



雲端運算與邊緣運算應用-期末報告

# 整合邊緣運算與 AWS 雲端之智慧 場域人流熱點分析系統

第五組

B11223211 余光正

B11223217 劉峻佑

B11223228 林元翔



# Overview

- ▶ 題目發想與動機 01
- ▶ 系統架構與流程 02
- ▶ AI 模型與硬體配置 03
- ▶ 實測展示 04
- ▶ 問題與解決 05





## 具體情境痛點

- 數據盲點：電商可追蹤「瀏覽時間」與「點擊率」，但實體店缺乏類似的精細數據。
- 傳統缺陷：即便使用高階的主動式紅外線或毫米波雷達，僅能偵測「人體存在」。
- 痛點場景：若顧客站在貨架前「滑手機」或「整理儀容」，感測器仍會持續觸發，導致後台數據誤導商業決策。

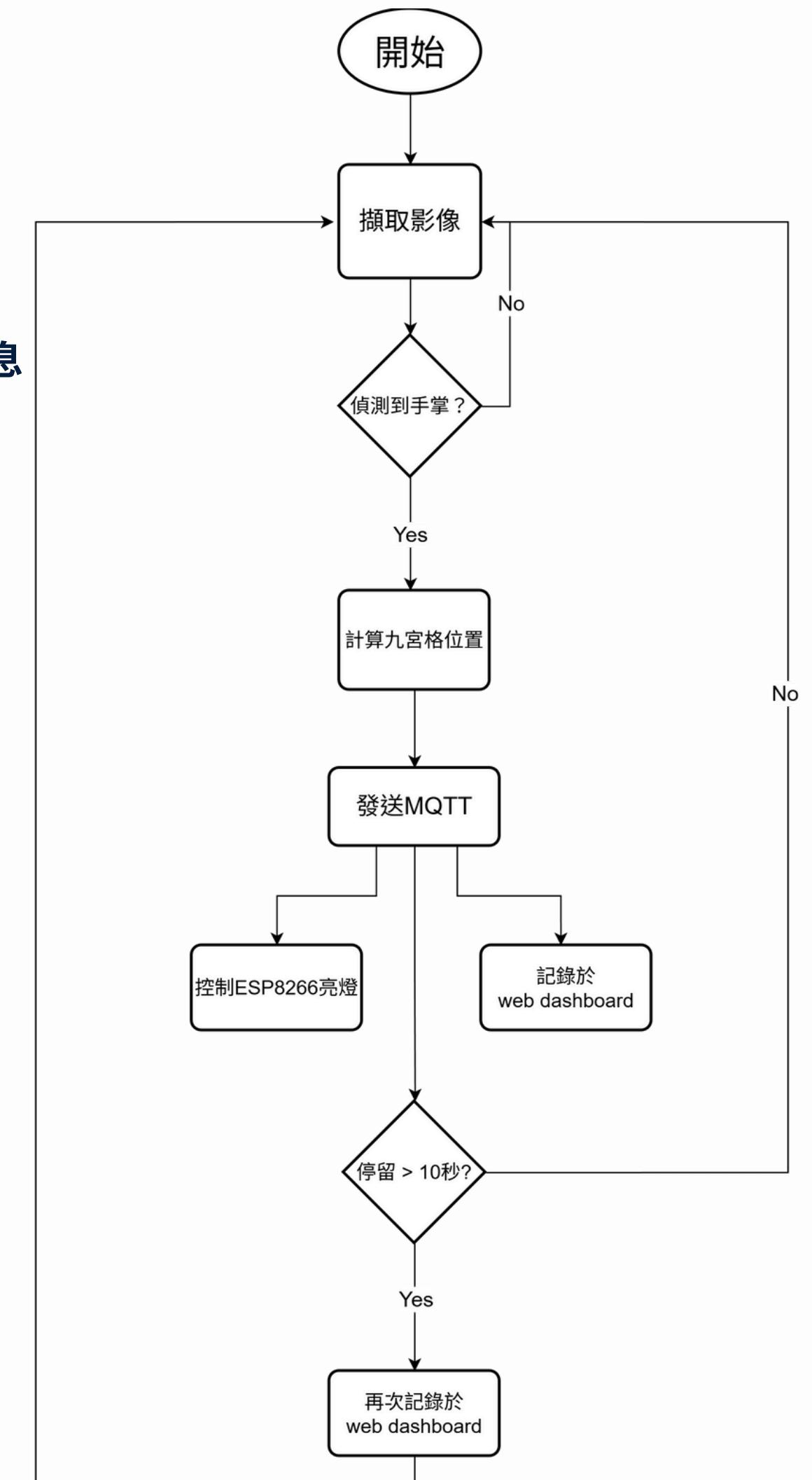
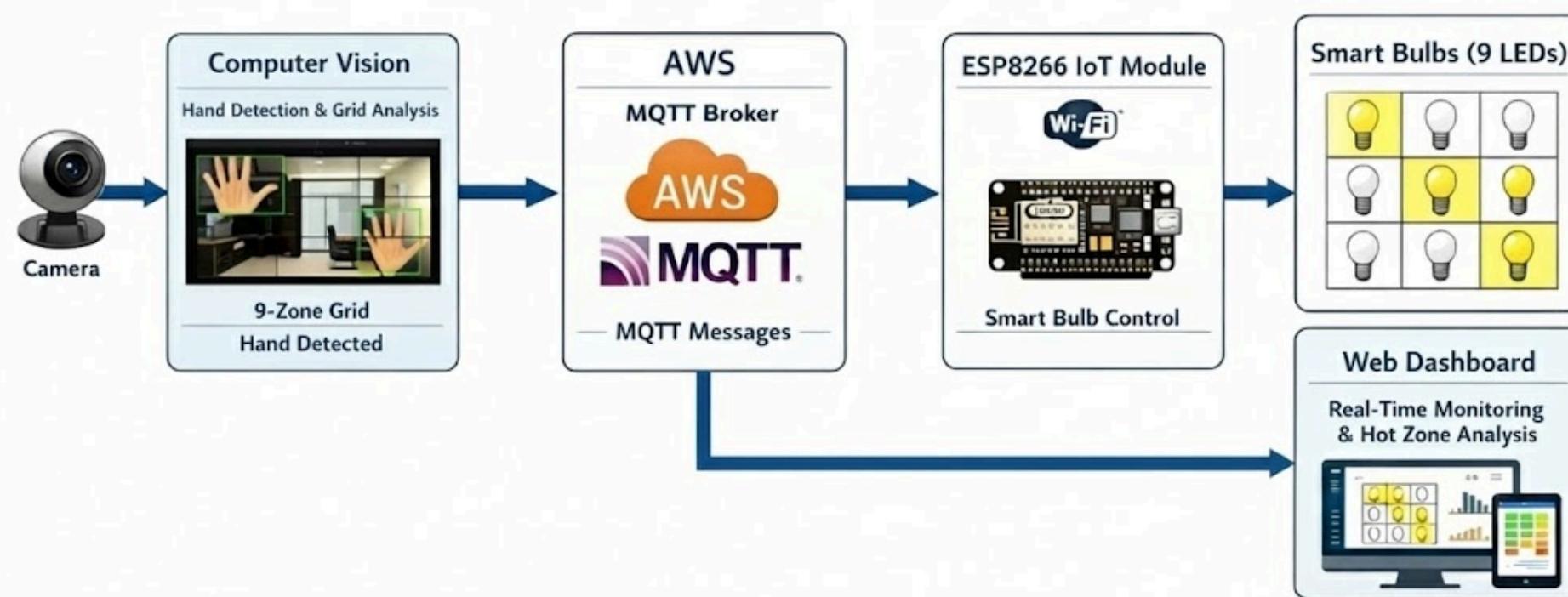


## 想解決的問題

- AI 視覺過濾：本系統利用 AI 偵測關鍵特徵（如人臉朝向）。
  - 低頭滑手機未偵測到正臉 視為無效（過濾）。
  - 抬頭看商品偵測到正臉 視為有效（紀錄）。
- 模擬：
  - 受限於現場展示空間，Demo 以「手掌」模擬真實場景中的「人臉視線」。
  - 手掌出現代表顧客「注視」貨架。
  - 手掌停留 10 秒 代表顧客「深度閱讀」商品資訊。

# 系統架構與流程

- AI 運算端：PC 負責辨識與分析。
- 通訊層：透過 AWS 進行溝通。
- IoT 控制端：ESP8266 負責接收AWS給的訊息並控制燈號。
- 監控端：Web 儀表板記錄並顯示狀態。



# AI 模型與硬體配置

## 1. AI策略：

- 使用的模型：**Mediapipe**
- 任務類型：物件偵測
- 為什麼選擇它？
  - 關鍵點偵測：不同於 YOLO 僅提供物件的「方框範圍」，MediaPipe 能精確捕捉手部 21 個關鍵點。
  - 邊緣運算效能：相較於 YOLO 通常依賴 GPU 算力，MediaPipe 即使在純 CPU 環境或樹莓派等輕量設備上，也能維持高 FPS 的即時流暢度，大幅降低硬體部署成本。
  - 追蹤機制：MediaPipe 採用「偵測 + 追蹤」的混合機制。一旦偵測到手部，下一幀便會切換為追蹤模式，直到目標遺失。這種方式比傳統逐幀偵測更穩定，能有效減少畫面抖動，確保亮燈反應無延遲。

## 2. 硬體配置：

- 感測器
  - 影像輸入：**PC CAM**
    - 功能：負責擷取場域影像，作為 AI 推論之輸入資料。
- 控制核心與致動器
  - IoT 控制核心：**ESP8266**
    - 功能：透過 Wi-Fi 接收 AWS 的 MQTT 指令。
  - 視覺回饋裝置：9顆 LED 燈
    - 腳位接法：使用 GPIO 控制 LED 燈 (D0~D8 腳位)。



實測展示

影片連結

# 問題與解決

問題：單純辨識手掌，位置準確度不足。

解決方案：三點重心法

- 採樣點：取用 Landmark 0 (手腕)、5 (食指根)、17 (小指根)。
- 算法：計算三點 Normalized (x, y) 平均值。
- 成效：獲得近似掌心的穩定座標。

問題：影像端持續偵測，導致 AWS IoT 數據重複累加。

解決方案：狀態鎖定機制

- 機制：記錄九宮格區域的「當前狀態」與「開始時間」。
- 判斷：僅在 **狀態改變** 或 **超過時間門檻** 時才視為新事件。
- 成效：排除連續訊號干擾，精準統計人流。



雲端運算與邊緣運算應用-期末報告

# THANK YOU!