# 光流測速

# Speed Estimation using Optical Flow


## 組別 : A86 組員 : 張鈞

## 指導教授 : 鄭桂忠


## 摘要(Abstract)


偵測速度在各種監控系統中是個不可或缺的部分，最常見的使用途徑包含了路上監控車輛超速或者是評估運動員的表現。

傳統上來說，速度偵測系統主要是依賴雷達感測器，其測速方法就是先將波向目標物發送，再利用反射回來的波來計算速度. 然而這種測速方式卻存在著本身的缺點，那就是雷達設備的高昂價格，造成這種系統的利用是被限縮的。

在這個專題中，提出一個利用影像為基礎的測速方式，其中這個系統的估測核心為光流技術. 這個系統需要一個經過實際測量真實大小的光流場，接著就可以估算目標物的速度. 因此，這個系統主要包含三個模組，處理初始影像的模組、追蹤目標物的模組以及真實速度轉換的模組。

為了瞭解整體計畫所需的知識，我先去讀了一些關於光流計算方式的論文及，包括了一些曾經利用類似方法來追蹤物體的文章，充分了解整體系統架構之後，開始一步一步地實作出來。

實作上的主要方向在於如何讓系統能夠穩定的定位運動物體，除了牽涉到光流算法本身的限制之外，還要考慮過於複雜的計算導致系統運作緩慢的問題，因此在設計上，後來加入了圖像金字塔幫助優化光流計算的部分，進而使最後系統能在處理速度及追蹤穩定度上取得不少進步。

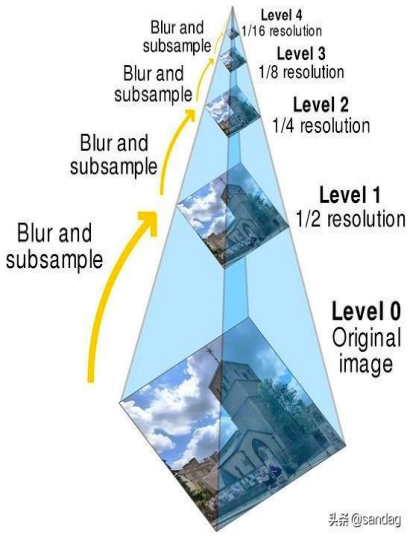
# 報告內容(Introduction)

## 一、架構與設計

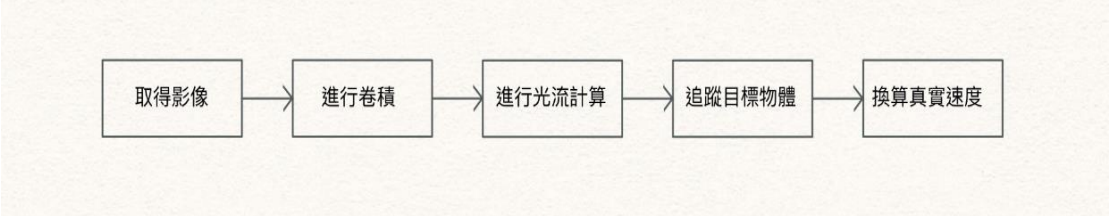
$$I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + H.O.T.,$$

$$\begin{aligned}
\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t &= 0 \text{ or} \\
\frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial t}\underbrace{\frac{\delta t}{\delta t}}_{=1} &= 0 \text{ and finally :} \\
\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t} &= 0.
\end{aligned}$$

Lucas-Kanade 算法原理圖(圖一)




圖像金字塔概念圖（圖二）




測速系統架構圖（圖三）

## 二、設計

### 1. 低通濾波器 ：

在系統實作上，首先取得影片的每一禎的影像，之後將影像送至圖像金字塔的模組中，在這個模組裡，會先利用低通濾波器進行處理，利用一個模糊化矩陣來對影像進行卷積，這個過程的目的就是進行特徵提取，這樣能夠消除相鄰像素之間的差異，經過這個處理過程，原本影像中的物體輪廓會更加被凸顯。

### 2. 圖像金字塔處理 ：

接著再透過刪除偶數行跟偶數列的部分，以每次縮小為原尺寸的 1/4 的速度來進行縮減像素，在這個過程中，為了不要損失太多影像的數據，在每次的刪減像素時，會使用刪減前的影像來做一個迭代的過程，讓刪減後的影像盡量保持原本的細節，在這個專題中，會將這個過程重複四次，將尺寸縮為原尺寸的 1/256，以利於接下來的Lucas-Kanade模組進行計算。

### 3. Lucas-Kanade 計算光流 ：

接著，此時已經選好需要追蹤的目標物，所以接下來就是利用Lucas-Kanade模組來計算出每一禎物標物的位置，這個過程利用前後影像的差異來獲取光流值；因為在圖像金字塔的過程中，像素被縮小了，所以在這個階段，計算光流時就需要依照這個縮小後的影像，放大以還原真正的光流值。在這個過程後，程式會在畫面上呈現出一個綠色圓點，藉此標記目標物的位置，讓使用者得知是否有持續跟蹤目標或是有跟丟的情況。

### 4. 換算真實速度 ：

原本是希望藉由Homography轉換來讓各種角度的拍攝都能有準確的結果，但是不料這個轉換的過程會使程式運作緩慢，在最後剩餘時間不足以改善運作速度的情況下，移除了這個過程，改成採用只需要知道球體運動長度及拍攝鏡頭與運動軌跡垂直距離、就能換算速度的方式來將像素還原成對應的實際長度。

## 三、實驗結果

### 1. 測試環境

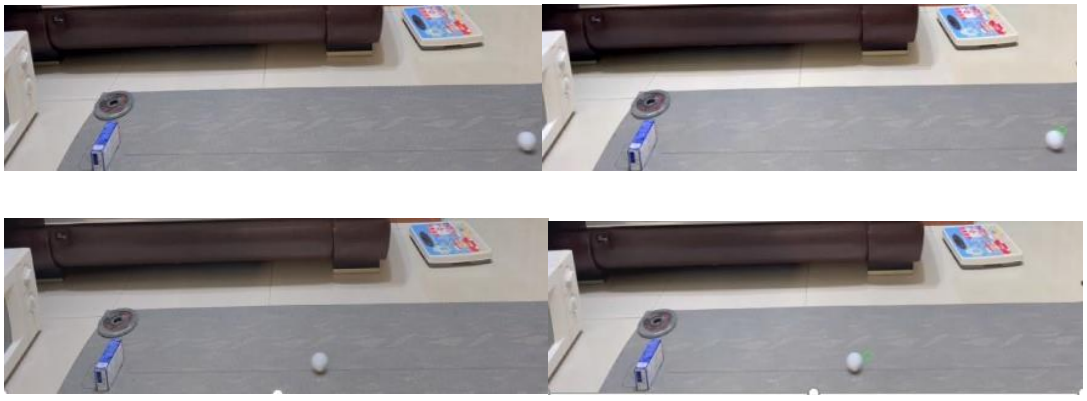
因為測速系統能夠分析的影片有所限制，原本有在戶外拍攝測試影片，但是卻發現戶外的環境變數太多，包括自然風造成相機晃動，導致追蹤結果不穩定，或是背景過於複雜，導致程式難以進行追蹤，所以最後在室內進行拍攝，因為室內空間變因易於控制，且測量也較為容易。

### 2. 追蹤目標物

在球體滾動時，程式需要追蹤球體的位置，在初期只單純使用 Lucas-Kanade 算法來進行追蹤時，只要球體滾動加快，程式就會跟丟，無法符合對於速度的要求; 在加入了圖像金字塔的運用之後，明顯的可以看出程式可以接受的最高速度提升了，也因此降低了跟丟的機率。

### 3. 換算真實速度

在前面的實驗原理有提到，因為拍攝影像的一些失真，所以要還原真實速度，需要先將影像經過 Homography 處理，但是在經過 Homography 的運算後，整體程式的運算速度降低許多，讓分析的時間變得太長，也失去了即時測速的意義，為了保持夠低的運算時間且兼顧分析的正確性，決定暫時放棄用 Homography 處理影像，為了保持正確性，鏡頭的拍攝畫面變成平行於運動軌跡，如此一來，就大幅降低因為拍攝角度造成的失真問題，而運算時間也得以不被拖慢。



拍攝的影像和經過追蹤分析的影像（圖四之一、圖四之二）

# 心得感想

在這個測速系統的實作過程中，大部分的結果都與理論相符，像是物體運動速度的限制，或者是對固定背景空間的要求等等，都能經由實作看出這些要求及限制的目的，然而在Homography轉換的部分，在理論上當我們把測試的背景轉換後能夠獲得最正確的空間條件，再拿來做分析就可以得到結果，但是在實際利用程式來進行轉換之後才發現這個轉換的過程花了不少時間，縱使最後的結果是對的，但是卻讓這個系統太慢，就失去了即時測速的基本要求，所以最後的實作結果仍然需要和現實妥協，而非單純的依理論照做。

這次的專題題目是指導教授實驗室沒有做過的題目，因此許多部份都需要從頭學起，自己尋找相關資料，在前期時經常遇到程式無法產出對應的功能、或是程式執行相當耗時等問題，一開始尋找解決方法時會多花點時間，但漸漸熟悉之後，就大概抓得道問題點在哪裡，也知道相關資料如何取得，解決問題的速度就變快了，比較可惜的是Homography轉換的部分，最後因為時間有限，所以沒有辦法將這個部分改進到可以實用的階段。

不過在經過多次修改光流算法、加入圖像金字塔之後，這個測速系統逐漸接近一開始希望能夠達成的目標，雖然可能欠缺了一些完整性，但有達成自己一開始的研究動機。