國立台北科技大學電子工程系110年度

實務專題成果報告書

高爾夫球軌跡估測系統

專題執行期間：2021年10月1日起至 2022年5月13日止

電子三乙　108360204　孫兆佑

電子三乙　108360208　張宸瑋

電子三乙　108360213　劉　珩

電子三乙　108360242　鄒鎧亦

高立人　副教授

中 華 民 國 111 年 5 月 13 日

摘要：

自2019年新冠肺炎首次在中國武漢首次被發現，並於隔年2020年開始在全球大流行以來，新冠肺炎的出現大大地改變現代人的生活方式，對於許多喜好運動的人來說，疫情影響下無法到戶外活動的日子無疑是令人抑鬱的。雖然隨著 VR 技術的進步許多運動項目藉由 VR 技術以新的姿態再次出現在大眾的眼前，但在遊玩 VR 時因為無法觀察周遭環境而導致的財產損失案例，以及相較於真實世界的觸感，VR 世界裡的觸感仍無法滿足世人。

本專題使用高速攝影機搭配影像辨識技術，使高速攝影機能夠取得使用者在室內擊出的高爾夫球數據，搭配球體拋物線物理公式與線性代數基礎知識計算出接近真實情況的拋物線數據，並存放於後端伺服器中，透過網頁的呼叫將球體拋物線軌跡呈現給使用者。

關鍵字：影像辨識、高斯背景建模 、霍夫曼轉換、線性回歸、網頁設計、資料庫架設

1. 專題簡介：

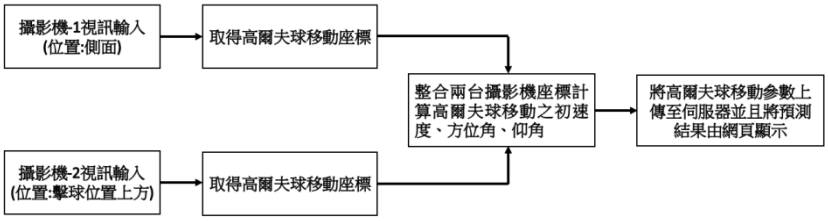
本專題主要目標為透過攝影硬體設備，結合軟體資源建構出一系統化高爾夫球落點判斷模組，在無需抵達球場的情況下，也能模擬真實擊球環境。在硬體方面，為了加速資料處理速度，我們使用 GPU 提升演算法運算效能，除了處理即時且大量的影像資料，同時也加速整個系統的運作。軟體方面，利用影像辨識演算法，減少真實環境中的非可控因子，計算出高爾夫球飛行軌跡的必要條件—初速度、仰角及偏向角，將產生數據與前端網頁透過資料庫結合；前端網頁結合球路軌跡方程式分析落點，顯示高爾夫球飛行動畫，並且延伸多元模式，透過驗證會員資訊可以查詢擊球紀錄，增添趣味性，同時達到人機整合之結果。

經影像辨識與物理模型的協作結果，產生數據精確度將趨近真實環境，用於競賽規劃，可發展為現今世界上球類競技常配備的鷹眼系統之基礎；用於遊戲系統，建立在發展迅速的實境技術上，身處室內卻如臨真實的球場，同時完成精準的球路判斷。

2. 研究方法：

本專題之系統架構主要分成4大部分:雙部相機之同步視訊輸入、移動球

體偵測以及球體座標取得、落點預測演算法、網頁端顯示預測結果，具體的系統架構圖如下圖(一)所示。

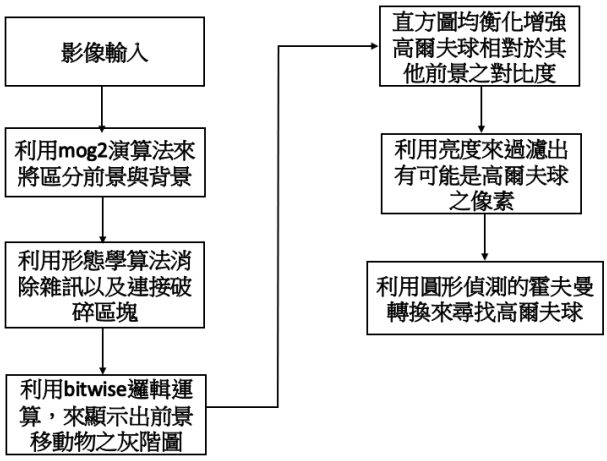


▲圖(一)系統架構圖

2.1 雙部攝影機之同步

在本系統架構的最初端，需同步接收兩台相機之偵測到高爾夫球位置，因此我們採用了多執行緒方法，利用 Python 提供的 Threading 模組將兩部攝影機執行程式分別以個別執行緒來執行，來達成多執行緒的平行計算，如此一來即可達成兩部攝影機之高爾夫球偵測位置的同步播放。

2.2 利用影像辨識之概念來捕捉高爾夫球移動位置



▲圖(二)影像處理流程圖

圖(二)為抓取高爾夫球之移動位置之影像處理流程圖，為了使攝影機能夠自動偵測追蹤到目標物，並對移動目標物進行追蹤，首先必須抓取到畫面中的移動目標，而我們利用的是背景減法的方式，利用圖像序列中目前的畫面與事先建立好的背景模型間的差異比較來確定移動目標物的位置，這是一種基於統計學的原理的目標檢測方法，在我們本次專題中利用到的 mog2演算法，是一種高斯混合模型為基礎的背景前景分割算法，而這個算法的一個特點為它會為每一個像素選擇一個合適數目的高斯分布，而這樣就可以針對亮度變化所引起的場景變化產生更好的適應。

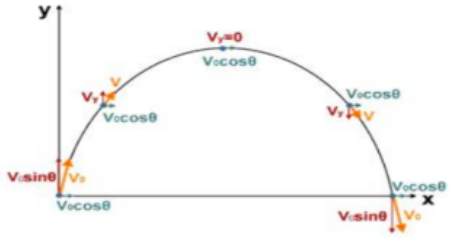
經過影像相減進行移動偵測後，球體內部有可能會有破碎的情形，因此我們利用形態學算法，先是以侵蝕的方式來消除雜訊，接著再以膨脹的方式來連接破碎的區塊，獲得較為完整的畫面。

利用背景相減得到的畫面與原本的輸入影像作 Bitwise 操作，來獲得有可能是高爾夫球之像素的灰階畫面，因為高爾夫球的表面，相對於其他有可能也是高爾夫球之像素而言，其亮度對比是較為突出的，因此我們利用此特徵，來加強高爾夫球對於其他有可能是高爾夫球之像素的對比，這裡使用的方式為限制對比度自適應直方圖均衡化(CLAHE)，此方式相對於普通直方圖均衡化使用從圖像直方圖導出的相同變換來變換所有像素。當整個圖像的像素值分布相似時，普通直方圖均衡化很有效。然而，當圖像包含明顯比大部分圖像更亮或更暗的區域時，這些區域的對比度將不會得到充分增強。而通過使用從鄰域區域計算出的變換函數來變換每個像素，來解決這個問題。但在對比度增強的同時，也放大了影像的噪音，因此利用此方法來改善噪音放大的問題。

利用上述的方法，大大的減少有可能是高爾夫球之像素，並且我們透過圓形偵測的霍夫曼轉換，來進行高爾夫球的偵測，圓方程式為$(x-a)^2+(y-b)^2=r^2$，其中(a，b)為圓心座標，r 為圓的半徑，用這個三維數據組，讓(a，b)在影像座標內不斷改變位置，找出所有可能的半徑 r，最後當這三維數據組的點數，超過我們定的閾值以及高爾夫球對應於畫面中理想的半徑，即為高爾夫球，並且取得圓心來當作捕捉到之高爾夫球參考座標。

2.3 利用偵測之高爾夫球位置進行落點預測

本專題中我們利用側面的攝影機來計算出高爾夫球最初的初速度以及飛行仰角，搭配由上而下的攝影機來計算出高爾夫球的飛行偏向角，利用這三個資訊來轉換成高爾夫球實際在真實世界的移動座標，以及計算出高爾夫球落地的位置，而初速度以及飛行仰角利用基本的物理概念，斜向拋射運動來計算高爾夫球移動的軌跡，在忽略空氣阻力的情況下，物體飛行時水平方向沒有受到外力作用，所以水平方向維持等速度運動，而鉛直方向受到重力作用，則做鉛直上拋運動，如下圖(三)所示。



水平方向：

$a=0$，$v_x=v_0\cos\theta$，$R=v_0\cos\theta t$

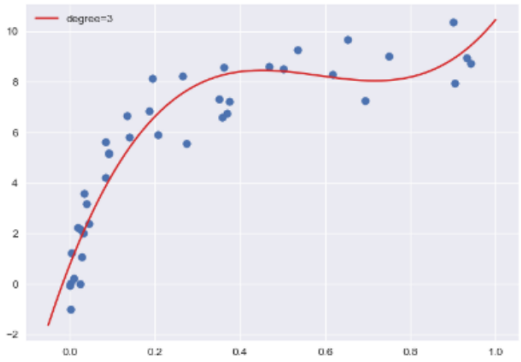
鉛直方向：

$a=g$，$v_y=v_0\sin\theta-gt$，$h=v_0\sin\theta t-\frac{1}{2}gt^2$

▲圖(三)斜向拋射運動基本模型

而從此模型中我們可以觀察到，要想預測高爾夫球的落點，我們必須了解它的初速度以及相對於地面的夾角，因此我們利用側面攝影機所觀察到的高爾夫球座標來模擬高爾夫球的運動軌跡。

這裡我們採用多項式迴歸方程式的方式來模擬自變數時間 t 與高爾夫球高度 z 之間的線性關係，多項式迴歸方程式是平面上許多數據點的一條最佳配適曲線，使用的方法稱為最小平方法。此方法是要使所有數據點到迴歸方程式的殘差平方和為最小，或者是說數據點到迴歸線的垂直距離平方和達到最小。多項式回歸中，數據不太具有線性關係，因此應尋找一些非線性曲線去擬合。我們可以建構許多新的特徵。如下圖(四)所示，用一條三次曲線去擬合數據效果更好。將三次函數看成$z(t)=at^3+bt^2+ct+d$。這樣就又變成解多元，其我們就是要找出 a、b、c、d 使其損失函數最小。



▲圖(四)線性回歸方程式之示意圖

透過此回歸方程式我們可得知高度 z 相對於時間的 t 的位置方程式，將此方程式透過一階微分來獲得速度方程式，再帶入當 t=0時，即可獲得的值為 z 方向在 t=0時的初速度。

而 x 方向的初速度假設無空氣阻力，水平方向維持等速度運動，因此只

要將最後時刻測量到的 x 位置與最初位置的 x 相減除以之間的時間差即可獲的 x 方向相對於 t=0的初速度。獲得 x 方向與 z 方向的初速度即可獲得高爾夫球移動之初速度以及起飛仰角。

利用在擊球點上方的攝影機所偵測到的高爾夫球位置，即可用來計算高爾夫球移動時的偏向角，根據理論，假設高爾夫球無受到任何阻力影響，會呈現固定角方向的移動，因此我們利用簡單的線性回歸方式，來計算初 x 方向相對於 y 方向的直線關係，利用此關係即可獲得高爾夫球移動之偏向角。

利用上述方法我們可獲得高爾夫球運動之初速度、起飛仰角、偏向角，但這裡的初速度為高爾夫球相對於呈現出來的畫面中為每單位時間移動之單位「像素」，因此我們必須將此速度轉換為實際在真實世界中的初速度，因此我們必須測量每個像素在實際的真實世界中占的實際長度為多少，接著再去換算出高爾夫球相對於真實世界之速度。

然而，為了更進一步推算真實環境下的高爾夫球位置，我們將空氣阻力納入考慮，若在空氣中的移動物速度不超過2.5馬赫，空氣阻力與其速度成正比，公式為$F = -k * v$，其中 k 為阻力係數，我們代入實心球體的阻力係數 0.42，不計風速的情況下，經由牛頓第二運動定律推算出動力學方程式如下:

$$m\ddot{x} = -k\dot{x}$$

$$m\ddot{y} = -k\dot{y} - mg$$

其中 m 為高爾夫球質量(約為73.5克)，g 為重力加速度9.8，經過移項整理和積分運算後，代入速度之初始條件: $x'(0) = v0cos\theta$及$y'(0) = v0sin\theta$，其中θ為飛行仰角，再經一次積分運算並考慮$x(0) = 0, y(0) = 0$的初始條件，即可得二維拋物線的軌跡方程式如下:

$$x(t) = v_0 \cos\theta \cdot \frac{m}{k}\left(1 - e^{-\frac{k}{m}t}\right)$$

$$y(t) = -\frac{mg}{k}t + \left(v_0 \sin\theta + \frac{mg}{k}\right) \cdot \frac{m}{k}\left(1 - e^{-\frac{k}{m}t}\right)$$

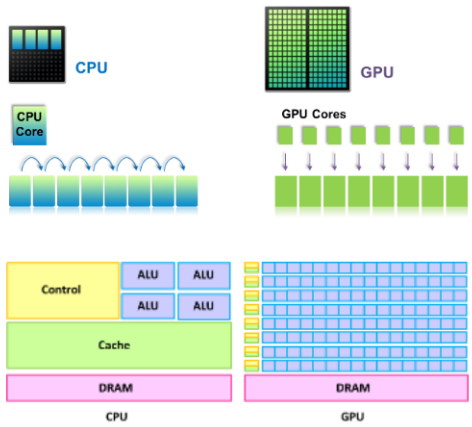
2.4 演算法加速

在本次的專題研究中，我們使用晒像科技之高速攝影機型號為 UI-3040CP-C-HQ-R2提供的 SDK 來做開發，而為了應對高速攝影機之高偵率，因此需要極短的影像處理時間，在這裡我們採用幾個方法來降低影像處理的資料量，來提升運算速度，我們利用了雙線性插值的方式，來進行影像壓縮，

透過維持長寬比的方式，來將原本的圖像資料縮小。

接著我們利用降低影像維度的方式，將原本影像從 RGB 空間轉換到灰階空間，因此相對處理的資料量就會變少。

最後我們利用硬體的方式來提升演算法的速度，也就是利用 GPU 來運算我們所設計的演算法，而這裡解釋一下，為什麼利用 GPU 可達到加速效果呢? GPU 和 CPU 本質上有很大的不同，CPU 具有較大的快取(cache)和較快的計算核心，可以處理較複雜的運算。然而 GPU 不具備這樣的條件，快取較小且單核心計算速度也沒有 CPU 快，只是取而代之的 GPU 具有極大量的計算核心(通稱 CUDA Core)，故其優勢在於適合進行同時間的大量運算。由下圖(五)所示，GPU 在計算時所有核心可高度平行進行計算，而 CPU 則是以順序性(sequential)的方式進行計算。GPU 是專為影像處理所需之計算密集、高度平行計算設計針對前述的應用，都具有計算資料量大且可高度平行化的特性，也就最適合使用 GPU 來計算。



▲圖(五)CPU 和 GPU 設計架構及計算特性

下表(一)為加速前與加速後平均一張影像之運算時間測試:

	加速前	加速後
平均時間	0.09602242040634155 s	0.027108615398406984s

▲表(一)加速前後對照表(測試樣本數量:10000偵)

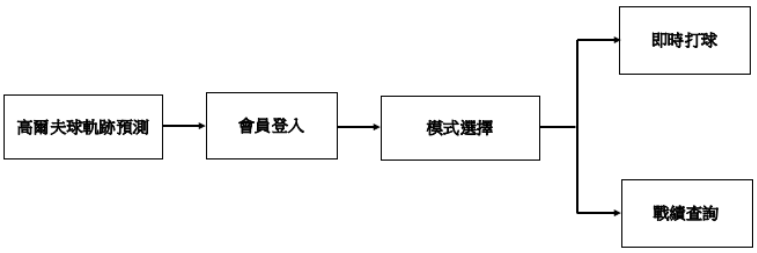
2.5 網頁動畫運行模組

前端網頁動畫整合是透過引入現有3D 模組資源—three.js，在環境及高爾夫球模組的呈現都顯為真實，只需取得擊球初速度、飛行仰角及偏向角，即可使球的位置依照軌跡方程式做變化，為了達到理想落點，高爾夫球於落地後還會做彈跳以此校正，而 three.js 在建置動畫時，須先完成設定視角

camera 以及環境 scene，每當動畫中新增一個物件時，則必須加進 scene 裡面，在球飛行過程中，可以任意觀察球的飛行狀態，最後停止時，視角便會動態跟上球，將此動畫匯入前端主要 HTML，查詢擊球紀錄時，也能以此動畫呈現。

2.6 前端網頁呈現

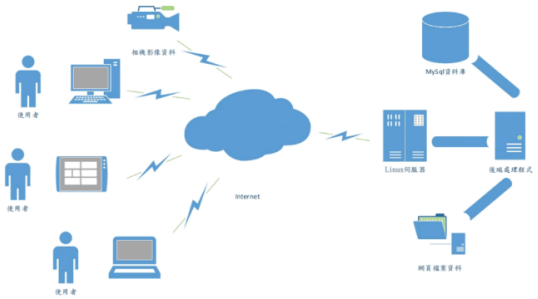
本專題使用 JavaScript-Fetch 作為網頁向伺服器送出請求的工作，如果成功的話，會得到一個帶有 Response(回應)物件值的已實現狀態的 Promise 物件。雖然 Fetch 可以由 Body 的方法來取得回應回來的內容，但因為 Body 屬性值本身是個 ReadableStream 的物件，需要再依照不同的內容資料類型使用對應的方法，才能真正取到資料物件，本次專題是轉換成 json 格式，再根據取得的資料進行其他動作。前端網頁結構圖如下圖(六)所示。



▲圖(六)前端結構圖

2.7 網頁及後端伺服器

為了回應使用者的網頁需求，本專題採用 JavaScript 作為後端處理之程式語言，並使用 MySql 資料庫儲存使用者數據。具體方式為在一台 Linux 作業系統中安裝 Node.js，Node.js 是一個能在伺服器中執行 JavaScript 的軟體，使用 Chrome V8作為核心，可以獲得較高效能；並搭配 Express 框架加速開發流程。而我們所撰寫的程式將透過判斷 Client 端所請求的路徑，給予正確的回應。

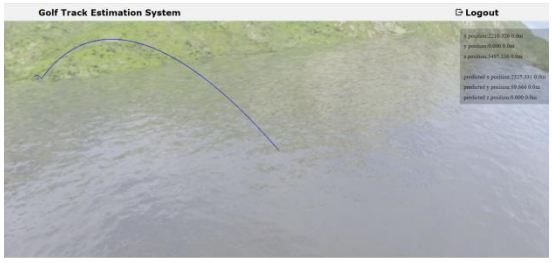


▲圖(七)網路設備結構圖

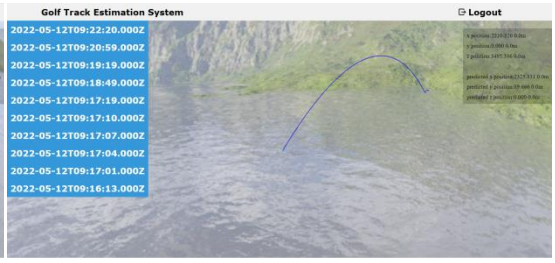
3. 研究成果：

（一）前端網頁呈現

下圖(八)為即時打球頁面，在網頁渲染當下網頁便會向後端請求一筆當下存取在伺服器中最新的數據並儲存，之後網頁每隔一段時間向伺服器請求當下存取在伺服器中最新的數據，並與儲存的數據進行比對，若不相同則將該筆數據渲染，由此達到即時呈現擊球軌跡。圖(九)為成績查詢頁面，圖中藍色部分為供使用者選取的歷史數據。



▲圖(八)即時打球頁面

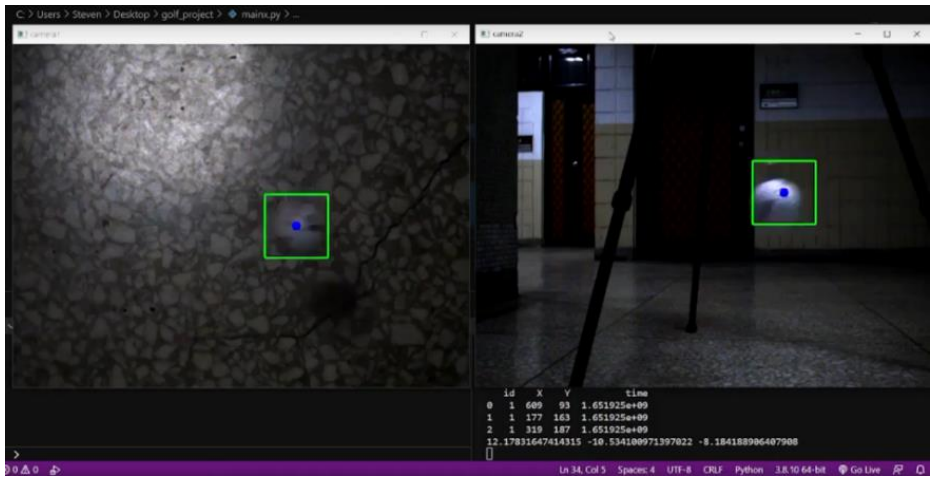


▲圖(九)成績查詢頁面

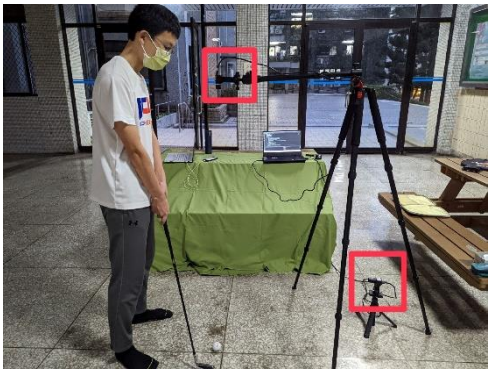
（二）高爾夫球夫球移動物偵測以及即時軌跡計算

圖(十)為偵測高爾夫球之主程式運作結果，且有一個背景運作程式，根據接受到的高爾夫球的位置，計算出高爾夫球軌跡移動時的重要參數。

圖(十一)及圖(十二)為記錄高爾夫球飛行畫面的高速攝影機擺放相對位置，擺放於球員側身之攝影機捕捉高爾夫球初速度及飛行仰角，擺放上方的攝影機捕捉高爾夫球偏向角。



▲圖(十)高爾夫球偵測並且及時計算初速度、起飛仰角、偏向角

▲圖(十一)、圖(十二)高速攝影機擺放位置

4. 結論與展望：

經過這次的專題製作過程，讓我了解到團隊合作的重要性，也讓我學習到整合的重要性以及困難度，因為每個人負責的項目不同，因此在最後需要透過整合來結合每個人負責的項目，也許在個別測試時沒問題，但在整合時可能就會出現許多未知的 bug，這也是最困難的地方，也是最有挑戰性的，而每個演算法都不一定是完美的，根據不同的情況來選擇最適當的方法是最重要的，而針對這次偵測高爾夫球偵測的部分，最大的問題就是，高爾夫球必須在相對於其他前景移動物時，其亮度對比是較明顯的，才會有比較好的效果，而預測軌跡方面，如果球速越快的情況下，能夠捕捉到的位置參考點也相對較少，因此所計算出來的初時移動參數有可能會有誤差，而在計算初速度時只套用了基本的物理模型，未考慮到其他方面，而網頁端呈現部份希望能夠增加更多不同的功能，來提供使用者更好的體驗，針對這些部分，希望未來可以去做改善。

參考文獻：

1、硬體加速搞不懂?CUDA 讓一切變得更簡單
https://www.computerdiy.com.tw/nvidia-cuda/

2、Using Fetch.（2022年3月18日）.MDN
https://developer.mozilla.org/zh-TW/docs/Web/API/Fetch_API/Using_Fetch

3、Circle Hough Transform
https://en.wikipedia.org/wiki/Circle_Hough_Transform

4. Efficient adaptivedensity estimation per image pixel for the task of background subtraction.