Panda 多功能自走車

專題學生：解昇梁
周庭瑜
林品逸

指導教授：鄭福炯 教授

大 同 大 學
資 訊 工 程 學 系

專題報告

中 華 民 國 110 年 1 月

Panda多功能自走車

大同資工系專題報告

解昇梁 周庭瑜 林品逸

大同大學

資訊工程學系

專題報告

Panda多功能自走車

解昇梁
周庭瑜
林品逸

經 考 試 合 格 特 此 證 明

專題考試委員　　指導教授

_____

_____

_____　　_____

中 華 民 國 110 年 1 月 13 日

摘要

目前社區及商業大樓多數以保全駐點為主，對於僅有少數幾位保全的大樓而言，必須面臨駐點或巡邏的問題，不論選擇哪個都無法避免社區漏洞的發生。

為了解決上述的社區漏洞，我們希望開發一種可降低社區人力、設備成本的多功能自走車，具有跨樓層巡邏且讓保全進行監控之功能，並搭配網頁及 App 做一個資訊整合平台來達成安全無死角防護。目前已實作出網頁遠端操控行駛、即時影像輸出、非接觸式電梯控制系統等功能。因應今年疫情，本專題之應用也可減少間接接觸的可能性，例如可完成防疫旅館之送餐問題、醫院內無菌手術設備運送等。

致謝

專題的完成，該感謝的有三方。第一方是鄭福炯鄭教授，教授在每次開會都會引導我們走到正確的方向以及給予建議，讓我們認知到自己的不足，並加以改善。同時，教授提供足夠的資源以及實驗室，讓我們可以專心地進行實作以及研究。第二方是物聯網技術中心的兩位工程師，他們不厭其煩地對我們講解一些實作上的經驗，這些資深的經驗幫助我們在實作以及研究的過程更加順遂。第三方則是研究生以及機械系博士班的學長，研究生們忙於自己的研究，但還是在百忙之中抽空帶我們進入這個領域。若沒有他們，我們的專題之路將會走得更加艱辛。感謝他們的支持以及教導，我們的專題才能如期完成。

目錄

圖目錄

表目錄

第一章　專題概述

1.1 背景

目前中小型社區及商業大樓多數以保全駐點為主，對於僅有少數幾位保全的大樓而言，必須面臨駐點或巡邏的問題，不論選擇哪個都無法避免社區漏洞的發生。例如：當有陌生人想進入社區，謊報其為住戶的親戚朋友，或是保全正在其他地方巡邏或處理事情時等情形，導致不明人士直接闖空門偷竊，進而造成無法及時處理的現象發生。

圖 1.1 社區大樓失竊案件

以上新聞(圖 1.1)為 2018 年新北市林口一處社區有一名小偷莫名持有住戶之社區門禁磁卡，闖入社區大肆搜刮[1]，**因為當時有門禁卡，保全也沒有再去做查證**，結果連闖三關進入被害人家中行竊，損失將近約一百多萬元。表 1.1 為警政署統計近五年的竊盜犯罪案件[2]。

表 1.1 警政署竊盜犯罪案件

年份	2016	2017	2018	2019	迄今
案件數	57606	52025	47591	42272	33558

為了解決前面所講的保全漏洞和外部因素等原因，本專題希望開發一種可以降低巡邏人力、設備成本的多功能自走車。其可以全區域時段性跨樓層巡邏並搭配保全進行監控，並利用現有的外部裝置，來實現手/自動導航與避障、影像身分辨識、即時影像輸出、危險警告並報警等功能並搭配手機 APP 來做一個資訊整合與查看系統來達成無死角安全社區。

1.2 動機

在實作的過程中，全世界陷入了疫情問題，因此本專題的整體架構因此改變。為了能符合防疫旅館的需求-降低間接接觸的可能性，本專題的目標新增兩個部分，一個是非接觸且非侵入式電梯控制系統，另一個則是可臨時增加巡邏任務。

疫情期間，防疫旅館裡面除了需要降低間接接觸的可能性外，還需要保護工作人員的生命安全，確保工作人員不會因此染疫。因此送餐任務交給 Panda 多功能自走車變成迫在眉睫的事情，危險的工作交給自走車，可以有效降低意外發生的可能性，並且讓防疫更加無死角。

在醫院手術的過程中，所有的器材都必須經過消毒才能進行使用，但運送的過程中，只要有人經過或者運送者本身有病原體，都可能導致意外的發生。因此將此任務交由多功能自走車來進行，可以將風險控制到最低，進而打造一個安全的手術環境。

1.3 專題目的

基於市面上中小型商業大樓保全之問題，本組希望開發出一個可以在室內自動巡邏之多功能自走車，同時其可以完成使用者臨時增加的任務，譬如:在防疫旅館內可進行送餐。本組希望其可以有以下的功能:網頁端控制車體、網頁端得知車體目前所有資訊、自動化巡邏。其中前兩項本專題已經完成，自動化巡邏在專題實作期間並沒有完成，期待未來可以接續完成。

因應疫情以及自走車的需要，本專題同時開發出一個非接觸且非侵入式電梯控制系統，其目的為二。其一為防止間接接觸的可能性，其二為為自走車提供可以藉由電梯上下樓的基礎，在未來完成自動化後，就可以進行室內每一樓層的地毯式巡邏。

1.4 遇到的問題及解決方法

本節將會討論本專題在專題實作的過程中，有遇到哪些技術上的困難點，並且說明如何克服這些問題。

1.4.1網頁端的互動該如何傳送至車體控制系統

一開始我們使用 FTP[3]將檔案傳至車體控制系統內的 Jetson Nano[4]，但是我們發現這種方式不能在穿網(ngrok)[5]之後使用，所以最後改使用 MQTT[6]來傳送資訊。

1.4.2網頁端的互動該如何傳送至非接觸且非侵入式電梯控制系統

經過了車體控制系統的設計後，我們一開始打算直接使用 MQTT 傳送至非接觸且非侵入式電梯控制系統，然而在我們實作上發現，該系統並沒有對外連線的功能。因此，採用 MQTT 傳送至光達及電梯控制系統，再由光達及電梯控制系統透過 CoAP[7]協定傳送至非接觸且非侵入式電梯控制系統。

1.4.3硬體設備使用過多

我們原先採用5塊樹莓派3b+[8]及1塊 Jetson nano 做為控制時的裝置，但是這種方式導致成本過高，所以我們將幾個可以合併的系統合併，最終使用3塊樹莓派3b+及1塊 Jetson Nano。未來與研究生的論文結合後，可以讓我們的架構最終只使用2塊樹莓派3b+及1塊 Jetson Nano。

1.4.4網路連線問題

光達及電梯控制系統必須接收非接觸且非侵入式電梯控制系統之 Wi-Fi，才能使用 CoAP 協定來傳送值。然而，此 Wi-Fi 是沒有對外連線能力的，所以必須要有一個具有對外連線能力的外接裝置來讓其具有對外連線能力，因此我們選用了4G 網卡來給予光達及電梯控制系統對外連線能力。

1.4.5Wi-Fi 連線安全及耗時問題

起初我們讓 ESP32[9]與樹莓派3b+來傳送訊息，但中間所透過的 Wi-Fi 並沒有加密，可能會讓有心人攔截或修改傳送的訊息。之後我們使用物聯網技術中心前輩所留下來的自動安全連線方式，自動產生密碼並加密，以保證連線的安全。

1.4.6即時影像延遲問題

由於使用 WebRTC[10]在傳輸即時串流時會耗費較多的網路資源，所以即時影像系統必須使用較大的頻寬，未來使用5G 裝置將會讓延遲問題消失不見。

1.4.7手機端登入問題

由於我們是使用瀏覽器來作為控制載體，所以我們使用了穿網技術(ngrok)，讓手機端可以進行展示。

1.5 專題專案管理與成本分析

本專題製作從 2020 年 2 月開始至 2021 年 1 月結束，共歷時 12 個月。如表 1.2 的專案進度表所示，分別有幾個不同的階段:蒐集 IOT 技術及相關應用、訂定專題題目、分析市面上產品優缺點、實作專題以及準備比賽資料、換成使用 Panda 而非 Rover、製作非接觸且非侵入式電梯控制系統、書面報告撰寫。本專案組員共 3 人，工作量累計共 7.5 個人月(詳見表 1.3)，工作分配如表 1.4 所示。本專案耗材、物品及設備費用共花費約 11 萬元(詳見表 1.5)，人事費用約 29 萬元，整體

專案執行成本共約 40 萬元(詳見表 1.6)。

表 1.2 專案進度表

年月 / 工作項目	2020年									2021年
	2~4月	5月	6月	7月	8月	9月	10月	11月	12月	1月
蒐集 IoT 技術及相關應用										
訂定專題題目										
分析市面上產品的優缺點										
實作專題以及準備比賽資料										
換成使用 Panda，而非 Rover										
製作非接觸且非侵入式電梯控制系統										
書面報告撰寫										

表 1.3 專案工作量

工作項目	工作量(人月)
設計階段	1
實作階段	4
測試階段	2.5
總計	7.5

表 1.4 工作分配表

工作內容	工作分配比率(%)		
	周庭瑜	解昇梁	林品逸
閱讀並蒐集相關資料	33%	33%	34%
網頁使用者介面程式設計	100%	-	-
MQTT 程式設計	-	100%	-
ROS 系統	-	50%	50%
Server 架構與程式設計	75%	25%	-
車體控制系統程式設計	-	100%	-
即時影像及人臉辨識系統	-	100%	-
Lua 程式設計	-	-	100%
自動安全連線設置	-	30%	70%
Coap 相關程式設計	-	-	100%
測試及修正	33%	33%	34%
系展海報設計	34%	33%	33%
專題報告撰寫	33%	34%	33%

表 1.5 耗材、物品及設備費用

金額單位：新臺幣元

項目名稱	說明	單位	數量	單價	金額
電腦	用於專題開發	臺	2	30,000	60,000
智慧型手機	用於實作測試	臺	2	5,000	10,000
Panda	用於實作環境	臺	1	15,000	15,000
光學雷達	用於實作環境	臺	1	2,500	2,500
Micro Camera	用於實作環境	個	2	900	1,800
樹梅派 3b+	用於專題開發	臺	3	1,300	3,900
Jetson nano	用於專題開發	臺	1	3,300	3,300
行動電源	用於實作環境	個	2	5,600	11,200
非接觸且非侵入式電梯控制系統	用於實作環境	個	1	2,000	2,000
電源供應器	用於實作測試	個	1	3,300	3,300
合計					113,000

表 1.6 整體專案執行成本

金額單位：新臺幣元

項目	金額
耗材、物品及設備費用	113,000
人事費用	7.5(人月) * 39,000(元/人月) = 292,500
總成本	405,500

第二章　相關應用及技術

本專題是由網頁、多功能自走車、非接觸且非侵入式電梯控制系統所組成的。其中網頁會使用到的技術有:HTML[11]、JavaScript[12]、CSS[13]、Ajax[14]、JSON[15]、PHP[16]等，巡邏車會使用到 FTP、Python[17]、Java[18]、C[19]、樹莓派、Jetson Nano、Intel NCS2[20]、WebRTC、ngrok、OpenVINO[21]、MQTT 以及 CoAP 等，非接觸且非侵入式電梯控制系統會使用到 ESP32、光敏感測器、紅外線偵測器、Lua[22]、自動安全連線等。上述之使用過程請參照第四章。

本章將介紹市面上的非接觸電梯控制系統、多功能自走車的相關比較以及本專題所利用到的技術。

2.1 相關應用

本小節將介紹目前已存在的非接觸電梯控制系統以及多功能自走車的相關比較，本專題使用商業模式圖 2.2、SWOT 圖 2.3 以及比較表 2.2 來分析市面產品與多功能自走車之差異性。

2.1.1非接觸電梯控制系統

下圖 2.1 為國立交通大學工程三館之非接觸電梯控制系統，其特色為用視覺辨識的方式來判斷手心向上為按上鍵、手心向下為按下鍵。其優缺點將由下表 2.1 來進行說明。

圖 2.1 非接觸電梯控制系統

表 2.1 非接觸電梯控制系統之優缺點

優點	缺點
可與電梯按鈕面板完全契合	不可適用於所有種類的電梯
手勢簡單明瞭	容易出現上下鍵皆觸發的情形

2.1.2多功能自走車的相關比較

本專題所假設的商業模式為出租機台以及幫忙架設警報網來做為主要的獲利來源，並且提供維修以及紀錄巡邏資訊之功能。圖 2.2 為商業模式圖。

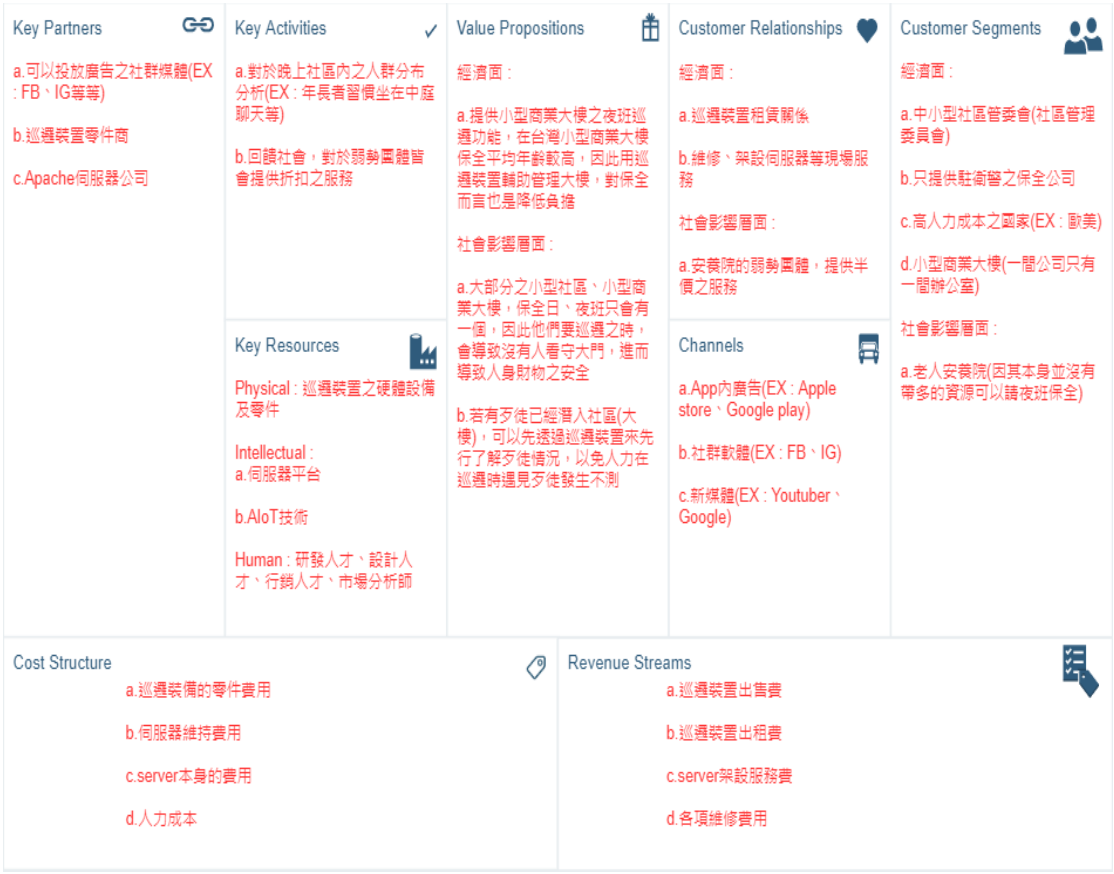

圖 2.2 商業模式圖

為了讓此作品能夠有商轉之機會，因此，本專題利用商業模式的關鍵詞，來製作一個 SWOT 分析圖，藉由圖 2.3 可以讓使用者更了解本專題的優缺點。

內部能力 / 外部因素	Strength a.造價便宜 b.減少巡邏時人力成本消耗	Weakness a.容易電量不足 b.知名度低
Opportunities a.保全平均年齡上升 b.人力成本上升 c.社會事件逐年增加	SO 由於保全平均年齡上升以及人力成本上升，因此減少巡邏時人力成本消耗將變成必要事項，也可同時降低保全在社區所滋生的社會事件。	WO 改良機體架構並以多顆備用電池的方式解決電力不足之問題，使小型社區僅需要一兩台即可減少多位人力資源浪費，對於保全公司來說，將有會有一定的吸引力。
Threat a.商品可取代性高 b.競品市占率高	ST 以低造價優勢及機體架構輕便為特點搶占其他相關市場，可大幅提升產品競爭力。	WT 利用適應高低起伏地形的一大特點來作為主打與其他產品不同的之處，並根據客戶的需求，在巡邏裝置上做出客製化的設計，以在市場上拓展知名度。

圖 2.3 SWOT 分析圖

本專題與現在的保全公司做一個比較，從眾多保全挑了兩家公司，這兩家公司以駐點保全為主，一個為家興保全，另一個則是國光保全；而本組從中利用這兩家保全之人力來與 Panda 對比分析。表 2.2 為 Panda 與駐點保全之比較。

表 2.2 與駐點保全之比較

	Panda	家興保全	國光保全
價格	v		
巡邏速度		v	v
防護措施	v		
監控範圍	v		
應變能力	v		
其他	v		

2.2 網頁層面

本小節會介紹本專題在網頁端所利用到的技術: HTML、JavaScript、CSS、Ajax、JSON 及 PHP，將在接下來的小節進行說明。

2.2.1 HTML

HTML (Hyper Text Markup Language)並非一種程式語言，而是一種網頁的基本語言，透過成對的標籤來實現所需要的功能。

2.2.2 JavaScript

JS(JavaScript)是一種直譯的程式語言。其可以在任何 HTML 基礎上的網頁進行插入，且可以做到互動式的效果，讓網頁與使用者更加密切。

2.2.3 CSS

CSS(Cascading Style Sheets)是一種用來為結構化的文件。其與 JS 相同，可以在任何 HTML 基礎上的網頁進行插入，但無法通過動態的方式呈現。

2.2.4 Ajax

Ajax(Asynchronous JavaScript and XML)是一種綜合多項網頁開發功能的開發技術。傳統的 Web 會在傳輸訊息時浪費許多頻寬，而 Ajax 可以僅取得或傳送需要的資訊給伺服器。

2.2.5 JSON

JSON(JavaScript Object Notation)是一種輕量級的資料交換語言，用來傳輸有屬性值或者有序的物件。

2.2.6 PHP

PHP(Hypertext Preprocessor)是一種通用的電腦語言。其可以嵌入 HTML 中使用，同時 PHP 可以讓開發者迅速的編寫出動態頁面。

2.3 Panda 多功能自走車

本小節會介紹本專題在自走車所利用到的技術及硬體: FTP、Python、Java、C、樹莓派、Jetson Nano、Intel NCS2、WebRTC、ngrok、OpenVINO、MQTT、CoAP，將在接下來的小節進行說明。

2.3.1 FTP

FTP(File Transfer Protocol)是一個用於電腦網路上在 Client 端與 Server 端之間進行檔案傳輸的協定。

2.3.2 Python

Python 是一種直譯的程式語言。其支援多種程式範式，包括物件導向、結構化、函數式等，同時也為現今最多人使用的程式語言。

2.3.3 Java

Java 是一種物件導向的程式語言。其程式語言寫法十分接近 C++語言，同時捨棄開發者容易引起錯誤的指標。

2.3.4 C

C 語言是一種廣泛應用於系統軟體及應用軟體開發的程式語言。其具有靈活的特性，讓使用者可以對記憶體中的每一個 Byte 進行操作，同時在特殊情況下還可以對每一個 bit 進行操作。

2.3.5 樹莓派

樹莓派是基於 Linux 作業系統的單晶片電腦，由英國樹莓派基金會所開發，目的是以低價硬體及開源軟體來促進基本電腦科學教育。

2.3.6 Jetson Nano

Jetson Nano 為 NVIDIA 所開發的一款小巧型電腦,其目的為增強邊緣運算以及人工智慧應用程式之效率所設計的。

2.3.7 Intel NCS2

Intel NCS2(Neural Compute Stick 2)為 Intel 所開發的一款邊緣運算的裝置,其可以安插在樹莓派上，使樹莓派可進行類神經模型之運算。

2.3.8 WebRTC

WebRTC(Web Real-Time Communication)是一個支援瀏覽器進行即時語音對話或者影像對話的 API(Application Programming interface)。本專題利用這個 API 來實作即時影像。

2.3.9 ngrok

ngrok(反向代理)通過安全隧道將位於 NAT 和防火牆後面的本地服務器公開到公共 Internet。

2.3.10 OpenVINO

OpenVINO(Open Visual Inference and Neural network Optimization)工具包是一個免費的工具包，可通過使用推理引擎將深度學習模型從框架和部署優化到 Intel 硬體上。

2.3.11 MQTT

MQTT(Message Queuing Telemetry Transport)訊息佇列遙測傳輸是 ISO 標準下基於發布/訂閱範式的訊息協定。

2.3.12 CoAP

CoAP(Constrained Application Protocol)是受限制的應用協議，是 IETF 組織發布的網際網路系列標準之一。

2.3.13 ROS

ROS(Robot Operating System)[23]是專為機器人軟體開發所設計出來的一套電腦作業系統架構。其提供類似於作業系統的服務，包括硬體抽象描述、底層驅動程序管理、共用功能的執行、程序間消息傳遞、程序發行包管理等等許多作業系統應有的功能。

2.3.14 Californium

Californium[24]可實現 Coap: udp, tcp 兩種傳輸層協議，單次請求傳輸可達到在 1M 以內。

2.4 非接觸且非侵入式電梯控制系統

本小節會介紹本專題在非接觸且非侵入式電梯控制系統所利用到的技術及硬體:Lua[24]、光敏感測器[25]、紅外線感測器[26]、ESP32 以及自動安全連線，將在接下來的小節進行說明。

2.4.1 Lua

Lua 是一個簡潔、輕量、可延伸的手稿語言。Lua 有著相對簡單的 C 語言 API，且容易嵌入應用中。很多應用程式使用 Lua 作為自己的嵌入式手稿語言，以此來實現可組態性、可延伸性。

2.4.2 光敏感測器

光敏感測器是可以感測光或是其他電磁能量的感測器，而光敏電阻會依照光線強度而改變其電阻，一般是光線強度越強，其阻值越小。

2.4.3 紅外線感測器

紅外線感測器[26]可分為主動式和被動式兩種。

主動式的紅外線感測器(紅外線避障模組)，感測器本身會發射紅外線光束，當紅外線光束被物體擋住後，紅外線光束就會反射，接收器接收反射的紅外線，做出動作。例如：廁所的自動沖水小便斗、感應式水龍頭。

被動式的紅外線感測器：紅外線動作感測器 (Passive Infrared Sensor) 又稱為 (PIR Motion Sensor)，感應器本身不會發射紅外線光束。是利用物體發射出來的紅外線的變化，來感應物體的移動。一般利用人體會發出紅外線的特性，常用來當作感應人體的感測器。本專題使用的紅外線感測器為被動式紅外線

2.4.4 ESP32

ESP32 是一系列低成本，低功耗的單晶片微控制器，整合了 Wi-Fi 和雙模藍牙。 ESP32 系列採用 Tensilica Xtensa LX6 微處理器，包括雙核心和單核變體，

內建天線開關、RF 變換器、功率放大器、低雜訊接收放大器、濾波器和電源管理模組。

2.4.5 自動安全連線

自動安全連線是連線端在搜尋 gateway 的同時會擷取被連線端的 ssid，並將之做 base64[27]的加密轉換，轉換後產生的密文就是該 gateway 的密碼以供連線使用，而其中的 base64 轉換的程式碼為本專題之指導老師鄭福炯老師所提供。

第三章　系統設計及架構圖

3.1 系統架構圖

本專題的整體系統分成三個部分，圖 3.1 為系統架構圖。第一部分為 Web/APP 請參照 4.1 小節、第二部分為 PANDA 多功能自走車、第三部分為非接觸且非侵入式電梯控制系統，其中第二部分又由三個子系統建構而成。在接下來的幾個小節，將會依序對二、三部分進行介紹。

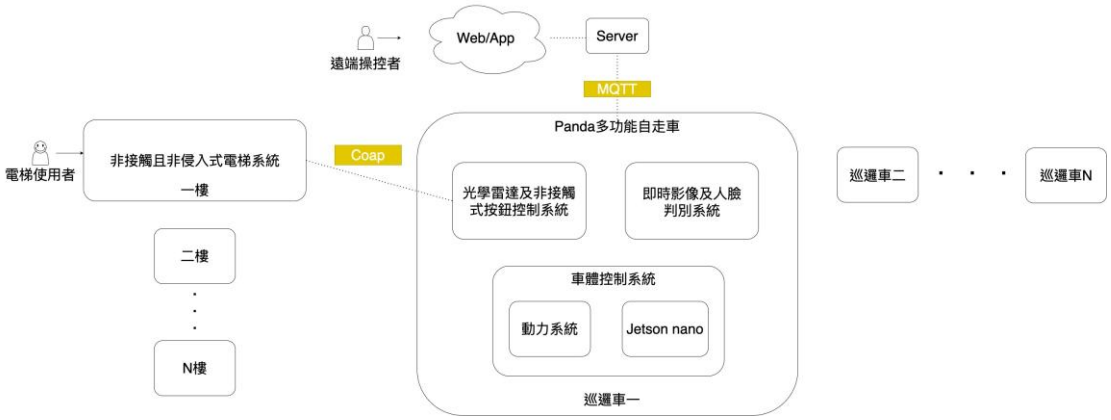

圖 3.1 系統架構圖

3.2 光學雷達及電梯控制系統

本組的光學雷達及電梯控制系統主要作為電梯控制傳送的中繼站，先讓其透過 4G 網路取得網頁所做出的動作資訊，再透過 ESP32 的 Wi-Fi 傳送資訊給電梯控制系統，其中分別使用 MQTT 以及 CoAP。

同時，這個系統上也承載著 ROS(Robot Operating System)，建圖以及未來要做的定位、導航都是靠其完成。

圖 3.2 光學雷達及非接觸式電梯控制系統

3.3 即時影像及人臉辨識系統

本專題透過 WebRTC 來讓即時影像顯示在樹莓派上的某 port 號，同時本組使用反向代理(ngrok)來讓網頁端可以取得即時影像的資訊，人臉辨識系統未來是要改成承載研究生的視覺辨識系統，目前暫時使用 OpenVINO 所公布的公開模組，並對其進行修改。其辨識能力，經過實測可以在 0.3 公尺至 1 公尺之間。

圖 3.3 即時影像及人臉辨識系統

3.4 車體控制系統

本專題透過 MQTT 將網頁端所做的動作值傳至 Jetson Nano 內我們所寫的 python 檔，在該檔內透過 MQTT 的訂閱端來取得資訊，並將其轉換成相映的值，同時調用機械系博士班學長所寫的 API，來將參數傳入動力系統。

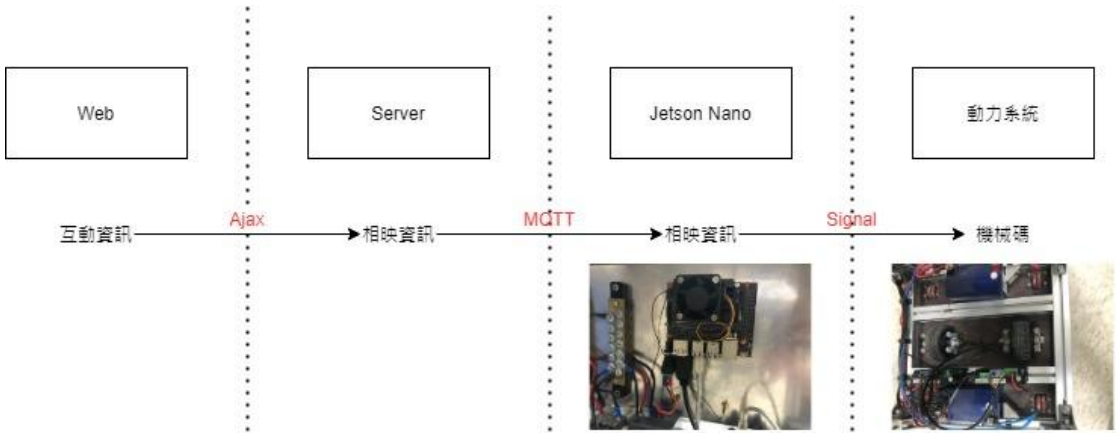

圖 3.4 車體控制系統

3.5 非接觸且非侵入式電梯控制系統

此臺電梯控制系統的機體架構是機械系博士班學長協助完成的，而內部的主要控制是由一塊 Esp32 的單晶片微控制器去控制致動器做所預想的動作，Esp32 是使用 lua 程式語言去完成程式的，當中編譯 lua 程式的 IDE 是使用 ESPlorer，在電梯控制系統裡還包含紅外線感測儀器及光敏感測儀器。

與此同時，電梯控制系統還裝有紅外線感測儀器，可以讓使用者在不接觸電梯按鈕的情況下使用電梯達到上下樓層的目的；光敏感測器是針對電梯控制系統在使用時，偵測致動器撞擊電梯按鈕時是否真的有撞擊成功。

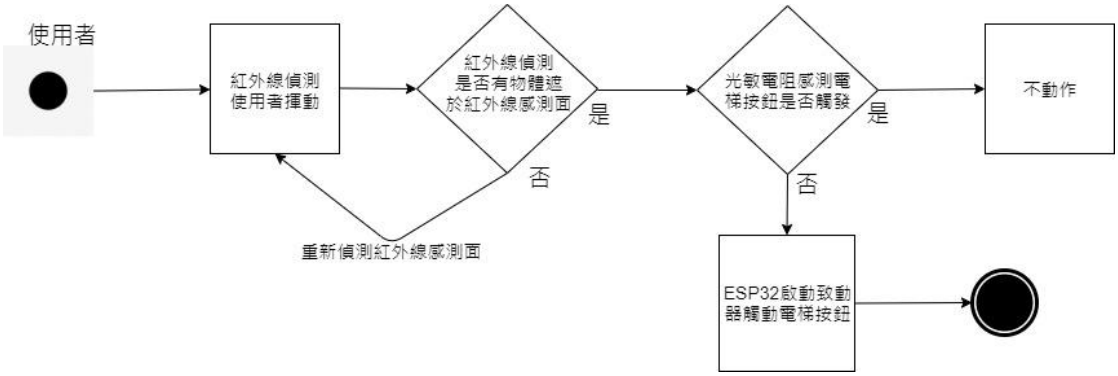

圖 3.5 非接觸且非侵入式電梯系統

第四章　系統實作過程

在實作的過程中，架構總共分成三個子架構，分別為網頁、Panda 及電梯控制系統。同時，自走車又分為三個子系統，分別為光學雷達及遠端控制電梯系統、即時影像及人臉辨識系統、車體控制系統。

4.1 網頁

利用網頁作為資訊整合平台，提供使用者查看資料。共有 4 個分頁，分別是首頁、即時影像、照片和樓層地圖。

4.1.1首頁

版面使用 Bootstrap[28]提供的素材製作，網頁需先註冊會員並登入後才能瀏覽內容。

下圖為網頁登入前(圖 4.1)及登入後(圖 4.2)的首頁選單。

圖 4.1 登入前首頁左方標籤

圖 4.2 登入後首頁左方標籤

Panda 簡介以及設計理念

在首頁的部分提供簡介(圖 4.3)和設計理念(圖 4.4)，讓瀏覽者能夠快速了解 Panda。簡介使用 Bootstrap 的 card 製作名片格式，html 程式碼請參照附錄 4.1.1。

圖 4.3 Panda 多功能自走車簡介

圖 4.4 Panda 多功能自走車設計理念

4.1.2註冊及登入

下圖為會員註冊(圖 4.5)及登入(圖 4.6)頁面。網頁會自動判斷帳號是否重複註冊、註冊時兩次輸入的密碼是否一致[29]，JavaScript 和 Ajax 程式碼請參照附錄 4.1.2。

圖 4.5 會員註冊頁面

圖 4.6 會員登入頁面

4.1.3即時影像

利用 WebRTC 套件來完成即時影像串流，詳細過程請參照 4.2.2 小節。下圖網頁顯示之畫面為即時影像畫面，右方按鍵則可以控制 Panda 行駛方向、查看目前所在樓層、顯示控制電梯控制系統之剩餘電量、控制電梯上下樓(圖 4.7)。HTML、JavaScript、CSS 和 Ajax 程式碼請參照附錄 4.1.3。

圖 4.7 即時影像頁面

4.1.4最近記錄下來的人臉辨識照片

本專題在樹梅派上使用 OpenVINO 以及 Intel 之開源資料，配合車體上的鏡頭來即時作出人臉辨識。同時，網頁每 30 秒更新 Panda 所拍攝的最新 9 張包含人臉的照片，照片右方會標註拍攝日期及時間。PHP 程式碼請參照附錄 4.1.4。

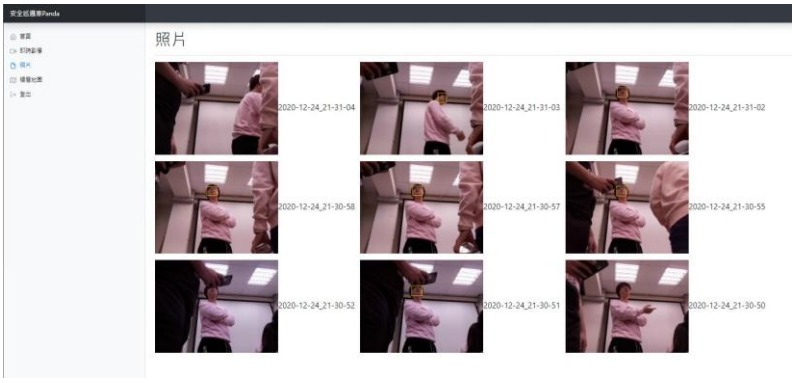

圖 4.8 照片頁面

4.1.5 ROS 所建立的樓層地圖

本專題利用 ROS 以及 hector slam[30]來建置地圖，未來將會完善定位以及導航功能，並且會同時在一個頁面上顯示，如此可讓使用者了解目前巡邏車的所在位置。

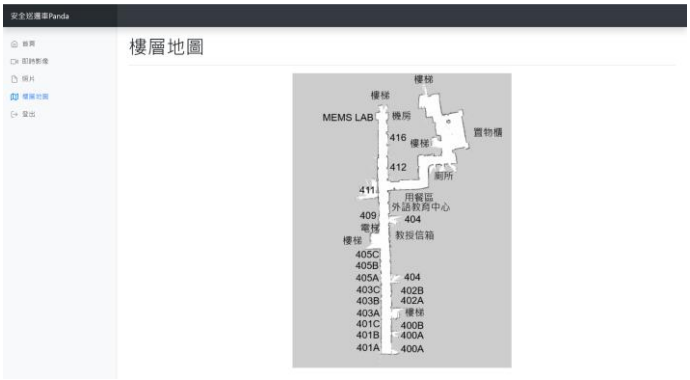

圖 4.9 樓層地圖頁面

4.1.6MQTT 通訊

網頁 Server 端作為 publisher 傳送資訊至物聯網技術中心的 broker[31]，再讓車體控制系統、光學雷達及遠端控制電梯系統 subscribe 相同的 topic，以取得網頁 server 端所傳送的資訊。PHP 程式碼請參照附錄4.1.6.1及4.1.6.2。

光學雷達及遠端控制電梯系統作為 publisher 傳送資訊至物聯網技術中心的 broker，再使網頁 Server 端 subscribe 相同的 topic，並取得光學雷達及遠端控制電梯系統所傳送的資訊。PHP 程式碼請參照附錄4.1.6.3。

4.2 Panda 多功能自走車

本小節將分別介紹三個子系統(光學雷達及遠端控制電梯系統、即時影像及人臉辨識系統、車體控制系統)的實作過程及步驟，同時並說明一些實作上的技巧以及注意事項。

4.2.1光學雷達及遠端控制電梯系統

(1)ROS 系統架設及應用[32]

ROS 是專門設計給機器人使用的整合軟體，其上面可以安裝不同的 slam[33]，來完成建置地圖、定位以及導航等功能。實作的過程中，參照 ROS 官方[34]所釋出的資訊來進行使用。

本專題使用了 hector slam 來做為我們建置地圖[35]、定位[36]以及導航[37]演算法，hector slam 一開始是被設計給凹凸不平的道路所使用，可用於救災的無人車上。正因如此，所以本專題選擇 hector slam，最終也透過其完成建置地圖及定位。

(2)遠端控制電梯

在遠端控制電梯的功能中，分成兩個部分，一個為取得網頁的 server 端資訊並傳送至電梯控制系統，一個為接收電梯控制系統所回傳的資訊，並傳至網頁的 server 端。第一部分，透過 MQTT 協定來實現網頁 server 端與光學雷達及遠端控制電梯系統的資訊傳遞，同時利用 CoAP 協定來實作出光學雷達及遠端控制電梯系統與電梯控制系統之間的資訊傳遞。Python 程式碼與 Java 程式碼請參照附錄 4.2.1.2.1。第二部分，也是透過 MQTT 協定來實作光學雷達及遠端控制電梯系統與網頁 server 端的資訊傳遞。Python 程式碼請參照附錄4.2.1.2.2。

為了使樹莓派和 ESP32 做 machine to machine 的溝通，實作過程中讓 Java(CoAP protocol)程式碼先跟該 ESP32 做與該裝置的本機連線，之後就可利用 Californium 框架做 CoAP 協定與溝通以達到傳值及取值的作用，程式碼請參照附錄 4.2.1.2.3。

(3) ROS 系統掃描地圖與建立地圖

在建立各層樓的地圖中，本專題是使用 ROS 系統的內建指令來進行的，以下指令均在樹莓派之終端機中輸入。

roscore //開啟 ROS 系統

roslaunch delta_lidar delta_lidar.launch //開啟 ROS 連接 lidar 之程序

roslaunch delta_lidar slam.launch //開始使用光學雷達掃描地圖

sudo apt-get install ros-kinetic-map server//下載存圖之套件

rosrun map-server map-saver -f ~/存圖之名稱-map//儲存掃描好之地圖

4.2.2即時影像及人臉辨識系統

(1)人臉辨識系統

本專題採用 openVINO 所公布的 open source 之模型，寫出一個使用上述模型之偵測程式，讓其可以透過模型來進行人臉辨識。在確認有人被偵測出後，會以時間為檔名紀錄所拍攝之人臉圖片。Python 程式碼請參照附錄4.2.2.1.1。

透過 Window10 並利用自動化 FTP，將圖片從樹莓派上取得至 server 端主機。.bat 程式碼請參照附錄4.2.2.1.2。

(2)即時影像

A.WebRTC 使用及安裝

- 第一步在樹莓派上安裝 UV4L

以下指令均在樹莓派之終端機內部依序輸入。

curl http://www.linux-projects.org/listing/uv4l_repo/lpkey.asc | sudo apt-key add -

sudo nano /etc/apt/sources.list

deb http://www.linux-projects.org/listing/uv4l_repo/raspbian/stretch stretch main

sudo apt-get update

sudo apt-get install uv4l uv4l-raspicam

sudo apt-get install uv4l-raspicam-extras

sudo apt-get install uv4l-server uv4l-uvc uv4l-xscreen uv4l-mjpegstream uv4l-dummy uv4l-raspidisp

sudo apt-get install uv4l-webrtc

- 第二步執行 UV4L

以下指令均在樹莓派之終端機內部依序輸入，且每次使用皆要執行。

uv4l --auto-video_nr --driver raspicam --encoding mjpeg --server-option '--port=9000‘

nano /etc/uv4l/uv4l-raspicam.conf

sudo service uv4l_raspicam restart

- 當完成第二步之後，即可在樹莓派 localhost 的 port 9000裡面看到控制介面
- 在 port 9000內部有一個 stream，代表著點進去即可觀看即時影像
- 在 port 9000內部有一個 control box，在裡面可以控制 stream 的參數

B.ngrok 使用

其安裝過程參照 ngrok 反向代理官方網站。我們可以透過 ngrok 反向代理將本機端的某些 port 號開放，讓其可以被外界用網址來進行觀看。在安裝完後，我們只需要在 ngrok.exe 所在的資料夾中開啟終端機，並在終端機輸入下述指令 ./ngrok http 9000 ，這時系統會給予兩個分別使用 http 和 https 協定的網址，這兩個網址都可以連接至本機剛才所開放的 port 上。

4.2.3車體控制系統

在實作車體控制系統時，Jetson Nano 扮演 MQTT 協定中的 subscriber，透過 subscribe 的動作來取得 server 端所 publish 至物聯網技術中心的資訊，同時調用

黃瑋晟所寫的 API，來將資訊傳換成機械碼，並傳送至動力系統使其運作起來。Python 程式碼請參照附錄4.2.3。

4.3 非接觸且非侵入式電梯控制系統

本系統在實作的過程中，樹莓派與網頁 MQTT 協定來進行訊息交換，同時樹莓派與 ESP32 使用 CoAP 協議[38]當中的 Californium 技術框架來實現 CoAP 協議的各種 request 及 response，使其可以讓樹莓派與 Esp32 做 machine to machine 的溝通。詳細請參照 4.2.1 小節。

本系統實作的過程中，使 ESP32 將讀取到的值去做致動器的觸動，並經由光敏感測器偵測按鈕是否有觸動成功，若先前已觸發過，後續欲觸發則不動作。ESP32 也可透過紅外線感測器去感測目前欲搭乘電梯之乘客的按鈕使用狀況，於上下按鈕間的紅外線感測區做揮動即可使上下按鈕動作，並經由光敏感測器偵測原電梯上的按鈕是否有按成功，若先前已觸發過，後續欲觸發則不動作，程式碼請參照附錄 4.3。

第五章　結論與未來展望

5.1 結論

本專題利用物聯網技術、網頁設計技術、ROS、公開影像處理技術實作出一套遠端控制多功能自走車及非接觸且非侵入式電梯控制系統。這兩個系統可以透過網頁端來控制 Panda 行走、電梯上下樓層，其中網頁可以在任何有對外連線能力的裝置上進行操控，非接觸且非侵入式電梯控制系統則可以讓使用者降低間接接觸的可能性，使得防疫效果可以進一步的提升。

在中小型社區、商業大樓之中，可以利用多功能自走車來保障住戶安全、保全人身安全與私人財產安全，並透過重重的監控機制，來建置整個社區、大樓的維護網，進而達成全面安全的成果，因此本組認為這種巡邏裝置有一定的市場。

相較於市面上人手不足的駐點警衛以及非接觸電梯控制系統，本專題可以有效降低人力成本，讓中小型社區、商業大樓也能達成安全無死角的防護。同時非接觸且非侵入式電梯系統可以應用於任意型號的電梯，靈活性比目前市面的非接觸電梯控制系統還要來的高，而且不容易出現同時觸發好幾個按鍵的情況。當未來自動導航完成之後，本專題將可以利用在防疫旅館、醫院等場所，可以全面提升安全性及便利性。

5.2 未來展望

本專題未來希望完成全自動多功能自走車且可讓使用者自行設定巡邏區域與時間，同時人臉辨識系統會替換成碩士班學長的視覺辨識系統，讓自走車可以同時利用光學雷達來導航，又可以利用視覺辨識進行微幅修正。如此一來，便可以讓這台自走車應用在中小型社區、商業大樓、防疫旅館、醫院等場所。以下為本組針對中小型社區防護的未來展望流程圖(圖5.1)。

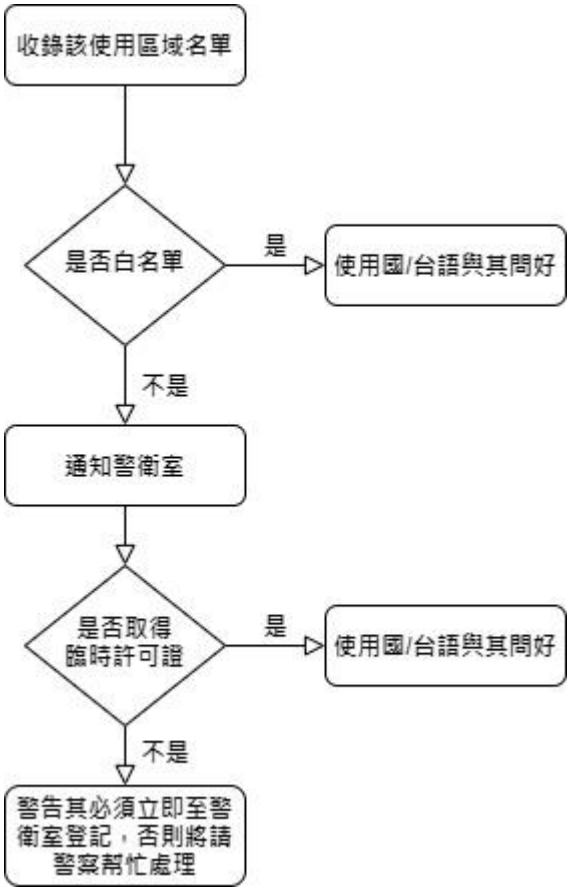
收錄該使用區域名單
是否白名單
是
使用國/台語與其問好
不是
通知警衛室
是否取得
臨時許可證
是
使用國/台語與其問好
不是
警告其必須立即至警
衛室登記，否則將請
警察幫忙處理

圖 5.1 未來展望流程圖

參考文獻

[1] TVBS。損失百萬！賊挑小年夜 持磁卡闖宅偷竊(2018-02-18)。檢自 https://makerpro.com.tw/2018/11/the-world-of-ros-to-practice-of-turtlesim/ (August 5,2020)

[2]警政署統計查詢網。https://ba.npa.gov.tw/npa/stmain.jsp?sys=100 (Jan 8,2021)

[3] FTP。檢自 https://zh.wikipedia.org/wiki/%E6%96%87%E4%BB%B6%E4%BC%A0%E8%BE%93%E5%8D%8F%E8%AE%AE (July 7,2020)

[4] Jetson Nano。檢自 https://www.nvidia.com/zh-tw/autonomous-machines/embedded-systems/jetson-nano/ (July 23,2020)

[5] ngrok。檢自 https://ngrok.com/ (Octorber 23,2020)

[6] MQTT。檢自 https://zh.wikipedia.org/wiki/MQTT(July 23,2020)

[7] CoAP。檢自 http://blog.ittraining.com.tw/2016/12/coap-vs-mqtt.html (July 23,2020)

[8]樹莓派。檢自 https://zh.wikipedia.org/wiki/%E6%A0%91%E8%8E%93%E6%B4%BE (July 8,2020)

[9] ESP32。檢自 https://zh.wikipedia.org/wiki/ESP32 (July 23,2020)

[10] WebRTC。檢自 https://zh.wikipedia.org/wiki/WebRTC (October 9,2020)

[11] HTML。檢自 https://zh.wikipedia.org/wiki/HTML (July 23,2020)

[12] JavaScript。檢自 https://zh.wikipedia.org/wiki/JavaScript (July 23,2020)

[13] CSS。檢自 https://zh.wikipedia.org/wiki/%E5%B1%82%E5%8F%A0%E6%A0%B7%E5%BC%8F%E8%A1%A8 (July 23,2020)

[14] Ajax。 檢自 https://zh.wikipedia.org/wiki/AJAX (July 23,2020)

[15] JSON。檢自 https://zh.wikipedia.org/wiki/JSON (July 23,2020)

[16] PHP。檢自 https://zh.wikipedia.org/wiki/PHP (July 23,2020)

[17] Python。檢自 https://zh.wikipedia.org/wiki/Python (March 13,2020)

[18] Java。檢自 https://zh.wikipedia.org/wiki/Java (July 6,2020)

[19] C 語言。 檢自 https://zh.wikipedia.org/wiki/C%E8%AF%AD%E8%A8%80 (July 7,2020)

[20] Intel NCS2。檢自 https://ark.intel.com/content/www/tw/zh/ark/products/140109/intel-neural-compute-stick-2.html (July 23,2020)

[21] OpenVINO。檢自 https://en.wikipedia.org/wiki/OpenVINO (July 23,2020)

[22] Lua。檢自 https://zh.wikipedia.org/zh-tw/Lua (October 9,2020)

[23] Ubuntu install of ROS Kinetic。 檢自 http://wiki.ros.org/kinetic/Installation/Ubuntu(July 17,2020)

[24] Californium。檢自 https://www.eclipse.org/californium/ (July 23,2020)

[25]光敏感測器。檢自 https://www.taiwaniot.com.tw/product/%E5%85%89%E6%95%8F%E9%9B%BB%E9%98%BB%E6%84%9F%E6%B8%AC%E5%99%A8%E6%A8%A1%E7%B5%84/ (Octorber 23,2020)

[26]紅外線感測器。檢自 http://blog.ilc.edu.tw/blog/index.php?op=printView&articleId=728064&blogId=868 (July 23,2020)

[27] Base64。檢自 https://zh.wikipedia.org/zh-tw/Base64 (January 10,2021)

[28] bootstrap。檢自 https://getbootstrap.com/(July 23,2020)

[29]蔡承洋 Youtube 頻道(August 27,2020)

https://www.youtube.com/c/%E8%94%A1%E5%AD%9F%E7%8F%82/videos

[30] hector slam。檢自 http://wiki.ros.org/hector_slam(July 17,2020)

[31]樹莓派安裝 Mosquitto 輕量級 MQTT Broker 教學，連結各種物聯網設備(2016-12-30)。檢自 https://blog.gtwang.org/iot/raspberry-pi/raspberry-pi-mosquitto-mqtt-broker-iot-integration/(July 30,2020)

[32] 凱森 張。Raspberry Pi 3b+ 安裝 Ubuntu 16.04 ROS 過程(2020-06-04)。檢自 https://vocus.cc/article/5ed7aa37fd89780001520946(July 11,2020)

[33] ROS 與 SLAM 入門教程-激光雷達(EAI F4)hector_slam 構建地圖(2017,02)。檢自 https://www.ncnynl.com/archives/201702/1365.html(July 17,2020)

[34]ROS 官網。檢自 https://www.ros.org/(July 17,2020)

[35] Rplidar 鐳射雷達使用 hector_slam 進行建圖(2018-12-06)。檢自 https://www.itread01.com/content/1544101622.html(July 21,2020)

[36] Kate cheng。[ROS#11]Localization(2019-09-26)。檢自 https://ithelp.ithome.com.tw/articles/10221092(July 20,2020)

[37] 晨星小子。Ros 的 navigation 之 amcl(localization)應用詳解(2015-11-25)。檢自 https://blog.csdn.net/chenxingwangzi/article/details/50038413(July 18,2020)

[38] 國立臺中教育大學資訊工程學系碩士論文(July 15,2020) (以 CoAP 為基礎運用於異質物聯網 環境中的自動服務整合與遞送機制 A

CoAP-Based Transparent Service Integration and Delivery Mechanism in Heterogeneous IoT Environment) 中華民國一百零三年七月

附錄

4.1.1

```
//使用 Bootstrap 的 card 製作名片格式
<div class="card" style="width:400px">
    <img class="card-img-top" src="car.jpg" alt="Card image">
    <div class="card-body">
        <h4 class="card-title">安全巡邏車 Panda</h4>
            <p class="card-text">1.即時影像:隨時查看巡邏車當前狀態並可使
            用網頁遠端操控行駛<br>
            2.非接觸式電梯控制系統:透過網頁連接 gateway 控制電梯上下樓
            <br>3.即時人臉辨識:每 30 秒更新巡邏車所拍的 9 張最新照片<br>4.
            樓層地圖:顯示巡邏車所在之樓層地圖</p>
    </div>
</div>
```

4.1.2

```
<script>
    $(document).ready(function(){
        //檢查帳號是否重複
        $("#username").keyup(function(){
            if($(this).val() != ''){
                $.ajax({
                    type:"POST", //表單傳送的方式
                    url:"checkname.php", //目標給哪個檔案
                    data:{ //為要傳過去的資料，使用物件方式呈現，因為變
                    數 key 值為英文的關係所以用物件方式傳送
                        'n':$(this).val() //代表要傳一個 n 變數，值為 username
                        文字方塊裡的值
                    },
                    dataType:'html' //設定網頁回應的是 html 格式
```

```
        }).done(function(data){
            if(data == "yes"){
                //此帳號已存在
                alert('此帳號已存在');
                $("#register button").attr('disabled', true);
            }
            else{
                //此帳號不存在
                $("#register button").attr('disabled', false);
            }
        }).fail(function(jqXHR, textStatus, errorThrown){
            alert("有錯誤產生，請看 console log");
            console.log(jqXHR, responseText);
        });
    }
});
$("#register").submit(function(){
    //確認兩次輸入的密碼是否相同
    if($("#password").val() != $("#checkpassword").val()){
        //兩者不同
        alert("密碼錯誤請重新確認");
    }
    else{
        //兩者相同
        $.ajax({
            type:"POST", //表單傳送的方式
            url:"add.php", //目標給哪個檔案
            data:{ //為要傳過去的資料，使用物件方式呈現，因為變
            數 key 值為英文的關係所以用物件方式傳送
```

```
                    'un':$("#username").val(),
                    'pw':$("#password").val(),
                    'n':$("#name").val(),
                },
                dataType:'html' //設定網頁回應的是 html 格式
            }).done(function(data){
                console.log(data);
                    if(data == "yes"){
                        alert("註冊成功，請按確認返回登入頁面");
                        window.location.href = "index.php";
                    }
                    else{
                        alert("註冊失敗");
                    }
            }).fail(function(jqXHR, textStatus, errorThrown){
                alert("有錯誤產生，請看 console log");
                console.log(jqXHR, responseText);
            });
        }
        return false;
    });
});
</script>
```

4.1.3

```
//操控 Panda 行駛之方向鍵
//使用 label 可以直接點擊圖片傳值
<form id="selection" method="get">
    <label><input type="radio" name="control" /><img src="empty.png"
    style="width:10px;height:10px;"></label>
```

```
<label><input type="radio" name="control" value="e" /><img src="on.png"
style="width:20px;height:20px;"></label>
<label><input type="radio" name="control" /><img src="empty.png"
style="width:20px;height:20px;"></label>
<label><input type="radio" name="control" /><img src="empty.png"
style="width:20px;height:20px;"></label>
<label><input type="radio" name="control" /><img src="empty.png"
style="width:30px;height:30px;"></label>
<label><input type="radio" name="control" value="q" /><img src="off.png"
style="width:20px;height:20px;"></label><br>
<label><input type="radio" name="control" /><img src="empty.png"
style="width:50px;height:50px;"></label>
<label><input type="radio" name="control" value="w" /><img
src="zumoup2.png" style="width:50px;height:50px;"></label><br>
<label><input type="radio" name="control" value="a" /><img
src="zumoleft2.png" style="width:50px;height:50px;"></label>
<label><input type="radio" name="control" value="x" /><img src="zimoc.png"
style="width:50px;height:50px;"></label>
<label><input type="radio" name="control" value="d" /><img
src="zumoright2.png" style="width:50px;height:50px;"></label><br>
<label><input type="radio" name="control" /><img src="empty.png"
style="width:50px;height:50px;"></label>
<label><input type="radio" name="control" value="s" /><img
src="zumodown2.png" style="width:50px;height:50px;"></label>
</form>
//隱藏 radio 選項
<style type="text/css">
    [type=radio] {
        position: absolute;
```

```
            opacity: 0;
            width: 0;
            height: 0;
        }
</style>
//接收 MQTT 傳的值
<script>
        $('#selection').change(function() {
            var selected_value = $("input[name='control']:checked").val();
            $.ajax( {
                //告訴程式表單要傳送到哪裡
                url: "new.php",
                dataType : "html",
                //使用 POST 方法
                method: "POST",
                cache: false,
                data: { selected_value : selected_value },
                success: function(response){
                    var test1 = "<p>"+response+"</p>";
                    $("h3").html(test1);
                }
            });
        });
</script>
//操控非接觸且非侵入式電梯控制系統之上下鍵
<form id="selection1" method="get">
        <label><input type="radio" name="control" /><img src="empty.png"
        style="width:50px;height:50px;"></label>
        <label><input type="radio" name="control1" value="t" /><img
```

```
        src="zumoup2.png" style="width:50px;height:50px;"></label><br>
        <label><input type="radio" name="control" /><img src="empty.png"
        style="width:50px;height:50px;"></label>
        <label><input type="radio" name="control1" value="b" /><img
        src="zumodown2.png" style="width:50px;height:50px;"></label>
</form>
//接收 MQTT 傳的值
<script>
        $('#selection1').change(function() {
                var selected_value = $("input[name='control1']:checked").val();
                $.ajax( {
                        //告訴程式表單要傳送到哪裡
                        url: "new1.php",
                        dataType : "html",
                        //使用 POST 方法
                        method: "POST",
                        cache: false,
                        data: { selected_value : selected_value },
                        success: function(response){
                                var test1 = "<p>"+response+"</p>";
                                $("h3").html(test1);
                        }
                });
        });
</script>
//查看目前樓層以及查看非接觸且非侵入式電梯控制系統剩餘電量之按鍵
<form id="message_form" method="POST" style="display:inline">
        <table id="message_table">
                <tr>
```

```
                <td>
                    <input type="button" id="submit_message" value="目前樓層"
                    onClick="messageGo();" />
                    <input type="button" id="submit_message1" value="目前電梯按
                    鈕電量" onClick="messageGo1();" />
                </td>
            </tr>
        </table>
</form>
//查看目前樓層
<script>
    function messageGo() {
        $.ajax({
        //告訴程式表單要傳送到哪裡
            url: "ll.php",
            //使用 POST 方法
            type: "POST",
            //接收回傳資料的格式，在這個例子中只要是接收 true 就可以了
            dataType: 'json',
            //傳送失敗則跳出失敗訊息
            error: function() {
            //資料傳送失敗後就會執行這個 function 內的程式，可以在這裡寫
            入要執行的程式
                alert("失敗");
            },
                //傳送成功則跳出成功訊息
                success: function() {
                    $.getJSON("ll.php", function(data) {
                        alert(data.floor);
```

```
                    })
                }
            });
        };
</script>
//查看非接觸且非侵入式電梯控制系統剩餘電量
<script>
    function messageGo1() {
        $.ajax({
        //告訴程式表單要傳送到哪裡
            url: "V.php",
            //使用 POST 方法
            type: "POST",
            //接收回傳資料的格式，在這個例子中只要是接收 true 就可以了
            dataType: 'json',
            //傳送失敗則跳出失敗訊息
            error: function() {
            //資料傳送失敗後就會執行這個 function 內的程式，可以在這裡寫
            入要執行的程式
                alert("失敗");
            },
            //傳送成功則跳出成功訊息
            success: function() {
                $.getJSON("V.php", function(data) {
                    alert(data.V);
                })
            }
        });
    };
```

```
</script>
```

4.1.4

```
<?php header('refresh: 30;url="photo.php"') ?>
<?php
	$folder = "Pic";
	$files = array();
	$handle = opendir($folder);
	while(($file=readdir($handle))!==false){
		if($file!='.' && $file!='..'){
			$hz=strstr($file,".");
			if($hz==".png") $files[] = $file;
		}
	}
	$filesTemp = array();
	if($files){
		foreach($files as $k=>$v){
			array_unshift($filesTemp,$v);
		}
	}
	$count = 0;
	if($files){
		foreach($filesTemp as $k=>$v){
			echo'<p style="display:inline-block">';
			if($count == 9) break;
			echo '<img width=320 height=240 src="'.$folder.'/'.$v.'">';
			echo substr($v,0,-4);
			if($count == 3 && $count == 6);
			$count++;
			echo'</p>';
```

```
            }
        }
?>
```

4.1.6.1

```
<?php
//phpMQTT.php 為官方所釋出的 opensource，專門設計給 php，讓其可以進行 mqtt 連線。
require("phpMQTT\phpMQTT.php");
//下方 IP 為物聯網技術中心之 broker IP
$server = "140.129.18.218";          // change if necessary
$port = 1883;                        // change if necessary
$username = "";                      // set your username
$password = "";                      // set your password
//這個部分非常重要，這是在 python 版本的 mqtt 不會出現的部分，每一個 publisher、subscriber 之 client_id 都必須要不同，否則會一直出現連線錯誤之情況。
$client_id = "phpMQTT-publisher"; // make sure this is unique for connecting to sever - you could use uniqid()
$mqtt = new Bluerhinos\phpMQTT($server, $port, $client_id);
 //以下程式碼為網頁按鈕介面傳值過來此頁面時才會出發，並進行一次 publish 之動作。
if($_SERVER['REQUEST_METHOD'] == 'POST')
    {
          $value = filter_input(INPUT_POST, "selected_value");
          if (isset($value))
          {
        if($mqtt->connect(true, NULL, $username, $password)) {
          $mqtt->publish("pub.140.129.18.218/controlPanda", $value);
          $mqtt->close();
        } else {
```

```
            echo "Time out!\n";
        }
            }
    }
?>
```

4.1.6.2

```
<?php
require("phpMQTT\phpMQTT.php");
$server = "140.129.18.218";          // change if necessary
$port = 1883;                                 // change if necessary
$username = "";                               // set your username
$password = "";                               // set your password
$client_id = "phpMQTT-publisher11"; // make sure this is unique for connecting to
sever - you could use uniqid()
$mqtt = new Bluerhinos\phpMQTT($server, $port, $client_id);
if($_SERVER['REQUEST_METHOD'] == 'POST')
    {
        $value = filter_input(INPUT_POST, "selected_value");
        if (isset($value))
        {
            if($mqtt->connect(true, NULL, $username, $password)) {
                //echo json_encode($value);
                $mqtt->publish("pub.140.129.18.218/ele/control", $value);
                $mqtt->close();
            } else {
                echo "Time out!\n";
            }
        }
    }
```

```
?>
```

4.1.6.3

```
<?php
ini_set("max_execution_time", "300");
//phpMQTT.php 為官方所釋出的 opensource，專門設計給 php，讓其可以進行 mqtt 連線。
require("phpMQTT/phpMQTT.php");
//方便測試使用，以便我們可以輕鬆看出是否有真的接收到正確的值。
$data = array("floor" =>5);
$data2 = array("V" =>3.3);
$level = "4";
$V = "3.3";
$count = 0;
//下方 IP 為物聯網技術中心之 broker IP
$server = "140.129.18.218";          // change if necessary
$port = 1883;                        // change if necessary
$username = "";                      // set your username
$password = "";                      // set your password
//這個部分非常重要，這是在 python 版本的 mqtt 不會出現的部分，每一個 publisher、subscriber 之 client_id 都必須要不同，否則會一直出現連線錯誤之情況。
$client_id = "phpMQTT-subscriber-level"'; // make sure this is unique for connecting to sever - you could use uniqid()
$mqtt = new Bluerhinos\phpMQTT($server, $port, $client_id);
set_time_limit(0);
if(!$mqtt->connect(true, NULL, $username, $password)) {
        exit(1);
}
$topics['pub/level'] = array("qos" => 0, "function" => "procmsg");
$mqtt->subscribe($topics, 0);
```

```
while($mqtt->proc()){
        global $level;
           $file = fopen("ll.txt","r+"); //開啟檔案
           fwrite($file,$level);
        fclose($file);
        $fileV = fopen("V.txt","r+"); //開啟檔案
           fwrite($fileV,$V);
        fclose($fileV);
}
$mqtt->close();
function procmsg($topic, $msg){
        global $count;
          //由於我們一次傳送兩個值，一個為目前樓層，一個目前樓層電梯控制系統剩餘電量，
      //所以多使用一個 count 來作判別。
        if($count % 2 == 0){
                  //將透過 mqtt 所接收的值紀錄至陣列，以供其他頁面調用
              $temp = (string)$msg;
              global $level;
              $level = substr($temp,0,1);
              global $data;
              $data = array("floor" =>$level);
              $count += 1;
        }else{
             //將透過 mqtt 所接收的值紀錄至陣列，以供其他頁面調用
              $temp = (string)$msg;
              global $V;
              $V = substr($temp,0,3);
              global $data2;
```

```
            $data2 = array("V" =>$V);
            $count -= 1;
?>
```

4.2.1.2.1

Python 程式碼

```
import sys, os, time, signal
import paho.mqtt.client as mqtt
import json
client = None
mqtt_looping = False
buttonValue = "x"
#Topic 一定要是唯一，否則將會跟其他的 mqtt 訊號相互干擾的問題。
TOPIC_ROOT = "pub.140.129.18.218/ele/control"
def on_connect(mq, userdata, rc, _):
      # subscribe when connected.
      mq.subscribe(TOPIC_ROOT + '/#')
def on_message(mq, userdata, msg):
  #將 Web/App 使用者透過網頁所傳的值紀錄至 LogF.txt
  #此值之後會再透過包含 CoAP 之 Java 程式傳送至非侵入且非接觸式電梯控制系統
      fp = open('LogF.txt', 'w+')
      s = str(msg.payload)[2]
      if s is "t":
         print ("1", file =fp)
      elif s is "b":
         print ("0", file =fp)
def mqtt_client_thread():
      global client, mqtt_looping
      client_id = "" # If broker asks client ID.
```

```
    client = mqtt.Client(client_id=client_id)
    # If broker asks user/password.
    user = ""
    password = ""
    client.username_pw_set(user, password)
    client.on_connect = on_connect
    client.on_message = on_message
    try:
        client.connect("140.129.18.218", 1883)
    except:
        print ("MQTT Broker is not online. Connect later.")
    mqtt_looping = True
    print ("Looping...")
    cnt = 0
    while mqtt_looping:
        client.loop()
        cnt += 1
        if cnt > 20:
            try:
                client.reconnect() # to avoid 'Broken pipe' error.
            except:
                time.sleep(1)
            cnt = 0
    print ("quit mqtt thread")
    client.disconnect()
def stop_all(*args):
    global mqtt_looping
    mqtt_looping = False
if __name__ == '__main__':
```

```
        signal.signal(signal.SIGTERM, stop_all)
        signal.signal(signal.SIGQUIT, stop_all)
        signal.signal(signal.SIGINT,    stop_all)    # Ctrl-C
        mqtt_client_thread()
        print ("exit program")
        sys.exit(0)
```

4.2.1.2.2

```
import paho.mqtt.client as mqtt
import time
import json
import random
MQTT_SERVER = "140.129.18.218"
MQTT_PORT = 1883
MQTT_ALIVE = 60
#每一個 mqtt 連線之 Topic 都要不同，否則將會互相干擾
MQTT_TOPIC1 = "pub/level"
mqtt_client = mqtt.Client()
mqtt_client.connect(MQTT_SERVER, MQTT_PORT, MQTT_ALIVE)

#將目前所在樓層、目前樓層的非接觸且非侵入式電梯控制系統之剩餘電量傳送
至 server 端
while True:
    fr = open("level.txt", mode = 'r')
    a=fr.read(1)
    fr.close()
    fr2 = open("V.txt", mode = 'r')
    b=fr2.read(3)
    fr2.close()
    payload1 = a
```

```
    payload2 = b
    #print(dataChnId1 + " : " + str(t0))
    mqtt_client.publish(MQTT_TOPIC1, payload1, qos=0)
    mqtt_client.publish(MQTT_TOPIC1, payload2, qos=0)
    time.sleep(10)
```

4.1.2.2.3

```
import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.sql.Time;
import java.util.Scanner;
import org.eclipse.californium.core.CoapClient;
import org.eclipse.californium.core.CoapResponse;
import java.util.concurrent.TimeUnit;
public class CoAPClient {
    public static void main(String[] args) {
        String push = "-1";              //先將push給裝置的值設為”-1”
        String batry = "-1";             //先將讀取電量變數的值設為”-1”
        while(true){//與該台Esp32連接的內網及port號
        String uri= "coap://192.168.4.1:5683/.well-known/core";
        先創各個CoapClient的物件client來供以下感測器做讀取作用
        CoapClient client = new CoapClient();
        CoapClient client1 = new CoapClient();//光敏1
        CoapClient client2 = new CoapClient();//光敏2
        CoapClient client3 = new CoapClient();//紅外線1
        CoapClient client4 = new CoapClient();//紅外線2
```

```
CoapClient client5 = new CoapClient();//電量
client.setURI("coap://192.168.4.1:5683/122/200");//致動器
client.setTimeout(3000);                    //timeout
client1.setURI("coap://192.168.4.1:5683/121/0/3");//光敏1
client2.setURI("coap://192.168.4.1:5683/121/1/3");//光敏2
client3.setURI("coap://192.168.4.1:5683/121/2/3");//紅外線1
client4.setURI("coap://192.168.4.1:5683/121/3/3");//紅外線2
client5.setURI("coap://192.168.4.1:5683/121/4/3");//電量
FileReader fr=null;
try {//創一個文字檔存目前收到之訊息
      fr = new FileReader("LogF.txt");
} catch (FileNotFoundException e2) {
      e2.printStackTrace();
}
CoapResponse response = null;
BufferedReader br = new BufferedReader(fr);
try {
      String temp = br.readLine();
      if(!temp.equals(push)){
            //將讀取之文字put進Esp32以讓致動器作動
            response = client.put(temp, 0);
            push = temp;
      }
} catch (IOException e1) {
      e1.printStackTrace();
}
try {
      br.close();
      fr.close();
```

```
            } catch (IOException e) {
                e.printStackTrace();
            }
            CoapResponse response_sensor5 = null;
            response_sensor5 = client5.get();
            if (response_sensor5!=null) {
                batry = response_sensor5.getResponseText();
                System.out.println(batry);
            }else{
                System.out.println("battery is error");
            }
            FileWriter ir=null;
            try {
                ir = new FileWriter("V.txt");//創一個文字檔以存取電量
                ir.write(batry);
            } catch (IOException e1) {
                e1.printStackTrace();
            }
            try {
                ir.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
            }
        }
}
```

4.2.2.1.1

```
import cv2 as cv
import time
```

```
#檔名的一部分，以及副檔名
video_name = 'pic_'
file_type = '.png'
#載入模型
net = cv.dnn.readNet('models/face-detection-retail-0004.xml',
'models/face-detection-retail-0004.bin')
#我們選擇 openVINO 所公布的 open source : face-detection-retail-0004
#設定要使用的目標裝置
net.setPreferableTarget(cv.dnn.DNN_TARGET_MYRIAD)
#一般在樹梅派上都是使用 index=0的相機
cap = cv.VideoCapture(0)
#取得相機的資訊:
def capProperties():
    print("[info] W, H, FPS")
    print(cap.get(cv.CAP_PROP_FRAME_WIDTH))
    print(cap.get(cv.CAP_PROP_FRAME_HEIGHT))
    print(cap.get(cv.CAP_PROP_FPS))
#調用方法，並設定
capProperties()
cap.set(3,320)
cap.set(4,240)
# 設定擷取影像的尺寸大小
#從這個時間點開始記錄所過的時間
start = time.time()
#紀錄一開始為零張圖片
frames = 0
# 使用 XVID 編碼
fourcc = cv.VideoWriter_fourcc(*'XVID')
network_width = 160
```

```
network_height = 120
def preprocess(source_cap):
    resized_cap = cv.resize(source_cap,(network_width,network_height))
    return resized_cap
#傳送影片時測試用，後來發現使用 WenRTC 更快
# 建立 VideoWriter 物件，輸出影片至 output.avi
# FPS 值為 20.0，解析度為 640x360
#out = cv.VideoWriter('output.avi', fourcc, 20.0, (640, 360))
#利用值來進行半段
picPrev = False
picNow = False
video_counter = 0
test = 0
while(cap.isOpened()):
    # 使用相機，並捕捉每一楨相片
    ret, frame = cap.read()
    tmp = test
    #preprocess
    resized = preprocess(frame)
    # Prepare input blob and perform an inference
    blob = cv.dnn.blobFromImage(resized, size=(300, 300), ddepth=cv.CV_8U)
    net.setInput(blob)
    out = net.forward()
    # Draw detected faces on the frame
    for detection in out.reshape(-1, 7):
     #作為判斷
        picPrev = picNow
        confidence = float(detection[2])
        xmin = int(detection[3] * resized.shape[1])
```

```
            ymin = int(detection[4] * resized.shape[0])
            xmax = int(detection[5] * resized.shape[1])
            ymax = int(detection[6] * resized.shape[0])
            cv.imshow('frame',frame)
            if confidence > 0.5:
                test = test + 1
                #判斷
                picNow =True
                cv.rectangle(frame, (xmin*2, ymin*2), (xmax*2, ymax*2), color=(0,
255, 255))
                #判斷
                if picPrev == False:
            #紀錄目前時間，之後會用來當作檔名
                    timestr=time.strftime('%Y-%m-%d_%H-%M-%S',
time.localtime(time.time()))
        #python 的字串直接相加代表第二個字串 append 在第一個後面
                    save_name = timestr + file_type
            #將這一楨寫入檔案
                    cv.imwrite(save_name, frame)
                    print('writing to ' + save_name)
                    video_counter = video_counter + 1
        if tmp == test:
            picNow = False
            if picPrev == True:
                timestr=time.strftime('%Y-%m-%d_%H-%M-%S',
time.localtime(time.time()))
                save_name = timestr + file_type
                cv.imwrite(save_name, frame)
                print('writing to ' + save_name)
```

```
                video_counter = video_counter + 1
        frames+=1
        if ret == True:
            # 寫入影格
            if cv.waitKey(1) & 0xFF == ord('q'):
                # 釋放所有資源
                cap.release()
                cv.destroyAllWindows()
                break
        else:
            break
```

4.2.2.1.2

ftpAuto.bat :

```
@echo off
ftp -s:D:ftpAuto.txt
timeout /t 5
:while
if 1 equ 1 (
    ftp -s:D:ftpAuto.txt
    timeout /t 30
    goto :while
)
```

//timeout /t 5代表等五秒、timeout /t 30代表等三十秒

//ftp -s:D:ftpAuto.txt 代表開啟 FTP，並將其每一行依序輸入 ftp>>後面。

//使用 goto :while 的原因是 bat 裡面沒有 while 指令，因此利用 goto 達成 while 效果。

ftpAuto.txt :

```
open 192.168.43.79
pi
```

```
pi
prompt off
lcd C:\xampp\htdocs\Pic
cd /home/pi/Desktop
mget *.png*
mdelete *.png
close
quit
//open 後面接要連結的 IP
//接著輸入帳號一行、密碼一行
//prompt 代表互動模式，有 off、on 可以選擇，我們這裡選擇 off
//lcd 代表進入本機端的某個檔案路徑、cd 代表進入連接端的某個檔案路徑
//mget *.png*代表本機端取得連接端所有副檔名為.png 的檔案
//mdelete *.png 代表連接端刪除所有副檔名為.png 的檔案
//close 關閉這一次 ftp 連線
//quit 可以離開 ftp>>
```

4.2.3

```
import sys, os, time, signal
import paho.mqtt.client as mqtt
import json
import serial
import time
import threading
#馬達初始設定
MotorL_COM_PORT = '/dev/ttyUSB1'       #左側馬達序列阜
MotorR_COM_PORT = '/dev/ttyUSB0'        #右側馬達序列阜
BAUD_RATE = 57600       #baud_rate
ser_L = serial.Serial(MotorL_COM_PORT,BAUD_RATE,timeout=0.2)
```

```
#設定 COM 傳輸參數
ser_R = serial.Serial(MotorR_COM_PORT,BAUD_RATE,timeout=0.2)
thread_state=0
ser_L.write(serial.to_bytes([0x02,0x00,0xc4,0xc6]))     #設定為 PC 控制速度模式
ser_R.write(serial.to_bytes([0x02,0x00,0x04,0xc6]))     #設定為 PC 控制速度模式
#mqtt 初始設定
client = None
mqtt_looping = False
#之後要訂閱的主題
TOPIC_ROOT = "pub.140.129.18.218/controlPanda"
#用來取得 payload 的全域型變數
global_var = 'x'
#mqtt  方法
def on_connect(mq, userdata, rc, _):
    # subscribe when connected.
    mq.subscribe(TOPIC_ROOT + '/#')
def on_message(mq, userdata, msg):
    global global_var
    #每種程式語言所使用 mqtt 傳出的格式不同，這裡是調整格式，取得需要的
值
    global_var = str(msg.payload)[2]
def mqtt_client_thread():
    global client, mqtt_looping, GlobalMsg
    client_id = "" # If broker asks client ID.
    client = mqtt.Client(client_id=client_id)
    # If broker asks user/password.
    user = ""
    password = ""
    client.username_pw_set(user, password)
```

```
client.on_connect = on_connect
client.on_message = on_message
try:
    #物聯網技術中心所架設的 broker 之 IP address
    client.connect("140.129.18.218", 1883)
except:
    print ("MQTT Broker is not online. Connect later.")
mqtt_looping = True
print ("Looping...")
cnt = 0
while mqtt_looping:
    client.loop()
    str1 = 'x'
    str1 = global_var
    print(str1)
    if str1 =='w':      #直行
        motor('l','g',80)
        motor('r','g',-80)
    elif str1 =='s':   #後退
        motor('l','g',-80)
        motor('r','g',80)
    elif str1 =='a':   #左轉
        motor('l','g',-20)
        motor('r','g',-20)
    elif str1 =='d':   #右轉
        motor('l','g',20)
        motor('r','g',20)
    elif str1 =='x':   #停止
        motor('l','g',0)
```

```
                motor('r','g',0)
            elif str1 =='q':   #關閉
                motor('l','q',0)
                motor('r','q',0)
            elif str1 =='e':   #啟動
                motor('l','s',0)
                motor('r','s',0)
            cnt += 1
            if cnt > 20:
                try:
                    client.reconnect() # to avoid 'Broken pipe' error.
                except:
                    time.sleep(1)
                cnt = 0
        print ("quit mqtt thread")
        client.disconnect()
def stop_all(*args):
        global mqtt_looping
        mqtt_looping = False
def motor_check():              #檢測馬達 讀取電池電壓
        while thread_state==1:
            ser_L.write(serial.to_bytes([0x80,0x00,0x80]))
            ser_R.write(serial.to_bytes([0x80,0x00,0x80]))
            time.sleep(0.5)
            while ser_L.in_waiting:
                res = ser_L.read(4)
                res_list = list(res)
                if res_list[0] == 225:
                    voltage = res_list[2]
```

```
                print("設備電壓為:",voltage)
            #print(res)
            #print(list(res))
#馬達值的傳輸
def motor(a,b,c):
#a::左輪或右輪(左為1,右為2)      b:: 0:stop   1:向前   2:向後     3:向左     4:向右
s:啟動      #q:停止      g:設定轉速     c::轉速(若無轉速設定直接下0即可)
    c=int(c)            #str -> int
    c=c/6000*16384     #依照輪鼓馬達 datasheet 計算是換算
    c=int(c)            #取整數
    if c<0:             #補數
        c=65536-abs(c)
    if b == '0' :
        c0=6             #c0 數據地址
        c1=0             #c1 數據高八位
        c2=0             #c2 數據低八位
        c3=6             #c3 數據較驗合(c0+c1+c2)*取低八位
    elif b == '1':
        #ser.write(serial.to_bytes([6,0,136,142]))
        c0=6
        c1=0
        c2=136
        c3=142
    elif b == '2':
        #ser.write(serial.to_bytes([6, 255, 120, 125]))
        c0=6
        c1=255
        c2=120
        c3=125
```

```
        elif b == 's':
            #ser.write(serial.to_bytes([0,0,1,1]))
            c0=0
            c1=0
            c2=1
            c3=1
        elif b == 'q':
            #ser.write(serial.to_bytes([0,0,0,0]))
            c0=0
            c1=0
            c2=0
            c3=0
        elif b == 'g':
            c0=6
            c1=c>>8
            c2=c-(c1<<8)
            c3=c0+c1+c2
            c3=c3&255
        if a =='l':
            ser_L.write(serial.to_bytes([c0,c1,c2,c3]))
        elif a=='r':
            ser_R.write(serial.to_bytes([c0,c1,c2,c3]))
#main 函數
if __name__ == '__main__':
        signal.signal(signal.SIGTERM, stop_all)
        signal.signal(signal.SIGQUIT, stop_all)
        signal.signal(signal.SIGINT,    stop_all)    # Ctrl-C
        mqtt_client_thread()
        print ("exit program")
```

```
        sys.exit(0)
```

4.3

```
--以下為”base64.lua” base64.lua 為 Esp32 開啟 ap mode 的程式
apCfg = {
   auth = wifi.WPA2_PSK,
   channel = 6,
   ssid = "A3-0401CCC/F+W/6#S;6,F/F3T1&#T0S",    --設 ssid
   pwd = "24:0a:2049"                                    --設置 pwd
}
wifi.mode(wifi.STATIONAP)
wifi.start()
wifi.ap.config(apCfg)
以下為”init.lua”程式碼，init.lua 為 lua 程式開機就會執行的檔案名稱
dofile("base64.lua")              --base64.lua 為開啟 ap mode 的程式
--設定各個 pin 腳至參數內
Act1 = 26                         --設置致動器 1(pin 腳 26)為 Act1
Act2 = 27                         --設置致動器 2(pin 腳 27)為 Act2
s1 = 34                           --設置光敏感測器 1(pin 腳 34)為 s1
s2 = 33                           --設置光敏感測器 2(pin 腳 33)為 s2
hw1 = 39                          --設置紅外線感測器 1(pin 腳 39)為 hw1
hw2 = 35                          --設置紅外線感測器 2(pin 腳 35)為 hw2
battery = 32                      --設置電池電量回傳(pin 腳 32)為 battery
--設定 input,output 至會用到之 pin 腳內
gpio.config({gpio={Act1},dir=gpio.OUT})
gpio.config({gpio={Act2},dir=gpio.OUT})
gpio.config({gpio={s1},dir=gpio.IN})
gpio.config({gpio={s2},dir=gpio.IN})
gpio.config({gpio={hw1},dir=gpio.IN})
gpio.config({gpio={hw2},dir=gpio.IN})
```

```
gpio.config({gpio={battery},dir=gpio.IN})
adc.setwidth(adc.ADC1,10)              --設置 ADC1 內所有 bits 數為 10
adc.setup(adc.ADC1,5,adc.ATTEN_11db) adc.setup(adc.ADC1,6,adc.ATTEN_11db)
adc.setup(adc.ADC1,3,adc.ATTEN_11db) adc.setup(adc.ADC1,4,adc.ATTEN_6db)
adc.setup(adc.ADC1,7,adc.ATTEN_11db)
gpio.write(Act1,1)                      --先將 Act1,Act2 的 output 設為 1
gpio.write(Act2,1)
check=-1                                --此行為為了檢查判斷式
function set_elevator(payload)          --esp32 透過 coap 會進此 function
   if (payload == "1") then                --payload=1 時
      gpio.write(Act1,0)                    --將 Act1 的 ouput 設為 0
      tmr.create():alarm(500, tmr.ALARM_SINGLE, function()
            gpio.write(Act1,1)
      end)              --延遲 0.5 秒後再將 Act1 的 output 設為 1
      check=1
   else
      if (payload == "0") then              --payload=1 時
         gpio.write(Act2,0)                   --將 Act1 的 ouput 設為 0
         tmr.create():alarm(500, tmr.ALARM_SINGLE, function()
            gpio.write(Act2,1)
         end)          --延遲 0.5 秒後再將 Act2 的 output 設為 1
         check=0
      end
   end
   return check
end
--以下 function 為紅外線感測使致動器啟動並用光敏感測器做節電作用
tmr.create():alarm(1000, tmr.ALARM_AUTO, function()
      a1 = adc.read(adc.ADC1,5)          --將 ADC1 通道 5 的電壓讀進 a1
```

```
a2 = adc.read(adc.ADC1,6)          --將 ADC1 通道 6 的電壓讀進 a2
h1 = adc.read(adc.ADC1,3)          --將 ADC1 通道 3 的電壓讀進 h1
h2 = adc.read(adc.ADC1,7)          --將 ADC1 通道 7 的電壓讀進 h2
if(h1 < 1000 && a1 < 1000) then--若 h1 小於 1000 同時按鈕沒按過
    gpio.write(Act1,0)             --將 Act1 的 ouput 設為 0
    tmr.create():alarm(500, tmr.ALARM_SINGLE, function()
        gpio.write(Act1,1)--延遲 0.5 秒後將 Act1 的 output 設為 1
    end)
    if(a1 < 1000)then
        gpio.write(Act1,0)         --若電梯按鈕按失敗則補按一次
        tmr.create():alarm(500, tmr.ALARM_SINGLE, function()
            gpio.write(Act1,1)
    end)
    else
    end
else
end
if(h2 < 1000 && a2 < 1000) then--若 h1 小於 1000 同時按鈕沒按過
    gpio.write(Act2,0)                 --將 Act2 的 ouput 設為 0
    tmr.create():alarm(500, tmr.ALARM_SINGLE, function()
        gpio.write(Act2,1)--延遲 0.5 秒後將 Act2 的 output 設為 1
    end)
    if(a2 < 1000)then                  --若電梯按鈕按失敗則補按一次
        gpio.write(Act2,0)
        tmr.create():alarm(500, tmr.ALARM_SINGLE, function()
            gpio.write(Act2,1)
    end)
    else
    end
```

```
        else
        end
end)
--以下 function 為測量電量之用
tmr.create():alarm(1000, tmr.ALARM_AUTO, function()--檢測電量
        bb = adc.read(adc.ADC1,4)
        bb = bb/1023*2.2
end)
--以下為 Esp32 與 Coap 協定連線相關 function--
deviceInfo = {
    flags = 0x00000001,
    classID = 0,
    deviceID = 120
}
function typeID(p)
    if p~=nil then
        deviceInfo.deviceID = p
    end
    return deviceInfo.deviceID
end
function flags(p)
    if p~=nil then
        deviceInfo.flags = p
    end
    return deviceInfo.flags
end
__cv = require "C4lab_CoAP210-ob"
__cv:startCoapServer(5683)
__cv:addActuator("/typeID","typeID")
```

```
__cv:addActuator("/flags","flags")


__cv:addActuator("/122/200","set_elevator")
__cv:addSensor("/121/0/3","a1")
__cv:addSensor("/121/1/3","a2")
__cv:addSensor("/121/2/3","h1")
__cv:addSensor("/121/3/3","h2")
__cv:addSensor("/121/4/3","bb")
__cv:postResource()
```