中華民國第 61 屆中小學科學展覽會

作品說明書

高級中等學校組　電腦與資訊學科

第三名

052507

雲端都市垃圾資源回收警示及清運排程系統之研究

學校名稱：臺北市立內湖高級工業職業學校

作者：	指導老師：
職二 何竑蓄	陳昭安
職二 張維中	王宛琦

關鍵詞：物聯網、 Line Bot、路徑規劃

摘要

本研究設計一個可以感測容量及感應開闔的垃圾桶，透過雲端接受各地垃圾桶的容量及使用狀態，回傳到中央戰情室統整訊息以編派人力進行回收，提高回收服務及效率。利用超音波模組感測到有人接近時驅動伺服馬達打開垃圾桶，以非接觸的方式讓使用者可以丟垃圾或回收物，另外於桶內放置超音波偵測容量，用紅藍綠三色 LED 顯示使用容量，桶滿的時候可控制不打開蓋子。垃圾桶的資訊會透過 ESP8266 WiFi 模組，利用 MQTT 通訊協定傳送到 Node-Red 建立即時的戰情室中央監控，用 Line Bot 發出即時訊息通知工作人員，整合的雲端物聯網系統，可以得知各個垃圾桶的狀況，達到減少人力成本及即時管理垃圾及回收物達到環境整潔並提高服務品質的目的。

壹、研究動機

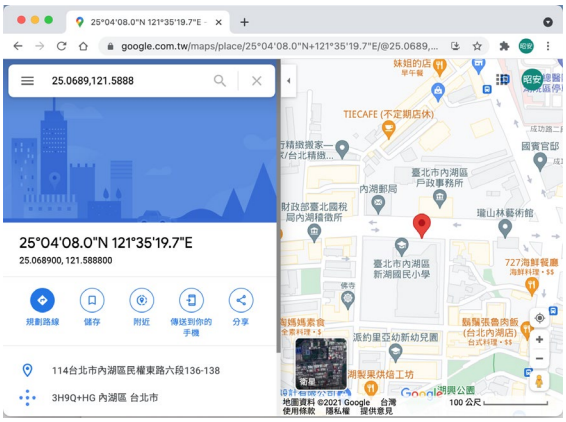
走在街頭的人行道上有時會發現公共垃圾桶滿出來卻沒人去收，造成垃圾桶附近散落一些因為無法投入垃圾桶而滿溢的垃圾的情況，如果垃圾桶容量可以保持在八成以下，讓路人可以順利丟入垃圾桶，就可以改善路邊髒亂的問題。如果街道的垃圾桶具有感測垃圾桶容量、自動開闔及網路連線的功能，再透過雲端收集各地容量及使用次數，並將整個台北市範圍公共垃圾桶的即時狀況(垃圾量、使用次數、使用時間)透過手機告知清潔人員，即可做到最佳的垃圾及資源回收的服務。

根據政府資料開放平台提供的「臺北市行人專用清潔箱」的下載資料，可以得知全台北市共有 1775 個清潔箱，資料內容共有除了路名與段數外，還有詳細的經緯度座標，資料格式如下圖 1 所示。將資料再利用 Google Maps API 去協助清潔人員規劃最佳的交通路徑，透過即時訊息通知清潔人員清運垃圾，便可做到最有效率及環保節能的服務。

行政區	路名	段號及其	經度	緯度
內湖區	民權東路	民權東路	121.5888	25.0689
內湖區	成功路	成功路2	121.5905	25.06688
內湖區	成功路	成功路2	121.5908	25.06774
內湖區	民權東路	民權東路	121.5883	25.06917

圖 1 台北市清潔箱及政府資料開放平台「臺北市行人專用清潔箱」的資料格式

圖 1 為目前台北市使用的清潔箱與資源回收箱一對，而透過開放資料平台「臺北市行人專用清潔箱」的查詢，全市共有 1774 個清潔箱，而資料中除了路段之外，也很清楚的標示清潔箱的位置，如下圖實際由地理座標搜尋再放個 Google 小人，確實可看到清潔箱。



為了得知清潔隊的實際清理清潔箱的方式與安排，我們透過電話詢問(內湖區文德分隊)，了解到清潔時程分四時段(早上、中午、下午、晚上)，正規每時段該分隊都會派出三台車去進行回收，而出發前彙整市民大型垃圾回收及舉報等需求進行路線的編排。另外也提到清潔隊的實際運作上，除了清潔車之外隊員用機車拖著拉車，機動的在市區巡查整理市容或處理市民的要求。以目前的方式，辛苦的清潔隊員採用定時排定清運的行程，或是在市街上巡邏檢視的收取，有些行人清潔箱滿桶或使用量少，都必須到了當地才能夠知道，也因此無法掌握移動到任何清潔箱可以得到的回收效益。訪談後清潔隊員給我們很高的期許，認為我們的構想對他們有三點幫助：

1、 可幫助他們在出發前提前得知垃圾量的變化，可以對他們的工作安排更有幫助。
2、 針對某些固定會垃圾量快速增加的清潔箱，可以安排稽查了解使用情況。
3、 防止民眾亂倒家庭垃圾，可以依照數據安排「嚇阻性的站點」。

整體系統流程構想結合 IoT 物聯網接收各地回傳的垃圾量，透過即時通訊的方式通知清潔人員。此外運用電子地圖的優勢，用顏色標示垃圾桶的容量，可以讓清潔隊員可以一目瞭然。比如說，若有紅色標誌的垃圾桶則垃圾容量代表快滿 (80%-100%)，藍色代表的是大約 40%-80%，綠色代表 0%-40%。系統會將資訊以即時且視覺化的方式傳送附近的清潔人員，使清潔隊員知道哪些垃圾桶已經快滿了，此外透過垃圾桶的地理位置，運用 Google Maps 協助清潔人員規劃最短路徑，以利清運效率，構想如下圖 2 所示。

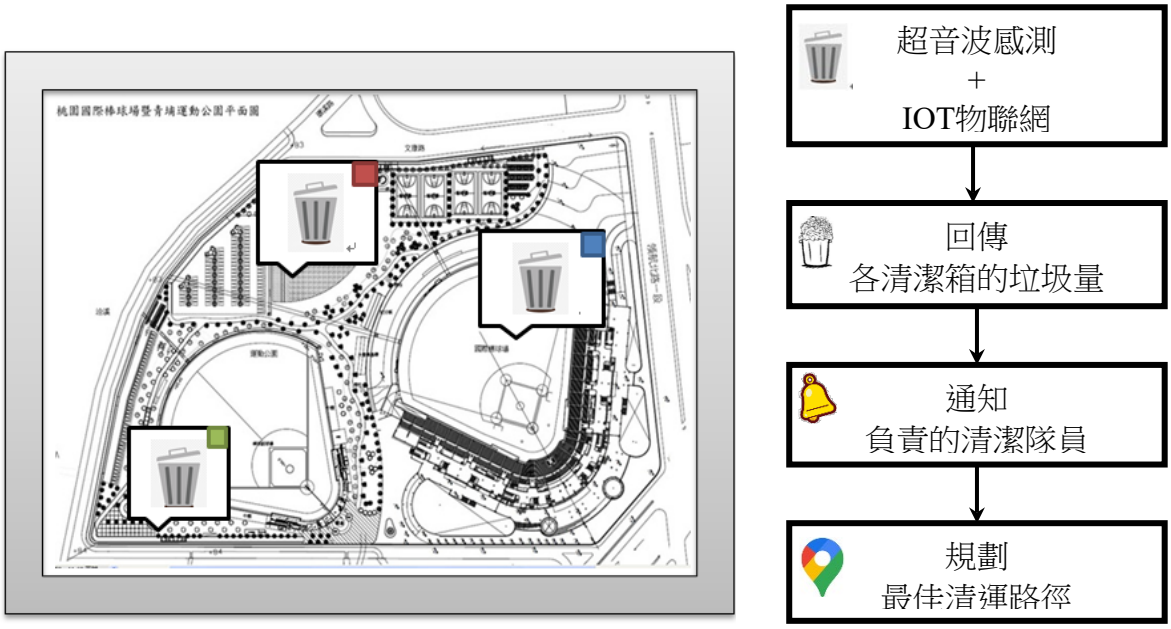


圖 2：使用物聯網建置的垃圾量監測畫面示意圖

貳、研究目的

本研究將研究一套具有偵測容量、使用次數，並可將這些資訊透過 MQTT 發佈的垃圾桶基礎裝置，主要是使用 ESP8266 去結合物聯網，利用 Node-Red 整合相關資訊，產生相關的警訊透過 LineBot 去通知清潔人員，最後用 Google Map API 去達到最佳路徑規劃，以方便清潔人員能用最少的時間和能源達到最高效率。主要的研究目的如下：

一、建構可偵測容量垃圾桶開闔並可用 MQTT 發送訊息的垃圾桶基礎裝置。

二、用 Node-Red 接收訊息並統整容量及使用量的資訊。

三、用 Node-Red 整合訊息，運用 LineBot 可以傳訊息給清潔人員。

四、運用 Google Map API 提供清潔人員規劃最佳清運路徑。

參、研究設備及器材

作品運用「程式設計實習」課程所學基礎，及「電子學實習」所學應用 Arduino、ESP8266、超音波感測器組裝電子電路。以 C++基礎自學 JavaScrip 透過搜尋 Google 雲端平台的 Google Maps API 相關範例，此外自學 Node-Red、LineBot 進行物聯網的訊息交換及處理，並於「專題實作」課程中完成此一專題。應用的硬體材料及軟體工具如下：

名稱	規格	數量
超音波模組	45x20x18mm	2
伺服馬達	21.5x11.8x22.7mm	1
Arduino WiFi 版	WeMos D1 R1	1
三色 LED 燈	RGB	1
Node-red	軟體	
Line Bot	軟體	

肆、研究方法

本作品規劃如圖 3 共分為基礎裝置設計及雲端服務兩個部分。基礎裝置設計是將垃圾桶裝上超音波模組和伺服馬達，再結合 ESP8266 網路通訊控制板達到物聯網的功用；雲端服務設計可分兩個部分，一個是 Node-Red 主要是運用 MQTT 通訊協定，將垃圾桶訊息發佈到 MQTT Broker Server，計畫使用 mqtt.cc 的外部伺服器，另外一個則是在 Node-Red 將資訊整合後，篩選垃圾量將滿的垃圾桶，用 LineBot 通知位置在這些垃圾桶附近的清潔員。另外根據垃圾桶的座標位置，利用 Google Map API 去規劃清運垃圾最佳路徑，提供清潔員去清運。

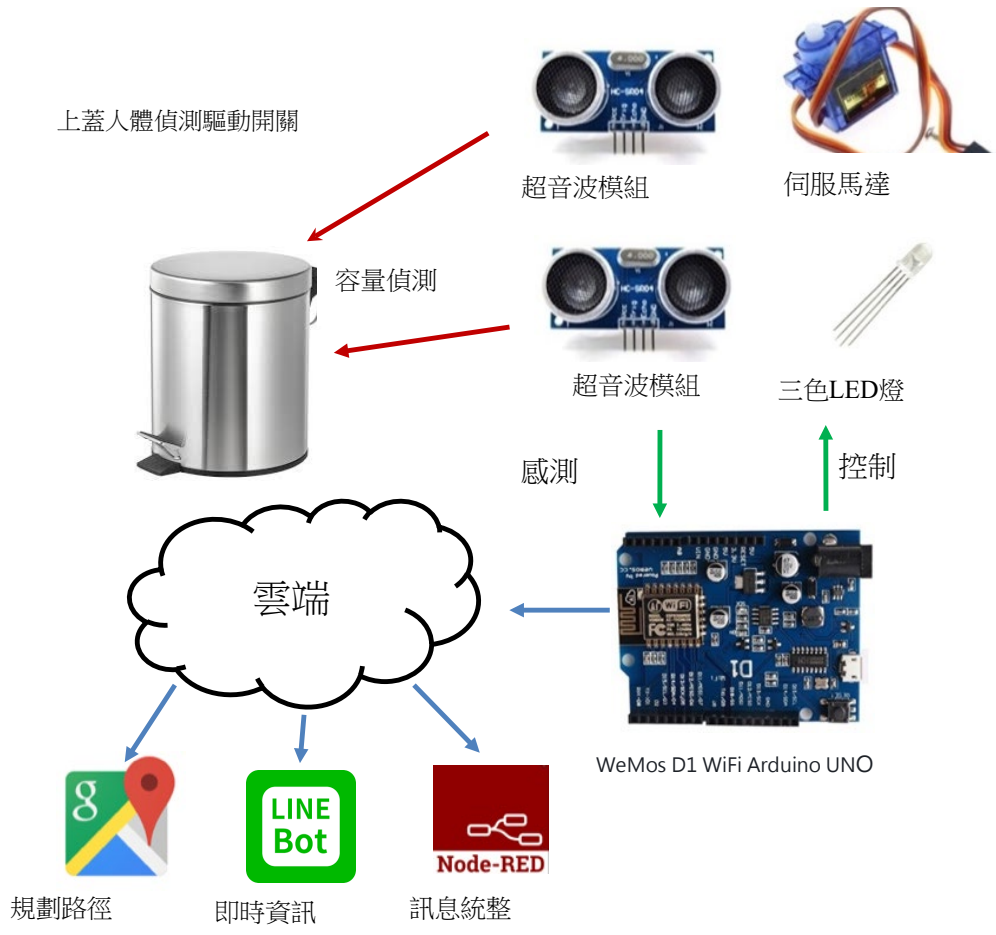


圖 3：系統概念圖(自繪)

整體的研究流程如圖 4 所示，可以得知裝置於垃圾桶上的超音波模組和伺服馬達是此作品的基礎設計，超音波模組是偵測人手部距離的驅動開關並能知道內部垃圾量，伺服馬達負責使垃圾桶蓋打開。本專題將重點放在 Google Map API(路徑規劃)、LineBot(即時回傳訊息)、Node-Red(訊息統整)再透過 WIFI 板與物聯網和上述做結合，製作一個能省時又省人力的垃圾桶清運系統。

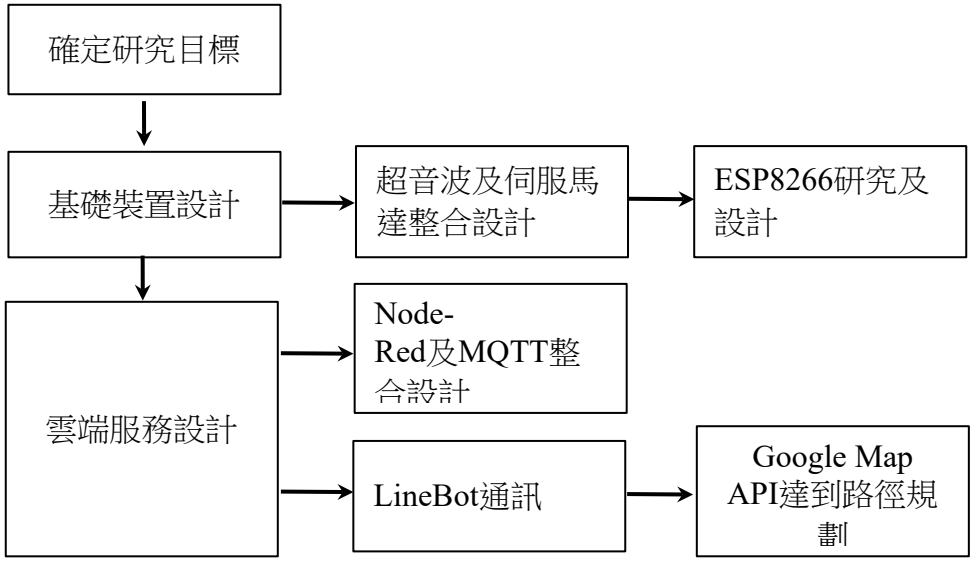
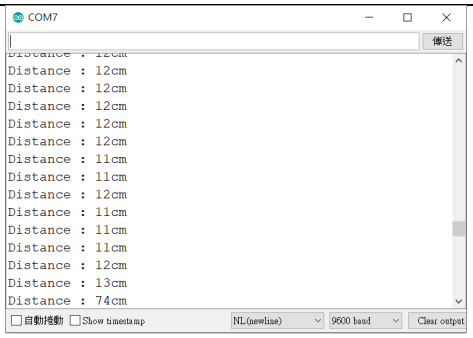


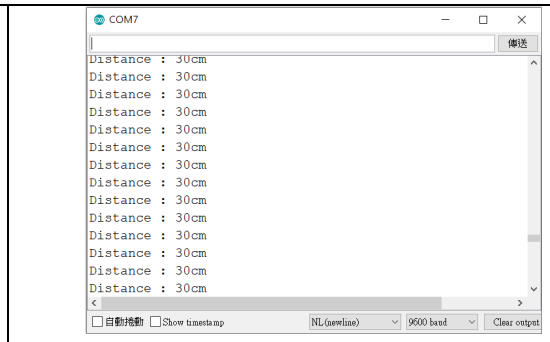
圖 4：整體研究流程圖 (自繪)

一、基礎裝置設計

(一)超音波模組

本研究使用兩個超音波模組主要用於偵測距離，其一個是偵測使用者是否接近，判斷是否可以打開垃圾桶蓋，另外一個功能是偵查垃圾量。當安裝電路並將程式寫入 Arduino 板開始進行實驗，使超音波模組便有功能，用手去遮擋前後移動測量距離，初步確定可以運作，但數據浮動劇烈，也無法確定距離是否正確。後來將超音波模組固定在一個地方，用尺測量超音波模組與障礙物的距離，經過多次的實驗之後，超音波模組測量頗為可靠，由於使用的解析度都還是在整數，以這樣的刻度距離來說，結果是可靠的，而本研究這樣的解析度也足夠了。





未固定時數字飄移劇烈	固定時數字穩定

圖 5：超音波模組透過序列步觀察測量數據

(二)伺服馬達

伺服馬達一開始選用 Micro Servo 9g A0090 這一顆，但是因為扭力不夠，所以在打開垃圾桶蓋的過程中，就會發現它推不動蓋子。所以換一顆扭力比較大的伺服馬達(SG90 Tower Pro 2kg)，原本配合腳踏型垃圾桶，想運用輔助工具去推動垃圾桶蓋，但是發現力量依然不夠，於時改成直接裝置伺服馬達於垃圾桶中間，就可以順利推開了，但也只能推開大概 45 度角左右。

將伺服馬達的塑膠旋臂改用鐵絲去推動垃圾桶蓋，但是鐵絲用久了會變形而且打開的角度比較小，於是再換成冰棒棍就會可以順利推開垃圾桶蓋了。

(三) 3 色 LED 燈

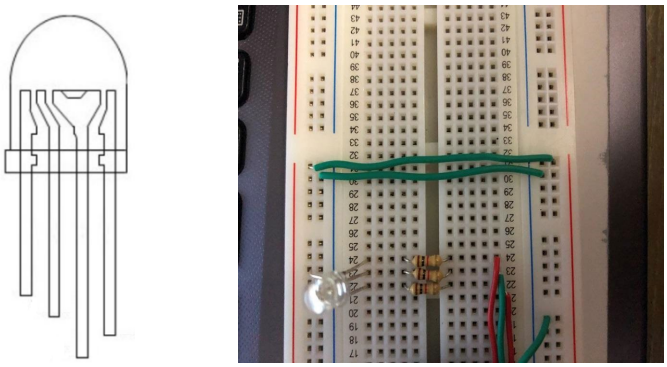


圖 6：三色 LED 燈接腳圖

3 色 LED 燈有分共陽極和共陰極，且共有四支接腳，分別為紅色、接地/高電位（長腳）、綠色、藍色。經過上課學到的知識，才知道要把接腳接在正極，即可以低態動作編寫程式，再依照超音波偵測出垃圾量的結果依序用紅、藍、綠呈現。

(四) ESP8266 Wifi 模組及 Node-Red

ESP8266 是一塊可以網路通訊控制板，採用範例去做連線的動作，如圖 7，#define STASSID "your-ssid"這行冒號裡面要改成自己網路的名稱，#define STAPSK "your-password"這行冒號裡面要改成自己網路的密碼，const char* host = "djxmmx.net";這行冒號裡面要改成自己電腦的 IP 位址，const uint16_t port = 17; 這行最後面要改成和 Node-Red 連接的埠一樣。

```
#include <ESP8266WiFi.h>

#ifndef STASSID
#define STASSID "your-ssid"
#define STAPSK  "your-password"
#endif

const char* ssid     = STASSID;
const char* password = STAPSK;

const char* host = "djxmmx.net";
const uint16_t port = 17;
```

網路名稱

網路密碼

電腦位址

序列埠要輸入17

圖 7：連接 WiFi 網路所需要設定的項目

連線成功使用時使用序列埠監控視窗觀察如圖 8，為了測試是否能在網路上看到訊息，所以用 Node-Red 這個網站來測試如圖 9，流程是先讓超音波模組測量距離，之後在顯示在序列埠監控視窗上，同時在 Node-Red 網站上可以看到這些資料一一顯示。

圖 8：連線成功後，和超音波模組做結合

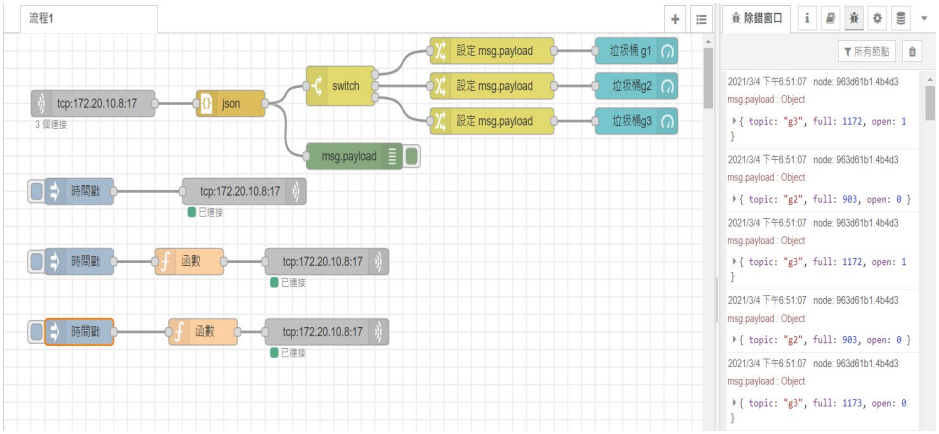


圖 9：Node-Red 編輯介面

```
Serial.println("sending data to server");
if (client.connected()) {
  char buff[80];
  sprintf(buff,"{\"topic\":\"\g1\",\"full\":%d}",cm);
  client.println(buff);
}
```

圖 10： Arduino 端的訊息透過 TCP 協定傳送到 TCP 伺服器

主要是要用 TCP 協定去連接網路，如下圖 13，第一步驟就是設定 TCP in 為監聽，不為 17，輸出為字串，而這個就是所謂的伺服器，TCP out 為連接，輸入自己的電腦位址，這樣子可以接收外部給的訊息，而第二步驟利用 JSON 去轉換資料形式，然後再用 Switch 分別給三個垃圾桶名稱，第三步驟就是設定垃圾桶的狀態，從沒有到滿了，第四步驟可以顯示出垃圾桶的垃圾量，你只要一直給他東西，它的圖就會慢慢的上升。

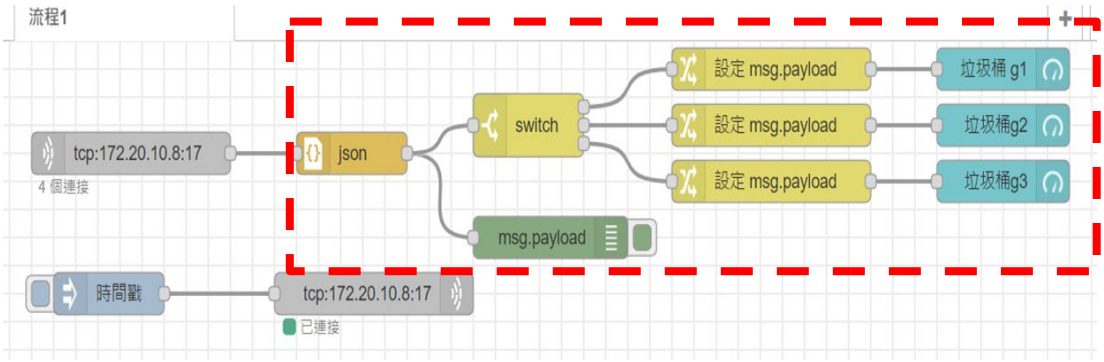


圖 11：設定函數去模擬垃圾桶的垃圾量

將圖 11 分為三個部分，分別是轉換 json 格式的物件，switch 依照 Topic 名稱是 g1、g2 或 g2，決定輸出 msg.payload 的路徑，再依據垃圾桶的容量資料設定垃圾桶的狀態，最後顯示垃圾桶的狀態。當輸入網址便可以看到整體的垃圾桶資訊。

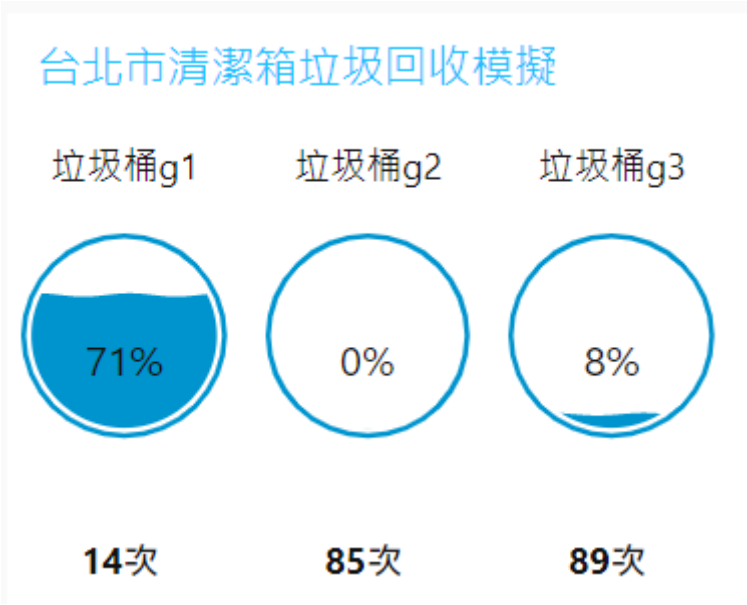


圖 12：使用 Dashboard UI 顯示資訊

二、MQTT 的通訊規劃

MQTT（Message Queuing Telemetry Transport，消息對列遥測傳輸協議），是一種基於發布/訂閱（publish/subscribe）模式的"輕量級"通訊協議，該協議構建於 TCP/IP 協議上，由 IBM 在 1999 年發布。MQTT 最大優點在於，可以以極少的代碼和有限的帶寬，為連接遠程設備提供實时可靠的消息服務。作为一種低開銷、低帶寬占用的即时通訊協議，使其在物聯網、小型設備、移動應用等方面有較廣泛的應用。

運用 MQTT 通訊協定，針對基礎裝置及後端，MQTT 會將傳送的訊息放在一個 Topic 內發送， Topic 規劃如：萬華區為 "taipei/a/xxx"、大同區為"taipei/b/xxx"，taipei 代表台北市，a~l 代表區別，而 xxx 代表流水號，由開放資料統計，清潔箱數量最多的是士林區有 210 個，最少的是文山區、信義區各 80 個，因此三個位數足夠使用。資訊透過 Topic 傳到 MQTT broker，在 Node-Red 訂閱各區的 Topic 以獲取垃圾桶容量及垃圾桶是否打開的資訊，藉此統計相關資訊，並將資訊整合後，將必要的資訊傳透過 Line Bot 傳送給清潔隊員以便得知即時情況。

分區	主題(Topic)	分區	主題(Topic)	分區	主題(Topic)
萬華區	taipei/a/xxx	大同區	taipei/b/xxx	信義區	taipei/c/xxx
大安區	taipei/d/xxx	松山區	taipei/e/xxx	中正區	taipei/f/xxx
內湖區	taipei/g/xxx	文山區	taipei/h/xxx	南港區	taipei/i/xxx
士林區	taipei/j/xxx	中山區	taipei/k/xxx	北投區	taipei/l/xxx

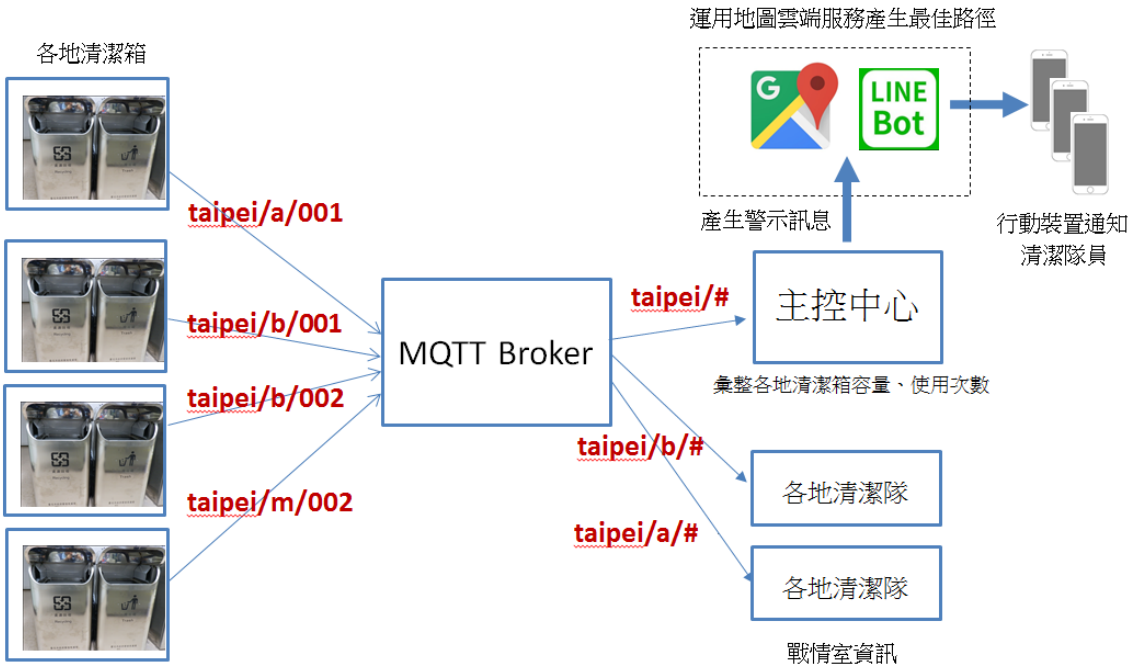


圖 12： MQTT 傳遞訊息架構圖

運用 MQTT 可以在接近即時的時間內，完成下列三項工作：

(一) 讓基礎裝置發佈容量及使用量的資訊，各區清潔隊可以接收容量及使用量。

(二)運用 LineBot 可以傳訊息給清潔人員。

(三) Google Map API 規劃最佳清運路徑。

三、資訊處理及觸發

(一)分區容量及使用量資訊統計

如圖所示，此系統所呈現的資訊包括垃圾桶容量、使用次數及使用時間，垃圾量容量是為了讓清潔人員提前得知垃圾桶情況，去避免浪費不必要的時間及油耗量。使用次數和使用時間則是為了瞭解分別在甚麼時段、地區，有較多人去使用垃圾桶，以做到更好的人力搭配。

下圖顯示為台北市各區的總表，這個資訊可以提供給台北市的相關主管單位，進行數據統計，監控全市清潔箱的使用狀況，由於全市共有 1774 個清潔箱，所以資訊處理上顯示各區目前累積的總容量及使用次數，另外也顯示各地垃圾桶容量大於 80%、90%及滿箱的清潔箱數量。而監控人員可以依照總表篩選分區及路段，顯示於如圖 13 上半部的位置，資訊顯示依照清潔箱已使用容量由大到小排序，藉以掌握每個清潔箱的狀態。

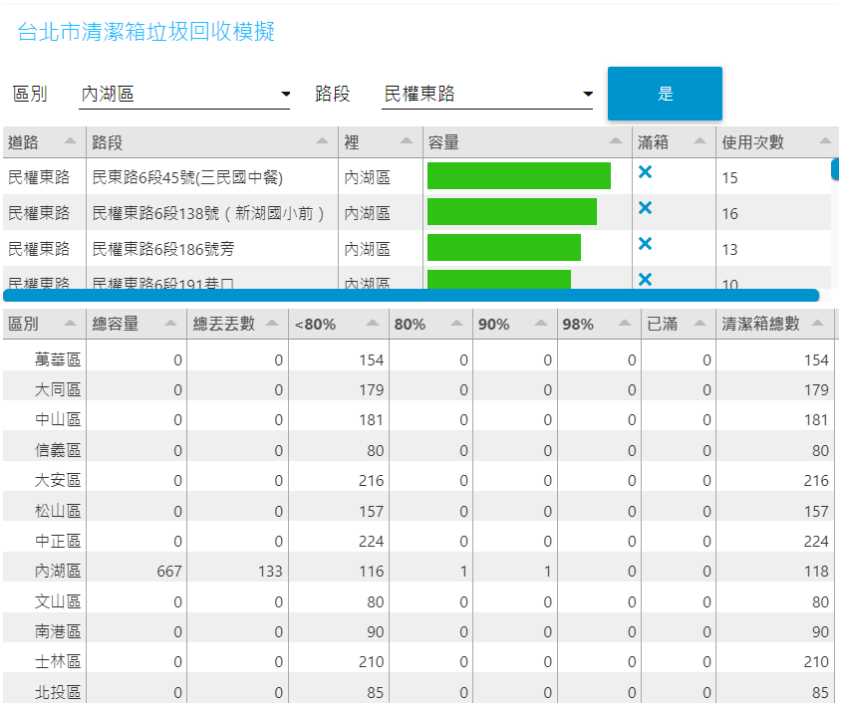


圖 13：各區各地的清潔箱容量及使用次數統計

(二)訊息發送觸發處理

經由電話詢問得知清潔人員的工作作息，於是設定垃圾桶容量超過 80% 以上的時候，便會觸發即時的通知。而配合清潔人員的時段，分別在(早、中、午、晚)固定時段發送整合資訊，使清潔人員在不同時段也能得知垃圾桶即時資訊，以方便進行清運路線的規劃。

四、Google Maps API 路線規劃設定

當操作者想從 A 地前往 B 地，能讓他們得知精準且省時的路線和即時路況分析，判斷車輛行駛的路徑，使操作者能更有效控管時間。API 也分為以下四種：

1. Directions API：規劃多個地點之間的精確路線。
2. Distance Matrix API：計算多個目的地的交通距離與時間。
3. Roads API：即時判斷車輛行駛路線。
4. Places API：搜尋或標示地理位置、地點或地址。

使用 Google 的 API 首先要登入到 Google API console 按下「啟用 API 和服務」如圖 18 所示。

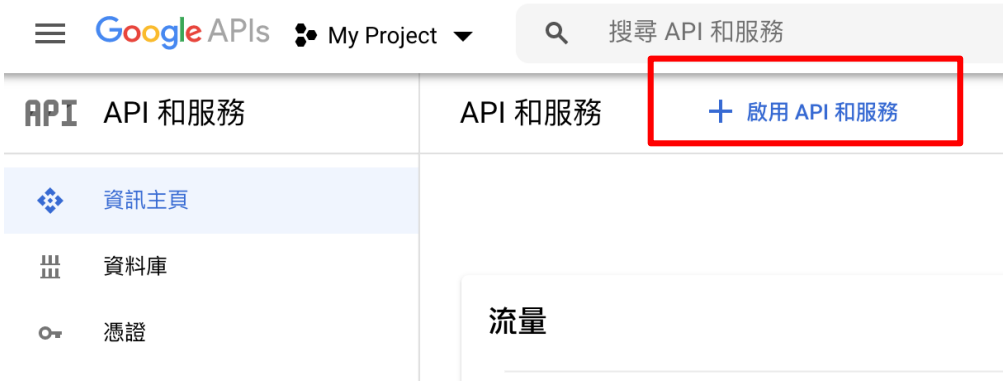


圖 14：啟用 API 的程序

再於清單中選取啟用 Directions API，接著啟用 Directions API，確定出現「API 已啟用」的訊息即已完成，如圖 19 所示。

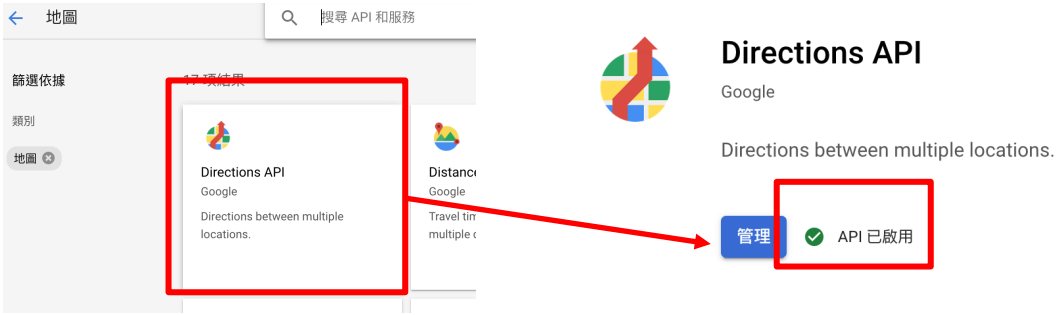


圖 15：搜尋「Directions API」點選並且啟用

接下來進行程式設定的程序，透過 directionsDisplay.setMap(map) 設定路線規畫的圖層，再經由 directionService.route 繪製。步驟過程如下：

1. 先載入路線服務與路線顯示圖層。
2. 製作出初始化地圖。
3. 放置路線圖層。
4. 進行路線相關設定。
5. 繪製路線。
6. 進行回傳(加入地圖標記、加入資訊視窗、加入地圖標記點擊事件)。

```
<iframe width="600" height="450" frameborder="0" style="border:0"
src="https://www.google.com/maps/embed/v1/place?key=你的 API key&q=地址"
allowfullscreen>
</iframe>
```

上面的這些都是把 Google Map 嵌入到網頁重要的程式，主要應用有 Place API 及 Directions API 設定，說明如下：

1. Place API：使用 place ?開頭，其後的參數設定有：

(1) q：地址、地點名稱，如：凱達格蘭大道一號、台北 101。

(2) center：地理座標，格式為"緯度，經度"，

如：25.083743522947394, 121.59424858252102。

(3) zoom：放大縮小，如：0~19。

2. Directions API：使用 directions?開頭，其後的參數說明：

(1) origin：起點，例如：國父紀念館

(2) waypoints：經過的地方，例如：台北市政府和台北 101

(3) destnation：終點，例如：台北市信義運動中心

以地點、地址定位模式	起點-經過的地方-終點的導航模式
	
key={YOUR API KEY}&q=內湖捷運站	key={YOUR API KEY} &origin=內湖捷運站 &waypoints=好樂迪 KTV&destination=湖光市場

採地理座標定位	以特定視野觀看當地的地景
	
key= {YOUR API KEY}&q=711&zoom=14& center= 25.0837435229473, 121.594248582521	key= {YOUR API KEY}&location=25.0837435, 121.594248&heading=60&pitch=30&fov=90

圖 16 ： Place API 及 Direction 的設定方式及結果

實際程式撰寫範例如下：

圖 17：利用 Place API 標示位置

圖 18：利用 Directions API 進行路徑規劃。

利用 Google Maps API 結合上述使用 Node-Red 取得各地的垃圾桶容量數據統計，可以得到下圖，其中藍色閃燈圖示上的數字表示該位置附近垃圾桶的數量，而垃圾桶容量則是運用不同的顏色去做顯示，此外藉由清潔隊員的手機，可以顯示垃圾車的正在進行的清運路徑，而當垃圾桶狀態改變後，清運路線也可以隨機調整。

圖 19： 採用直覺的視覺圖示顯示目前區域的使用狀態

五、最高效益回收路線評估

利用 Distance Matrix API 產生距離矩陣及時間距震，推算最短的距離或最短時間。從資料中找出四個垃圾桶的四個座標 P1~P4，其座標 (lat,lng) 分別為：

P1 = (25.12333,121.468244) ,

P2 = (25.1233,121.468736) ,

P3 = (25.12359,121.468902) ,

P4 = (25.12262,121.469363) ，

則 API 呼叫格式如下：

https://maps.googleapis.com/maps/api/distancematrix/json?origins=25.12333,121.468244|25.1233,121.468736|25.12359,121.468902|25.12262,121.469363&destinations=25.12333,121.468244|25.1233,121.468736|25.12359,121.468902|25.12262,121.469363&key={KEY}

得到 JSON 結果，四個點可以組合出 4x4 個距離或時間組合，如下的數據為其中一列的數據：


```
{
                (略)
  "rows" : [
    {
      "elements" : [
        {
          "distance" : { "text" : "1 公尺", "value" : 0 },
          "duration" : { "text" : "1 分鐘", "value" : 0 },
          "status" : "OK"
        },
        {
          "distance" : { "text" : "0.6 公里", "value" : 635 },
          "duration" : { "text" : "2 分鐘", "value" : 137 },
          "status" : "OK"
        },
        {
          "distance" : {"text" : "0.6 公里","value" : 577},
          "duration" : {"text" : "2 分鐘","value" : 126},
          "status" : "OK"
        },
        {
          "distance" : {"text" : "0.5 公里","value" : 455},
          "duration" : {text" : "2 分鐘","value" : 122 },
          "status" : "OK"
        }
      ]
    },
    (略)
}
```


將 JSON 的數據整理成距離矩陣如下，便可以用這個距離矩陣組合出 6 個路徑，其中有一個路徑為最短路徑，如下表所示。

	P1	P2	P3	P4
P1	0	635	577	455
P2	50	0	627	504
P3	108	58	0	563
P4	928	878	820	0

如果以 P1 為起點，每一點都不重複拜訪的話，另使用容量方面，P1=0.5、P2=0.3、P3=0.9、 P1=0.1，總容量 V=1.8 ，共有 6 條路徑則六條路徑的長度及成本效益 C 分別為：

S1 = P1 > P2 > P3 > P4> = 635+627+563 = 1,825　　C= 1.8/1.825 = 0.99

S2 = P1 > P2 > P4 > P3 = 635+504+820 = 1,959　　C= 1.8/1.959 = 0.92

S3 = P1 > P3 > P2 > P4 = 577+58+504 = 1,139 (最短)　　C=1.8/1.139 = 1.58

S4 = P1 > P3 > P4 > P2 = 577+563+878 = 2,018　　C=1.8/2.018. = 0.89

S5 = P1 > P4 > P2 > P3 = 455+878+627 = 1,960　　C=1.8/1.960 =0.92

S6 = P1 > P4 > P3 > P2 = 455+820+58 = 1,333　　C=1.8/1.333. =1.35

由上述假設所有的清潔箱都要清運，經距離矩陣計算 S3 只需 1.139 公里便可以走完，如果以全程都拜訪的情況 S3 應該是成本最低，以成本來計算共有 1.58。然而其中有兩個垃圾桶只用掉 10%或 30%的容量，是否可以考慮不要收取，如此是否可以將成本效益提高。

假設 50% (含)以下的不收，則路線 S1 所得到的成本效益為 2.25，遠大於 1.58，而清潔隊員也可以因此工作輕鬆許多。

S1 = P1>P3 = 635　　C=(0.5+0.9)/577 = 2.25

假設 30% (含)以下的不收，則路線將有：

S2= P1>P2>P3= 635+627=1,262　　C=(0.5+0.9+0.3) / 1.262 = 1.34

S3= P1>P3>P2= 577+58=635　　C=(0.5+0.9+0.3) / 0.635 = 2.67

我們很驚訝地發現，當我們選擇 S3 的路線，順便收了 P2 的 30%容量，可以獲得更高的成本效益。根據這三條路線分析，S1 每個垃圾桶都有收到，可能是清潔人員的原始路線；S2、S3 都少收一個垃圾桶，因為 P4 清潔箱容量還很少所以不收；S3 少收 P4 的垃圾桶，又可能因為順路的關係，所以也順便收 P2 的垃圾桶，還可以得到更好的成本效益。

下圖是實際使用計算成本效益後規劃的路線圖，Google Map 標示清潔箱的位置，也用圖示顯示目前清潔箱的垃圾量。透過已規劃完整的最佳路徑，讓清潔隊員可以有效率得去清運垃圾，也可以獲得更好的工作品質。

圖 20：從起點到終點的最佳路徑

六、LineBot 即時通訊設定

LineBot 即時通訊息可從 Line Developer 官網登入後進行設定，步驟為先創建一個機器人，然後將圖 14 機器人的 User ID 及 Channel access token 分別複製，貼到 Node-Red 程式如圖 15 所示，並寫程式讓機器人能在固定的時間傳送高達 80%的垃圾桶資訊，傳給清潔人員知道。

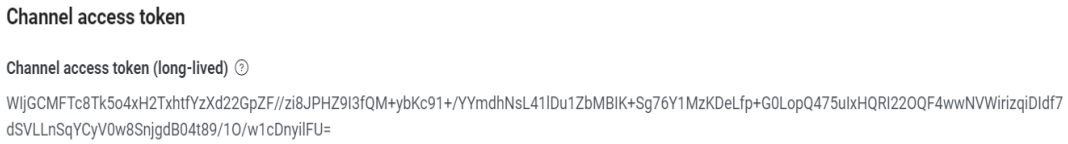


圖 21：Channel access token 是 linebot 的帳號

圖 22：User ID 是 linebot 的密碼

```
CHANNEL_ACCESS_TOKEN = 'WIjGCMFTc8Tk5o4xH2TxhtfY
USER_ID = 'U023fc17426e6a9d7d2e0cf06ef515bc3';
message = {
    type:'text',
    text:"垃圾桶情形: " + msg.payload
};
```

圖 23：關於 Line Bot 在 Node-Red 傳送訊息的設定

伍、研究結果

一、基礎裝置設計

圖 24：垃圾桶外觀

圖 25：內部構造

圖 26：接線圖

二、利用 Node-red 統整訊息

我們可以運用 Node-red 這個網站，可以把 ESP8266 所傳送的訊息傳到這個網站上，如果在這個網址後面加 ui，就可以產生如下圖的這個網站。

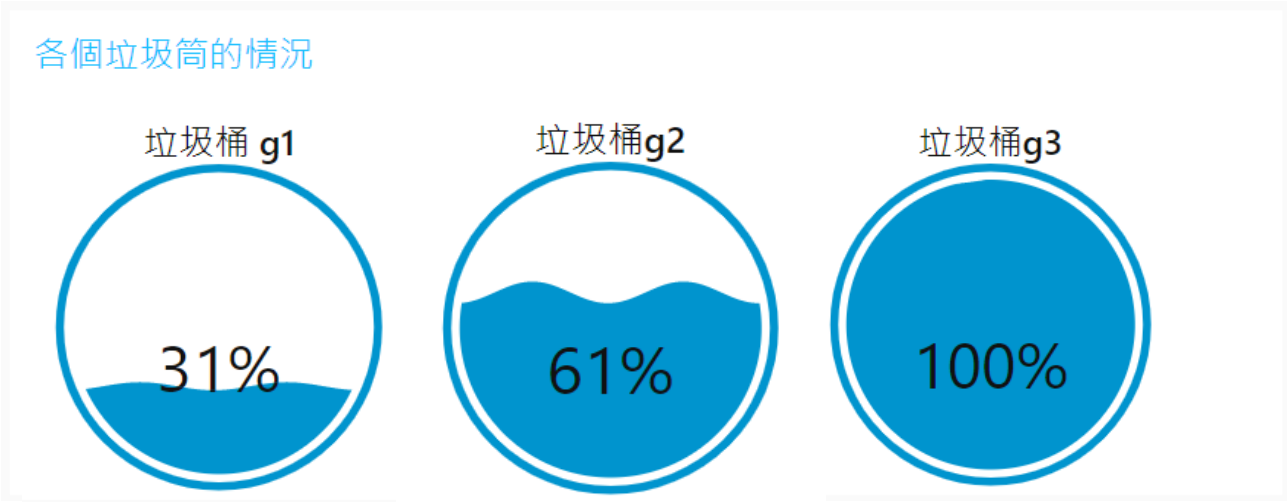


圖 27：垃圾桶的垃圾量顯示情形

三、LineBot 回傳垃圾桶資訊

主要做這個 LineBot，用一種開發工具(Node-Red)，拉方塊，並把程式寫上去，就可以傳送訊息給清潔人員，這樣子就可以讓清潔人員很清楚知道哪些垃圾桶要去清理。

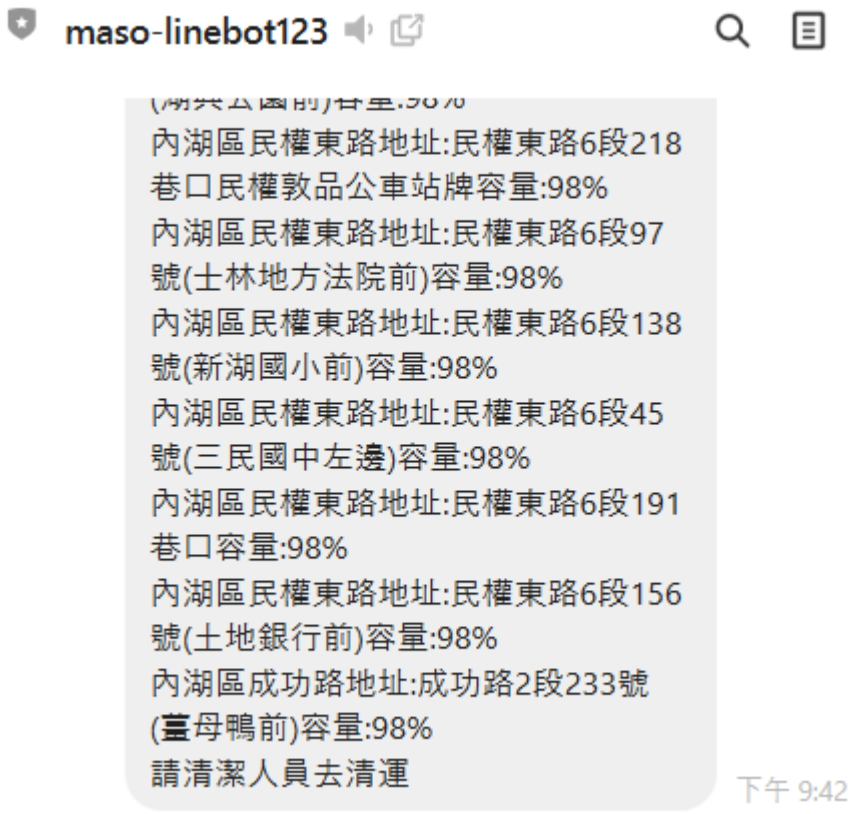


圖 28：這樣子可以知道目前垃圾桶的狀況

四、Google 垃圾清運路線規劃

此圖是模擬清潔人員的回收過程，首先將三個垃圾桶設置於不同位置，地點分別為全聯(g1)、捷運內湖站(g2)、湖光市場(g3)，垃圾量分配如圖 21 所示。

一開始假設人員位置位於清白公園，已透過 Line Bot 得知三個位置垃圾量，可以發現 g2 和 g3 已達到需回收的標準，所以透過 Google API 去計算出最短路徑(不包含 g1)，使清潔人員能以最快速度及最短距離去完成收垃圾的動作。

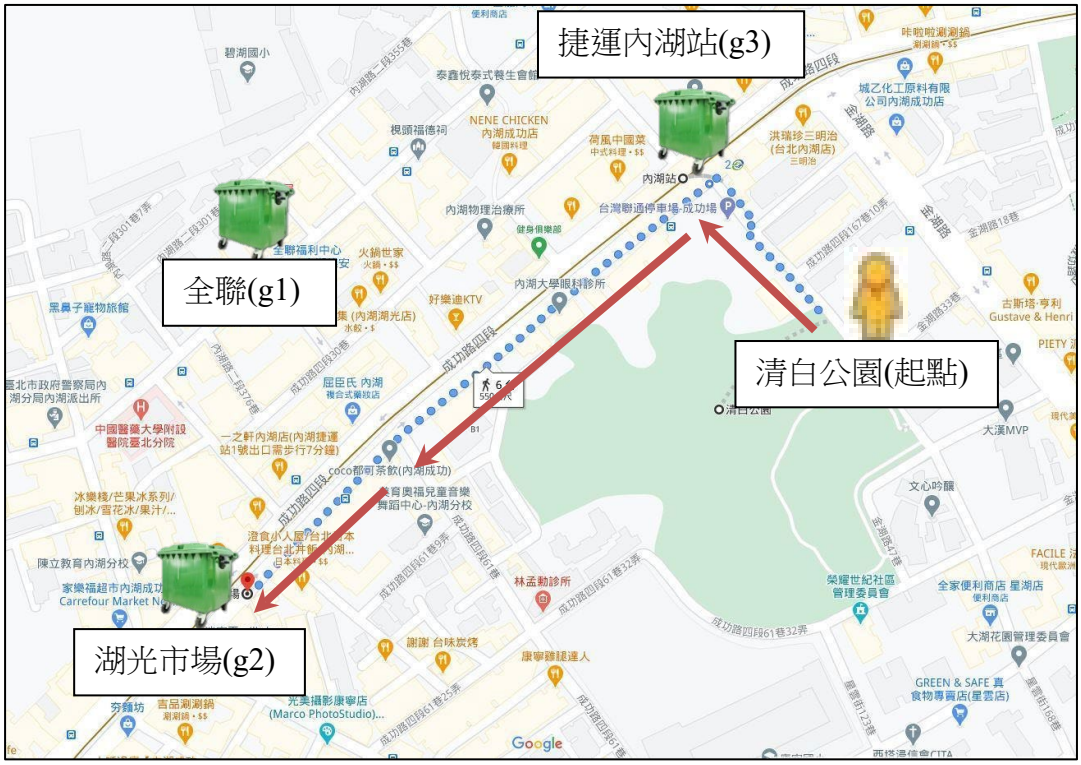


圖 29：路徑規劃結果圖

陸、討論

根據本研究將都市清運排程系統分成基礎裝置設計及雲端服務兩個部分來進行討論：

一、基礎裝置設計：

(一) 超音波模組：經過實驗之後，雖然參考網路上的資料，它能偵測距離的範圍是 2cm~400cm，但是實際上能偵測的距離是 3cm~100 多 cm。如果物體的距離小於 3cm，它會判定這個物體在很遠的地方，使數值顯示出來 1000 多 cm。

(二) 伺服馬達：因為我們買的是腳踏型垃圾桶，所以原本是要利用輔助器具去推垃圾桶蓋，但是所需的力太大，而且伺服馬達的扭力不足，所以沒辦法使垃圾桶打開，後來我們去買 2kg 扭力的伺服馬達，但是也有實驗過，結果也是沒辦法，所以不用輔助器具，讓伺服馬達直接裝在挖的空間裡面。

二、雲端服務：

(一)ESP8266：一開始在做這個實驗的時候，花了蠻多的時間才順利地連上網路，連成功之後就可以和我們的超音波模組做結合，然後這塊板子上面的腳位跟其他板子不太一樣，所以要特別記下來，在做實驗的途中，做出來的這

個資料又可以結合我們的 Node-Red，寫三段程式讓我們的 Node-Red 可以得到訊息。

(二)垃圾量的百分比：我們一開始用超音波模組測距離，用的單位是公分，但在 Node-Red 介面，所顯示的圖畫是用百分比來呈現，如果要換算的話，距離大概 12 公分，就換算成百分比為 30%；距離大概 7 公分，就換成百分比為 60%；大概距離 3 公分，就換成百分比為 100%，因為超音波模組如果距離小於 2 公分，它會顯示錯誤。

(三)Google Map API：路徑規劃可以幫助清潔人員以最有效率的情況下收垃圾，除此之外，它有很多好處，例如：可以在地圖上標示出附近的垃圾桶在哪裡，如果垃圾桶滿了，這樣子更方便通知給清潔人員精確的位置，還有它還可以隱藏地圖上不需要的資訊，讓整個地圖變得更清楚，清運方式規劃愈周密愈能得到民眾支持，施政成功率將大為提昇。

柒、結論

為了做到能夠省時且環保的垃圾桶，本研究設計的方向是能做到一個能夠回報即時訊息且能自主規劃路徑的智慧垃圾桶，也因為這樣本專題整合以 LineBot、Google API 及 Node-Red 等雲端物聯的技術，整合功能如下：

(一) 智慧垃圾桶：偵測手部距離還有得知內部垃圾量並自動開啟垃圾桶。

(二) LINE Bot：把垃圾桶內的垃圾量即時發送回傳給清潔人員，使清潔人員方便清理。

(三) Google API：為了回收垃圾量已滿且不同位置的垃圾桶，所以讓 API 去計算回收垃圾的最短路徑，以節省時間和減輕垃圾車的油耗量。

(四) Node-Red：連結 Wifi 板使垃圾桶能連上網路，並負責接收訊息，並模擬多個垃圾桶去觀察其他垃圾量情形。

此垃圾桶的功用是透過 Node-Red 顯示垃圾量，並由 Line Bot 即時回傳訊息至附近的清潔人員，再運用 API 規劃回收各地已滿垃圾桶的最短路徑，使清潔人員能迅速掌握情況，這樣的話，能節省垃圾車的油耗量，也因為 API 計算出了最短路徑，達到省時和環保的概念。

本研究未來將採用獨立電池配合太陽能電池供電及充電，如此就不用配接市電，亦可節能減碳。將本研究的運作模式亦可套用到可回收的資源回收桶。垃圾桶連結到網路有距離上的限制，盡量讓裝置放在 WiFi AP 的中心點，才不會有網路延遲的問題。

捌、參考文獻資料

曾吉弘譯(2019)。實戰物聯網開發：使用 ESP8266。碁峰資訊

臺北市行人專用清潔箱。https://data.gov.tw/dataset/121355。政府資料開放平臺。

CH.Tseng (2015)。DIY – Arduino 智慧型垃圾桶。引用自
https://chtseng.wordpress.com/2015/11/28/automatic-trash-can-docx/

台南市教育局科技教育網。**超音波原理**。引用自
http://maker.tn.edu.tw/modules/tad_book3/page.php?tbdsn=201

By Kiran Daware 。How Does A Servo Motor Work?。引用自
https://www.electricaleasy.com/2015/01/how-does-servo-motor-work.html

大年君(2018)。什麼是 PWM 信號，如何實現 PWM 信號輸出？。引用自
https://kknews.cc/zh-tw/news/xq2a8mo.html

Time-distance matrix and three ways to use it. from
https://www.geoapify.com/time-distance-matrix-and-three-ways-to-use-it

Oxxo(2018)。Google Maps API - 路線規劃。引用自
https://www.oxxostudio.tw/articles/201810/google-maps-19-directions.html

Oxxo(2017)。Google Maps API - 網頁載入地圖 (起手式)。引用自
https://www.oxxostudio.tw/articles/201707/google-maps-1.html

w3schools。HTML。引用自
https://www.w3schools.com/

MQTT 入門介紹
https://www.runoob.com/w3cnote/mqtt-intro.html

政府開放資料平台
https://data.gov.tw/dataset/121355

【評語】052507

本作品設計一個可以感測容量及感應開闔的垃圾桶，並搭配雲端控制系統，決定何時調派人力進行回收，並規劃最短路徑。此作品完整度尚屬完整，實作與理論兼具。建議未來可以在垃圾桶容量量測精準度上，增加實驗資料與創新方法。

作品簡報

中華民國第61屆全國中小學科學展覽會

雲端都市垃圾資源回收警示及清運排程系統之研究

組別：高級中等學校組
科別：電腦與資訊學科

前言：研究動機

常常看到行人道清潔箱滿出

沒人即時回收造成環境髒亂

如果清潔箱可偵測容量

雲端

將清潔箱容量資訊透過
雲端統整訊息給清潔隊

預知清潔箱
的容量資訊

即時通知
立即行動
提高效率

提高市政
服務品質

前言：研究構想

傳統固定清運排程

定時清運

清潔箱的容量資訊完全不知

必須檢查並清理每個垃圾桶

雲端動態清運排程

定時清運

即時警訊

規劃最佳清運路徑

僅清運超過80%的清潔箱

研究方法：基礎裝置設計

製作具有容量偵測及人體感測的清潔箱取代傳統的清潔箱，透過的公開的容量資訊提供給清潔員或大眾應用

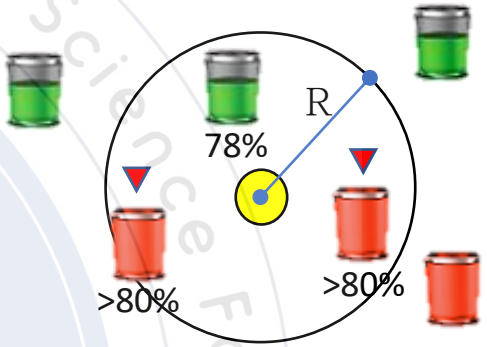
研究方法：Line警示訊息

以隊員座標搜尋發出訊息附近容量80%以上的清潔箱

篩選附近已滿的清潔箱

篩選方法

以清潔隊員為圓心搜尋半徑 R 範圍內的清潔箱



▼ 被篩選到的清潔箱

Webhook

Line Bot

>80%
清潔箱

隊員座標

清潔隊員

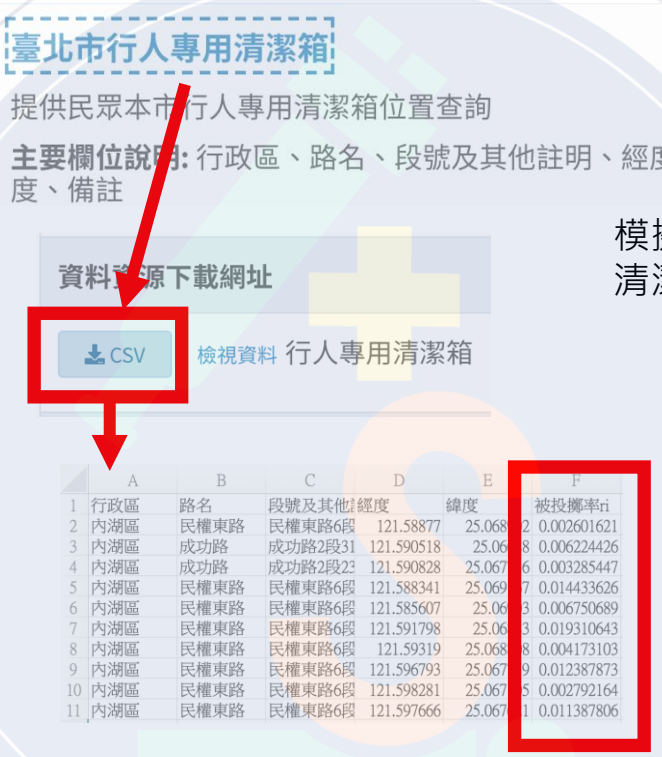
研究方法：雲端都市垃圾清運模擬資料

探討清運效能是否優於傳統，以台北市進行整體的垃圾清運模擬。

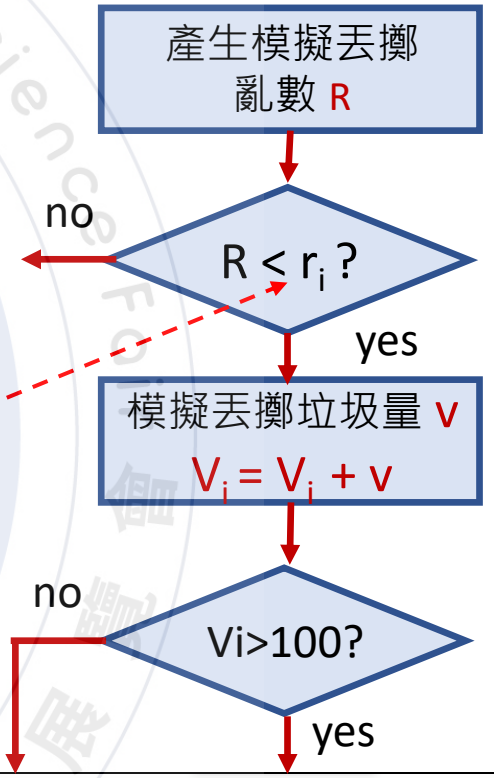
應用開放資料平台
12區 1774筆 清潔箱
資料含地理座標

亂數定義每個清潔箱的被丟擲垃圾的機率 r_i，依地區人流模擬不同的垃圾增加量

產生的亂數R小於 r_i 時表示丟擲垃圾，則亂數產生垃圾量v，累加各清潔箱的容量Vi



	A	B	C	D	E	F
1	行政區	路名	段號及其他註	經度	緯度	被投擲率ri
2	內湖區	民權東路	民權東路6段	121.58877	25.068 2	0.002601621
3	內湖區	成功路	成功路2段31	121.590518	25.06 8	0.006224426
4	內湖區	成功路	成功路2段23	121.590828	25.067 6	0.003285447
5	內湖區	民權東路	民權東路6段	121.588341	25.069 7	0.014433626
6	內湖區	民權東路	民權東路6段	121.585607	25.06 3	0.006750689
7	內湖區	民權東路	民權東路6段	121.591798	25.06 3	0.019310643
8	內湖區	民權東路	民權東路6段	121.59319	25.068 8	0.004173103
9	內湖區	民權東路	民權東路6段	121.596793	25.067 9	0.012387873
10	內湖區	民權東路	民權東路6段	121.598281	25.067 5	0.002792164
11	內湖區	民權東路	民權東路6段	121.597666	25.067 1	0.011387806



道路	路段	分區	容量	滿箱	使用次數
中山北路	五段6號圓山保齡球館停車出口南...	士林區		✕	72
中山北路	五段380號北側	士林區		✕	65
中山北路	五段190巷對面	士林區		✕	61
中山北路	六段785號(日本學校)(一)	士林區		✕	59

研究方法：最佳路徑規劃

最佳路徑規劃前

最佳路徑規劃後

各清潔箱座標 $P = [P_0 \quad P_1 \quad \dots \quad P_N]$

各清潔箱容量 $V = [V_0 \quad V_1 \quad \dots \quad V_N]$

將P 代入***DistanceMatrix API*** 的***origins***

及***destinations*** 產生距離矩陣 D、時間矩陣T

$$D = \begin{bmatrix} d_{oo} & d_{o1} & \dots & d_{oN} \\ d_{1o} & d_{11} & \dots & d_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ d_{No} & d_{N1} & \dots & d_{NN} \end{bmatrix} \qquad T = \begin{bmatrix} t_{oo} & t_{o1} & \dots & t_{oN} \\ t_{1o} & t_{11} & \dots & t_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ t_{No} & t_{N1} & \dots & t_{NN} \end{bmatrix}$$

清潔隊清運路徑一定從起點P_0 到起點 P_0

假設路徑為$P_0 \rightarrow P_2 \rightarrow P_3 \rightarrow P_0$

則其距離 $D_T = Sum([d_{02}\ d_{23}\ d_{30}])$

則其時間 $T_T = Sum([t_{02}\ t_{23}\ t_{30}])$

容量 $V_T = Sum([V_2\ V_3])$

$$Cost = \frac{V_T\ (\text{總容量})}{D_T(\text{總距離}) \times T_T(\text{總耗時})}$$

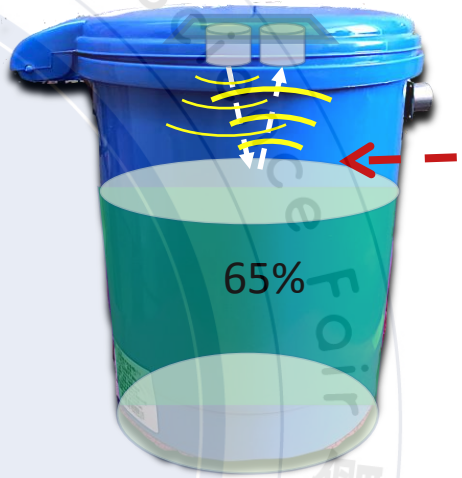
以距離優先列舉所有路徑取 ***Cost*** 最大者為最佳路徑

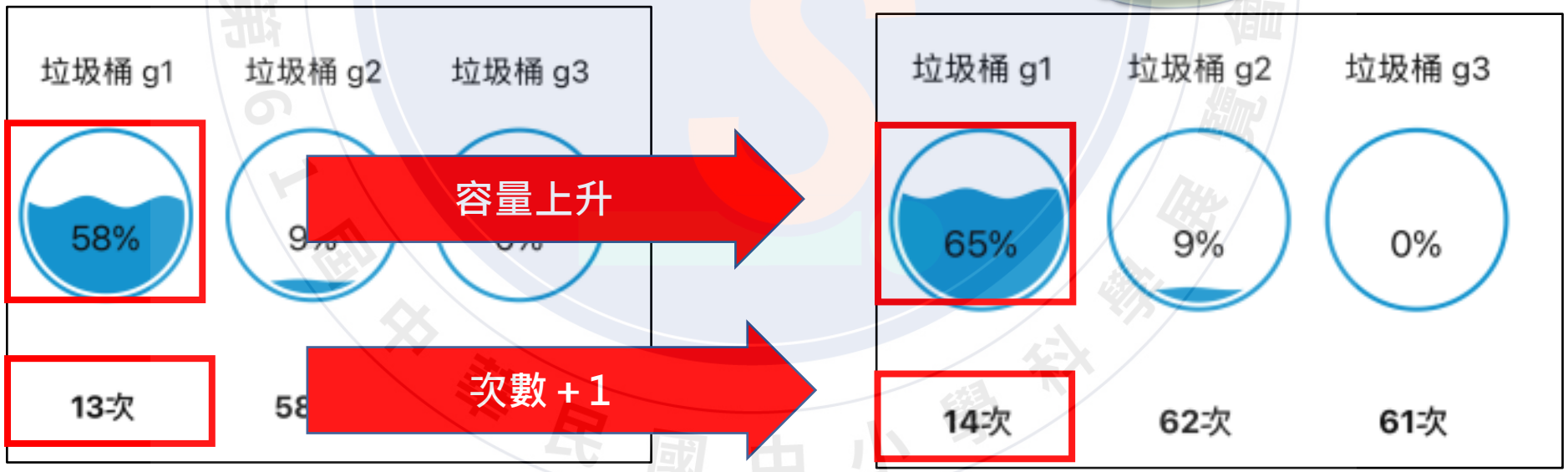
研究結果：基礎裝置測試及訊息發送測試

上傳使用次數

開啟上蓋

上傳使用容量





投入垃圾前

投入垃圾後

研究結果：Line警示訊息測試

清潔隊員發送查詢，依隊員座標搜尋那些清潔箱已滿需要回收

研究結果：最佳路徑規劃

固定路徑：
每個清潔箱拜訪並收取

動態最佳路徑規劃：
容量少的清潔箱可以不收

容量太少選擇不收

距離(distance):	8.86 公里
時間(duration):	28.83 分鐘
總容量(vol):	799.28
成本效益(cost):	187.79

距離(distance):	7.13 公里
時間(duration):	22.82 分鐘
總容量(vol):	522.16
成本效益(cost):	192.53

討論：系統模擬運作效能評估

趟次	容量	固定路徑			動態規劃		
	vol	distance' (km)	duration' (min)	cost'	distance (km)	duration (min)	cost
1	530.0	8.8	22.0	164.3	6.3	18.0	280.4
2	470.0	8.8	22.0	145.7	6.0	19.0	247.4
3	390.0	8.8	22.0	120.9	5.4	12.0	361.1
4	500.0	8.8	22.0	155.0	7.8	20.0	192.3
5	630.0	8.8	22.0	195.3	8.8	22.0	195.3
平均	504.0	8.8	22.0	156.2	6.9	18.2	255.3

成本效益

65%

計算區間	固定路徑		動態規劃	
	機車	汽車	機車	汽車
一日	53元	317元	41元	248元
一週(5工作日)	265元	1585元	205元	1240元
一年(260工作日)	13780元	82420元	10660元	64480元

油耗成本節省

22%

結論

本研究以雲端都市為藍圖，建構人行道清潔箱清運警示及排程系統：

1. 可即時通報附近的清潔人員，可提供更好的都市清潔品質。
2. 可動態規劃最佳的清潔箱垃圾清運路徑，可降低營運成本。
3. 以本例的一個清潔隊分隊估計可以省下2萬元油耗，若以台北市估計50個清潔分隊一年將可以在油耗上省下近100萬元的油耗。

未來展望

1. 採用獨立電池配合太陽能電池供電及充電不需配接市電。
2. 將本研究的運作模式亦可套用到可回收的資源回收桶。
3. 垃圾桶連結到網路有距離上的限制，盡量讓裝置放在WiFi AP的中心點，才不會有網路延遲的問題。
4. 若納入悠遊卡或一卡通，可以建立使用者付費的機制。

參考資料來源

臺北市行人專用清潔箱。政府資料開放平台。https://data.gov.tw/dataset/121355