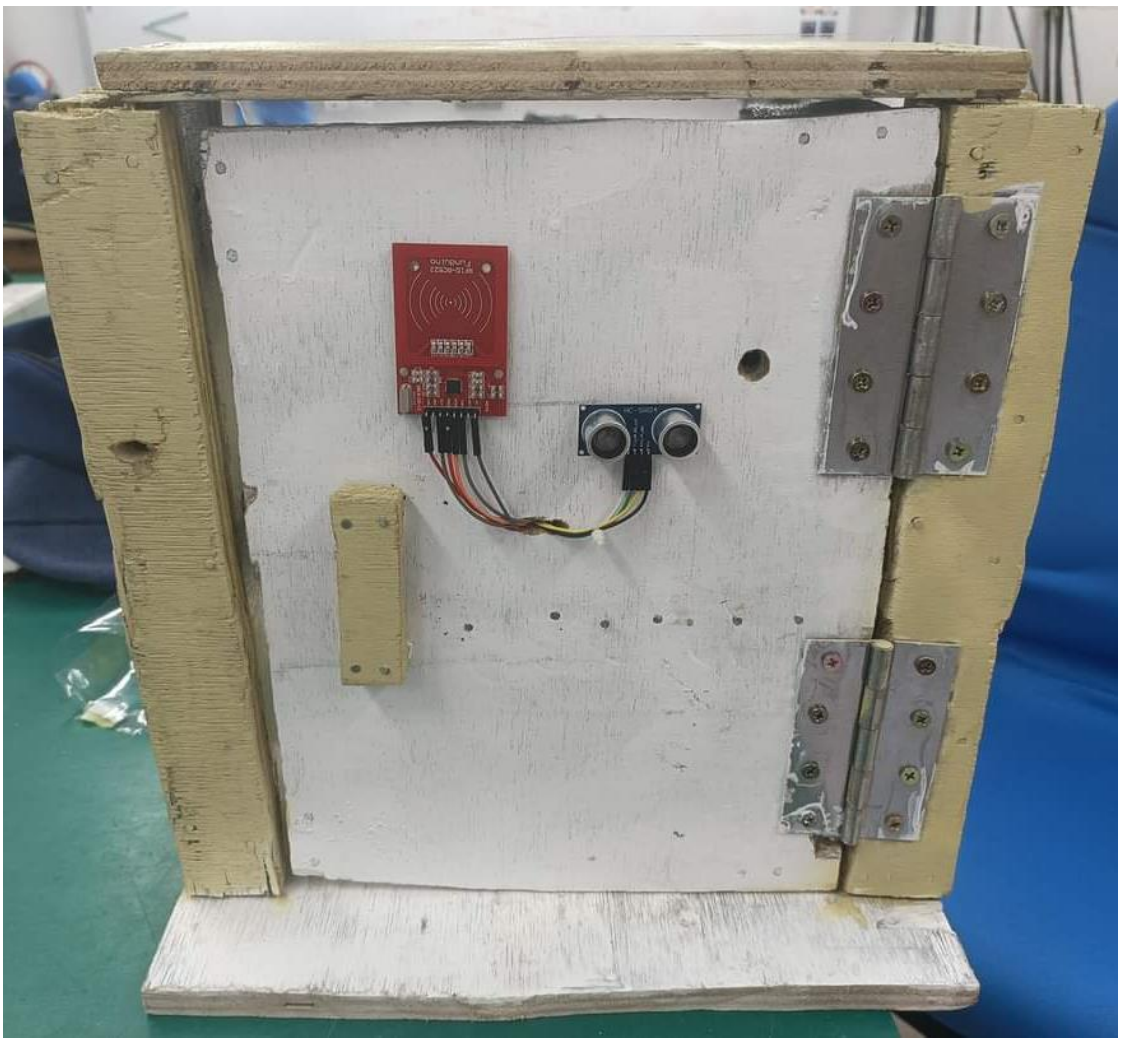
中華民國第 63 屆中小學科學展覽會

作品說明書



科　　別：工程學科（一）

組　　別：高級中等學校組

作品名稱：雲端智能防盜門鎖系統

關 鍵 詞：防盜鎖、雲端、人臉辨識

編號：

目錄

圖目錄

表目錄

摘要

闖空門是現今社會經常發生的事故，社會新聞上出現因住宅遭人入侵而導致錢財、隱私和生命危險的案件層出不窮。所以我們製作出雲端智能防盜門鎖系統，利用 Linkit 7697 與雲端及 WiFi 還有人臉辨識技術及各元件結合檢測是否有人入侵住宅與開鎖運用，另外還可拍攝照片後發送照片與訊息至屋主的手機，當特定情況時蜂鳴器會響起提醒鄰居及屋主或是嚇阻竊賊。

與傳統門鎖比起來我們的防盜門鎖系統更為安全，就算是被偷竊，還是能捕捉到嫌疑人的面貌以便在日後交給警察證據，也較不容易被解開，相比於傳統門鎖安全許多。

壹、 研究動機

闖空門是現今社會經常發生的事故，社會新聞上出現因住宅遭人入侵而導致錢財、隱私和生命危險的案件層出不窮，人們總是在事故發生後才知道自己的自我保護意識有多糟，如果沒有被入侵住宅就不會釀成損失錢財和隱私外流等事故。甚至有些入侵者會威脅人質生命好換取更多的錢，為了避免這些事情發生，故想出利用發送訊息及拍攝照片在屋主離開房子後偵測是否有人嘗試想進入屋內，或是屋主熟睡時也能第一時間醒來，蜂鳴器也可以驚嚇入侵者，讓對方無功而返。

貳、 研究目的

一、研究目的

根據圖 1，我們可以知道以「破壞門鎖」及「開鎖進入」的方式入侵住宅的案件，佔全體的 15.47%，因此我們便決定從改良門鎖的方向下手，為了達到「居住安全」這個目的，我們利用遠端及 RFID 與人臉開鎖，這個方式是要避免因鎖頭露在外頭而遭造破壞或解鎖，當有人試圖入侵時，會傳遞訊息及拍攝影像至主人手機。希望可藉由本次研究，打造更安全的雲端智能防盜門鎖系統。

（一）、利用 RFID 開鎖

（二）、當住宅被人侵入或嘗試侵入會發送通知訊息及照片

（三）、人臉辨識開鎖

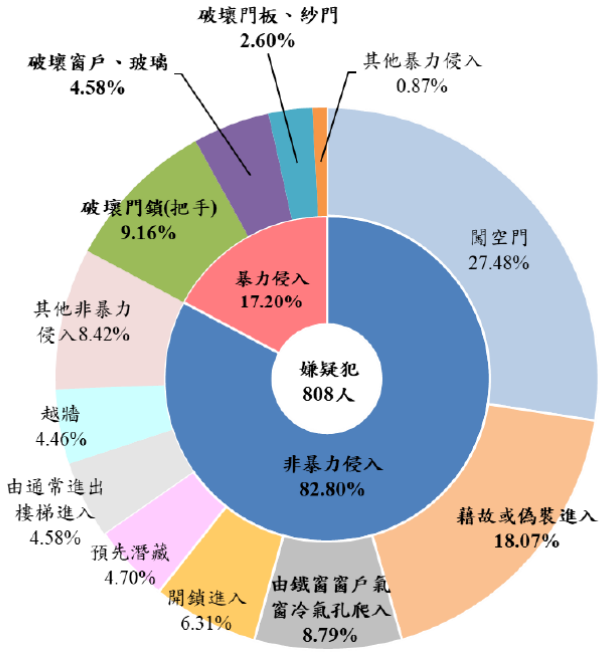


圖 1、侵入住宅竊盜嫌疑犯—按犯罪方法別分 109 年 1-10 月

二、文獻回顧

（一）、甚麼是互聯網？

「物聯網（Internet of Things，簡稱 IoT）是一個基於網際網路、傳統電信網等訊息承載體」（中興大學樂齡學習網，2023），讓所有物理對象都能利用網絡來去實現相互連通並且能夠被獨立尋址。IoT 的架構可分為為感知層、網路層與應用層，在 IoT 上，每個人都可以把現實世界的物體利用電子標籤與網路聯結。通過 IoT 可以對機器、設備及人員執行集中管理與控制，也可以對家庭設備、汽車進行遠端控制，應用範圍可說是非常地廣泛。IoT 的應用領域主要包括物流領域、醫療領域、智能環境領域與個人和社會領域等，都擁有很大的市場及前景。台灣的網通產品競爭力可以說是不錯的，但是在跨領域、跨產業整合的資料分析及萃取能力方向仍有缺陷需要補足。

IoT 常見的三種無線通訊技術可分為 Wi-Fi、藍芽及 ZigBee，本次研究所使用的無線通訊技術是 Wi-Fi。「Wi-Fi 的傳輸速率遠高於其他無線傳輸技術，但由於需包含 TCP/IP 協議的標準，因此通訊設備必須包含 MCU 與大量記憶體」（林文恭，2020），因此就提高了 Wi-Fi 的製作成本。

（二）、雲端智能防盜門鎖系統如何對付竊賊？

「竊賊幾乎都是以破壞工具行竊，例如電鑽、鐵橇、大型螺絲起子、鋼鋸、油壓剪等」(臺北市政府警察局通信隊，2023)，這些都是竊賊有可能使用的行竊工具，利用這些工具破壞門鎖後侵入盜竊。失竊率高的門鎖依次為喇叭鎖、輔助鎖及普通分離式多段鎖，這些門鎖防盜能力較差，很容易會被竊賊破壞後開啟，如果住宅只靠一個喇叭鎖鎖住大門，那麼極有可能會被盜竊。如果住戶安裝一個較安全的門鎖，像是材質堅固、破壞需耗費比較長時間的門鎖，通常竊賊就會因為需要花費比較龐大的時間去破壞，導致被警察逮捕的機率變高而不敢盜竊。而雲端智能防盜門鎖系統是會將整個系統都藏於門內部，但是不會深到 RC522-RFID 無法感應的程度，因為沒有裸露在外的部分，所以除非竊賊透過敲擊門體的方式判斷系統藏於何處並鑿洞，否則很難破壞到系統並入侵住宅。

參、 研究設備及器材

一、研究硬體

表 1、研究硬體

電腦	行動電源	Micro USB 傳輸線
行動載具	電路板	電烙鐵（架）
斜口鉗	吸錫器	美工刀
紙箱	A4 紙	剪刀
膠帶	泡棉膠	熱熔膠
彩色筆	水彩筆	水彩顏料

二、研究軟體

表 2、研究軟體

名稱	規格
Arduino	LinkIt 7697 環境、ESP32-CAM 環境
MCS Lite	MIT License
Line Notify	API

三、研究材料

表 3、研究材料

名稱	規格	數量
Linkit 7697 開發板	ARM Cortex-M4	1
Linkit 7697 擴充板	5V/3.3V/2.5V	1
RC522-RFID 板	13-26 mA/直流 3.3V	1
電磁鎖	直流 12V	1
繼電器	3V/5V	1
有緣蜂鳴器	S8550 三極管驅動	1
ESP32-CAM	內置 520 KB SRAM，外部 4MPSRAM	1
超音波測距模組	HC-SRO4	1
變壓器	DC12V	1
DC 電源接頭	外徑 5.5mm、內徑 2.5mm	1
杜邦線	公對母、母對母、公對公	12、9、3
焊錫	0.8mm	若干
電路板	80mm*60mm	1

肆、　研究過程或方法

一、研究架構與過程

本次專題我們利用 Linkit 7697 控制繼電器、電磁鎖、蜂鳴器、RFID-RC522、ESP32-CAM、超音波感測模組，透過這些模組元件來製造我們的防盜門鎖，我們利用 RFID-RC522 來控制 Linkit 7697，再讓 Linkit 7697 去控制其他的元件。我們透過使用雲端技術讓行動載具可以控制智慧門鎖系統，讓 RFID-RC522 可以接上電源並開始工作。此外，當有人想要不透過雲端或 RFID-RC522 強行開鎖侵入民宅，屆時，ESP32-CAM 會將此人的影像擷取下來，然後發送警告訊息讓屋主知曉並使用 LINE 將此人的照片傳送給屋主，雖然門鎖可能會因為不夠結實牢固，導致民宅遭人侵入進而發生財物損失的問題，但因為有透過超音波感測模組判斷後利用 ESP32-CAM 將入侵者的影像記錄下來，所以可以保留入侵者的影像當作證據待事發後報案使用。蜂鳴器我們當作警報器使用，當有人想利用強行開鎖的方式侵入室內，將會觸發蜂鳴器的電路使蜂鳴器開始鳴叫，並且會將入侵者的影像記錄下來傳給屋主，達到嚇阻入侵者及警告鄰居和屋主的目的，使屋主能夠盡可能的將損失降到最小。

為了完成我們要開發的系統，（如圖 2）：這次的研究主要可分成「內部電路」、「外部硬體」、「程式撰寫」，最後將所有研究成果結合，並透過不斷的測試、改良，使這個系統更加完整。

內部電路：為了讓系統能擁有雲端開鎖、磁扣開鎖、人臉辨識開鎖、照相以及新增刪除磁扣使用者的功能，我們參考了許多不同的文章，並且尋找能夠支援我們所有需求的開發版。起初我們將 LinkIt 7697 板與 NANO 擴充板結合後再利用麵包板與各個元件結合，但是為了將整個電路可以放在外部硬體內，我們在中途將麵包板換成了電路板去結合，並且將接電源與接地各焊一排，在製作程式撰寫完成前，我們是沒有加裝超音波感測模組的，但是有一位組員在程式撰寫完成後提議再加裝超音波感測模組下去，可以在有人破壞門鎖之前先拍攝照片並傳給主人，就可以更好的防止犯罪，所以才打算增加超音波感測模組。

外部硬體：為了能將整個系統裝進外殼內並且保持美觀，我們將殼做得有點厚度，

裡面留了許多空間用以架設系統電路，與此同時，在調整內部電路時也不斷改變外殼的內部構造，使整個系統在出現故障時能夠方便維修。

程式撰寫：我們將在課程學到的程式結合並且活用，像是超音波、有緣蜂鳴器、繼電器以及 MCS 都是我們在課程上有學習到的，但是我們在課程並沒有學習到利用 LINE Notify 來傳遞 LINE 訊息與 ESP32-CAM、RFID-RC522 及電磁鎖，於是我們參考了許多有關於這些元件的不同文章，將範例或文章上的程式理解後才將他們結合並且活用，要不然抓下來的程式僅僅是抄襲，不理解的話下次還是不會懂有關於這些程式運作的思路、原理。

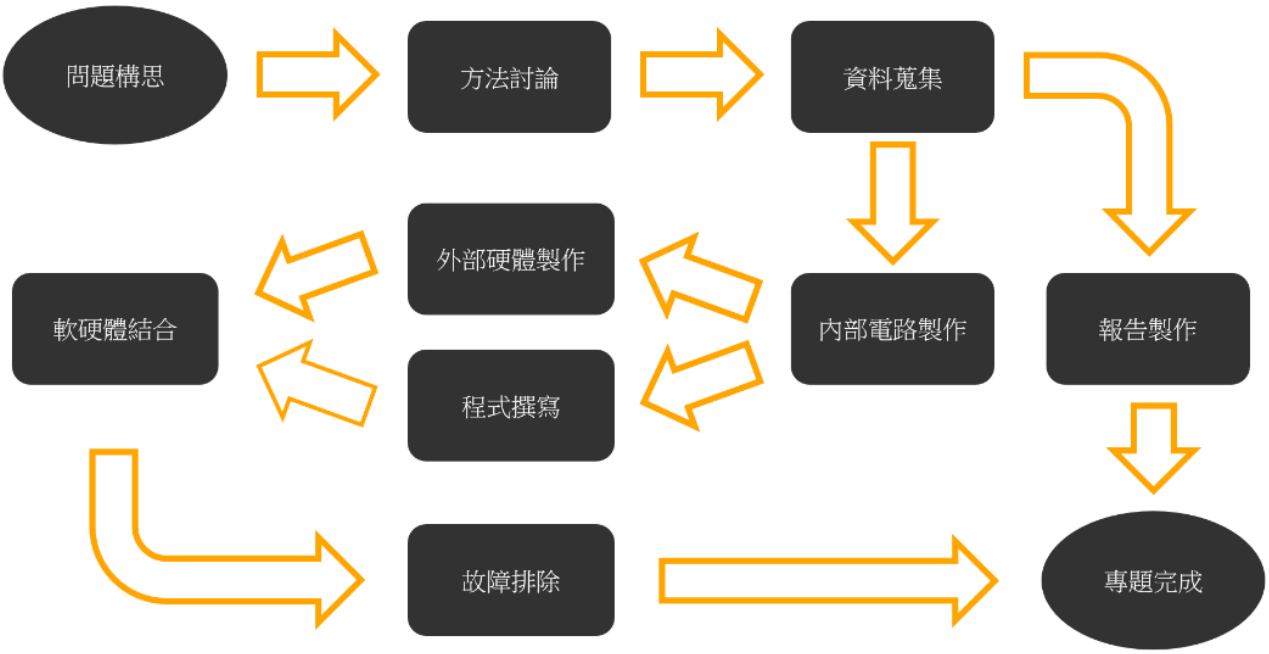
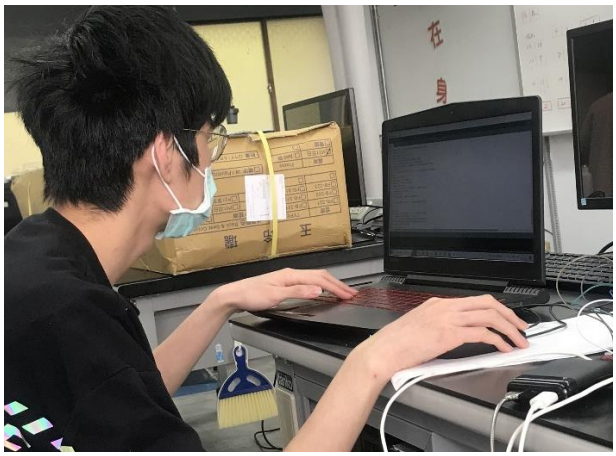
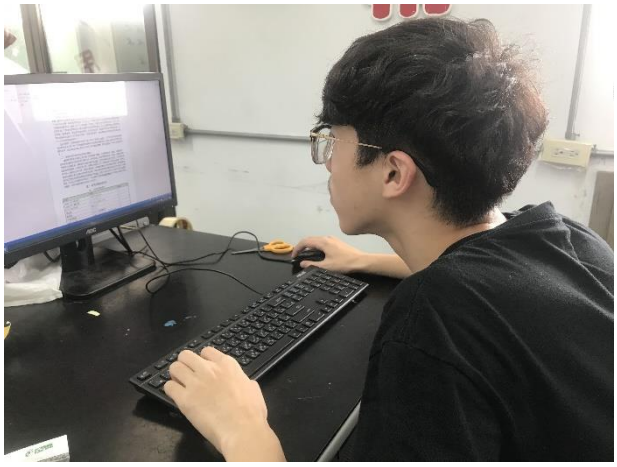
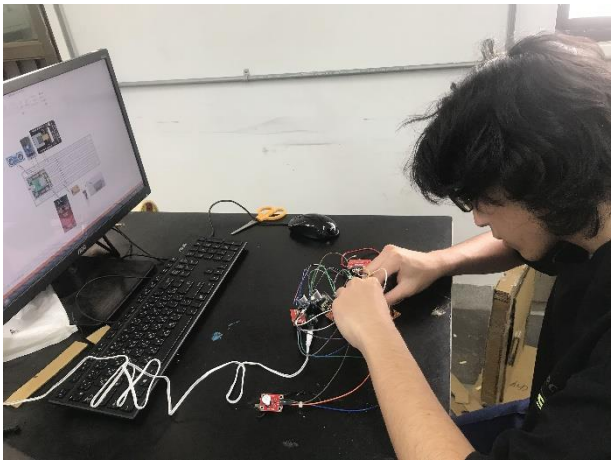


圖 2、研究流程與架構

二、製作過程

我們在製作過程中，每個人都會專門負責幾項來做，但是當自己分內的事情進度做的差不多時我們也會去幫助其他組員一起趕進度。製作過程如圖 3~圖 6 所示。

<table>
<tr><td>  </td><td>  </td></tr>
<tr><td>圖 3、程式撰寫及修改</td><td>圖 4、硬體機設計</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td>圖 5、報告製作</td><td>圖 6、電路設計與檢修</td></tr>
</table>

三、接線與內部電路

我們將有緣蜂鳴器接至 PIN2 接腳，把 ESP32-CAM 的 GPIO14 接腳對接至 PIN4 接腳，GPIO15 接腳對接至 PIN8 接腳，把變壓器接上 DC 電源接頭後，DC 電源接頭 GND 與電磁鎖 GND 對接，則 VCC 與單路繼電器 COM（輸入電壓）接腳對接，電磁鎖 VCC 則與單路繼電器 NO 接腳（輸入高電位時會輸出高電位之接腳）對接，再將單路繼電器接至 PIN5 接腳，把超音波測距模組 Trig 接腳（用來發送訊號）接至 PIN6 接腳，Echo 接腳（用來接收訊號）接至 PIN7 接腳，把 RFID-RC522 板 NSS 接腳（用於晶片選擇）接至 PIN10 接腳，MISO（用於射頻芯片傳輸數據至主控板）接至 PIN12 接腳，MOSI（用於主控板傳輸數據至射頻芯片）接至 PIN11 接腳，SCK（用於控制時脈）接至 PIN13 接腳，但是 RST（用於重置）與 IRQ（用於中斷）接腳是不接的，因為在此裝置中沒有重置與中斷的需求。最後把各元件的 VCC 與 GND 各焊一排至電路板上，並且將 LinkIt 7697 板的 VCC 與 GND 也接至電路板，接線如圖 7 所示。

我們將整個電路板與各個元件實體安裝至門內空間裡。超音波感測器與 RFID-RC522 板元件我們都將其放置外面，RFID-RC522 板因為木頭太厚隔著木頭無法感測，至於超音波感測器是因為那兩個洞不好鑽，所以才放至外頭。內部電路如圖 8 所示。

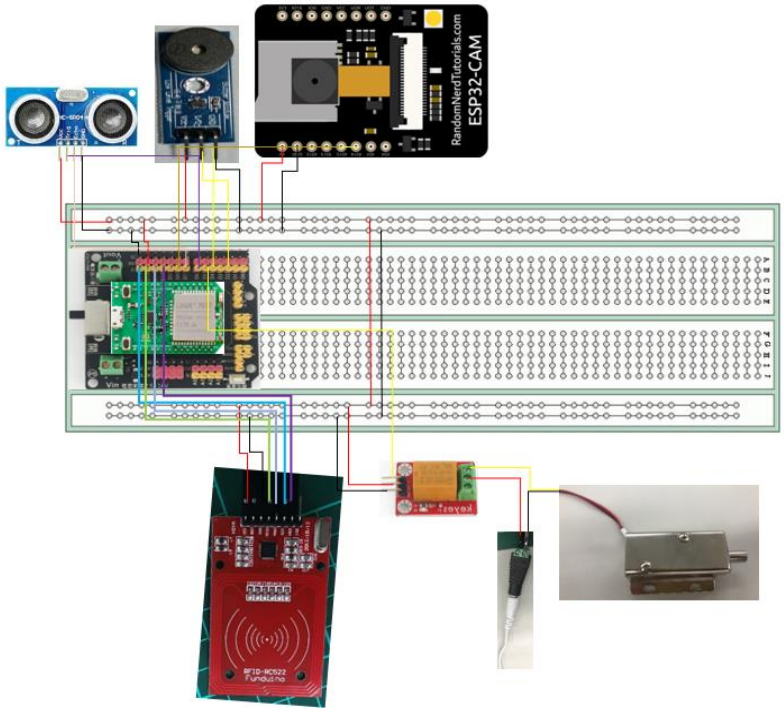


圖 7、接腳佈線圖

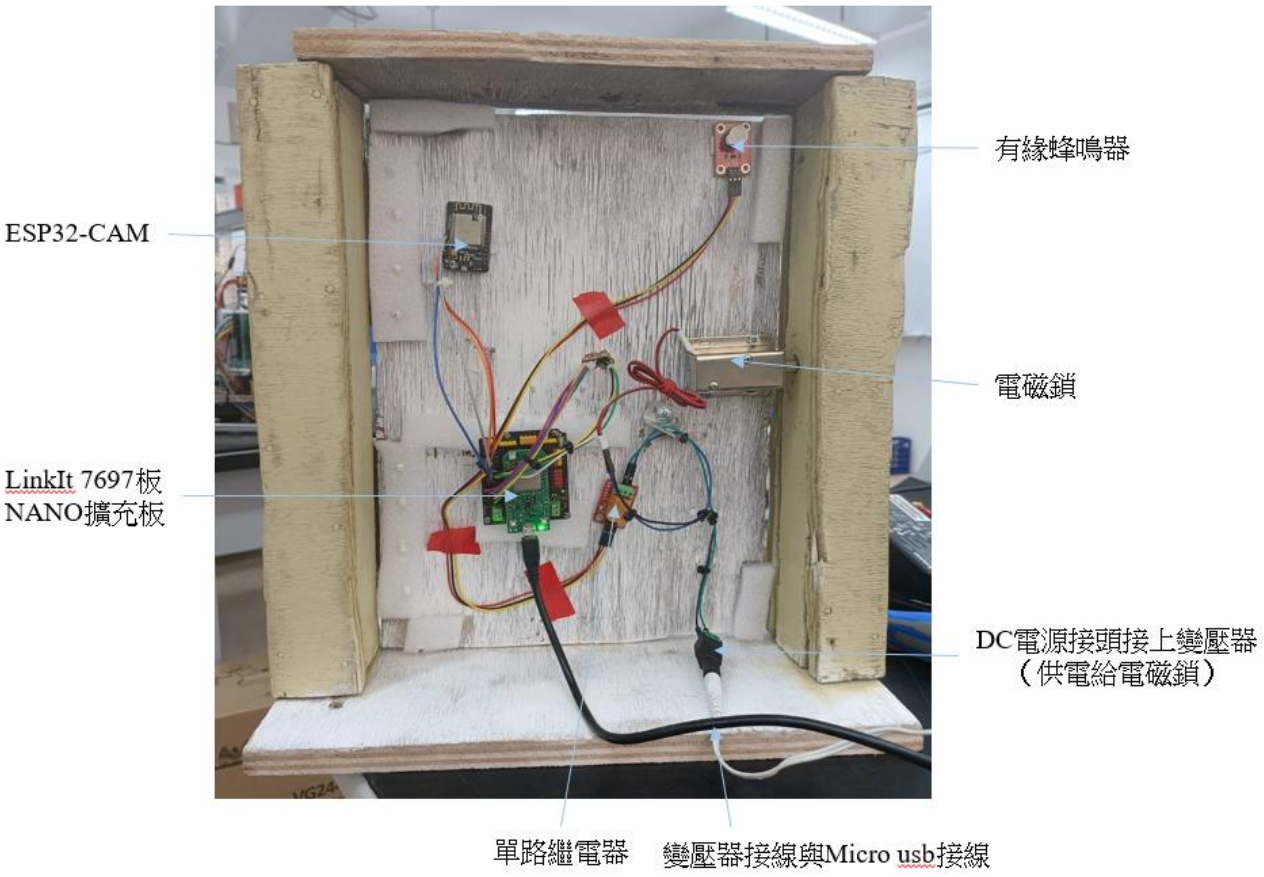


圖 8、內部電路圖

伍、 研究結果

一、動作流程

裝置動作流程為：通電後門鎖連線雲端與 Wi-Fi 並利用 ESP32-CAM 開啟即時影像後待命。當在關鎖時超音波判斷有人離門鎖 60 公分（以手臂量測距離）以內時並且持續 5 秒會傳遞訊息與影像。有三種開鎖方式，第一種利用雲端開關，開啟時打開門鎖並傳遞訊息。關閉時關閉門鎖後傳遞訊息。第二種利用人臉辨識，成功時打開門鎖傳遞訊息，5 秒後關閉門鎖後傳訊息，可透過雲端提早關鎖。第三種利用磁扣感應，當 RFID 感應到磁扣，判斷為哪種模式，如為新增使用者模式則判斷是否為已新增磁扣，如果不是則新增磁扣且傳遞新增成功的訊息。如為刪除使用者模式則判斷是否為已新增磁扣，如果是則刪除磁扣且傳遞刪除成功的訊息。如為感應模式則判斷是否為已新增磁扣，如果是則與人臉辨識成功一樣的動作。如果感應錯誤則累積失敗一次且傳遞訊息給屋主。失敗至 3 次或以上會傳遞訊息與圖片，且蜂鳴器響起，10 秒後會關閉，當門鎖開鎖時也會關閉。

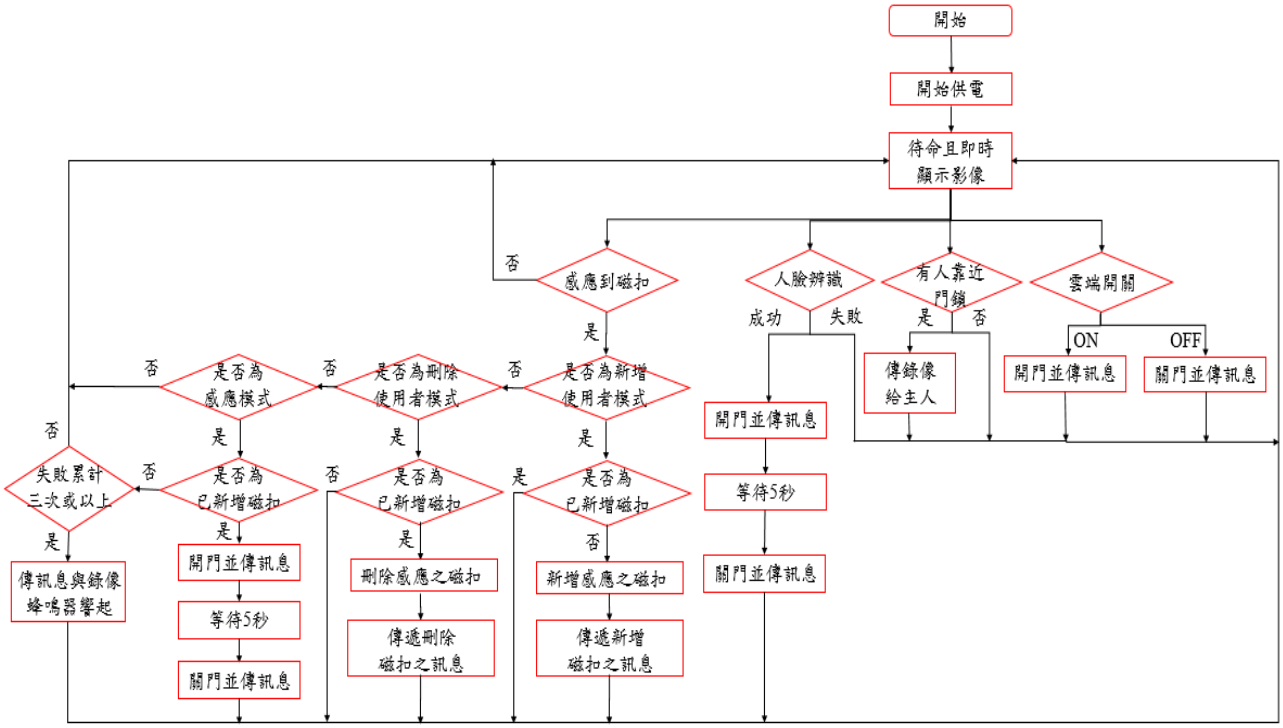


圖 9、雲端智能防盜門鎖系統之流程圖

二、LinkIt 7697 程式

本研究裝置使用 Arduino IDE 軟體撰寫程式，當 MCS 雲端開關打開或關閉時，或著是 RFID 感應到磁扣時，回傳給 LinkIt 7697 板利用條件判斷式做判斷，人臉辨識時則讓 ESP32-CAM 板去做判斷，辨識成功後再回傳給 LinlIt 7697 板，再以繼電器傳送訊號（1 或 0）給外部負載（電磁鎖），並傳遞訊息至主人的手機，且依情況來傳遞訊號（1 或 0）給蜂鳴器及 ESP32-CAM，還可依超音波測量距離來判斷是否有人並決定是否傳遞訊號（1 或 0）給 ESP32-CAM，達到開鎖關鎖與防盜作用。圖 10~圖 17 為程式內容。

```
#include <LWiFi.h>//引入WiFi連線函式庫標題檔
#include "MCS.h"//引入MCS雲端連線函式庫標題檔
#include <SPI.h>//引入序列通訊函式庫標題檔
#include <MFRC522.h>//引入RFID函式庫標題檔
#include <Ultrasonic.h>//引入超音波函式庫標題檔

String lineToken = "7OgiismZ85IZ3FZOOVDrYNmZUwDCT5dEL3ups1Y6EhZ";//設置LINE權杖
String RfidNo, rfids[100]{};//宣告RFID磁扣名稱與磁扣陣列
int loss = 0, lock = 0, tweet = 0, AutoLock = 0, SensFlag = 0, SensDelay = 0, rfid_quan = 1;
//宣告磁扣連續感應失敗的累積次數與開鎖旗標與蜂鳴器計時器與自動鎖定計時器與測距旗標與測距延遲秒數計時與新增磁扣數量
float dist;//宣告超音波測量距離
Ultrasonic ultrasonic_6_7(6, 7);//宣告超音波接腳
char _lwifi_ssid[] = "xixa3333";//WiFi帳號
char _lwifi_pass[] = "asdfghjkl";//WiFi密碼
MCSLiteDevice mcs("S1qcQJ8Ri", "b02ec897df624871fc9d0b090b9674f38e60266e6647223dcd1183bfe6800998", "192.168.205.9", 3000);
//取得MCS雲端ID與Key與IP位址和埠做連線
MCSControllerOnOff switchs("switchs");//取得雲端開關MCS控制通道
MCSControllerOnOff rfidadd("rfidadd");//取得新增使用者MCS控制通道
MCSControllerOnOff RfidDel("RfidDel");//取得刪除使用者MCS控制通道
MFRC522 rfid(/*SS_PIN*/ 10, /*RST_PIN*/ UINT8_MAX);//宣告RFID腳位
```

圖 10、引入函式庫及宣告

```
String mfrc522_readID()//RFID讀取
{
  String ret;
  if (rfid.PICC_IsNewCardPresent() && rfid.PICC_ReadCardSerial())
  //判斷有感應到新的磁扣與有讀取磁扣資料
  {
    MFRC522::PICC_Type piccType = rfid.PICC_GetType(rfid.uid.sak);
    // 根據卡片回應的SAK值判斷磁扣類型
    for (byte i = 0; i < rfid.uid.size; i++) { // 逐一取得UID值
      ret += (rfid.uid.uidByte[i] < 0x10 ? "0" : "");
      ret += String(rfid.uid.uidByte[i], HEX); // 取得UID值
    }
  }
  rfid.PICC_HaltA();// 讓磁扣進入停止模式
  rfid.PCD_StopCrypto1();//可重複感應磁扣
  return ret;//回傳磁扣UID值
}
```

圖 11、RFID 讀取副程式

```
void sendLineMsg(String myMsg) {//LINE連接與傳遞訊息
  static TLSClient line_client;//創建LINE的區域變量
  myMsg.replace("%","%25");//轉換字符
  myMsg.replace("&","%26");
  myMsg.replace("§","&");
  myMsg.replace("\\n","\n");
  if (line_client.connect("notify-api.line.me", 443)) {//LINE是否連接成功
    line_client.println("POST /api/notify HTTP/1.1");//傳遞給LINE資料
    line_client.println("Connection: close");
    line_client.println("Host: notify-api.line.me");
    line_client.println("Authorization: Bearer " + lineToken);
    line_client.println("Content-Type: application/x-www-form-urlencoded");
    line_client.println("Content-Length: " + String(myMsg.length()));
    line_client.println();
    line_client.println(myMsg);
    line_client.println();
    line_client.stop();//停止傳遞
  }
  else Serial.println("Line Notify failed");
}
```

圖 12、LINE 連接與傳遞訊息副程式

```
void LockSub(){//門鎖鎖定副程式
  lock=0;//開鎖旗標為0
  digitalWrite(5, LOW);//門鎖自動鎖定
  Serial.println("已鎖定");
  sendLineMsg(String("message=\n")+"智慧門鎖已鎖定");
}

void UnLockSub(){//門鎖開鎖副程式
  tweet=0;//蜂鳴器計時器停止計時
  loss=0;//失敗次數重置為0
  digitalWrite(5,HIGH);// 給繼電器高電位使電子鎖開鎖
  digitalWrite(2,LOW);// 蜂鳴器停止鳴叫
  Serial.println("通過");
  sendLineMsg(String("message=\n")+"智慧門鎖已開鎖");
}

void Photograph(){//拍照副程式
    digitalWrite(4, HIGH);//傳遞高電位給ESP32-CAM拍照並傳遞照片
    delay(100);//延遲0.1秒
    digitalWrite(4, LOW);//傳遞低電位給ESP32-CAM停止拍照
}
```

圖 13、門鎖鎖定、開鎖、拍照副程式

```
void setup()//初始化各值
{
    Serial.begin(9600);//序列埠初始化
    mcs.addChannel(switchs);//新增雲端開關MCS控制通道
    mcs.addChannel(rfidadd);//新增新增使用者MCS控制通道
    mcs.addChannel(RfidDel);//取得刪除使用者MCS控制通道
    Serial.println("WIFI連線中");//傳遞訊息至序列埠並換行
    while (WiFi.begin(_lwifi_ssid, _lwifi_pass) != WL_CONNECTED) { delay(1000); }
    //判斷WiFi是否連線成功
    Serial.println("WIFI已連線");
    Serial.println("MCS連線中");
    while (!mcs.connected()) { mcs.connect(); }//判斷MCS是否連線成功
    Serial.println("MCS已連線");
    Serial.begin(9600);//序列埠初始化
    SPI.begin();//序列通訊初始化
    rfid.PCD_Init();//初始化RC522讀卡機模組
    pinMode(2, OUTPUT);//初始化腳位連接蜂鳴器
    pinMode(4, OUTPUT);//初始化腳位連接ESP32-CAM
    pinMode(5, OUTPUT);//初始化腳位連接繼電器
    pinMode(8, INPUT);//人臉辨識傳輸電位
    rfids[0] = "8cc13d63";//將原使用者的磁扣碼存至陣列
}
```

圖 14、初始化各值

圖 15、MCS 同步、超音波測距、計時倒數器

圖 16、感應模式、刪除使用者模式

圖 17、新增使用者模式、其餘開鎖方式

三、ESP32-CAM 程式

ESP32-CAM 之程式也是利用 Arduino IDE 的預設環境修改成 ESP32-CAM 的環境後去撰寫程式，通電後開啟 ESP32-CAM 的 IP 位址伺服器並做設定並宣告各值及定義腳位再去初始化鏡頭後網路連線，在 LinkIt 7697 傳來高電位後開始拍攝照片之動作，拍攝後讀取照片位置並判斷傳送照片數據包，傳完後判斷是否傳送成功並且清除緩衝區。圖 18~圖 26 為程式內容。在圖 19 程式碼第 44 行因為程式碼過長，所以在這邊顯示：「String head = "--Taiwan\r\nContent-Disposition: form-data; name=\"message\"; \r\n\r\n" + msg + "\r\n--Taiwan\r\nContent-Disposition: form-data; name=\"imageFile\"; filename=\"esp32-cam.jpg\"\r\nContent-Type: image/jpeg\r\n\r\n";」。

圖 26 為人臉辨識函數，輸入圖像矩陣和檢測到的人臉框的結構體。函數先將人臉對齊，再根據 is_enrolling 判斷進行人臉登記或識別。識別成功時顯示歡迎詞和顏色標記以及將電位回傳給 LinkIt 7697 板去做開鎖動作，失敗時顯示警報信息並返回-1。最後釋放內存並返回匹配到的人臉 ID。

```
#include "esp_camera.h"//引入ESP32-CAM開發版的相機庫之標題檔
#include <WiFi.h>//引入ESP32-CAM開發版的WiFi庫之標題檔
#include <WiFiClientSecure.h>//引入建立與網站的HTTPS通訊函式庫之標題檔
#include "soc/soc.h"//引入客戶端與服務器程序函式庫之標題檔
#include "soc/rtc_cntl_reg.h"//引入操作實時時鐘控制寄存器函式庫之標題檔
#define CAMERA_MODEL_AI_THINKER
//定義CAMERA_MODEL_AI_THINKER 的符號常量，可利用這個符號常量表示相機型號
#include "camera_pins.h"
//將此標頭檔案引入進程式碼中，此標頭檔案定義了與相機引腳相關的常量和設置
const char* ssid = "xixa3333";//WiFi帳號
const char* password = "asdfghjkl";//WiFi密碼
String myLineNotifyToken = "70giismZ8SIZ3FZ00VDrYNmZUwDCT5dEL3ups1Y6EhZ";
//LINE權杖
```

圖 18、標題檔與宣告及定義

```
String sendImage2LineNotify(String msg) {//傳遞LINE訊息與照片之副程式
  camera_fb_t * fb = NULL;//宣告指針變量為fb，類型為camera_fb_t*，並初始化為NULL
  fb = esp_camera_fb_get();//取得相機影像放置fb
  if (!fb) {//判斷沒取得相機影像
    delay(100);//延遲0.1秒
    Serial.println("Camera capture failed, Reset");
    ESP.restart();//重置ESP開發板
  }

  WiFiClientSecure client_tcp;//啟動SSL wificlient
  Serial.println("Connect to notify-api.line.me");
  if (client_tcp.connect("notify-api.line.me", 443)) {//判斷LINE與TCP在443端口連接成功
    Serial.println("Connection successful");
    String head = "--Taiwan\r\nContent-Disposition: form-data; name=\"message\"; \r
    //宣告字串用於與HTTP協議進行文件上傳或表單提交
    String tail = "\r\n--Taiwan--\r\n";//宣告字串用於與HTTP協議要求的消息結束標示符
    uint16_t imageLen = fb->len;//獲取fb指針指向len屬性的值，表示為圖像文件數據長度
    uint16_t extraLen = head.length() + tail.length();//獲取頭部和尾部字符串的長度之和
    uint16_t totalLen = imageLen + extraLen;//獲取圖像文件數據與附加長度之和
```

圖 19、副程式片段 1-拍攝照片並讀取照片位置後判斷連接獲取各值字符與長度

```
//開始POST傳送訊息
client_tcp.println("POST /api/notify HTTP/1.1");
client_tcp.println("Connection: close");
client_tcp.println("Host: notify-api.line.me");
client_tcp.println("Authorization: Bearer " + myLineNotifyToken);
client_tcp.println("Content-Length: " + String(totalLen));
client_tcp.println("Content-Type: multipart/form-data; boundary=Taiwan");
client_tcp.println();
client_tcp.print(head);
uint8_t *fbBuf = fb->buf;//獲取指向圖像文件數據的指針
size_t fbLen = fb->len;//獲取圖像文件數據長度
Serial.println("Data Sending....");
//照片，分段傳送

for (size_t n = 0; n < fbLen; n = n + 2048) {//將圖像文件數據分包發送給服務器
  if (n + 2048 < fbLen) {//判斷數據包無超過2048字節
    client_tcp.write(fbBuf, 2048);//傳送2048字節的數據包給服務器
    fbBuf += 2048;//將指針移動到下個數據包起始位址
  }
  else if (fbLen % 2048 > 0) {//判斷還有剩下的數據包
    size_t remainder = fbLen % 2048;//取得剩下的數據包長度
    client_tcp.write(fbBuf, remainder);//將剩下的數據包發送給服務器
  }
}
client_tcp.print(tail);
client_tcp.println();
```

圖 20、副程式片段 2-傳遞訊息與數據包

```
    String getResponse = "", Feedback = "";//宣告字符串
    boolean state = false;//將state宣告類型為布林，並初始化為false
    int waitTime = 3000;   // 依據網路調整等候時間，3000代表，最多等3秒
    long startTime = millis();//獲取當前程序的運行時間
    delay(1000);//延遲1秒
    Serial.print("Get Response");

    while ((startTime + waitTime) > millis()){//等待TCP客戶端響應
      Serial.print(".");
      delay(100);//延遲0.1秒
      bool jobdone=false;//將jobdone宣告類型為布林，並初始化為false
      while (client_tcp.available())
      {//當有收到回覆資料時
        jobdone=true;//可退出迴圈之標記為true
        char c = client_tcp.read();//取得客戶端發送的數據
        if (c == '\n')//判斷字符讀取到換行符時
        {
          if (getResponse.length() == 0) state = true;//如果字符串長度為0則標記為true
          getResponse = "";//將字符串清空
        }
        else if (c != '\r')//判斷讀取到的字符不為回車符
          getResponse += String(c);//將客戶端傳送之資料丟給字符串
        if (state == true) Feedback += String(c);//判斷標記為true則將客戶端傳送之資料丟給字符串
          startTime = millis();//獲取當前程序的運行時間
      }
      if (jobdone) break;//有收到回覆資料即可退出迴圈
    }

    client_tcp.stop();//關閉與TCP服務器的連接
    esp_camera_fb_return(fb);//清除緩衝區
    return Feedback;//回傳客戶端傳送之資料
  }

  else {//否則連接失敗時
    esp_camera_fb_return(fb);//釋放相機緩存區內存
    return "Send failed.";//回傳傳送失敗之字符串
  }
}
```

圖 21、副程式片段 3-回傳判斷與清除緩衝區與連接失敗之處理

```
void startCameraServer();
//啟動一個相機伺服器，讓客戶端通過網路連接到相機，獲取即時影像或者訪問相機的設置
void setup() {
    Serial.begin(115200);//序列埠初始化
    Serial.setDebugOutput(true);
    /*啟用Serial物件的調試輸出模式，Serial物件可輸出調試信息到所有已註冊的串口，
    這些信息可以用於調試程序或查找錯誤*/
    Serial.println();
    camera_config_t config;//宣告相機的配置參數
    config.ledc_channel = LEDC_CHANNEL_0;//將LEDC通道設置為0
    config.ledc_timer = LEDC_TIMER_0;//將LEDC計時器設置為0
    config.pin_d0 = Y2_GPIO_NUM;//宣告腳位
    config.pin_d1 = Y3_GPIO_NUM;
    config.pin_d2 = Y4_GPIO_NUM;
    config.pin_d3 = Y5_GPIO_NUM;
    config.pin_d4 = Y6_GPIO_NUM;
    config.pin_d5 = Y7_GPIO_NUM;
    config.pin_d6 = Y8_GPIO_NUM;
    config.pin_d7 = Y9_GPIO_NUM;
    config.pin_xclk = XCLK_GPIO_NUM;
    config.pin_pclk = PCLK_GPIO_NUM;
    config.pin_vsync = VSYNC_GPIO_NUM;
    config.pin_href = HREF_GPIO_NUM;
    config.pin_sscb_sda = SIOD_GPIO_NUM;
    config.pin_sscb_scl = SIOC_GPIO_NUM;
    config.pin_pwdn = PWDN_GPIO_NUM;
    config.pin_reset = RESET_GPIO_NUM;
    config.xclk_freq_hz = 20000000;//設置相機模塊的時鐘頻率為20MHz
    config.pixel_format = PIXFORMAT_JPEG;//設置相機輸出圖像格式為JPEG格式
```

圖 22、初始化鏡頭

```
if (psramFound()) {//判斷開發版是否搭載PSRAM外部記憶體
    config.frame_size = FRAMESIZE_UXGA;//設置相機輸出圖像分辨率為UXGA
    config.jpeg_quality = 10;//設置輸出的圖像質量
    config.fb_count = 2;//分配相機圖像緩存數量
}
else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
}
esp_err_t err = esp_camera_init(&config);
/*esp_err_t類型是一個枚舉類型的錯誤碼，用於指示操作的成功或失敗
如果相機初始化成功，則該函數返回ESP_OK。反之則函數返回相應的錯誤碼*/
if (err != ESP_OK) {//判斷初始化失敗
    Serial.printf("Camera init failed with error 0x%x", err);
    return;//退出此程式
}
sensor_t* s = esp_camera_sensor_get();//獲取相機模組的傳感器物件指針
if (s->id.PID == OV3660_PID) {//判斷傳感器裝置ID是否為OV3660_PID
    s->set_vflip(s, 1);//垂直翻轉
    s->set_brightness(s, 1);//調整亮度
    s->set_saturation(s, -2);//調整飽和度
}
s->set_framesize(s, FRAMESIZE_QVGA);//設置傳感器的幀大小為FRAMESIZE_QVGA
```

圖 23、判斷與調整相機模組

```
    WiFi.mode(WIFI_STA);//WiFi作為客戶端連接到指定的WiFi網路
    long int StartTime = millis();
    WiFi.begin(ssid, password);//嘗試連接WiFi網路
    while (WiFi.status() != WL_CONNECTED) {//開始迴圈直到WiFi連接成功
        delay(500);//延遲0.5秒
        Serial.print(".");
        if ((StartTime + 10000) < millis()) break;//超過10秒也會退出迴圈
    }
    if (WiFi.status() != WL_CONNECTED) {//判斷WiFi連接失敗
      Serial.println("Reset");
      delay(1000);
      ESP.restart();//連線不成功，則重新開機
    }
    Serial.println("");
    Serial.println("WiFi connected");
    startCameraServer();
    Serial.print("Camera Ready! Use 'http://");
    Serial.print(WiFi.localIP());//即時影像IP位址
    Serial.println("' to connect");
    pinMode(14, INPUT);//14接4
    pinMode(15, OUTPUT);//15接8
    digitalWrite(15, HIGH);
}
```

圖 24、網路連線及初始化設定

```
void loop() {
    if (digitalRead(14) == true) {//高電位時，傳圖片
        Serial.println("starting to Line");
        String payload = sendImage2LineNotify("照片傳送中，即時影像網址:http://"+WiFi.localIP().toString());
        //呼叫傳遞訊息與圖片之副程式，並將回傳值丟至字串payload
        Serial.println(payload);
        delay(2000);//延遲2秒
    }
    delay(100);
}
```

圖 25、主程式-判斷 LinkIt 7697 傳來高電位後執行

```
static int run_face_recognition(dl_matrix3du_t *image_matrix, box_array_t *net_boxes){
    dl_matrix3du_t *aligned_face = NULL;
    int matched_id = 0;

    aligned_face = dl_matrix3du_alloc(1, FACE_WIDTH, FACE_HEIGHT, 3);
    if(!aligned_face){
        Serial.println("Could not allocate face recognition buffer");
        return matched_id;
    }
    if (align_face(net_boxes, image_matrix, aligned_face) == ESP_OK){
        if (is_enrolling == 1){
            int8_t left_sample_face = enroll_face(&id_list, aligned_face);

            if(left_sample_face == (ENROLL_CONFIRM_TIMES - 1)){
                Serial.printf("Enrolling Face ID: %d\n", id_list.tail);
            }
            Serial.printf("Enrolling Face ID: %d sample %d\n", id_list.tail, ENROLL_CONFIRM_TIMES - left_sample_face);
            rgb_printf(image_matrix, FACE_COLOR_CYAN, "ID[%u] Sample[%u]", id_list.tail, ENROLL_CONFIRM_TIMES - left_sample_face);
            if (left_sample_face == 0){
                is_enrolling = 0;
                Serial.printf("Enrolled Face ID: %d\n", id_list.tail);
            }
        } else {
            matched_id = recognize_face(&id_list, aligned_face);
            if (matched_id >= 0) {
                Serial.printf("Match Face ID: %u\n", matched_id);
                rgb_printf(image_matrix, FACE_COLOR_GREEN, "Hello Subject %u", matched_id);
                Serial.println(matched_id);
                digitalWrite(15, LOW);
                delay(100);
                digitalWrite(15, HIGH);
            } else {
                Serial.println("No Match Found");
                rgb_print(image_matrix, FACE_COLOR_RED, "Intruder Alert!");
                matched_id = -1;
            }
        }
    } else {
        Serial.println("Face Not Aligned");
        //rgb_print(image_matrix, FACE_COLOR_YELLOW, "Human Detected");
    }

    dl_matrix3du_free(aligned_face);
    return matched_id;
}
```

圖 26、人臉辨識函數

四、製作測試結果

我們製作完程式和電路與最後的門之後，開始把這三個結合並且測試，先靠近門鎖測試超音波感測判斷後拍攝影像傳送，再測試雲端開關讓電磁鎖開或關並傳遞訊息，接下來測試拿磁扣感應 RFID-RC522，感應成功後開鎖且傳遞訊息，15 秒後自動關鎖也傳送訊息，然後測試感應失敗會傳遞訊息，累積達三次或以上時拍攝影像並傳遞且蜂鳴器響起 30 秒，開鎖可提早關閉，再測試開啟新增使用者模式可新增磁扣，開啟刪除使用者模式可刪除磁扣，再測試人臉辨識，辨識失敗，擷取人臉素材後即可辨識成功，並將門鎖開鎖，關鎖動作與磁扣感應成功後執行動作相同。本次測試過程如圖 27~圖 49。

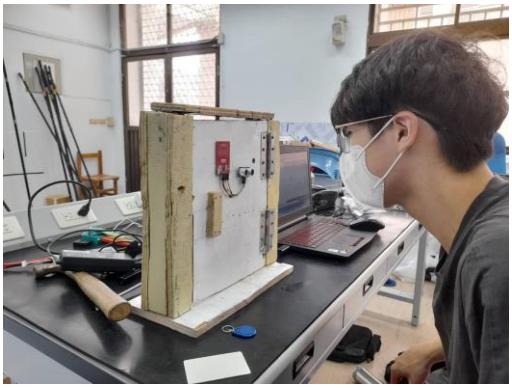


圖 27、超音波判斷距離測試

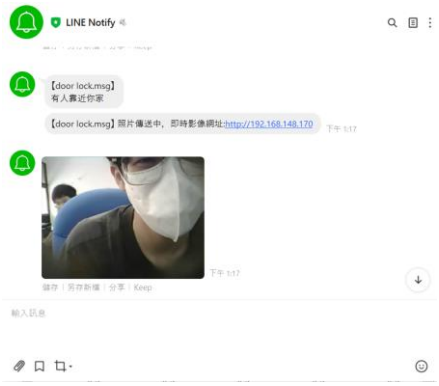


圖 28、超音波判斷後傳遞照片測試

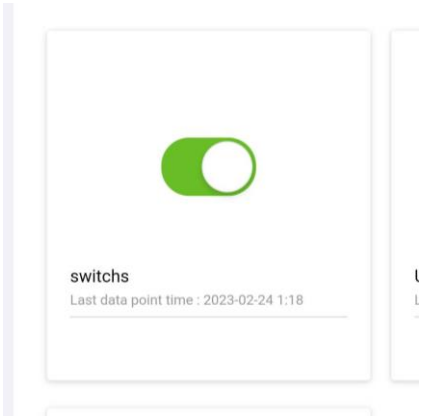


圖 29、雲端開關-開啟測試

圖 30、雲端開鎖測試

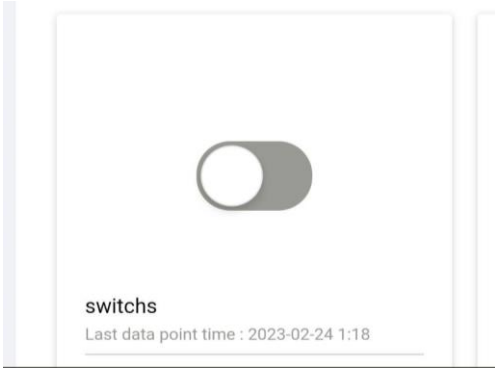


圖 31、雲端開關-關閉測試

圖 32、雲端鎖定測試

圖 33、RFID 開鎖測試

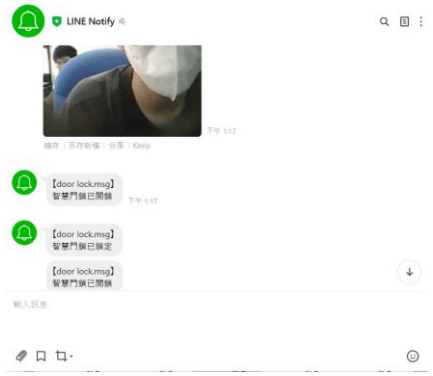


圖 34、門鎖開鎖傳遞訊息測試

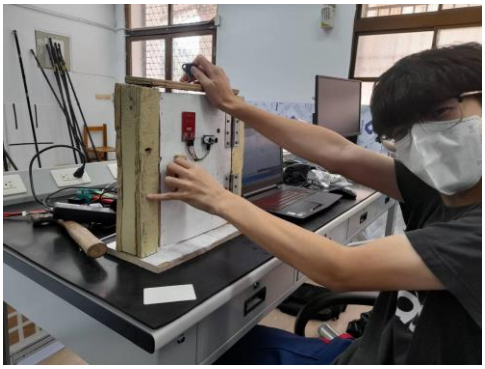


圖 35、RFID 自動鎖定測試

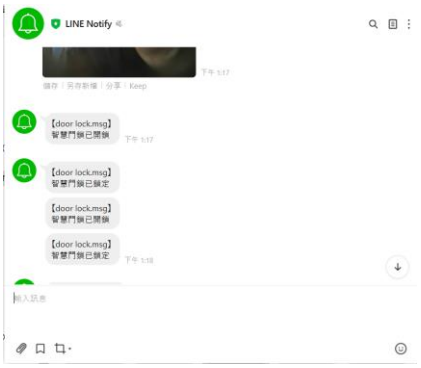


圖 36、門鎖鎖定傳遞訊息測試

圖 37、RFID 感應失敗測試

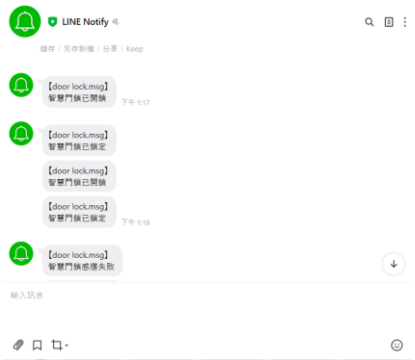


圖 38、RFID 感應失敗傳遞訊息測試

圖 39、RFID 感應失敗連續達三次測試

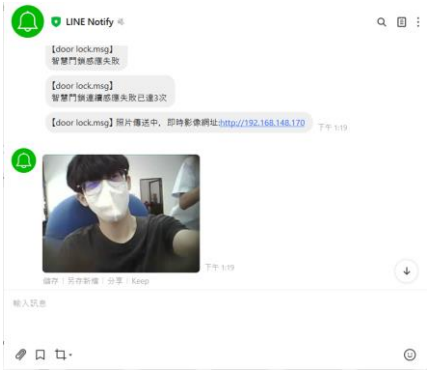


圖 40、RFID 感應失敗三次傳訊息測試

圖 41、開啟新增使用者模式

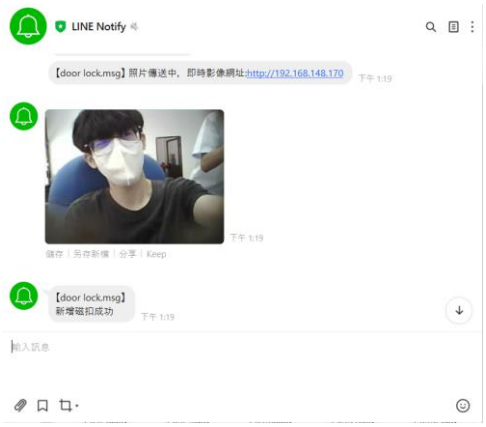


圖 42、新增磁扣成功訊息

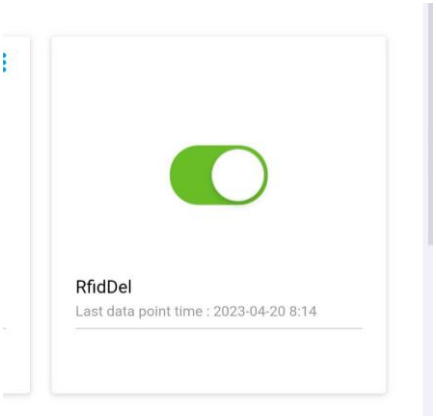


圖 43、開啟刪除使用者模式

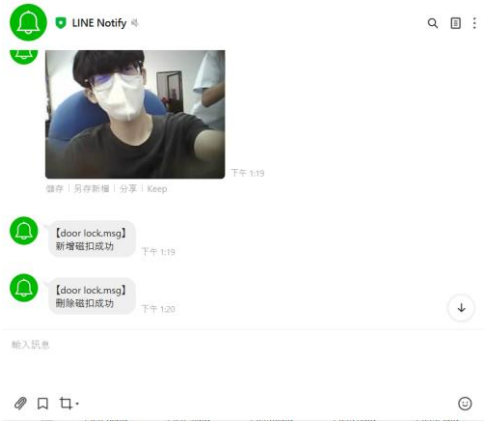


圖 44、刪除磁扣成功訊息

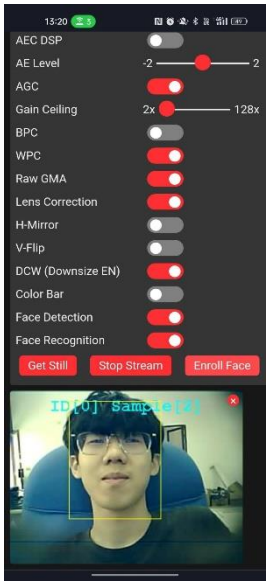


圖 45、人臉辨識素材擷取

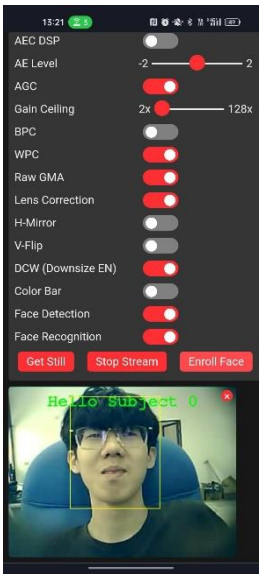


圖 46、人臉辨識成功

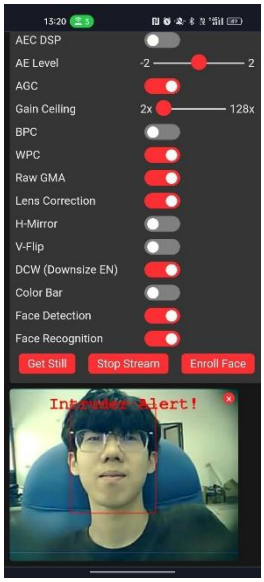


圖 47、人臉辨識錯誤

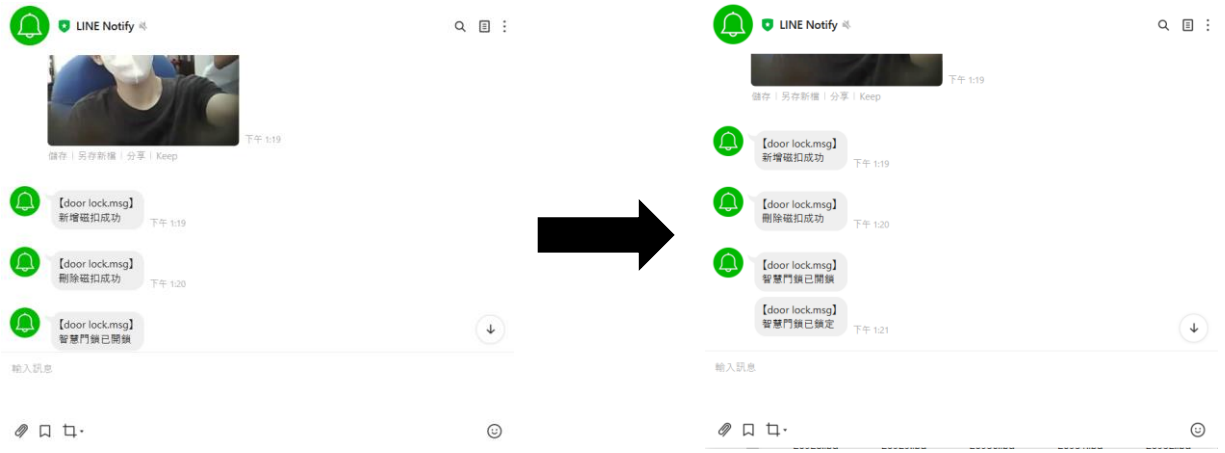


圖 48、人臉辨識成功開鎖　　圖 49、門鎖自動鎖定

五、遭遇困難&解決困難

（一）、ESP32-CAM 與 LINE 連接失敗

在撰寫 ESP32-CAM 程式時因與 LINE 連接失敗，因此無法傳遞 LNE 訊息與影像，於是去尋找範例程式或是網路上找尋程式，結果都相同，連接失敗。於是開始找尋為何 ESP32-CAM 跟 LINE 連接會失敗，找尋了許久，都找不到蛛絲馬跡，但是在找尋途中有找到另一種方法，架設網站伺服器後，利用 ESP32-CAM 傳遞圖像至網站伺服器，再傳至屋主手機上面。我們就開始試試看尋找到的方法，但是發現架設公開的網站伺服器不是免費的，於是放棄了這個方法，又回頭繼續尋找原因。再次找尋了許久，終於找到在 FB 上有關 ESP32-CAM 拍攝並傳遞影像至 LINE 程式之

貼文下面有一則留言說著如果 ESP32-CAM 環境下載的版本太新便會傳遞失敗，於是立馬下載舊版本（1.0.3 版）才終於與 LINE 連接成功。原本的版本是最新版 1.0.6 版。

（二）、RFID-RC522 無法感應

在測試 RFID-RC522 時拿磁扣去感應卻沒有反應，於是尋找是否有接線錯誤，結果都是正確的，接下來開始尋找程式是否有錯誤，但找了好幾次，都沒有錯誤，就找 RFID-RC522 的範例程式，編譯燒錄後還是一樣。再來把與 RFID-RC522 對接的杜邦線都拿去換一批，依然一樣的結果，於是測試杜邦線是否正常，測完後都是正常的。到最後只能拿去詢問老師，老師尋找一番過後也找不出問題所在，於是老師向我們索取 RFID-RC522 去測試。幾天後老師就跟我們說這 RFID-RC522 是壞掉的，於是拿去給廠商退貨，並索取新的 RFID-RC522，取得後拿去測試，終於可以感應磁扣了。非常感謝老師的幫忙。但是在之後在有一次測試時有時感應的到磁扣有時感應不到，或是感應很久才有反應，於是開始重燒好幾次，發現還是一樣，然後將 RFID-RC522 板拿起後才測試正常，原來是因為 RFID-RC522 板放在地上時會因為受到雜訊干擾而感應不良，但是在之後裝上外部硬體測試時一切正常。

（三）、超音波測距模組測距顯示為 0cm

在測試超音波測距模組量測距離時卻沒有反應，把數值顯示在序列埠上也為 0，於是去尋找範例程式測試，還是一樣的結果，接下來就換確定是好的超音波測距模組，依然是一樣的結果，就開始把矛頭指向杜邦線，跟同學拿杜邦線來用後，依然還是一樣。就取與其他元件對接的杜邦線來與超音波測距模組對接，測試結果才正常有在序列埠上顯示出測量距離數值，於是與同學再借杜邦線，並且測試與同學借的杜邦線，直到測試成功的杜邦線足夠，才終於可以繼續將程式製作下去。感謝同學的幫助。

陸、 討論

表 4、有無裝防盜門鎖系統的差別

名稱	一般民宅的鎖	雲端智能防盜門鎖系統
優點	1、價格便宜 2、安裝容易	1、與市面的防盜鎖相比價格較低 2、較不容易遭到外部因素破壞 3、自帶錄像功能
缺點	1、容易受破壞 2、容易被破解	1、需要定期維護 2、需要電源才能工作
改進的部分		1、可以自動配對位址

柒、 結論

一、運用超音波偵測是否有物體靠近門鎖

本組利用超音波感測器模組來偵測是否有物體靠近門鎖，經過利用手臂測量後，最適的感測距離為 60cm，當關門期間偵測到物體時利用 ESP32-CAM 將其拍照並利用 LINE Nofity 傳遞給屋主可使屋主防範，但在測試時有時超音波感測器會無偵測到物體 0.幾秒，於是我們設定了無偵測到物體時延遲避免錯誤。

二、透過自製模型，設計實驗測試產品可行性程度

本組利用紙箱與白紙製作成門的小模型，在門裡面留了許多空間放置電路，並在門框挖了一個洞用於放置電磁鎖之鎖芯，當關門時將會伸至裡面，開門時電磁鎖將會把鎖芯吸起即可開門。

三、結合 Wi-Fi 與雲端跟人臉辨識技術開發出具有完整防盜功能的門鎖

透過 LinkIt 7697 板設計，可利用 MCS 雲端或是 RFID 跟人臉辨識開鎖，還可新增與刪除磁扣，發生任何事件也會將訊息傳遞給屋主，當開鎖錯誤太多次時也會拍照後傳遞給屋主並讓蜂鳴器響起，即可警告屋主與鄰居或是威嚇竊賊。

四、後續研究建議

本研究期望在未來，待我們所學能夠更加精進時，能夠持續更新更多功能，達到更

加完善之防盜功能，與讓使用者能更方便的開啟門鎖，也期望能在未來將此門鎖實質化成為真正的門鎖並讓民眾願意購買。

五、總體結論

對於自己居住的人來說，更多人選擇在外面租房子，自己買房的人也隨之變少，雖然有鄰居及保安但也不一定能百分百的確認自己的人身安全以及物品被偷竊等危害，所以我們想到可以嘗試用科技解決，在製作專題的過程中了解許多相關的知識，以及課堂上所教的材料及程式來製作這次專題，但過程依舊充滿困難與挫折，在製作過程中我們學會如何去分配工作和團隊合作，各司其職，缺一不可，在過程中有許多意見交流，討論出更好的方案，途中也會有意見不合的時候，並一直改動找不到最合適及便宜方法來製作的門鎖系統，在這一次的專題中，我們活用了在學校學到的知識，也學到了許多在學校沒教過的知識，更是成為我們寶貴的經驗。

捌、 參考文獻資料

[1]內政部警政署統計室（2020 年 12 月 9 日）。警政統計通報。https://reurl.cc/d7Wjb8

[2]中興大學樂齡學習網（2023 年 3 月 8 日）。物聯網，是什麼？。https://reurl.cc/zArm9N

[3]林文恭（2020 年 10 月 27 日）。物聯網之智能商務。碁峰資訊股份有限公司。

[4]臺北市政府警察局通信隊（2023 年 2 月 24 日）。防竊宣導。https://reurl.cc/V8R0Q6

[4]曾希哲（2019）。電腦科學 LinkIt 設計物聯網應用。翰吉文化。

[5]曾希哲的學習天地（2023 年 3 月 8 日）。RFID 辨識。 https://reurl.cc/lvErE9

[6]吉哥的分享（2021 年 4 月 16 日）。Line 通知積木。https://reurl.cc/3O8nrl

[7]夜市小霸王（2021 年）。ESP32CAM 感測超音波後傳 Line 通知。https://reurl.cc/qk9R8N

[8]WebduinoRFID（2017 年）。RFID 感應。https://reurl.cc/qkZnb3

[9]夜市小霸王（2019 年）。ESP32CAM 影像伺服器及臉部辨識。https://reurl.cc/EGq0pA

[10]孫允武（2003 年 6 月 27 日）。電磁學。Wayback Machine (archive.org)

[11]嘉里達科教具總匯（2009 年 2 月 9 日）。日本磁力學儀器 嘉里達(理腦思)中小學教具及創意科學實驗儀器設備。https://www.kelabhk.com/newweb/JAPANWEB/PHYMAGNETIC.htm