國立虎尾科技大學
資訊管理系

專題製作報告

力與速
Power and Speed

中華民國 106 年 12 月

國立虎尾科技大學
資訊管理系
專題製作報告

力與速 Power and Speed

成果發表
評審委員：_____ _____ _____

指導老師：胡念祖 老師、游立椿老師

學　　生：**40341102** 呂倩綾

40341107 張瑞珊

40341108 許巧欣

40341112 程佩璇

40341141 廖耕瑲

中華民國 106 年 12 月

力與速 Power and Speed

摘要

田徑，是所有運動項目的根本，故有運動之母之稱號。選手為了達到更好的成績表現，需仰賴教練的經驗指導才能達到最理想的起跑模式。若要搭配市售田徑運動分析平台蒐集選手表現的量化數值，而所需設備均要價不斐，一般基層學校很少有經費可購入使用。

因此本研究建立一套田徑短距離起跑模式與即時回饋平台。以「降低成本、簡化步驟、快速架設、輕鬆測試」為目標，並搭配近年來科技趨勢「物聯網」進行樹莓派、Windows 系統及各項感測裝置之整合。以簡單的設備讓使用者一個人即可佈設完畢也不需透過穿戴式裝備就能測得各項數據，並透過 Django 架設網頁利用 Highcharts 圖表方式將資料視覺化呈現，讓使用者可以透過手持式裝置即時了解數據意義，並做進一步調整訓練計畫。

有鑑於田徑老師在訓練選手時所遇到的培育困難，本研究期望藉由物聯網技術結合田徑科學，與體育進行異界合作，為田徑教練及選手帶來更科學的訓練方式。期望未來這套系統可以廣泛應用在各單位，不論是基層學校或者專業訓練團隊都能使用這套系統，將田徑科學之觀念深植於各大田徑相關單位。

誌謝

感謝專題指導老師 胡念祖老師，在專題製作時給了我們許多寶貴的建議及指導，讓我們將一開始的發想實踐成一個專題計劃。在我們對於專題感到徬徨時，不斷的給我們鼓勵，讓我們更有信心去解決專題所遇到的困難，並且順利地完成此篇專題。也感謝虎尾科大田徑隊指導教練游立椿老師，為我們提供了田徑方面相關知識及技術的指導，並且帶領田徑隊替我們的專題進行更精準的測試。

此外，也要感謝這一年多來協助我們完成專題的學長們。謝謝博元學長及勇任學長，總是在我們迷茫的時候為我們指點方向，不辭辛勞地為我們提供了技術及文件撰寫上的協助，不論白天或者晚上皆即時地提供我們解決的方式。也謝謝景翔學長，在測試時，總是作為我們的測試對象，不辭辛勞地為我們進行一次又一次的實測。再次謝謝學長們在學業繁忙之餘，為我們提供了最即時的幫助，讓我們能夠順利地完成專題，謝謝您們。

最後，感謝一起製作專題的好夥伴，因為有你們的包容及體諒，才能讓這專題順利的完成。

呂倩綾、張瑞珊、許巧欣、程佩璇、廖耕瑲　謹誌於

國立虎尾科技大學資訊管理系

中華民國 106 年 12 月

力與速 Power and Speed

目錄

頁數

摘要 i

誌謝 ii

目錄 iii

圖表目錄 v

第一章 導論 1

1.1 研究動機與研究問題 1

1.2 研究方法及架構 3

第二章 文獻探討 4

2.1 田徑與百公尺加速度 4

2.2 物聯網 6

2.3 樹莓派 7

2.4 Linux Debian 7

2.5 Python 8

2.6 Highcharts 9

2.7 OpenCV 10

2.8 Socket 11

2.9 FTP 12

2.10 C# 13

2.11 Django 14

第三章 研究設計 15

3.1 系統架構 15
3.2 資料來源收集 17
3.3 製作 Highcharts 圖表 19
第四章 系統實作 20
4.1 系統環境 20
4.2 系統實作 21
4.2.1 RPI 操作 21
4.2.2 Microsoft Visual Studio 操作 24
4.2.3 SQL Server 操作 34
4.2.4 手持式智慧裝置操作 37
第五章 結論與未來研究方向 43
5.1 結論 43
5.2 未來研究方向 44
參考文獻 45
作者簡介 47

圖表目錄

頁數

圖 3.1 系統架構 17
圖 3.2 左腳壓力感測數值 18
圖 3.3 右腳壓力感測數值 18
圖 3.4 樹莓派所感測之速度數值 19
圖 3.5 Highcharts 選擇圖表樣式頁面 19
圖 4.1 系統開發環境 20
圖 4.2 Raspberry Pi 3 Model B 21
圖 4.3 Rpi 22
圖 4.4 Rpi_socket 23
圖 4.5 Rpi_thread 23
圖 4.6 C#_runner 24
圖 4.7.1 Cost Time button 25
圖 4.7.2 Save Numbers button 26
圖 4.7.3 Start Time button 26
圖 4.7.4 End Time button 26
圖 4.7.5 React Time button 27
圖 4.8.1 ssh button 28
圖 4.8.2 ssh button 程式碼 28
圖 4.8.3 Rpi_status button 28
圖 4.8.4 Rpi_status button 程式碼 29
圖 4.8.5 Rpi_py_kill button 29

圖 4.8.6 Rpi_py_kill button 程式碼 29
圖 4.8.7 Camera button..... 30
圖 4.8.8 Camera_finish 程式碼 30
圖 4.8.9 Camera_mph 程式碼 31
圖 4.9.1 Start button..... 32
圖 4.9.2 C#視窗所呈現雙腳足部壓力資料..... 32
圖 4.9.3 Ready button..... 32
圖 4.9.4 Go button..... 33
圖 4.9.5 Stop button..... 33
圖 4.9.6 Run Django button..... 33
圖 4.10 C#_data_recv 視窗 34
圖 4.11 左腳壓力數值..... 35
圖 4.12 右腳壓力數值..... 36
圖 4.13 測試速度數值..... 36
圖 4.14 查詢視窗..... 37
圖 4.15.1 選擇跑者 ID 選單..... 38
圖 4.15.2 選擇該跑者測試 ID 選單..... 39
圖 4.16.1 畫圖囉按鈕..... 39
圖 4.16.2 畫圖囉按鈕顯示左腳壓力數值..... 40
圖 4.16.3 畫圖囉按鈕顯示右腳壓力數值..... 40
圖 4.16.4 查圖囉按鈕..... 41
圖 4.16.5 查圖囉按鈕顯示畫面..... 41
圖 4.16.6 起跑圖按鈕..... 42

圖 4.16.7 起跑圖按鈕顯示畫面 42

第一章 導論

1.1 研究動機與研究問題

近年來隨著各大體壇賽事的興起,各項運動競賽越來越受到矚目。尤其是田徑，田徑是奧運的核心項目，也是奧運最多獎牌的項目。田徑運動來自人類的基本體能活動，選手較量的是速度、力量、體力及耐力，比賽的規則簡單，卻最能直接觸動人心。而在眾多田徑項目中，又以百公尺項目最為人津津樂道。而這些在運動場上的田徑選手們，都是從小經過各項訓練才得以上場發揮他們的實力。但現今的學校單位或者田徑訓練單位，因為田徑運動分析平台過於昂貴而未採購，教練使用傳統的田徑訓練模式利用肉眼進行觀察以及過去的經驗做為訓練選手的依據，使許多選手從小就接觸了不適合自己的訓練方法。此外，有新進學員時，教練因為不清楚學員過去的訓練狀況，所以會使用主觀化的訓練建議做為訓練選手的依據，此種狀況也會對選手造成運動傷害。

為此，本研究以物聯網技術結合田徑為發想創建「田徑短距離起跑模式與即時回饋平台」,此系統有別於目前市售昂貴的田徑運動分析平台,不僅可以降低設備成本且選手不需穿戴特別裝置即可輕鬆測試，而且也能利用測試時之數據研究探討百公尺「起跑出發階段」與「加

速度階段之關係」。將結合起跑架與壓力感測器，偵測選手在起跑時的足部壓力分佈與起跑架架攝錄影設備，讓教練可參考足部發力狀態及選手起跑狀態去調整選手起跑時的姿勢。本研究亦會使用物聯網無線感測裝置—樹莓派整合攝影機元件設計速度偵測之設備，可以即時拍攝選手起跑後加速度時的區間瞬時速率，並且利用無線基地台作為溝通回傳至資料庫。最後以 Highcharts 製成圖表讓使用者能利用圖表的即時呈現，可以即時查詢得知選手的訓練狀況，讓教練更清楚且容易發現數據間的變化，得以做出更好的決策判斷。而這些數據將存於 SQL Server 中，讓教練及選手之後可以透過智慧型手持式裝置進行查詢。

此系統將統整從起跑到終點所使用的各項田徑數據偵測設備，改善以往因為偵測設備過於零散不易架設的問題。並且整合動力學及運動學所需數據，提供整理數據分析功能。此外，此系統不僅可以降低以往田徑運動分析平台昂貴的建置成本，也能降低偵測設備的架設時間，讓教練及選手可以更充分的運用訓練時間。

我們希望藉由此系統，可以讓教練及選手免去以往田徑運動分析平台繁雜的架設過程以及移除過度繁雜的穿戴式設備減輕選手測試時的負擔。並且在測試當下可以獲取最新的測試資料回饋至網頁，讓選手及教練可以運用更科學化的方式進行訓練計畫的改進。

1.2 研究方法及架構

本研究透過文獻探討，將建立系統的方法步驟，分為資料收集、視覺化呈現、成果展示等作為本研究方法及架構：

(一)資料收集

首先，本研究由(1)樹莓派、(2)壓力感測器、(3)特徵資料、(4)訓練資料所蒐集之數據，匯入 SQL Server 的資料庫。

(二)視覺化呈現

使用 Django 架設網站，將 SQL Server 內數據以 Highcharts 技術製作圖表，讓使用者得以利用圖表，更明確地看出左右腳在壓力感測器上之關係，亦放上圖片讓使用者了解到其預備姿勢。除此之外，每一個通過點的速率及姿勢，也以圖片的方式來呈現。而圖表方式的呈現可更清楚且容易發現資料間的變化，以便讓田徑教練能做出更好的決策判斷。

(三)成果展示

教練可以透過下拉式選單選擇跑者代號與跑者測試代號，查詢該次測試者的足部壓力數值所製作的可視化圖表，讓田徑教練可以更簡易從圖表中了解跑者在起跑時的準備時間與足部壓力分佈之關係，也可以查看測試者在通過每一個樹莓派時的速率。藉此針對各選手的競賽狀況進行調整。

第二章 文獻探討

本章將在第一節中回顧田徑與百公尺加速度相關文獻；從第二節始將回顧物聯網及本研究所使用相關技術之相關文獻。

2.1 田徑與百公尺加速度

田徑是奧運的核心項目，也是奧運最多獎牌的項目。在 1896 年第一屆現代奧林匹克運動會，就有田徑這個項目。田徑運動來自人類的基本體能活動，選手較量的是速度、力量、體力及耐力，比賽的規則簡單，卻最能直接觸動人心。而在眾多田徑項目中，又以百公尺項目最為人津津樂道。四年一次的夏季奧運會，誰是全球跑的最快的男人跟女人，往往是賽會最矚目的焦點所在。

百公尺技術分為四個階段：一、起跑出發階段；二、起跑後加速度階段；三、最高速度維持階段；四、衝線階段，本研究的主要目的在探討第一階段與第二階段之銜接性問題。許樹淵提到在田徑百公尺競賽，選手一踏上比賽跑道，選手第一個動作就是將起跑架拿起來後再調整自己的起跑架模式，目的是能在起跑出發階段獲得最大的水平速度。百公尺起跑出發階段次跑者對起跑架水平推蹬產生的反作用力，推動身體向前，短暫的反應時間和巨大的推進力為快速起跑的必要條件。[1]所以，起跑架模式調整得宜，在百公尺比賽中，是非常重

要的一環技術表現。而在起跑後加速度階段，當跑者前腳蹬離踏板後，即進入加速度階段，為避免加速度產生的過程中造成減速的狀態發生，加速度的延伸性技術往往是跑者最重要的技術課題所在。

Harland 及 Steele 兩位學者曾提過蹲踞式起跑是田徑短距離非常重要的技術之一，目前針對起跑模式的模式中，不外乎短式、中式、長式三種起跑模式，三種起跑模式各具其優缺點，整理如下：

	短式	中式	長式
重心高度	A	B	C
反應時間	A	B	C
身體平衡狀態		A	
身體放鬆狀態	C	B	A
踏板反作用力	C	B	A
手臂支撐	C		

短式起跑適用於肌力強且爆發力大的選手，長式起跑適用於身材高肌力爆發力小的選手，中式則適用於兩者之間的選手，但是，每位選手都是獨一無二的個體，各肢段長度也不同，若能明確依肢段長度

找出選手所使用的起跑模式，絕對有助成績表現。而好的起跑模式對於起跑後加速度階段的銜接也存在著高度關聯性，起跑模式的好壞，似乎已決定起跑後加速度的銜接。[2]

2.2 物聯網

物聯網是網際網路、傳統電信網等資訊承載體，讓所有能行使獨立功能的普通物體實現互聯互通的網路。物聯網一般為無線網，而由於每個人周圍的裝置可以達到一千至五千個，所以物聯網可能要包含500 兆至一千兆個物體。在物聯網上，每個人都可以應用電子標籤將真實的物體上網聯結，在物聯網上都可以查出它們的具體位置。通過物聯網可以用中心電腦對機器、裝置、人員進行集中管理、控制，也可以對家庭裝置、汽車進行遙控，以及搜尋位置、防止物品被盜等，類似自動化操控系統，同時透過收集這些小事的資料，最後可以聚整合大資料，包含重新設計道路以減少車禍、都市更新、災害預測與犯罪防治、流行病控制等等社會的重大改變。[3]

物聯網將現實世界數位化，應用範圍十分廣泛。物聯網拉近分散的資訊，統整物與物的數位資訊，物聯網的應用領域主要包括以下方面：運輸和物流領域、健康醫療領域範圍、智慧型環境（家庭、辦公、工廠）領域、個人和社會領域等，具有十分廣闊的市場和應用前景。[3]

2.3 樹莓派

本研究主要以樹莓派上結合攝影機元件利用無線基地台與Windows 應用程式進行溝通，利用 Python 藉由 OpenCV 實作瞬時速度偵測與跑者影像擷取的部分系統。

樹莓派是由 Raspberry Foundation（樹莓派基金會）這一非盈利組織進行打造。提供 1-4 個 USB 插口、HDMI 接口、網絡接口、音頻接口，使用 Python 作為主要語言，同時還支持 C、C++、PHP、Java、Perl、Ruby、Squeak Smalltalk 在內的各種編程語言，再加上極低的售價，讓 Raspberry 成為了編程入門者的最好教材，同時也成為了很多智能設備最簡便的大腦之選。[4]

2.4 Linux Debian

Debian 系統目前採用 Linux 核心或者 FreeBSD 核心，而 Linux 是一個最初由 Linus Torvalds 創建，目前由全球成千上萬的程式師共同維護的軟體，FreeBSD 是一個包括核心和其它軟體的作業系統。[5]

然而，讓 Debian 支持其他核心的工作也正在進行，最主要的就是 Hurd。Hurd 是一組在微核心（例如：Mach）上執行的、提供各種不同功能的伺服器。Hurd 是由 GNU 計畫所設計的自由軟體。[5]

這個作業系統中的大部分基本工具來自於 GNU 計畫；因此我們把它們命名為 GNU/Linux、GNU/kFreeBSD 和 GNU/Hurd。這些工具同樣都是自由的。[5]

Debian 帶來了超過 51000 個套件，為了能在電腦上輕鬆地安裝，這些套件都已經被編譯包裝為一種方便的格式，一個管理器（APT），幫助電腦上管理數千個的工具，過程就如安裝一個應用程式那麼簡單。而這些全都是自由軟體。[5]

2.5 Python

Python 自 1990 年由 Guido van Rossum 在荷蘭的 CWI 開始發展以來，從 0.9 進步到今天的 2.4.2，不但累積了相當完整的標準程式庫(模組)，更有無以計數的非標準模組，而且絕大部分都是開放原始碼的。單以內建的模組來講，從簡單的數學運算、字串處理、網際網路協定連線、網際網路資料處理、各種壓縮格式，以及 POSIX 與主要作業系統的支援功能等等，含括的範圍非常地廣泛。[6]

各種主要的作業系統都支援 Python。Python 程式常常不需要修改，便可以同時在 Linux 與 Windows 平台上執行，即使撰寫 GUI 程式(透過 PyGTK，wxPython 等 binding)也是一樣。所撰寫的 Python 程式透過標準的 distutils(模組)進行包裝後，用標準的方式即可安裝於各種平台；

在 Windows 下更可以自動產生方便的可執行 installer。[6]

Python 算是執行效率不錯的直譯式語言，但畢竟比不上 C 和 Fortran。然而只要我們想，大可以 C/C++或 Fortran 撰寫高效率的模組；這些模組的使用方式，與內建模組以及用 Python 撰寫的模組完全一樣。最好的是，撰寫的方法並不困難。[6]

2.6 Highcharts

吳孟春&丁嵐提出 Highcharts 為挪威 Highsoft AS 於 2009 年推出的一項圖表工具產品[7]，圖表庫裡的圖表皆使用 JavaScript 語言，提供簡單的使用方法讓網頁或應用程式增加互動性，可很方便的為 Web 應用添加直觀的、動態互動式圖表，支援 IE、火狐、Google Chrome 等主流的瀏覽器以及 iPhone/iPad 手機或平板電腦桌面顯示，在 iOS 和 Android 下，支持多點觸控提供了一個無縫的用戶體驗。[8]

它有一個很重要的特性那就是可以對源碼進行下載、編輯，不需要安裝其他插件，只需要兩個 JS 檔即可執行，不受版權限制，可在個人使用、學校團體或非營利組織下遵守創用 CC 規則的話皆可免費使用，實現真正意義上的代碼開源。[9]

由於是純 Java 語言做腳本實現，故不需要在用戶端安裝外掛程式，也不需要事先在伺服器部署即可實現多功能圖表。支持折線圖、

圓形圖、柱狀圖、時速表等多樣圖表類型，並可把多類型圖表集合成在同一個圖表上顯示，更是同時支援極地圖表轉換與多軸數據來做比較。[9]

2.7 OpenCV

OpenCV 全名是 Open Source Computer Vision Library，是一個影像處理函式庫，由 Intel 發起並參與開發，以 BSD 授權條款發行，可在商業和研究領域中免費使用，目前是非營利的基金組織 OpenCV.org 在維護。[10]

OpenCV 1.0 版於 2006 年釋出，以 C 語言作為開發主體，當使用 OpenCV 函式庫時，程式設計師要自行注意記憶體管理，因此在開發大型程式時較不方便。[10]

OpenCV 在影像處理方面應用廣泛，可以讀取儲存圖片、視訊、矩陣運算、統計、影像處理等，可用在物體追蹤、人臉辨識、傅立葉轉換、紋理分析、動態視訊的影像處理等。[10]

OpenCV 提供簡單的 GUI 介面，像將影像顯示在螢幕上，在視窗上加上滑動桿和偵測滑鼠和鍵盤輸入，方便我們驗證或呈現結果。但因為 OpenCV 當初設計的時候著重在演算法的處理，所以關於系統硬體的

支援，和介面元件的完整度都不高，所以假使想要開發完整的 C/C++ 應用程式，還是需要像 Qt、wxWidets 之類的應用程式框架。[10]

OpenCV 提供的函式方便我們推演更進階的影像處理演算法，就好像 MATLAB 的功用，但是執行速度比 MATLAB 快上許多，通常也比我們自己用 C/C++寫的函式還快，而除了 C/C++之外，OpenCV 也提供其他語言的支援，像 Java 或 Python 等。[10]

2.8 Socket

Socket 是一組具體的 UNIX 系統呼叫，是 80 年代美國的研究單位研究在 UNIX 中接收 TCP／IP 軟體的問題，並使其亦適應於其他場合。所以研究者們創造一種新的通訊介面，盡可能使用現有的 UNIX 系統呼叫。為支援那些不易被整合於現有函式庫的 TCP／IP 函數，新定義了一些系統呼叫函數，這便是承接口介面(Socket Interface) 至今已被廣泛的認可與應用，並成為一種標準。[11]

在作業系統中，通常會為應用程式提供一組應用程式介面（API），稱為插座介面（英語：socket API）。應用程式可以通過插座介面，來使用網路插座，以進行資料交換。[12]

最早的插座介面來自於 4.2 BSD，因此現代常見的插座介面大多源自 Berkeley 插座（Berkeley sockets）標準。在插座介面中，以 IP

位址及通訊埠組成插座位址（socket address)。遠端的插座位址，以及原生的插座位址完成連線後，再加上使用的協定（protocol)，這個五元組（five-element tuple)，作為插座對（socket pairs)，之後就可以彼此交換資料。[12]

例如，再同一台電腦上，TCP 協定與 UDP 協定可以同時使用相同的 port 而互不干擾。作業系統根據插座位址，可以決定應該將資料送達特定的行程或執行緒。這就像是電話系統中，以電話號碼加上分機號碼，來決定通話對象一般。[12]

2.9 FTP

FTP 全名為 File Transfer Protocol (client and server)，它是一種獲得網際網路世界普遍採用的通訊協定之一。FTP 協定是由一組非商業組織的學者們，在 1985 年的時候提出了一種開放的協定，提供給想要製作檔案傳輸相關應用的電腦軟體設計者們參考，讓大家可以依照這個標準，獨立製作出支持 FTP 協定的檔案傳輸軟體，卻又可以確保互相能夠相容。[13]

FTP 是一種主從式的架構，也就是 Client and Server 架構(主從式架構)，這個協定要運行一定由 FTP Server 及 FTP Client 組成，唯有這兩組軟體搭配，才能達成 FTP 檔案傳輸的功效。[13]

FTP Server 就像是錄影帶出租店，裡面放著很多錄影帶(想像為電腦中的檔案)，等著客戶來租借或是還片，錄影帶店的客人就像是 FTP Client，會到錄影帶出租店中借影帶(想像為 FTP 中的下載動作)或是把片子還給出租店(想像為 FTP 中的上傳動作)。[13]

2.10 C#

C#是微軟公司發布的一種面向對象的、運行於.NET Framework 之上的高級程序設計語言。並定於在微軟職業開發者論壇(PDC)上登台亮相。C#是微軟公司研究員 Anders Hejlsberg 的最新成果。C#看起來與 Java 有著驚人的相似；它包括了諸如單一繼承、接口、與 Java 幾乎同樣的語法和編譯成中間代碼再運行的過程。但是 C#與 Java 有著明顯的不同，它藉鑑了 Delphi 的一個特點，與 COM（組件對像模型）是直接集成的，而且它是微軟公司.NET windows 網絡框架的主角。[14]

C#是一種安全的、穩定的、簡單的、優雅的，由 C 和 C++衍生出來的面向對象的編程語言。它在繼承 C 和 C++強大功能的同時去掉了一些它們的複雜特性。C#綜合了 VB 簡單的可視化操作和 C++的高運行效率，以其強大的操作能力、優雅的語法風格、創新的語言特性和便捷的面向組件編程的支持成為.NET 開發的首選語言。[14]

力與速 Power and Speed

C#是面向對象的編程語言。它使得程序員可以快速地編寫各種基於 MICROSOFT .NET 平台的應用程序，MICROSOFT .NET 提供了一系列的工具和服務來最大程度地開發利用計算與通訊領域。[14]

C#使得 C++程序員可以高效的開發程序，且因可調用由 C/C++編寫的本機原生函數，因此絕不損失 C/C++原有的強大的功能。因為這種繼承關係，C#與 C/C++具有極大的相似性，熟悉類似語言的開發者可以很快的轉向 C#。[14]

2.11 Django

Django 是一個開放源代碼讓 Web 開發工作更加有高效率的 Web 應用框架，由 Python 寫成，利用最小的代價建構出高質量的 Web 應用程式，並以最小的成本維護。採用了 MVC 的框架模式，即模型 M，視圖 V 和控制器 C。它最初是被開發來用於管理勞倫斯出版集團旗下的一些以新聞內容為主的網站的，即是 CMS（內容管理系統）軟件。並於 2005 年 7 月在 BSD 許可證下發布。這套框架是以比利時的吉普賽爵士吉他手 Django Reinhardt 來命名的。[15]

力與速 Power and Speed

第三章 研究設計

3.1 系統架構

系統以物聯網做為開發技術，在樹莓派上結合攝影機元件利用無線基地台與 Windows 應用程式進行溝通，利用 Python 藉由 OpenCV 實作瞬時速度偵測與跑者影像擷取的系統。系統所使用之物聯網技術由應用層、網路層、感知層所組成，以下為各層架構之說明。

(一) 應用層：大量訊息經過大數據的分析後，做出反應並給予各裝置相對的指令。在此系統應用層主要是由筆記型電腦及智慧型手持裝置所構成。

1. 筆記型電腦：作為系統伺服器，作業系統為 Windows 7，並使用 C#製作圖形化操作視窗，透過 ssh 技術對樹莓派進行遠端操控，並利用 C#結合 Socket 技術進行資料間的傳輸。
2. 智慧型手持裝置：教練及選手或者其他使用者，皆可以透過連網的智慧型手持裝置進行各項數據的查詢。例如:選手在起跑時的足部壓力施力狀況、起跑後各偵測點速度及影像及起跑時之狀態錄影……等。

(二) 網路層：透過有線與無線的網際網路將訊息傳遞。在此系統感

知層主要是由無線基地台(Access Point，AP)所構成。

無線基地台:負責筆記型電腦、智慧型手持裝置及樹莓派之間的溝通連結。

（三） 感知層:物體透過辨識系統、感應技術,偵測外界環境的變化，接收各類的訊息。在此系統感知層主要由起跑架的足部壓力感測器以及起跑後偵測速度及拍攝影像用的樹莓派所構成。

1. 起跑架足部壓力感測器：利用感測器測得足部的壓力值，藉由資料擷取（Data Acquisition，DAQ）技術，將感測器所回傳之訊號經由訊號放大器將訊號放大之後，透過類比轉換器將類比訊號轉化為數值資料。
2. 樹莓派：採用以 Linux Debain 為基礎的 Raspbian 系統，使用 OpenCV 進行影像處理。將相機所拍攝到的影片進行圖片擷取過後，加上選手經過樹莓派的瞬時速度，並利用 Python 結合 Socket 與 FTP 將所獲取到的資料與照片傳回至伺服器。

本研究團隊以圖 3.1 說明物聯網之系統架構圖。

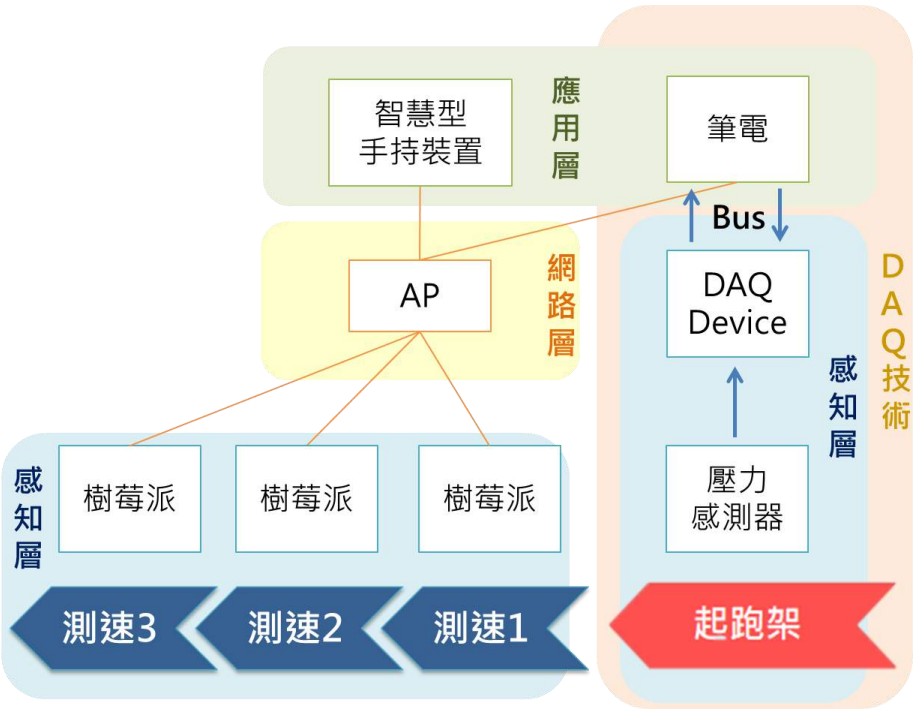


圖 3.1 系統架構

3.2 資料來源收集

本研究資料來源為選手在起跑時足部發力所測得之瞬間壓力、起跑後樹莓派所測得各階段速度、教練在選手測試前所輸入的特徵資料(如：人體各肢段長度、訓練年齡、最佳成績、慣用腳、身高體重年齡……等。)及訓練資料(如：起跑線到前後踏板距離、前後踏板角度……等。)。

力與速 Power and Speed

結果 訊息

	Runner_id	Test_id	Time	Kg1	Kg2	Kg3	Kg4	Kg5	Kg6	Kg7	Kg8	pic
1	1	1	0.98	0.029	0.039	0.114	0.056	0.027	0.021	0.054	0.112	10
2	1	1	1.47	0.044	0.066	0.122	0.051	0.049	0.021	0.041	0.112	15
3	1	1	1.96	0.063	0.1	0.077	0.051	0.07	0.021	0.014	0.09	20
4	1	1	2.45	0.048	0.072	0.07	0.062	0.07	0.033	0.014	0.101	25
5	1	1	2.94	0.029	0.039	0.144	0.045	0.027	0.01	0.054	0.112	30
6	1	1	3.43	0.067	0.106	0.092	0.022	0.081	0.01	0.054	0.067	35
7	1	1	3.92	0.029	0.039	0.099	0.051	0.027	0.021	0.041	0.112	40
8	1	1	4.41	0.048	0.072	0.048	0.045	0.049	0.055	0.014	0.112	45
9	1	1	4.42	0.052	0.079	0.099	-0.01	0.07	0.021	0.041	0.045	45
10	1	1	4.43	0.029	0.039	0.099	0.068	0.027	0.021	0.041	0.124	45
11	1	1	4.44	0.037	0.052	0.099	0.011	0.027	0.021	0.041	0.078	45
12	1	1	4.45	0.071	0.113	0.077	0.045	0.081	0.021	0.014	0.09	45
13	1	1	4.46	0.029	0.038	0.129	0.039	0.027	0.021	0.067	0.112	45
14	1	1	4.47	0.056	0.086	0.129	0.017	0.049	0.044	0.067	0.09	45
15	1	1	4.48	0.044	0.066	0.085	0.011	0.049	0.055	0.041	0.067	45
16	1	1	4.49	0.067	0.106	0.07	0.011	0.081	0.055	0.027	0.078	45
17	1	1	4.5	0.041	0.059	0.085	0	0.038	0.055	0.041	0.067	45
18	1	1	4.51	0.052	0.079	0.085	-0.01	0.049	0.044	0.041	0.045	46
19	1	1	4.52	0.033	0.045	0.077	0	0.027	0.01	0.041	0.056	46
20	1	1	4.53	0.056	0.086	0.122	0.022	0.049	0.055	0.081	0.09	46
21	1	1	4.54	0.018	0.018	0.114	0.034	0.006	-0.01	0.041	0.09	46
22	1	1	4.55	0.052	0.079	0.136	0.011	0.049	0.055	0.067	0.078	46

圖3.2 左腳壓力感測數值

結果 訊息

	Runner_id	Test_id	Time	Kg1	Kg2	Kg3	Kg4	Kg5	Kg6	Kg7	Kg8	pic
1	1	1	0.97	0.015	0.025	0.075	0.072	0.038	3.29	-0.1	0.083	10
2	1	1	1.46	0.069	0.089	0.036	0.017	0.082	3.326	-0.13	0.049	15
3	1	1	1.95	0.019	0.03	0.032	0.067	0.049	3.281	-0.14	0.083	20
4	1	1	2.44	0.01	0.02	0.055	0.034	0.038	3.281	-0.13	0.049	25
5	1	1	2.93	0.051	0.068	0.032	0.012	0.071	3.317	-0.13	0.049	30
6	1	1	3.41	0.001	0.009	0.06	0.045	0.027	3.272	-0.13	0.049	35
7	1	1	3.9	0.006	0.015	0.07	0.028	0.027	3.299	-0.11	0.06	39
8	1	1	4.4	0.001	0.009	0.06	0.045	0.027	3.281	-0.13	0.049	44
9	1	1	4.42	0.019	0.03	0.06	0.061	0.049	3.281	-0.13	0.06	45
10	1	1	4.42	0.06	0.078	0.027	0.017	0.082	3.317	-0.15	0.026	45
11	1	1	4.43	0.015	0.025	0.075	0.023	0.038	3.326	-0.1	0.049	45
12	1	1	4.44	0.065	0.083	0.041	0.028	0.082	3.29	-0.13	0.037	45
13	1	1	4.45	0.019	0.03	0.06	0.045	0.049	3.29	-0.13	0.049	45
14	1	1	4.46	0.047	0.062	0.036	0.023	0.049	3.326	-0.13	0.049	45
15	1	1	4.47	0.01	0.02	0.06	0.045	0.038	3.29	-0.13	0.049	45
16	1	1	4.48	0.051	0.068	0.027	0.023	0.082	3.29	-0.15	0.037	45
17	1	1	4.49	0.037	0.052	0.046	0.017	0.049	3.317	-0.12	0.026	45
18	1	1	4.5	0.01	0.02	0.065	0.028	0.027	3.299	-0.11	0.06	45
19	1	1	4.51	0.028	0.041	0.027	0.056	0.06	3.272	-0.15	0.072	46
20	1	1	4.52	0.037	0.052	0.036	0.05	0.06	3.272	-0.14	0.049	46
21	1	1	4.54	0.042	0.057	0.046	0.045	0.071	3.299	-0.13	0.049	46
22	1	1	4.55	0.06	0.078	0.036	0.023	0.082	3.299	-0.14	0.037	46

已成功執行查詢。 (local) (1

圖3.3 右腳壓力感測數值

力與速 Power and Speed

結果 訊息

	Runner_id	Test_id	StartTime	RPi1S	RPi2S	RPi3S	RPi1T	RPi2T	RPi3T
1	3	1	2017/01/16 11:30:28	197.564575457	293.994598718	314.918938366	2017-04-13 12:33:08.000	2017-04-10 03:33:07.000	2017-05-10 11:05:27.00
2	3	2	2017/01/16 11:30:28	214.746428083	314.005456618	176.851938451	2017-04-13 12:34:16.000	2017-04-10 03:34:15.000	2017-05-10 11:06:35.00
3	3	4	2017/01/16 11:30:28	182.398159509	82.7259965152	270.874065722	2017-04-13 12:37:09.000	2017-04-10 03:38:23.000	2017-05-10 11:09:29.00
4	3	6	2017/01/16 11:30:28	189.994369734	321.936259886	142.300326572	2017-04-13 12:44:10.000	2017-04-10 03:44:09.000	2017-05-10 11:16:29.00
5	5	3	2017/01/16 11:30:28	586.74481412	560.900736544	206.455614941	2017-04-13 12:41:16.000	2017-04-10 03:41:27.000	2017-05-10 11:41:30.00
6	13	4	2017/01/16 11:30:28	2.75077528145	1.52218158578	0.28580235777	2017-04-14 10:57:28.000	2017-04-11 12:52:39.000	2017-05-11 08:41:59.00
7	13	7	2017/01/16 11:30:28	0.200244855722	2.11331065363	0.5763765972 81	2017-04-14 12:40:53.000	2017-04-11 02:36:09.000	2017-05-11 10:25:21.00
8	13	8	2017/01/16 11:30:28	1.75364985378	3.21511464102	0.198386565653	2017-04-14 12:50:25.000	2017-04-11 02:45:36.000	2017-05-11 10:34:57.00
9	15	2	2017/01/16 11:30:28	7.10648062999	12.6796220268	1.54499209102	2017-04-20 05:26:35.000	2017-04-17 07:26:27.000	2017-05-11 10:26:56.00
10	15	3	2017/01/16 11:30:28	5.86906366853	6.97309216449	5.92153744173	2017-04-20 05:31:07.000	2017-04-17 07:30:59.000	2017-05-11 10:31:24.00
11	15	4	2017/01/16 11:30:28	4.01263181738	10.4359052373	5.82244867793	2017-04-20 05:33:37.000	2017-04-17 07:33:30.000	2017-05-11 10:33:59.00
12	21	1	2017/01/16 11:30:28	2.60748214813	3.51492260579	2.27626814746	2017-04-20 05:53:39.000	2017-04-17 07:53:36.000	2017-05-11 10:54:12.00
13	22	1	2017/01/16 11:30:28	5.48970332206	5.90371796509	3.15591216572	2017-04-20 05:55:56.000	2017-04-17 07:55:53.000	2017-05-11 10:56:30.00
14	22	2	2017/01/16 11:30:28	2.52853204556	3.84429153769	2.11830849018	2017-04-20 06:02:12.000	2017-04-17 08:02:10.000	2017-05-11 11:02:32.00
15	22	3	2017/01/16 11:30:28	2.626803445	6.38178450253	1.19211074056	2017-04-20 06:03:22.000	2017-04-17 08:03:18.000	2017-05-11 11:04:01.00

圖3.4 樹莓派所感測之速度數值

3.3 製作 Highcharts 圖表

在抓取完資料後，本研究結合上述來源資料以網頁結合 Highcharts 圖表方式做呈現，讓使用者可以利用下拉式選單選擇跑者代號與跑者測試代號，查詢該次測試者的足部壓力數值所製作的可視化圖表，讓田徑教練可以更簡易從圖表中了解跑者在起跑時的準備時間與足部壓力分佈之關係。

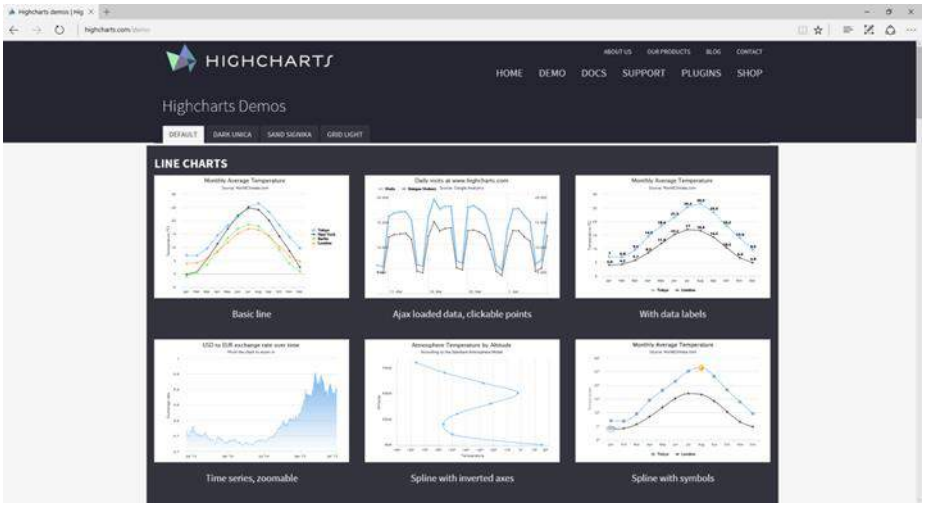


圖3.5 Highcharts選擇圖表樣式頁面

第四章 系統實作

4.1 系統環境

本研究的程式開發環境是使用：

1. Server 端

Lenovo 2320AXV 筆記型電腦做為系統平台伺服器，硬體部分使用的中央處理器(CPU)為 Intel(R) Core(TM) i5-3360M CPU @ 2.80GHz (4 CPUs),~2.8GHz，記憶體為 4G，作業系統採用 Windows 7 旗艦版 64 位元(6.1,組建 7601),使用程式為 Microsoft Visual Studio 2012，使用資料庫為 Microsoft SQL Server 2012。

圖4.1 系統開發環境

力與速 Power and Speed

2. 樹莓派端

樹莓派端所使用為樹莓派 — Raspberry Pi 3 Model B，Raspberry Pi 3 Model B 使用的系統基於 Linux 下的 Raspbian 作業系統，只有一張信用卡大小的單板電腦（Single Board Computer，SBC）。為 64 位元處理器，四核心 1.2GHz 的 Cortex-A53 晶片，內建 USB2.0、Wi-Fi 及藍牙功能，作業系統由 MicroSD 記憶卡啟動。

圖4.2 Raspberry Pi 3 Model B

4.2 系統實作

本研究將系統實作分成四個部分，依序分為 RPI 操作、Microsoft Visual Studio 操作、SQL Server 操作及手持式智慧裝置操作。

4.2.1 RPI 操作

本系統在樹莓派部份採用以 Linux 為基礎的 Raspbian 系統，使透過 Opencv 技術讓相機模組進行錄影時，畫出一區塊進行辨認，當跑者

力與速 Power and Speed

行經過那一區塊，便利用其公式與技術，將跑者行經時當下的瞬時速度與照片擷取下來，並利用 Python 結合 Socket 與 FTP 將所擷取到的資料與照片傳回伺服器。

圖 4.3 Rpi

圖 4.3 為樹莓派在操場架設的實際畫面，我們藉由行動電源供電給樹莓派，透過相機腳架將樹莓派架高，以利樹莓派拍攝。

```
import socket, sys
import thread_function as tf
import time
import thread

def threadWork(client):

    msg = client.recv(1024)
    tf.thread_test(msg)
    client.send("exit")
    client.close()



try:
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
except socket.error, msg:
    sys.stderr.write("[ERROR] %s\n" % msg[1])
    sys.exit(1)

sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1) #reuse tcp
sock.bind(('192.168.1.2', 8110))
sock.listen(5)

while True:
        (csock, adr) = sock.accept()
        threadWork(csock)
```

圖 4.4 Rpi_socket

圖 4.4 為樹莓派利用 Socket 協定傳輸資料的程式碼，透過此程式碼跟 Server 端進行溝通。

```
def thread_test(b):
    a = {'1':rpi_status , '2':ram_status ,'3':ct.camera_carspeed , '4':kill_camera}
    data = b.split(",")

    if len(data)  == 1:
        a[b]('test')
    else :
        socket_Client("camera open!")
        a[data[0]](data[1],data[2])
```

圖 4.5 Rpi_thread

圖 4.5 為當樹莓派收到 Server 所傳來的指令，便會透過此程式碼進行動作，可以看到當樹莓派收到不同指令分別會有不同動作：

1. 收到 1 時：回傳樹莓派狀態

2. 收到 2 時：回傳樹莓派 RAM 的狀態

3. 收到 3 時：開啟相機模式

4. 收到 4 時：關閉所有程式

4.2.2 Microsoft Visual Studio 操作

本系統透過Microsoft Visual Studio製作圖形化介面讓使用者可以輸入選手ID藉此區別資料，透過ssh技術對樹莓派進行遠端操控，並利用C#結合Socket技術進行資料間的傳輸。而在壓力感應器上我們藉由資料擷取（Data Acquisition，DAQ）技術，將感測器所回傳之訊號經由訊號放大器將訊號放大之後，透過類比轉換器將類比訊號轉化為數值資料，將所接收到之資料存入SQL Server。

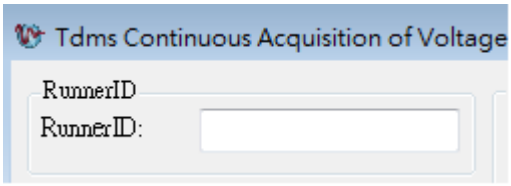


圖 4.6 C#_runner

圖4.6為教練輸入選手ID，藉此區別不同選手，在此系統會自動區別此選手進行第幾次測試，會給予其排序的Test ID，透過此排序可讓手持式智慧裝置觀看查詢資料時更為明確。

圖4.7.1至圖4.7.5為Save Result的內容，以下分別為：

1. Cost Time：測試所花費時間。

2. Save Numbers：儲存編號。

3. Start Time：跑者起跑當下時間。

4. End Time：跑者跑完最後點的當下時間。

5. React Time：End Time減去Start Time的時間，也就是跑者的測試時間

圖 4.7.1 Cost Time button

力與速 Power and Speed

圖 4.7.2 Save Numbers button

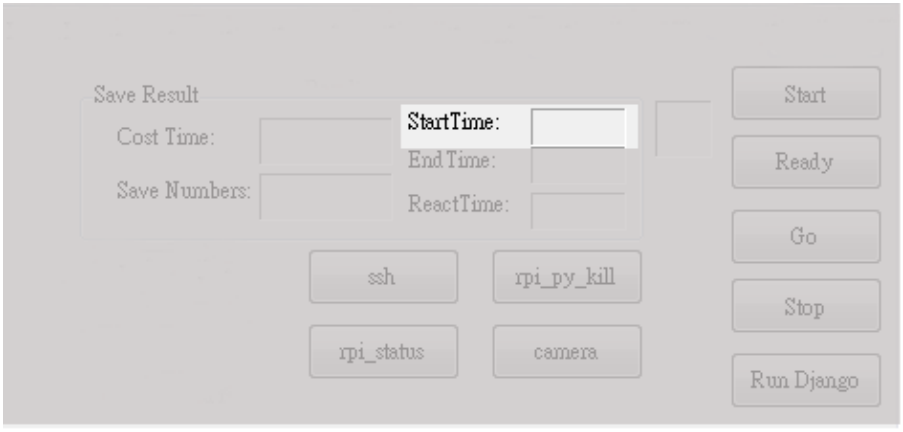


圖 4.7.3 Start Time button

圖 4.7.4 End Time button

力與速 Power and Speed

圖 4.7.5 React Time button

在 Save Result 其下方分別有六個按鍵藉由其四個按鍵用以控制樹莓派：

1. ssh：透過 ssh 技術遠端操控樹莓派並下指令開啟其程式。
2. Rpi_status：透過傳送指令要求樹莓派回傳現在的狀態及資訊。
3. Rpi_py_kill：假如因當天其訊號不良，或其他因素造成其問題無法進行測試，可藉由此鍵將樹莓派上的程式關閉，即可再次重啟。
4. Camera：用以開啟樹莓派上的相機，並會回傳其資訊確認相機皆已就位，跑者便可進行測試，在跑者測試完畢後，會確認所有資料皆有回傳完畢，便會關閉相機。

力與速 Power and Speed

圖 4.8.1 ssh button

```
private void ssh_open()
{
    Thread t_ssh_open1 = new Thread(() => ssh_command("192.168.1.1"
                                    , "nice -n 0 python /home/pi/Desktop/SendAndCesv/formal/socket_send_recv.py"));
    t_ssh_open1.Start();
    Thread t_ssh_open2 = new Thread(() => ssh_command("192.168.1.2"
                                    , "nice -n 0 python /home/pi/Desktop/SendAndCesv/formal/socket_send_recv.py"));
    t_ssh_open2.Start();
    Thread t_ssh_open3 = new Thread(() => ssh_command("192.168.1.3"
                                    , "nice -n 0 python /home/pi/Desktop/SendAndCesv/formal/socket_send_recv.py"));
    t_ssh_open3.Start();
}
```

圖 4.8.2 ssh button 程式碼

圖 4.8.3 Rpi_status button

```
2279        private void ssh_rpi_status()
2280        {
2281            Thread t_ssh_status1 = new Thread(ss.socket_client);
2282            String rpi_1 = "1" + "+192.168.1.1";
2283            t_ssh_status1.Start(rpi_1);
2284            Thread t_ssh_status2 = new Thread(ss.socket_client);
2285            String rpi_2 = "1" + "+192.168.1.2";
2286            t_ssh_status2.Start(rpi_2);
2287            Thread t_ssh_status3 = new Thread(ss.socket_client);
2288            String rpi_3 = "1" + "+192.168.1.3";
2289            t_ssh_status3.Start(rpi_3);
2290        }
2291
```

圖 4.8.4 Rpi_status button 程式碼

圖 4.8.5 Rpi_py_kill button

```
private void ssh_kill_python()
{
    Thread t_ssh_kill1 = new Thread(() => ssh_command("192.168.1.1", "pkill -f python"));
    t_ssh_kill1.Start();
    Thread t_ssh_kill2 = new Thread(() => ssh_command("192.168.1.2", "pkill -f python"));
    t_ssh_kill2.Start();
    Thread t_ssh_kill3 = new Thread(() => ssh_command("192.168.1.3", "pkill -f python"));
    t_ssh_kill3.Start();
    //t_ssh_kill1.Abort();  //close Thread()
}
```

圖 4.8.6 Rpi_py_kill button 程式碼

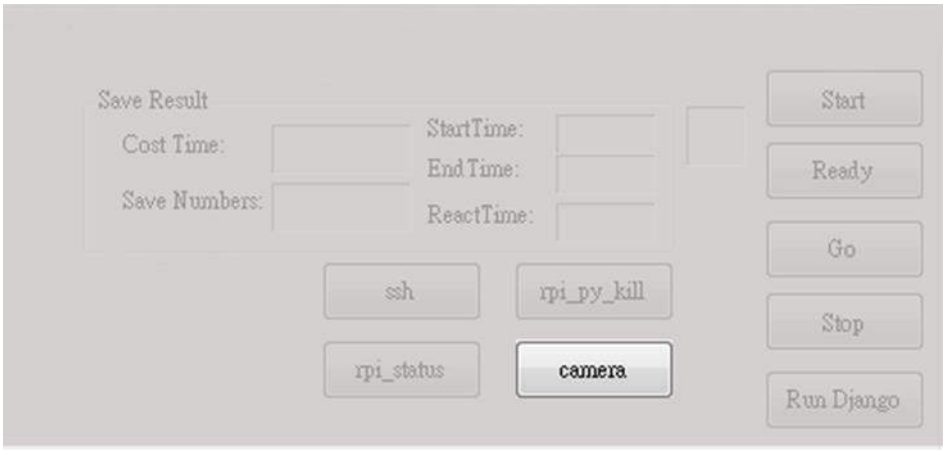


圖 4.8.7 Camera button

```
134
135         public static void camera_finish(string data)
136         {
137             for (int i = 0; i <= 2; i++)
138             {
139                 if (data == camera_open[i])
140                 {
141                     if (id_list[i] != "null")
142                     {
143                         camera_data[i] = "null";
144                         camera_date[i] = "null";
145                     }
146                     else
147                     {
148                         id_list[i] = i.ToString();
149                     }
150                 }
151                 if (id_list.SequenceEqual(id_pass))
152                 {
153                     result = "Camera is finish!";
154                     sb_result.Append(result + "\r\n");
155                     Console.WriteLine("Camera is finish!");
156                     id_clean();
157                     camera_check = "True";
158                     break;
159                 }
160             }
```

圖 4.8.8 Camera_finish 程式碼

```
163         public static void camera_mph_date(string data)
164         {
165             string[] words = data.Split('\'');
166             string mph_date = words[5] + " m/s:" + words[3] + " date:" + words[1];
167             result = mph_date;
168             sb_result.Append(result + "\r\n");
169             Console.WriteLine(mph_date);
170             for (int i = 0; i <= 2; i++)
171             {
172                 if (words[5] == id_check[i])
173                 {
174                     camera_data[i] = words[3];
175                     camera_date[i] = words[1];
176                     id_list[i] = i.ToString();
177                 }
178             }
179             if (camera_data[0] != "null" && camera_data[1] != "null" && camera_data[2] != "null")
180             {
181                 commitSql();
182                 result = "Camera data is ok!";
183                 sb_result.Append(result + "\r\n");
184                 Console.WriteLine("Camera data is ok!");
185                 id_clean();
186                 camera_check = "False";
187                 for (int i = 0; i <= 2; i++)
188                 {
189                     camera_data[i] = "null";
190                     camera_date[i] = "null";
191                 }
192 
193             }
194         }
```

圖 4.8.9 Camera_mph 程式碼

而在視窗左上方有五個按鍵，如圖 4.9.1 至圖 4.9.6：

1. Start：當按下時壓力感測器便會開時收取數值，此時便可檢視壓力感測器是否有正確的收入資料。
2. Ready：確定好便有正確收入資料時，按下讓測試員準備就緒。
3. Go：確認測試員準備就緒後即可按下，此時系統便會以隨機秒數倒數 3、2、1 並發出槍聲。
4. Stop：當選手跑過最後測試點時，即可按下結束此次測試。
5. Run Django：用以架設 Django 網站。

力與速 Power and Speed

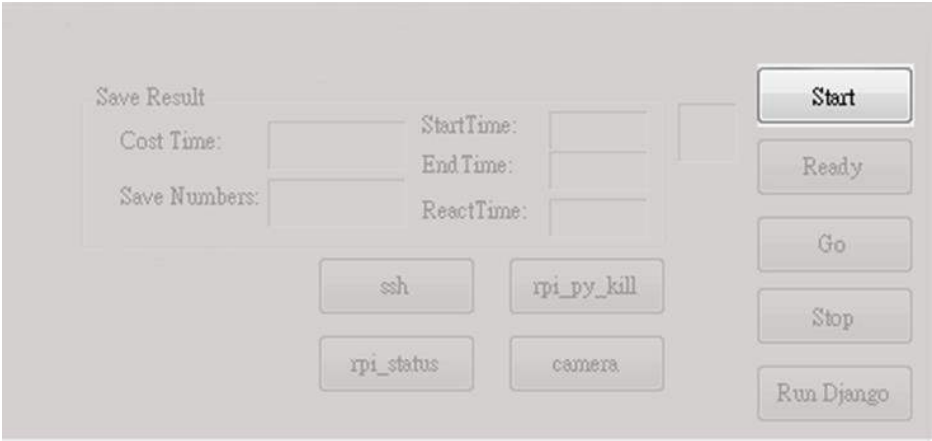


圖 4.9.1 Start button

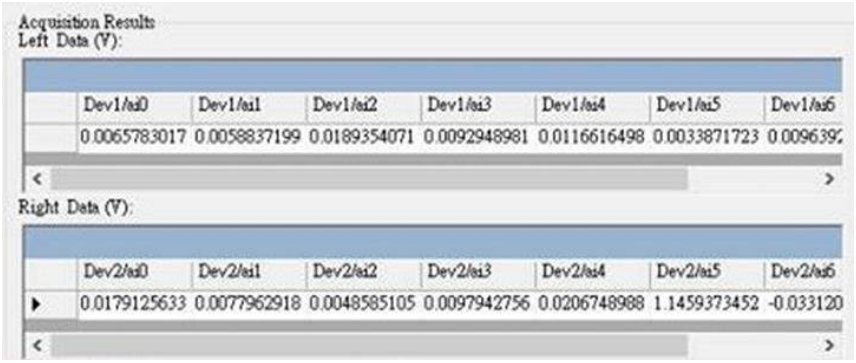


圖 4.9.2 C#視窗所呈現雙腳足部壓力資料

圖 4.9.3 Ready button

力與速 Power and Speed

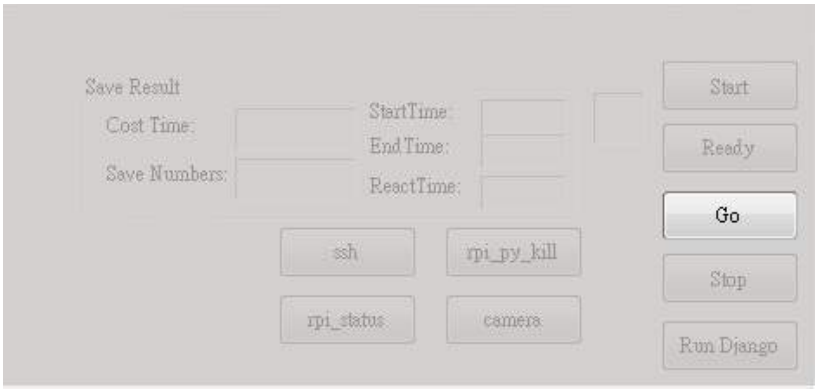


圖 4.9.4 Go button

圖 4.9.5 Stop button

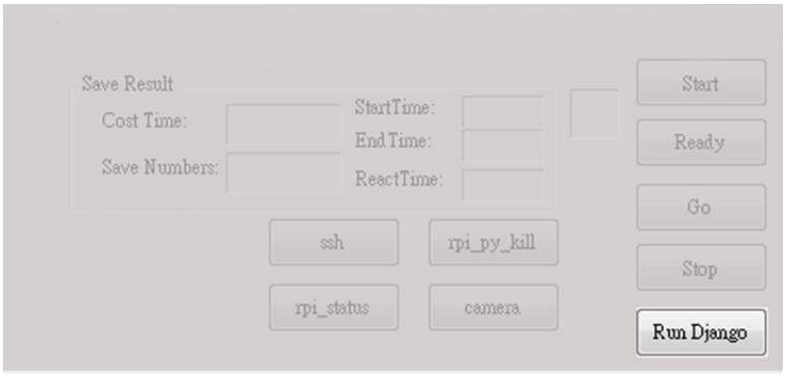


圖 4.9.6 Run Django button

在整個視窗最右邊，有一個視窗，用以即時觀看樹莓派端回傳其所有資料，便能更佳的掌握樹莓派的狀態。如圖 4.10。

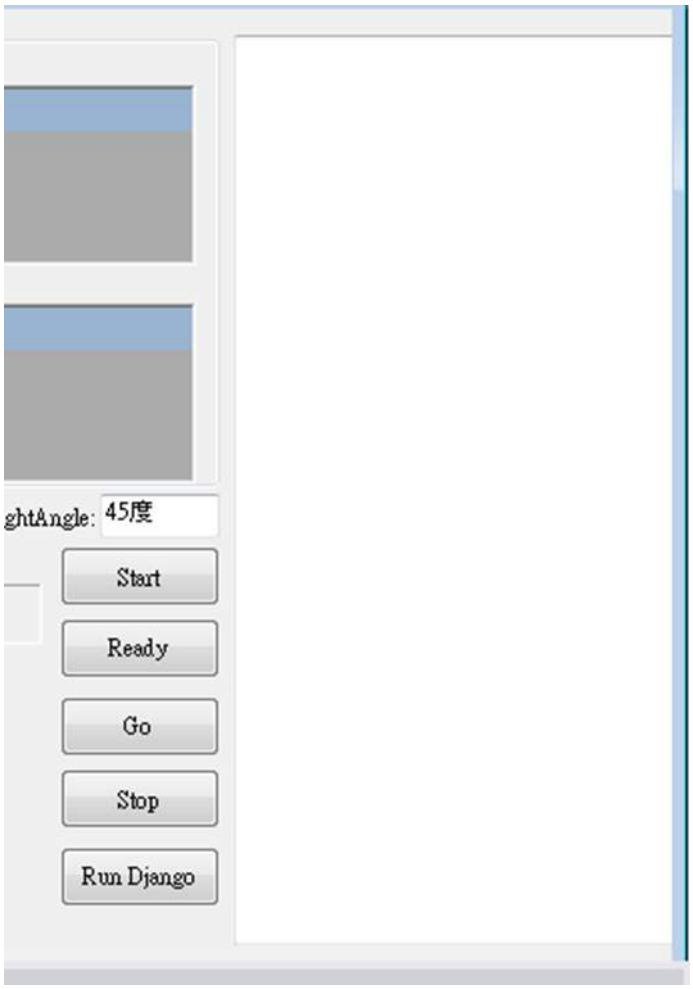


圖 4.10 C#_data_recv 視窗

4.2.3 SQL Server 操作

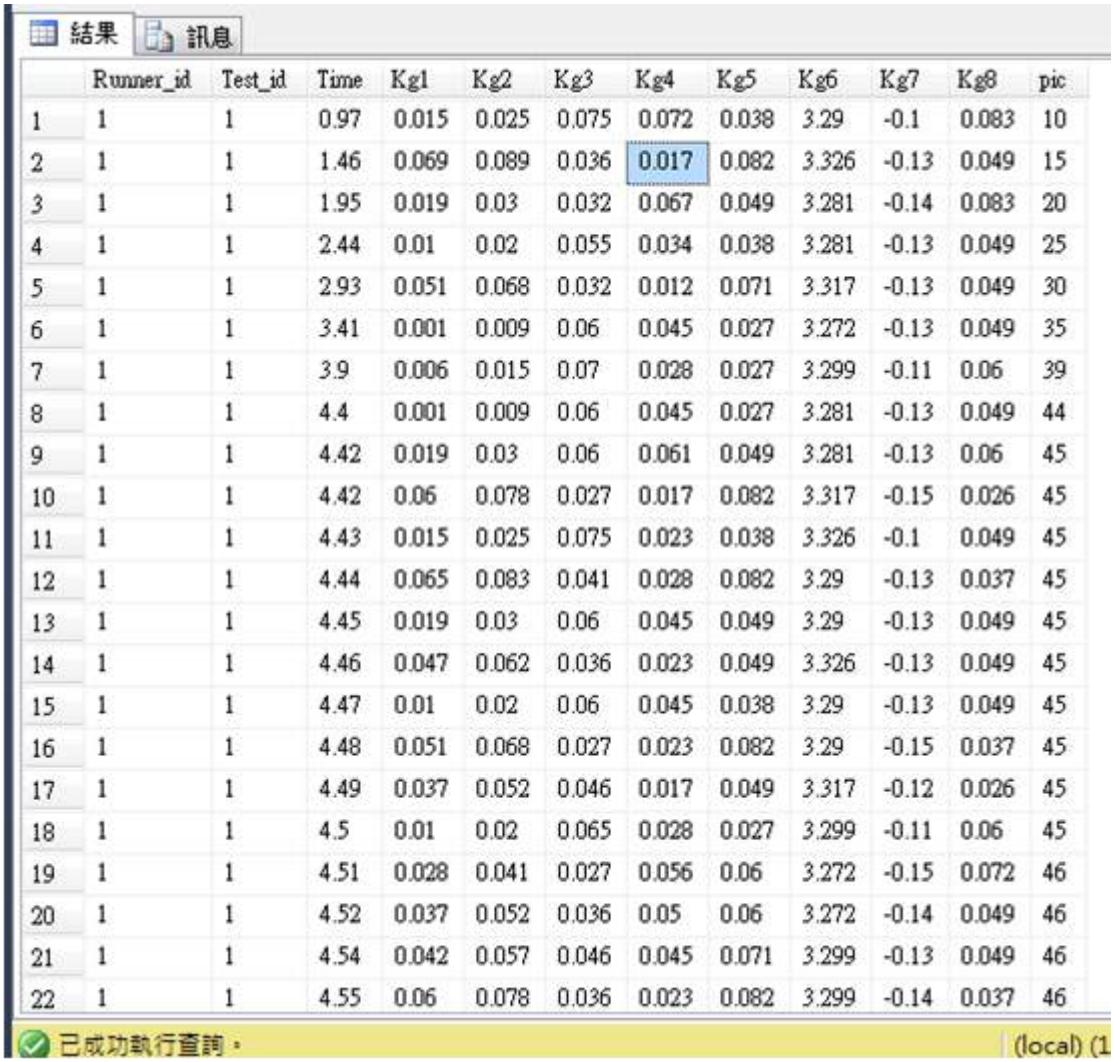
本研究將樹莓派及透過 DAQ 所擷取下的壓力感測器資料，以 Socket 的方式回傳至 Server，並同時以預存程序存進資料庫中以便後續的作業。

結果 訊息

	Runner_id	Test_id	Time	Kg1	Kg2	Kg3	Kg4	Kg5	Kg6	Kg7	Kg8	pic
1	1	1	0.98	0.029	0.039	0.114	0.056	0.027	0.021	0.054	0.112	10
2	1	1	1.47	0.044	0.066	0.122	0.051	0.049	0.021	0.041	0.112	15
3	1	1	1.96	0.063	0.1	0.077	0.051	0.07	0.021	0.014	0.09	20
4	1	1	2.45	0.048	0.072	0.07	0.062	0.07	0.033	0.014	0.101	25
5	1	1	2.94	0.029	0.039	0.144	0.045	0.027	0.01	0.054	0.112	30
6	1	1	3.43	0.067	0.106	0.092	0.022	0.081	0.01	0.054	0.067	35
7	1	1	3.92	0.029	0.039	0.099	0.051	0.027	0.021	0.041	0.112	40
8	1	1	4.41	0.048	0.072	0.048	0.045	0.049	0.055	0.014	0.112	45
9	1	1	4.42	0.052	0.079	0.099	-0.01	0.07	0.021	0.041	0.045	45
10	1	1	4.43	0.029	0.039	0.099	0.068	0.027	0.021	0.041	0.124	45
11	1	1	4.44	0.037	0.052	0.099	0.011	0.027	0.021	0.041	0.078	45
12	1	1	4.45	0.071	0.113	0.077	0.045	0.081	0.021	0.014	0.09	45
13	1	1	4.46	0.029	0.038	0.129	0.039	0.027	0.021	0.067	0.112	45
14	1	1	4.47	0.056	0.086	0.129	0.017	0.049	0.044	0.067	0.09	45
15	1	1	4.48	0.044	0.066	0.085	0.011	0.049	0.055	0.041	0.067	45
16	1	1	4.49	0.067	0.106	0.07	0.011	0.081	0.055	0.027	0.078	45
17	1	1	4.5	0.041	0.059	0.085	0	0.038	0.055	0.041	0.067	45
18	1	1	4.51	0.052	0.079	0.085	-0.01	0.049	0.044	0.041	0.045	46
19	1	1	4.52	0.033	0.045	0.077	0	0.027	0.01	0.041	0.056	46
20	1	1	4.53	0.056	0.086	0.122	0.022	0.049	0.055	0.081	0.09	46
21	1	1	4.54	0.018	0.018	0.114	0.034	0.006	-0.01	0.041	0.09	46
22	1	1	4.55	0.052	0.079	0.136	0.011	0.049	0.055	0.067	0.078	46

圖 4.11 左腳壓力數值

力與速 Power and Speed

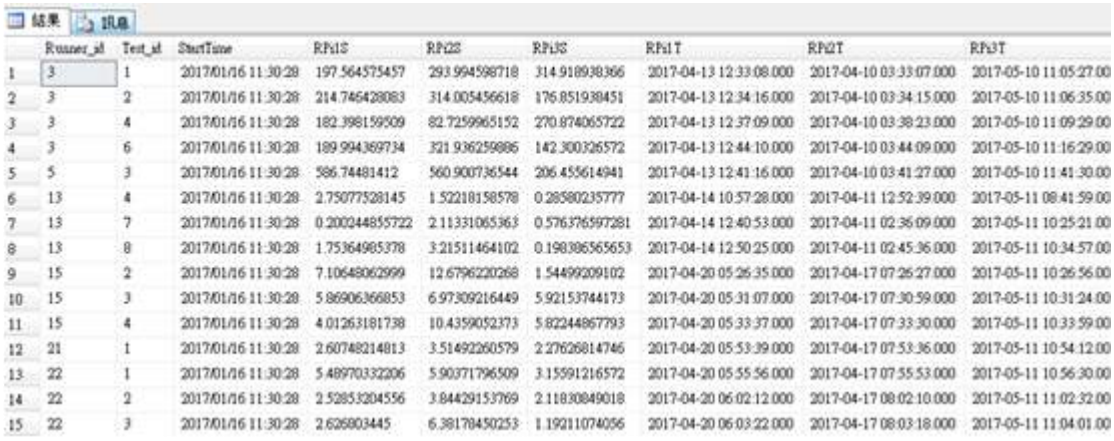


結果 訊息

	Runner_id	Test_id	Time	Kg1	Kg2	Kg3	Kg4	Kg5	Kg6	Kg7	Kg8	pic
1	1	1	0.97	0.015	0.025	0.075	0.072	0.038	3.29	-0.1	0.083	10
2	1	1	1.46	0.069	0.089	0.036	0.017	0.082	3.326	-0.13	0.049	15
3	1	1	1.95	0.019	0.03	0.032	0.067	0.049	3.281	-0.14	0.083	20
4	1	1	2.44	0.01	0.02	0.055	0.034	0.038	3.281	-0.13	0.049	25
5	1	1	2.93	0.051	0.068	0.032	0.012	0.071	3.317	-0.13	0.049	30
6	1	1	3.41	0.001	0.009	0.06	0.045	0.027	3.272	-0.13	0.049	35
7	1	1	3.9	0.006	0.015	0.07	0.028	0.027	3.299	-0.11	0.06	39
8	1	1	4.4	0.001	0.009	0.06	0.045	0.027	3.281	-0.13	0.049	44
9	1	1	4.42	0.019	0.03	0.06	0.061	0.049	3.281	-0.13	0.06	45
10	1	1	4.42	0.06	0.078	0.027	0.017	0.082	3.317	-0.15	0.026	45
11	1	1	4.43	0.015	0.025	0.075	0.023	0.038	3.326	-0.1	0.049	45
12	1	1	4.44	0.065	0.083	0.041	0.028	0.082	3.29	-0.13	0.037	45
13	1	1	4.45	0.019	0.03	0.06	0.045	0.049	3.29	-0.13	0.049	45
14	1	1	4.46	0.047	0.062	0.036	0.023	0.049	3.326	-0.13	0.049	45
15	1	1	4.47	0.01	0.02	0.06	0.045	0.038	3.29	-0.13	0.049	45
16	1	1	4.48	0.051	0.068	0.027	0.023	0.082	3.29	-0.15	0.037	45
17	1	1	4.49	0.037	0.052	0.046	0.017	0.049	3.317	-0.12	0.026	45
18	1	1	4.5	0.01	0.02	0.065	0.028	0.027	3.299	-0.11	0.06	45
19	1	1	4.51	0.028	0.041	0.027	0.056	0.06	3.272	-0.15	0.072	46
20	1	1	4.52	0.037	0.052	0.036	0.05	0.06	3.272	-0.14	0.049	46
21	1	1	4.54	0.042	0.057	0.046	0.045	0.071	3.299	-0.13	0.049	46
22	1	1	4.55	0.06	0.078	0.036	0.023	0.082	3.299	-0.14	0.037	46

已成功執行查詢。 (local) (1

圖 4.12 右腳壓力數值



結果 訊息

	Runner_id	Test_id	StartTime	RPs1S	RPs2S	RPs3S	RPs1T	RPs2T	RPs3T
1	3	1	2017/01/16 11:30:28	197.564575457	293.994598718	314.918938366	2017-04-13 12:33:08.000	2017-04-10 03:33:07.000	2017-05-10 11:05:27.00
2	3	2	2017/01/16 11:30:28	214.746428083	314.005456618	176.851938451	2017-04-13 12:34:16.000	2017-04-10 03:34:15.000	2017-05-10 11:06:35.00
3	3	4	2017/01/16 11:30:28	182.398159509	82.7259965152	270.874065722	2017-04-13 12:37:09.000	2017-04-10 03:38:23.000	2017-05-10 11:09:29.00
4	3	6	2017/01/16 11:30:28	189.994369734	321.936259886	142.300326572	2017-04-13 12:44:10.000	2017-04-10 03:44:09.000	2017-05-10 11:16:29.00
5	5	3	2017/01/16 11:30:28	586.74481412	560.900736544	206.455614941	2017-04-13 12:41:16.000	2017-04-10 03:41:27.000	2017-05-10 11:41:30.00
6	13	4	2017/01/16 11:30:28	2.75077528145	1.52218158578	0.28580235777	2017-04-14 10:57:28.000	2017-04-11 12:52:39.000	2017-05-11 08:41:59.00
7	13	7	2017/01/16 11:30:28	0.200244855722	2.11331065363	0.576376597281	2017-04-14 12:40:53.000	2017-04-11 02:36:09.000	2017-05-11 10:25:21.00
8	13	8	2017/01/16 11:30:28	1.75364985378	3.21511464102	0.190386565653	2017-04-14 12:50:25.000	2017-04-11 02:45:36.000	2017-05-11 10:34:57.00
9	15	2	2017/01/16 11:30:28	7.10648062999	12.6796220268	1.54499209102	2017-04-20 05:26:35.000	2017-04-17 07:26:27.000	2017-05-11 10:26:56.00
10	15	3	2017/01/16 11:30:28	5.86906366853	6.97309216449	5.92153744173	2017-04-20 05:31:07.000	2017-04-17 07:30:59.000	2017-05-11 10:31:24.00
11	15	4	2017/01/16 11:30:28	4.01263181738	10.4359052373	5.82244867793	2017-04-20 05:33:37.000	2017-04-17 07:33:30.000	2017-05-11 10:33:59.00
12	21	1	2017/01/16 11:30:28	2.60748214813	3.51492260579	2.27626814746	2017-04-20 05:53:39.000	2017-04-17 07:53:36.000	2017-05-11 10:54:12.00
13	22	1	2017/01/16 11:30:28	5.48970332206	5.90371796509	3.15591216572	2017-04-20 05:55:56.000	2017-04-17 07:55:53.000	2017-05-11 10:56:30.00
14	22	2	2017/01/16 11:30:28	2.52853204556	3.84429153769	2.11830849018	2017-04-20 06:02:12.000	2017-04-17 08:02:10.000	2017-05-11 11:02:32.00
15	22	3	2017/01/16 11:30:28	2.626803445	6.38178450253	1.19211074056	2017-04-20 06:03:22.000	2017-04-17 08:03:18.000	2017-05-11 11:04:01.00

圖 4.13 測試速度數值

4.2.4 手持式智慧裝置操作

在手持式智慧裝置上，本系統透過 Django 來架設網頁，並使用 Bootstrap 讓網頁成為互動式網頁，在圖表表現部份利用了 Hightcharts 讓圖表能動態顯示，使教練在觀看選手數據時更能一目了然。

圖 4.14 查詢視窗

此為查詢視窗畫面，在畫面上方有兩個下拉式選單，方便讓使用者選取，如圖 4.15.1 至圖 4.15.2；下方有三個按鈕，方便讓使用者選擇所需要的資料，如圖 4.16.1 至圖 4.16.7：

力與速 Power and Speed

1. 選擇跑者 ID：此為各個選手不同編號，藉此可以區分選手。
2. 選擇該跑者測試 ID：此為選擇跑者的幾次測試的選單，讓使用者可以明確了解，此次所查詢的為第幾次測試。
3. 畫圖囉:此按鍵可以利用 Highcharts 畫出足部壓力值之圖表。
4. 查圖囉：可利用此按鍵觀看跑者在測試中，經過測試點之圖片以及通過時瞬時速度。
5. 起跑圖：可利用此鍵觀看起跑時腳出發的動作。

圖 4.15.1 選擇跑者 ID 選單

力與速 Power and Speed

圖 4.15.2 選擇該跑者測試 ID 選單

圖 4.16.1 畫圖囉按鈕

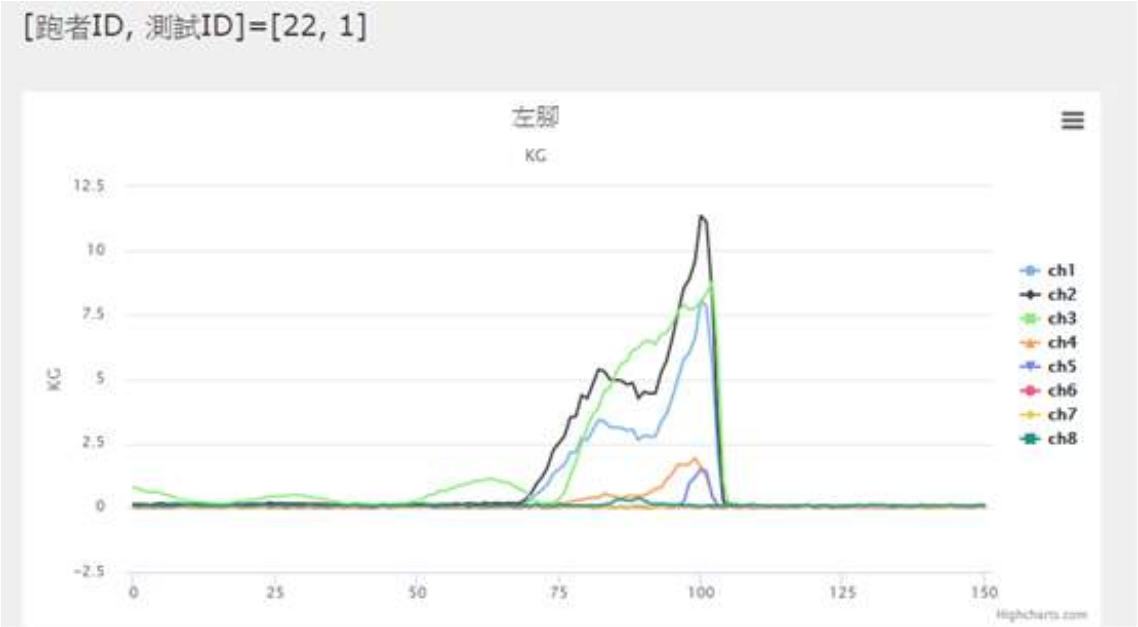


圖 4.16.2 畫圖囉按鈕顯示左腳壓力數值

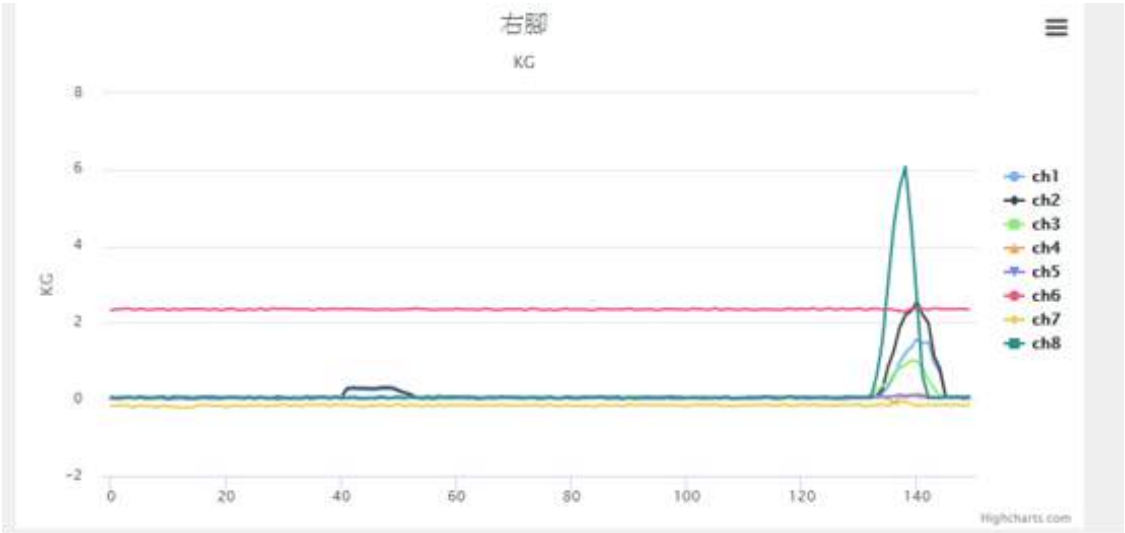


圖 4.16.3 畫圖囉按鈕顯示右腳壓力數值

圖 4.16.4 查圖囉按鈕

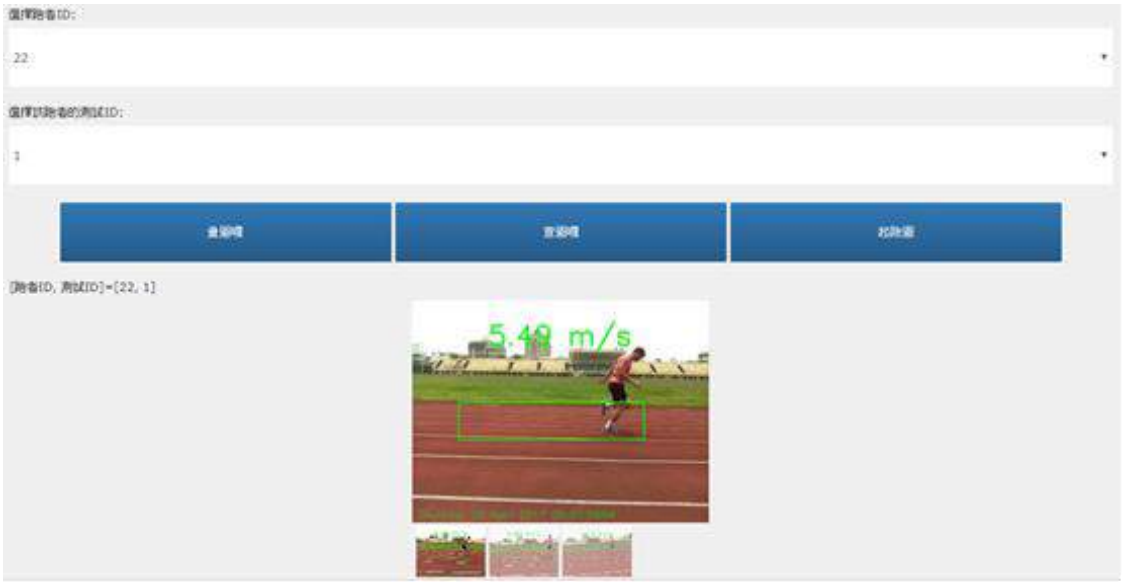


圖 4.16.5 查圖囉按鈕顯示畫面

圖 4.16.6 起跑圖按鈕

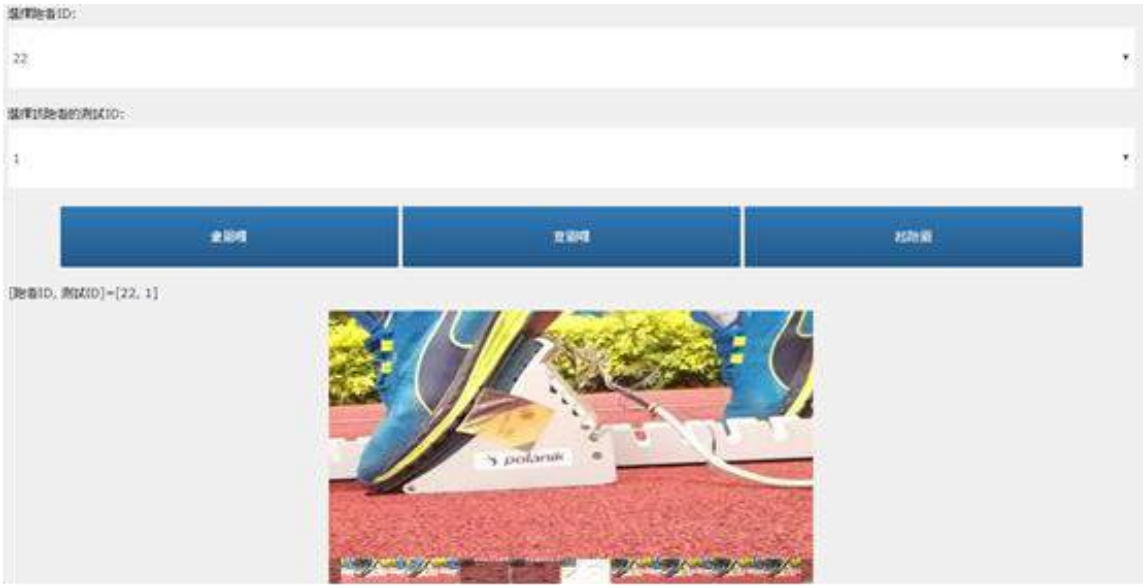


圖 4.16.7 起跑圖按鈕顯示畫面

第五章 結論與未來研究方向

5.1 結論

本研究考慮到基層學校經費有限，無法購入市面上較為昂貴的田徑運動分析平台，因此一開始以減輕市售昂貴的田徑運動分析平台帶來之經費壓力及使用者能夠輕鬆使用系統進行測試為目標，並以「使田徑訓練方式具有科學依據性」為設計動機及搭配物聯網技術建置本系統。

透過田徑短距離起跑模式與即時回饋平台，教練可以透過智慧型手持式設備即時地取得選手當下測試之資料，且選手測試時不需使用繁雜的設備及程序即可測得有根據性之運動數據，使傳統的田徑訓練方式具有科學依據性且更人性化。

在目前,系統已降低以往田徑運動數據蒐集平台昂貴的建置成本，統整過去各項田徑數據偵測設施，改良過往零散的偵測設施所造成的麻煩並整合動力學及運動學所需數據，提供整體數據分析之功能。此外也降低系統架設時間，讓教練及選手可以更充分的運用訓練時間並且接收起跑架足部壓力分佈數值、各樹莓派擷取動態的畫面所測之速度與選手測試時之擷取畫面，將數值與擷取畫面存入至資料庫，讓教練及選手能夠隨時查看每一次測試的資料，並且讓教練能夠透過所測

得數據為依據，去調整每一位選手的狀況。

5.2 未來研究方向

在系統穩定發展後，未來展望為透過大數據的資料收集，利用蒐集之資料進行分析，期望能將此系統發展為田徑運動數據分析平台。為每一位選手定義出合適的起跑模式，改善田徑選手的競賽表現，利用完善的系統功能與架構使選手可以運用更科學的方式進行訓練，將田徑科學能夠普及化於台灣各階級學校。

參考文獻

[1]許樹淵(1996)・田徑論・台北市:偉彬體育研究社。

[2]Harland, H. J.and Steele, J. R.(1997).Biomechanics of the sprint start. Sport Medicine,23(1),11-20.

[3]維基百科一物聯網，

https://zh.wikipedia.org/wiki/%E7%89%A9%E8%81%94%E7%BD%91

[4]製造全球最便宜電腦的它，以 8.7 億美元的身價被買下，

http://tech.fanpiece.com/ifanr/%E8%A3%BD%E9%80%A0%E5%85%A8%E7%90%83%E6%9C%80%E4%BE%BF%E5%AE%9C%E9%9B%BB%E8%85%A6%E7%9A%84%E5%AE%83-%E4%BB%A5-8-7-%E5%84%84%E7%BE%8E%E5%85%83%E7%9A%84%E8%BA%AB%E5%83%B9%E8%A2%AB%E8%B2%B7%E4%B8%8B-c1232079.html

[5]關於 Debian，

https://www.debian.org/intro/about

[6]Python 程式語言簡介，

https://yungyuc.github.io/oldtech/python/python_intro.html

[7] 吳孟春、丁嵐，「High Charts 組件在氣象業務中的開發和應用」，計算機與網路，第十二期，第 65—68 頁，民國一零三年。

[8] HighCharts，http://www.highcharts.com/products/highcharts，2015 年 11 月。

[9] 張建軍、劉虎、倪芳英，「基於 SSH 與 Highcharts 整合架構的 Web 應用研究」，計算機技術與發展，第二十三卷，第九期，第 245—247 頁，民國一零二年九月。

[10] OpenCV 介紹，
http://monkeycoding.com/?p=514

[11] Socket，
http://pws.niu.edu.tw/~ttlee/os.101.1/day/socket/

[12] 維基百科－網路插座，
https://zh.wikipedia.org/wiki/%E7%B6%B2%E8%B7%AF%E6%8F%92%E5%BA%A7

[13] FTP 是什麼，
http://www.raidenftpd.com/tw/ftp.html

[14] Baidu 百科－C#，
https://baike.baidu.com/item/C%23

[15] Baidu 百科－Django，
https://baike.baidu.com/item/django/61531

作者簡介

姓　　名：呂倩綾

學　　號：40341102

畢業學校：國立斗六高級家事商業職業學校

姓　　名：張瑞珊

學　　號：40341107

畢業學校：國立大甲高級中學

姓　　名：許巧欣

學　　號：40341108

畢業學校：國立斗六高級家事商業職業學校

姓　　名：程佩璇

學　　號：40341112

畢業學校：私立東海高級中學

姓　　名：廖耕瑆

學　　號：40341141

畢業學校：私立大成高級商工職業學校