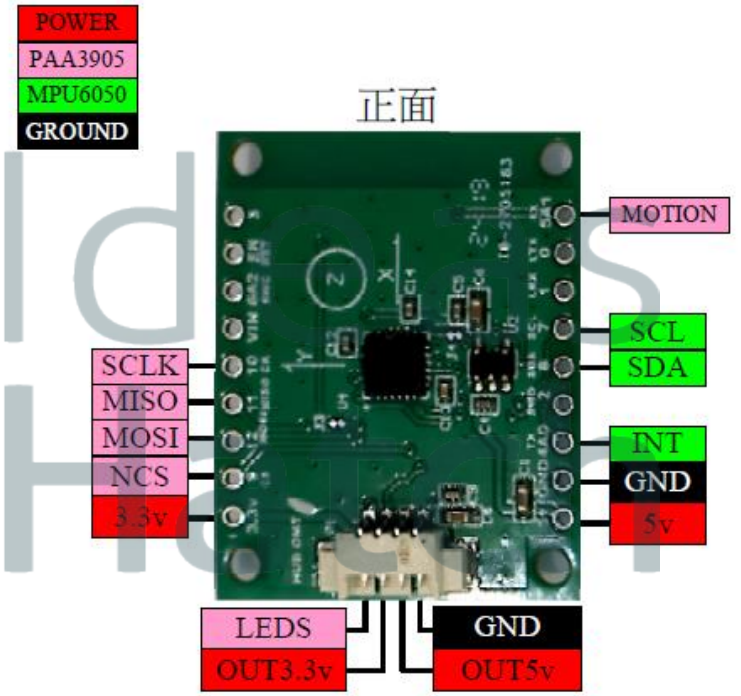
Ideas Hatch

PAA3905E1-Q 簡介

- PAA3905E1-Q 是一款為動態追蹤及穩定懸浮中的無人機打造的光學動態追蹤晶片。奠基於光學導航技術，此晶片不僅能測量方位的轉變及幅度的變化 (delta x 和 delta y)，更具有 80mm 至無限遠的廣泛操作範圍，是遠場 (far-field) 追蹤應用的理想選擇。
- 此晶片的主要優勢之一，是能在極低的光照條件下作用 (即 5 lux)，同時可根據環境的亮度，自動切換3種不同的模式。除此之外，使用數字信號處理系統 (DSPS / Digital Signal Processing System) 的PAA3905E1-Q 不需任何電信信號即可處理 XY 定位，因此能夠在任何環境提供定位服務，包括無GPS信號的地區。
  - 可支援 80mm 至無限遠的廣泛工作距離
  - 鏡頭不需調焦
  - 可自動偵測各種表面條件，例如：棋盤、條紋、光滑的表面和原地旋轉
  - 具內建振盪器
  - 可透過寄存器讀取幀數據
  - 低功耗
  - 支援多晶片同步操作
  - 可自動切換操作模式

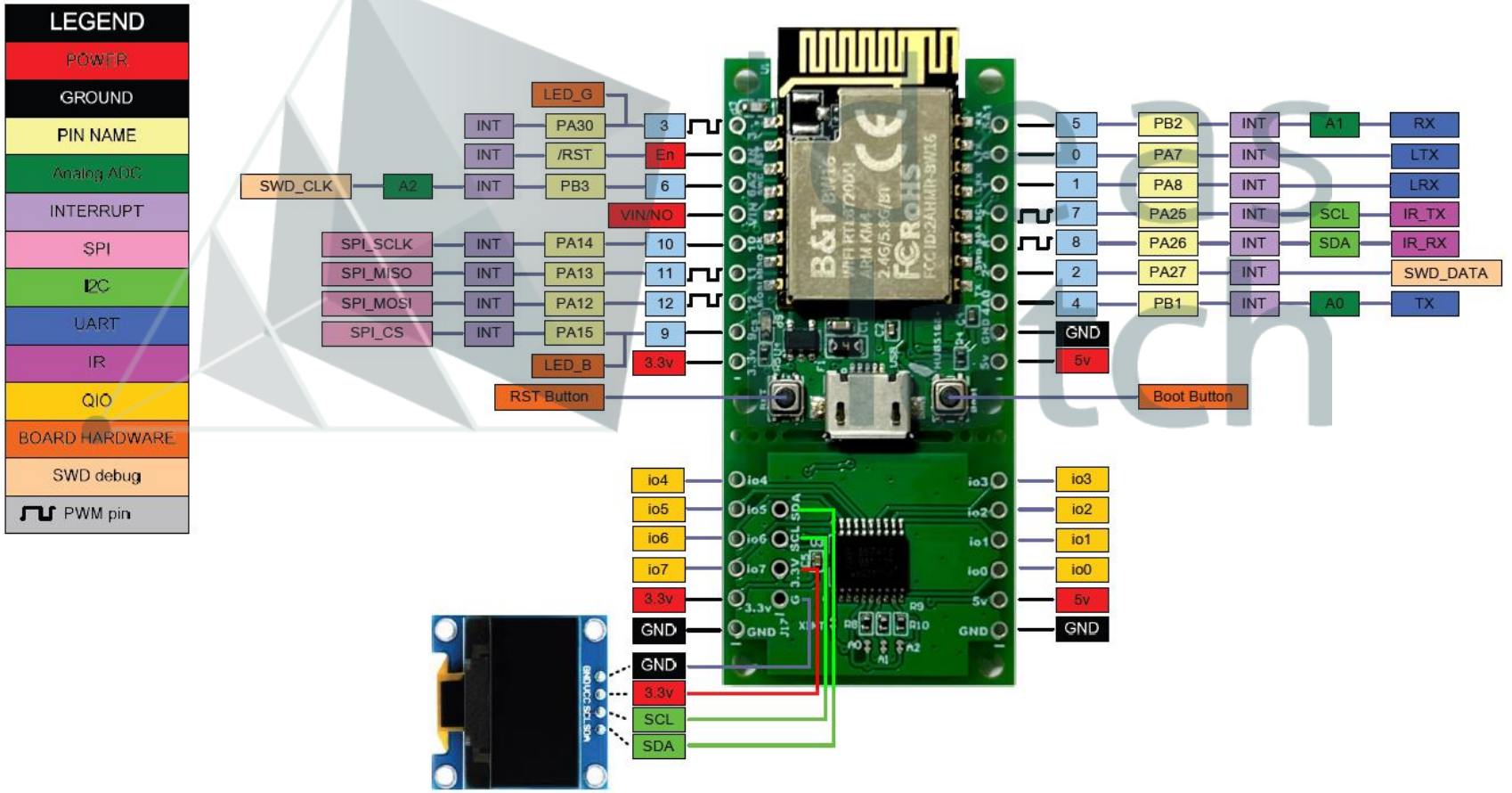
HUB PAA3905 OMT

HUB-PAA3905-OMT 是由 PAA3905 結合 G-Sensor(MPT-6050)來偵測在立體空間移動的狀況，利用G-Sensor 來調整PAA3905 對地的方向，由PAA3905 影像擷取來感知位置的移動，在使用方面設定好內部參數，就能透過SPI 介面得知位置移動的X-Y 方向的差異值(無水平旋轉時)，另G-sensor 是透過I2C 的介面來得知運動姿態。



HUB 5168+

- HUB5168+ 具有2.4G & 5G WIF 雙頻IOT 微控器，內部記憶體高達512Kbytes，3組ADC、1 個硬體SPI、2 組UART、1 組I2C 及5 個PWM 輸，運轉頻率高達200MHz，可以有效的執行各種IOT 的任務。



案例說明

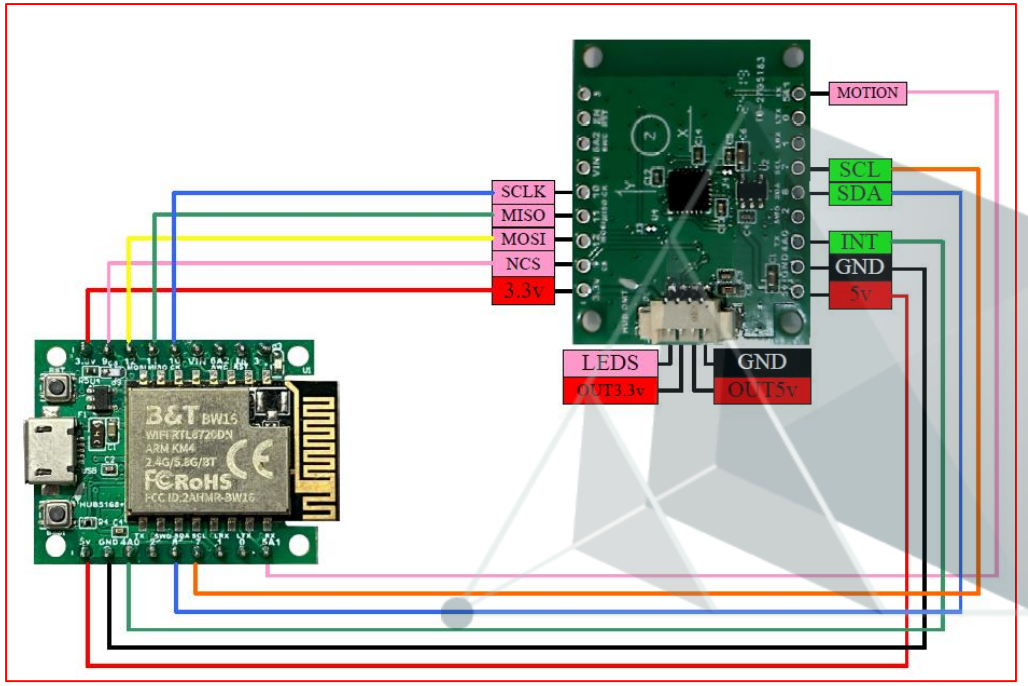
- 運用HUB OMT + HUB 5168+ 進行移動距離計算，並將結果顯示與OLED上。
- 移動一段時間/距離後靜止會持續顯示移動的距離；再次移動則開始新的cycle
- 使用CPI(Counts Per Inch) 方式進行移動距離估算，(dx,dy)為OMT輸出的移動量

$$CPI = \frac{12.198}{\text{Height (m)}} \text{x } (RESOLUTION + 1) \text{x} \frac{200}{8600}$$

$$\text{Distance (inches)} = \frac{\text{dx}}{\text{CPI}} \quad \text{and} \quad \frac{\text{dy}}{\text{CPI}}$$

HUB 5168+ 安裝方式參考： https://github.com/ideashatch/HUB-5168-Plus_examples
HUB-PAA3905-OMT Lib: https://github.com/ideashatch/HUB-PAA3905-OMT

接線方式



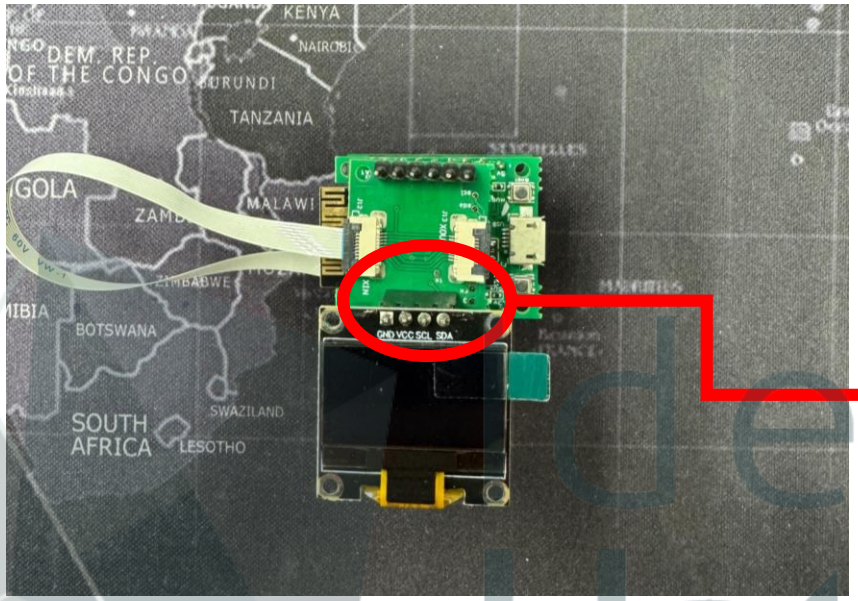
方法一

方法二

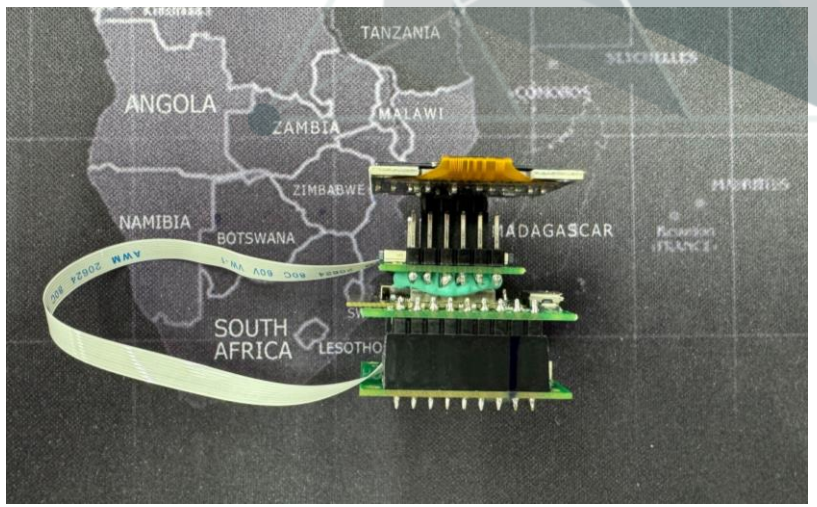
需注意腳位！

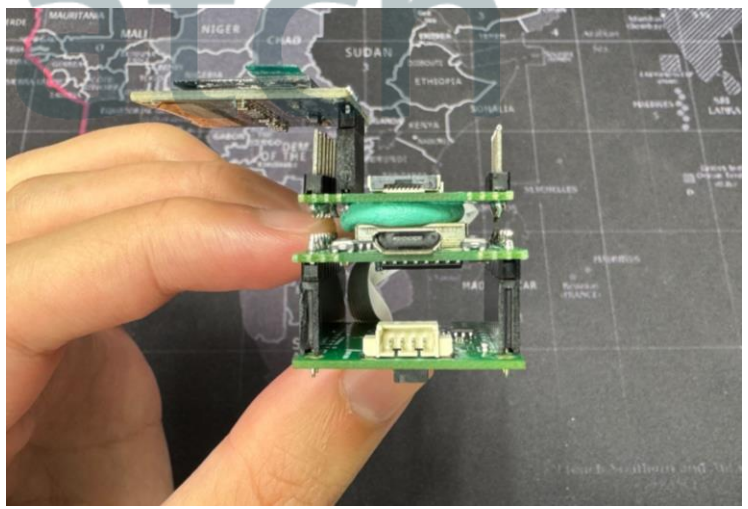
擴充小板有專用的OLED接口！

接線展示



擴充小板有專用的OLED接口！





程式重點

```
float currentHeight = 0.095;

if (millis() - timer > UPDATE_INTERVAL)
{
    SensorData sensor;
    MotionData motion;

    sensor.dx = sensor.dy = 0;
    sensor.squal = 0;
    sensor.shutter = 0;

    // Read PAA3905 data
    sensor.op_mode = paa.readMotion(&sensor.dx, &sensor.dy);
    if (sensor.op_mode) paa.getSqualShutter(&sensor.squal, &sensor.shutter);

    // Read MPU6050 angles and accelerations
    sensor.angleX = mpu.getAngleX();
    sensor.angleY = mpu.getAngleY();
    sensor.angleZ = mpu.getAngleZ();
    sensor.accelX = mpu.getAccX();
    sensor.accelY = mpu.getAccY();


    if (!hasMovementStopped(&sensor))
    {
```

本案例採用9公分左右的工作高度

取得dx,dy數據

取得陀螺儀數據

程式重點

```
if (hasMovementStopped(&sensor))  // Movement has stopped
{
    if (isMoving)  // Was it moving previously
    {
        // Display the total distance moved
        isMoving = false;  // Reset the movement flag
        display.clearDisplay();

        // Display the total distance in meters and cm
        display.setTextSize(1);
        display.setTextColor(SSD1306_WHITE);
        display.setCursor(0, 0);
        display.println("Total Distance Moved:");
        display.setCursor(0, 10);
        display.print(total_distance_meters);
        display.println(" meters");
        display.setCursor(0, 20);
        display.print(total_distance_meters * 100);  // Convert to cm
        display.println(" cm");
        display.display();  // Update the OLED display
    }
}
else  // Movement is ongoing
{
    if (!isMoving)  // Just started moving
    {
        // Reset total distance accumulation
        Serial.println("Resetting total_distance to 0 for new movement...");
        total_distance = 0;
        total_distance_meters = 0;
        isMoving = true;  // Mark movement as active
    }

    // Update resolution based on current height and movement
    updateResolutionBasedOnHeight(currentHeight, &sensor);
}
```

hasMovementStopped()用以判斷是否sensor還有在移動，用以持續顯示最後結果

開始新動作時，重設數值

計算及累加移動的距離

程式重點

```
// Function to detect when movement stops
bool hasMovementStopped(SensorData *sensor)
{
    // Define a threshold to detect if movement has stopped
    const int movement_threshold = 3;

    // Check if both dx and dy values are below the threshold
    if (abs(sensor->dx) < movement_threshold && abs(sensor->dy) < movement_threshold)
    {
        // If no significant movement for 500 ms, consider movement stopped
        if (millis() - lastMovementTime > 300)  // 500 ms delay to confirm stop
        {
            return true;  // Movement has stopped
        }
    }
    else
    {
        // If movement is detected, update the last movement time
        lastMovementTime = millis();
    }

    return false;  // Movement is ongoing
}
```

OMT在靜止的狀況，依舊有可能有小數值輸出，因此設一個threshold 過濾非明顯動態

程式重點

```ino
void updateResolutionBasedOnHeight(float height, SensorData *sensor) {
    int resolutionRegValue = 42;  // Default value (0x2A)
    float cpi = calculateCPI(height, resolutionRegValue);

    Serial.print("Calculated CPI: ");
    Serial.println(cpi);

    // Calculate and accumulate distance
    calculateAndAccumulateDistance(sensor, cpi);
}
```

```ino
// Function to calculate CPI based on height and resolution register value
float calculateCPI(float height, int resolutionRegValue) {
    return (CPI_BASE / height) * (resolutionRegValue + 1) * CPI_MULTIPLIER;
}
```

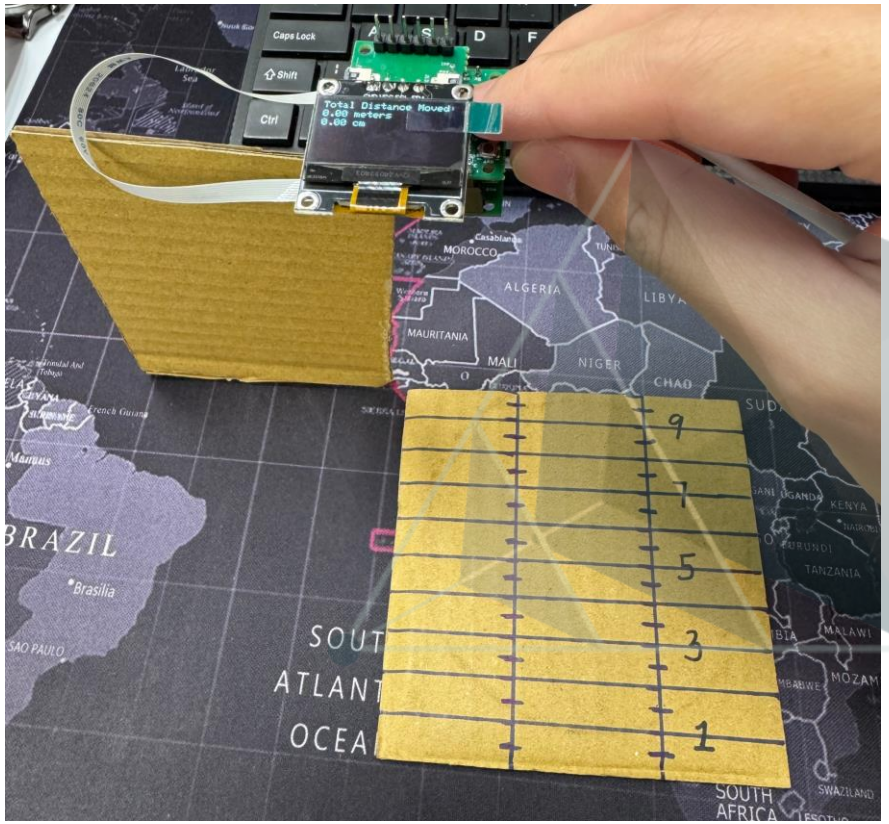
計算CPI, 預設resolution = 42

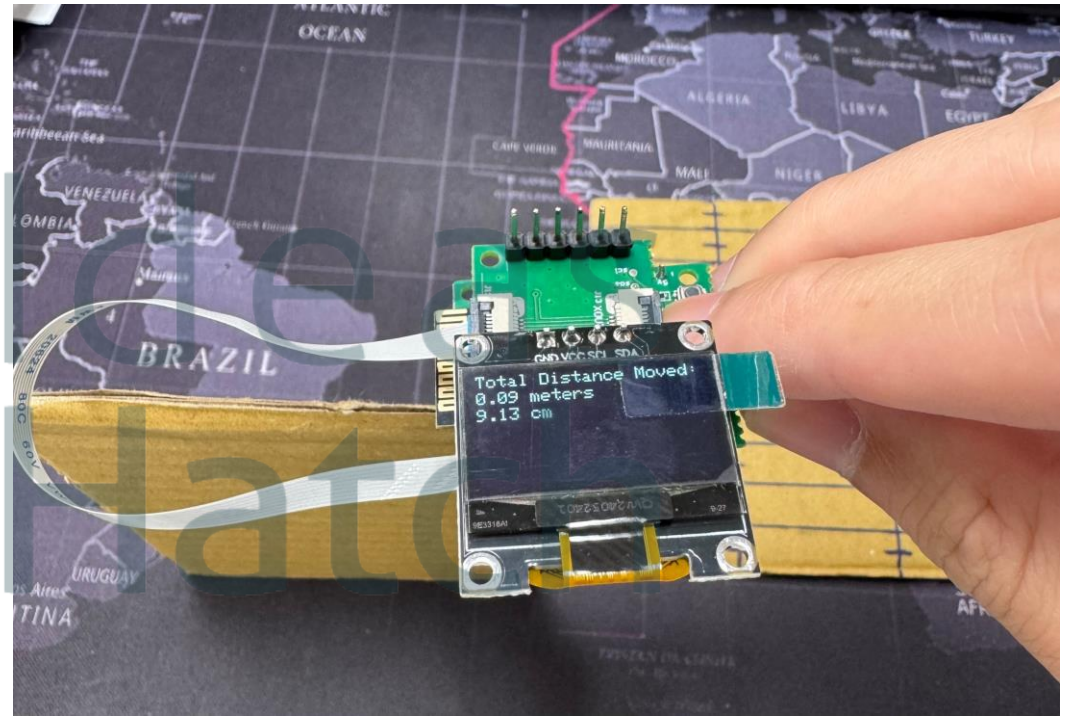
```ino
// Function to calculate real-world distance moved and accumulate total distance
void calculateAndAccumulateDistance(SensorData *sensor ,float cpi)
{
    const int MOVEMENT_THRESHOLD = 3;

    if(abs(sensor->dx) > MOVEMENT_THRESHOLD || abs(sensor->dy) > MOVEMENT_THRESHOLD)
    {
        float distanceX_inches = sensor->dx / cpi;
        float distanceY_inches = sensor->dy / cpi;

        float distanceX_meters = distanceX_inches * INCHES_TO_METERS;
        float distanceY_meters = distanceY_inches * INCHES_TO_METERS;

        float distance_meters = sqrt(distanceX_meters * distanceX_meters + distanceY_meters * distanceY_meters);

        total_distance_meters += distance_meters;

        // Optionally convert to millimeters for display purposes
        float total_distance_cm = total_distance_meters * 100.0;

        // Print the accumulated distance
        Serial.print("Total Distance (meters): ");
        Serial.println(total_distance_meters);
        Serial.print("Total Distance (mm): ");
        Serial.println(total_distance_cm);

        // Display total distance on the OLED
        display.clearDisplay();
        display.setTextSize(1);
        display.setTextColor(SSD1306_WHITE);
        display.setCursor(0, 0);
        display.println("Total Distance Moved:");
        display.setCursor(0, 10);
        display.print(total_distance_meters);
        display.println(" meters");
        display.setCursor(0, 20);
        display.print(total_distance_cm);
        display.println(" cm");
        display.display();
    }

}
```

為了計算上方便，只採用X-Y兩方向，總距離使用畢氏定理計算

如需要更高精度，可再結合陀螺儀數據做sensor fusion矯正

Demo

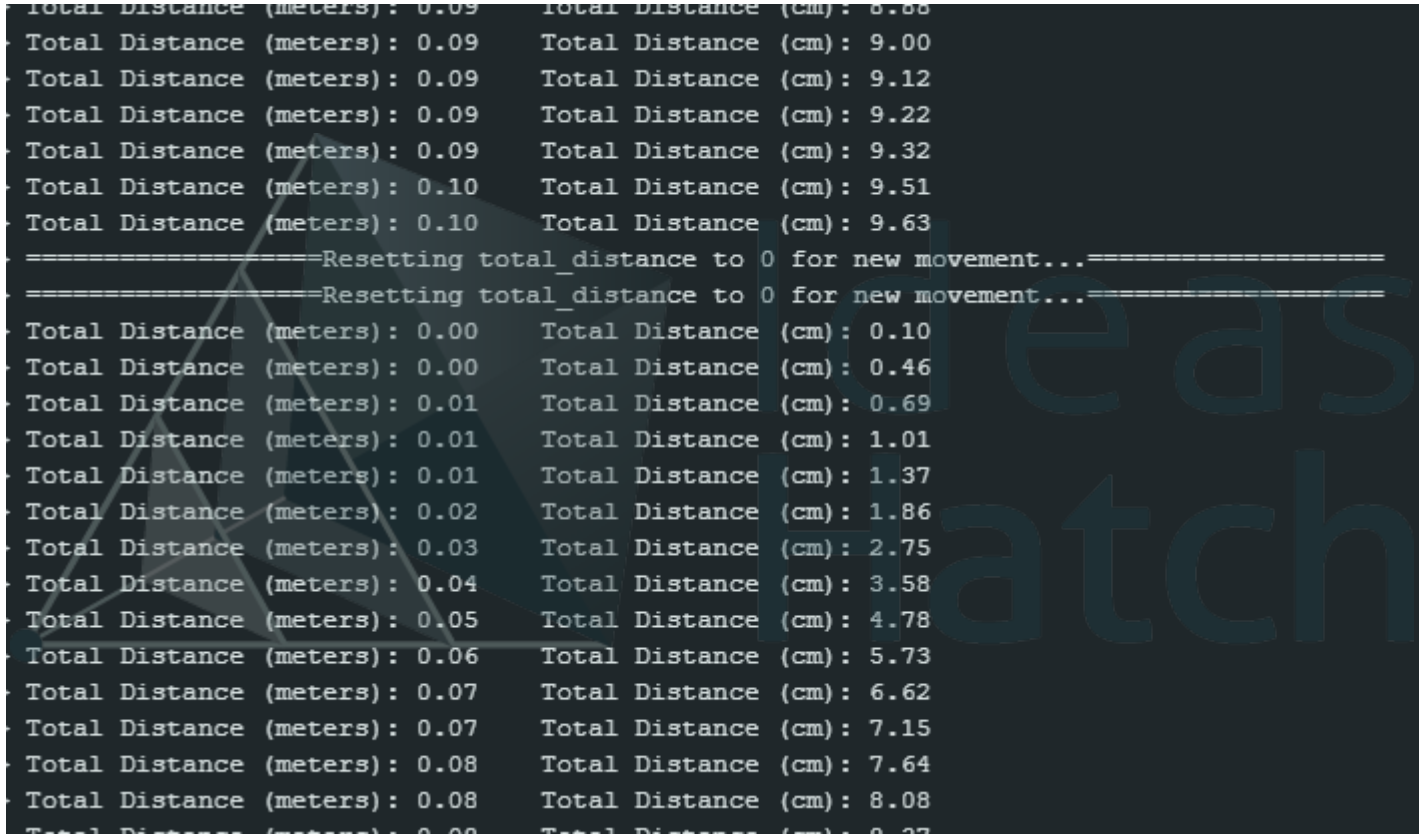




以高度為9cm左右進行測試，
誤差範圍落在5%～15%左右。

主要影響因素：FOV內的參照物變化、移動速度、環境光強度

Demo



Thank you

物聯網智造基地
IOT SERVICE HUB