行政院國家科學委員會補助

大專學生參與專題研究計畫研究成果報告

＊ ********* ** ＊
＊ 計 畫 ＊
＊ ： 結合軌跡追蹤與擴增實境之 XNA 遊戲設計 ＊
＊ 名 稱 ＊
＊ ********* ** ＊

執行計畫學生： 劉書銘
學生計畫編號： NSC 99-2815-C-216-006-E
研 究 期 間 ： 99 年 07 月 01 日至 100 年 02 月 28 日止，計 8 個月
指 導 教 授 ： 陳建宏

處理方式： 本計畫涉及專利或其他智慧財產權，2 年後可公開查詢

執 行 單 位： 中華大學資訊工程學系

中華民國 100 年 03 月 31 日

目　　錄

圖 表 目 錄

摘　　要

現今娛樂市場發展成熟，遊戲內容越來越豐富，畫質也越來越精細，但始終操作模式，依然是使用機械式的控制器來進行遊戲，亦或是藉由控制器的無線偵測產生出不同的動作。如果能擺脫使用控制器的限制，直接對攝影機做直覺化的動作分析，那玩遊戲不再局限於雙手操作，而是整個人體動作模擬於遊戲之中。

本研究計畫，內容運用均值位移演算法（Mean-Shift Tracking Algorithm）之軌跡追蹤概念，在微軟 XNA 遊戲設計平台上實現自然人體操控之擴增實境（Augmented Reality, AR）娛樂遊戲。此研究可以使玩家擺脫鍵盤、滑鼠、等其他控制器，達到高互動且無負擔之娛樂性。

本研究計畫的概念，藉由實現一個結合人體操控，來更深入了解圖形識別與擴增實境等相關技術之原理與應用。如下，[圖一]。

[圖一] 研究結果之概念圖

關鍵字：Microsoft XNA 3.1, Mean-Shift Algorithm, Kernel Mean-Shfit Algorithm, Bhattacharyya Coefficient, Augmented Reality, Haar Cascade

1 研究背景及目的

在許多電影或是概念影片中，常常看見裡面的人物，藉由身體部位的移動，可以不靠任何控制器或是外部裝置，操作某些物品或是機器。很多人認為這只是動畫或是假想概念，在現實生活中是不可能發生的事情。但在國外，類似的人體操控技術卻從以前就開始構想，因在國內少有文獻研究，絕大多數的人還認為那是虛幻，不可實行的。假若此項技術能應用於生活中，擺脫舊有的實體操作介面控制，帶給多數人的便利，那這在未來絕對是造福人群的新技術。

本計劃以微軟 XNA 遊戲開發平台，進行系統設計，將這項人體操控之概念應用於娛樂控制上，改變其現有的控制器操作模式，把滑鼠與鍵盤無形化，只靠攝影機就能達到人機控制，偵測追蹤玩家特定部位之動作、藉由均值位移演算法(Mean-Shift Tracking Algorithm) 來記錄其特定部位之運動軌跡，結合擴增實境概念產生實體加虛擬的效果來建構一項自然操控的遊戲架構。

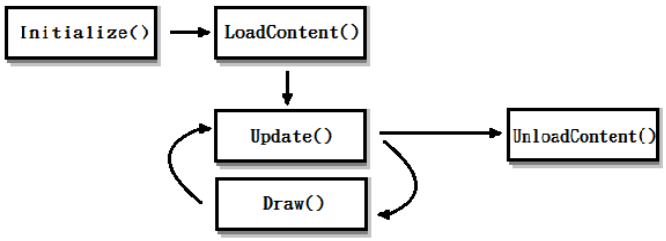
2 研究技術簡介

傳統的遊戲和現代目前流行的 PS3、XBox360、Wii 來說，這些遊戲方式主要還是搭配使用額外的控制器來進行遊戲，但是如果不想用控制器玩樂怎麼辦？雖然微軟近期內也推出 Kinect 體感遊戲攝影機，但其設備價錢昂貴，還是沒辦法普遍性可以人手一台？所以為了解決這些問題，我們研究使用低價位的攝影機做為應用媒介，達到高娛樂的互動性；透過在攝影機前偵測人體部位的特徵和移動軌跡，並行處理相關遊戲動作與效果。

物體追蹤透過比對連續影像間，物體的相似度來完成，其中的議題涵蓋兩種 1. 如何建立物體的特徵與相似程度的判別，2. 如何在影像中快速尋找目標物。為了解決問題，我們實作均值位移演算法(Mean-Shift Tracking Algorithm)[6-13]，這種區域式追蹤方法(Region-Based Tracking)，可將影像變動的區域視為有移動物體的存在，藉著偵測這些變動區域的位置，達到追蹤的目標。

擴增實境(Augmented Reality，AR)，是一種計算攝影機影像的位置或角度，並加上相應圖像的技術，附加一些資訊。這種技術目標是在螢幕上，把虛擬世界套在現實世界中進行互動。使用者看到是由電腦系統處理好、已結合虛實影像，這也是本研究計畫預期將引入的概念。

XNA 的全名是 XNA Game Studio[1]，在 2006 年底微軟公司所發表的一個跨平台的次世代遊戲開發介面，提供一個免費的遊戲開發專屬環境，適用於 Microsoft Visual Studio 的整合開發環境(IDE)延伸模組，以.NET Framework 為基礎，加入支援遊戲應用所需的函式庫(XNA Framework)，為開發工作者提供一個 XNA 遊戲的基本架構作為起跑點，並賦予遊戲開發者有能力以 XNA 平台來自訂與發展遊戲內容，進而建立出屬於自己的遊戲作品。如果玩家擁有 Xbox360 遊戲主機，還可以透過專屬 XNA 開發者俱樂部(XNA Creator's Club)[2]，與其他玩家共享彼此開發的遊戲程式。[3-5]。當建立一個 XNA 遊戲專案，預先設置開發者常用的五個重要的函數如圖二。



［圖二］XNA 遊戲主要架構流程

Initialize( )：遊戲初始化，在此函數大部分處理為物件初始化、視窗大小設定等。

LoadContent( )：載入相關圖形內容，如：3D模型、2D貼圖、音效檔、文字設定檔…等，這些在遊戲中出現的資源。編譯時會經由 Content Pipeline 轉換成 XNA 支援的某種內部格式，也就可以將這些資源載入進來。

Update( )：處理更新狀態， 如：玩家輸入、偵測碰撞、人工智慧演算法…等。

Draw( )：將所有物件納入遊戲中，使這些物件運用圖形裝置，繪製畫面。

UnloadContent( )：進行LoadContent的資源釋放，像是結束遊戲時就會自動執行的函數。

3 研究方法及進行步驟

3.1 節介紹背景相減法(Background Subtraction)，將影像分離成前景與背景，減少繁瑣計算，增加追蹤效率。

3.2 節介紹均值位移演算法(Mean-Shift Tracking Algorithm)之方法流程，依序從最基本的直方圖(Histogram)的運用、相似係數(Bhattacharyya Coefficient)的比較、Basic 均值位移演算法步驟、還有基於 Kernel 模型的均值位移演算法應用；最後能追蹤，就可以開始記錄移動軌跡，進而對應資料庫，產生遊戲效果。

3.3 節介紹擴增實境(Augmented Reality)的應用，將一些虛擬的資訊、圖片付加在實體影像中，藉由 Haar Cascade 方法來實作。

3.1 Background Subtraction

背景相減法(Background Subtraction)[10]，是一個很常見的移動物偵測方法。其在連續影像中，先建立環境背景，做為學習的判斷依據。將輸入影像的每個像素點(Pixel)色彩 RGB 值，與背景影像的 RGB 值相減，把結果取絕對值與某個限制範圍(Threshold)做比較，即可將輸入影像分成前景點(Foreground Pixel)與背景點(Background Pixel)。其運用公式(1)如下：

$$F_k(\mathrm{x},\mathrm{y}) = \begin{cases} I_\mathrm{k}(\mathrm{x},\mathrm{y}), & |I_\mathrm{k}(\mathrm{x},\mathrm{y}) - B_\mathrm{k}(\mathrm{x},\mathrm{y})| > T_\mathrm{d} \\ \mathrm{R}=0,\ \mathrm{G}=0,\ \mathrm{B}=0, & \text{otherwise} \end{cases} \quad (1)$$

其中$F_\mathrm{k}(\mathrm{x},\mathrm{y})$代表第 K 個影格(Frame)時，前景影像$F_\mathrm{k}$上的點(x,y) RGB 值；$I_\mathrm{k}$為第 K 個影格的影像；$B_\mathrm{k}$為背景影像；$T_\mathrm{d}$為差異強度的限制值(Threshold)。

［圖三］ B_k背景

［圖四］ I_k追蹤物

［圖五］ F_k前景

3.2 Mean-Shift Tracking Algorithm

3.2.1 Attribute Modeling with Histogram

在電腦視覺中，最好的應用就是直方圖(Histogram)，它是一種二維統計圖表，可分析色彩的頻率分佈；也就是所有影像點(Pixel)值統計於直方圖的容器(Bin)中；其可觀察色彩分佈。再利用正規化(Normalization)方法，如公式(2)：可以縮小直方圖比例，以利影像處理之圖像計算。

$$\text{Normalization} = \frac{\text{the value of each bin}}{\text{the total value of the bins}} \tag{2}$$

3.2.2 Histogram Matching Bhattacharyya Coefficient

為了分辨追蹤物，常利用(Minimum Bounding-Box)方法，找出追蹤區塊(Tracking Area)，再分析成直方圖。因為時間因素，追蹤則有先後順序：時間點較前者為原始直方圖(Target Histogram)，時間點較後者為目標直方圖（Candidate Histogram）。

有了那兩種直方圖，即可使用相似係數(Bhattacharyya Coefficient)[7]做比較，判斷出在兩個時間點上，追蹤區塊的相似程度與差異性。其方法先將兩個直方圖正規化計算(Histogram Normalization)，依序把每個容器的兩個比例值，也就是(Target Frequency, p_u)與(Candidate Frequency, q_u)，相乘再開根號連加；即可求出在不同時間點的追蹤區塊之相似程度ρ，如公式(3)。註：u 為容器的編號。

$$\rho = \sum_{u=1}^{m} \sqrt{p_u q_u} \tag{3}$$

3.2.3 Basic Mean-Shift Tracking

均值位移演算法(Mean-Shfit Tracking Algorithm)於二維離散影像中，連續每一回合都有兩張影格(Target Frame, Candidate Frame)處理。先分析各個追蹤區塊的直方圖，比較相似係數(Bhattacharyya Coefficient)，利用均值位移疊代法(Mean-Shift Iteration)，迴圈式的在影格(Candidate Frame)中找尋最接近，且最相似於影格(Target Frame)的追蹤區塊。找到後，對追蹤區塊計算權重值，求得新的質心位置，即可更新追蹤點，產生均值位移向量(Mean-Shift Vector)。

在此歸納基本的均值位移演算法簡易步驟，其 Step 3~5 做為均值位移疊代(Mean-Shift Iterations)：

Step 1. 在原始影格中(Target Frame)，針對某個追蹤區塊，產生(Target Model, Q)，計算其直方圖，正規化縮小比例(Normalization)。

Step 2. 針對下一張影格(Candidate Frame)，同樣與Q相同的追蹤位置(x)，再產生(Candidate Model, P)，計算其直方圖，正規化縮小比例。

Step 3. 在那追蹤區塊(Region of Interest, ROI)中，用 i 來代表每個像素點(Pixel)，且 u 代表在直方圖中每個容器(Bin)的編號值。運用公式(4)，計算每個像素點(Pixel)的權重值。

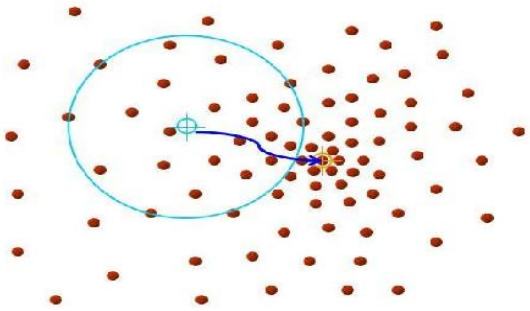
$$w_i = \sqrt{\frac{q_u}{p_u}} \tag{4}$$

Step 4. 更新均值位移向量(Mean-Shift Vector)，如公式(5)。

$$M(x) = \frac{\sum_i w_i x_i}{\sum_i w_i} - x \tag{5}$$

Step 5. 更新目前追蹤位置且開始疊代(Iteration)，如公式(6)。

$$x \leftarrow \left(x + M(x) = \frac{\sum_i w_i x_i}{\sum_i w_i} \right) \tag{6}$$



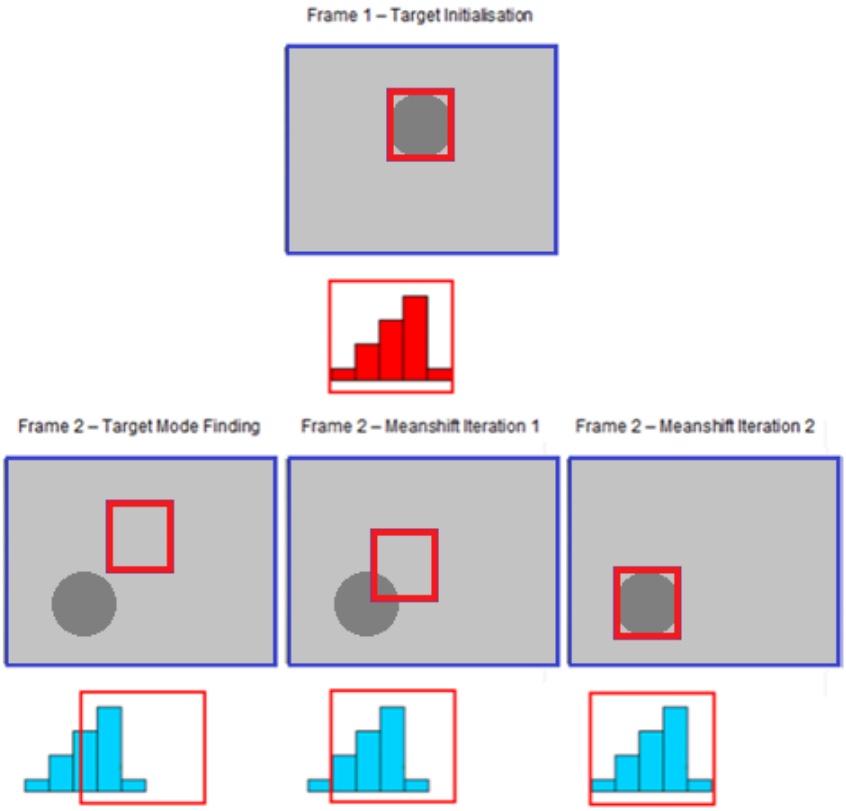
[圖六] Mean-Shift Tracking

為了更清楚說明均值位移演算法，在此也利用示意圖表示，如圖七。

Frame1 為一張影像，計算(框框)追蹤區塊的直方圖(Target)。

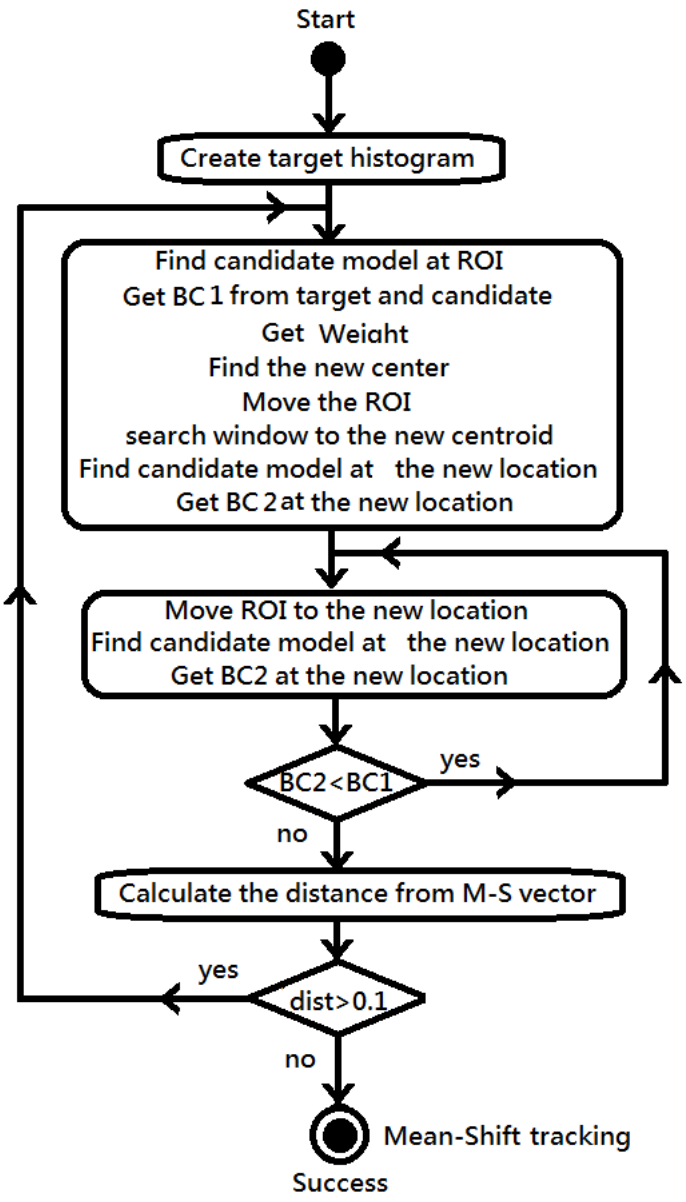
Frame2 為下一張影像，也計算(框框)追蹤區塊的直方圖(Candidate)。

利用相似係數判斷 Frame1 與 Frame2 的追蹤區塊;若每次追蹤相似度較前一次低，則開始疊代(Mean-Shift Iteration)，並更新均值位移向量，其判斷條件為移動距離誤差小於某個限制範圍(Threshold)，例：0.1 單位。經過幾次的疊代(Iteration)，就可以找到相似度最高且最吻合的追蹤物。



［圖七］均值位移演算法示意圖

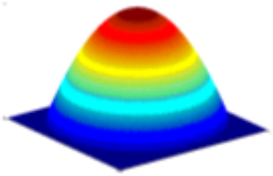
本系統中，均值位移演算法流程圖，如圖八。利用攝影機偵測影像，每秒15個影格(Frames)的動態背景下，做到移動物體的自動追蹤，可以有更多的彈性，且準確又快速的找尋追蹤物。



［圖八］均值位移演算法流程圖

3.2.4 Kernel-based Mean-Shift Tracking

基於 Kernel 模型的均值位移演算法為進階應用[8]，從追蹤區塊(ROI)，判斷其核心 Kernel 模型在整個範圍的分佈情況。在此使用 Epanichnikov Kernel，作為追蹤模型，如圖九，公式(7)；其形狀越接近中心權重越大，以防止邊緣受到背景影響，故可以增加追蹤的準確度。



［圖九］ Epanechnikov Kernel 形狀

$$K_E(x) = \begin{cases} c(1-\|x\|^2) & \|x\| \le 1 \\ 0 & otherwise \end{cases} \tag{7}$$

Kernel 與 Basic 方法差異在，原本的 Basic 是利用真實的 RGB 色彩值去做分析。在直方圖中，對於某個容器(Bin)累加 1；但是在 Kernel 方法中是判斷每個像素點，相較於模型中所在位置，套用公式計算權重值(Kernel Value)，累加於某個容器 (Bin)中。所以 Kernel 模型可以決定直方圖的分佈，讓追蹤效果更好。

Epanechnikov Kernel，是一個很常使用的模型。在此有幾個重要步驟如下：

Step 1. 找尋追蹤區塊(ROI)的中心位置，如公式(8)：

$$X_c = \frac{\text{Left} + \text{Right}}{2}$$
$$Y_c = \frac{\text{Top} + \text{Bottom}}{2} \tag{8}$$

Step 2. 針對每個像素點 I(x, y)找尋正規化後的 X 與 Y 座標，如公式(9)：

$$X_n = \frac{2 * (x - X_c)}{\text{Width of ROI}}$$
$$Y_n = \frac{2 * (y - Y_n)}{\text{Width of ROI}} \tag{9}$$

Step 3. 針對每個像素點(Pixel)，找尋 Kernel Distance，如公式(10)：

$$\text{Dist} = \sqrt{X_n^2 + Y_n^2} \tag{10}$$

Step 4. 找尋 Kernel Value，為 Epanechnikov Kernel 的計算結果

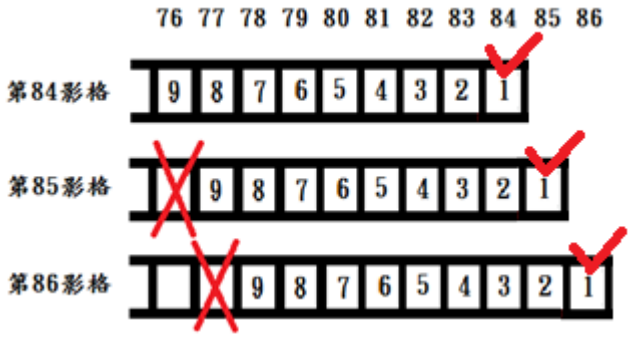
If $\text{Dist}^2 < 1$ $K = \frac{2(1-\text{Dist})}{\pi}$

else $K = 0$

3.2.5 Trajectory Tracking

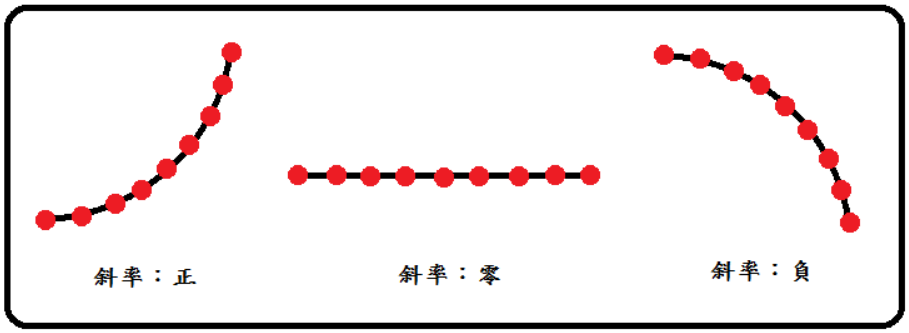
在電腦硬體設備中，攝影機的擷取畫面，通常每一秒有 30 張影格(Frame)，平均每張間隔約 33 豪秒(ms)，其時間過得非常短暫，影格間的變化是人眼感覺不出來。然而每秒 30 次，計算量較大，會降低追蹤效率。經過我們測試發現，以 66 毫秒(ms)來擴大計算的緩衝時間，是不會影響追蹤效率，故本系統以每秒 15 個影格(frame)做為一秒頻率；將每個追蹤區塊的中心點之座標位置記錄下來，做為軌跡判讀的依據。

以本系統來說，軌跡的分析每次只取樣前 9 個記錄點(也就是以當下時間為基準，前 594 毫秒 = 0.594 秒 = 9 個記錄點)；每新增一個，則捨棄最舊的記錄點，如圖十。



［圖十］系統取樣記錄點示意圖

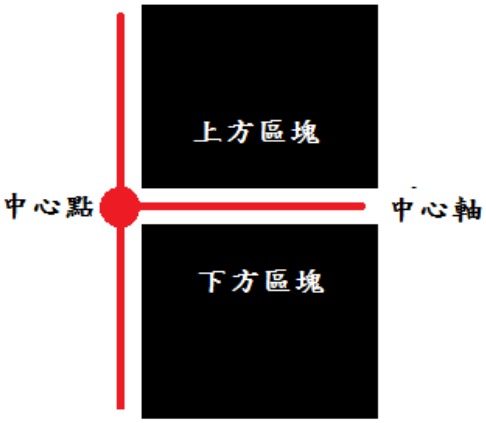
本系統應用在賽車娛樂遊戲上，其控制的主要效果為 1.方向控制、2.速度控制；而玩家以右手做為遊戲的追蹤目標，使用均值位移演算法追蹤，將軌跡分類成三種判斷，如圖十一。



［圖十一］玩家右手的軌跡分類

如果 9 個記錄點的軌跡移動總距離，少於某個限制值(Threshold)，例如距離 60 單位，則判定為不改變效果，持續不變。若大於限制值，則判定追蹤物有移動軌跡可以更新效果。

遊戲系統的**方向控制**：模擬真實開車操控方向盤，以中心軸分上下區塊，如圖十二：區塊的軌跡移動(所有取樣記錄點的斜率變化)產生不同效果，如表一。



［圖十二］可追蹤的範圍

右手控制虛擬方向盤		
斜率辨別	**中心軸上方區塊**	**中心軸下方區塊**
左彎效果	由負趨近於零	由趨近於零變正的
	斜率變大	斜率變大
右彎效果	由趨近於零變負的	由正趨近於零
	斜率變小	斜率變小
直線前進效果	斜率為零	

［表一］軌跡影響的遊戲控制效果

遊戲系統的**速度控制**：以右手(追蹤物)與自行設定的中心點，距離的控制，在某個限制距離(Threshold)內為玩家停止不動的效果，大於則有加速度的功能。也就是說，距離越大速度越快，距離越小速度越慢。

3.3 Augmented Reality

3.3.1 Haar Cascade Classifiers

在此使用 Haar Cascade 技術實作中，主要基於哈爾分類器(Haar Classifier)的物件偵測特徵(Haar-like Features)。從特徵中有兩三個群組影像點，伴隨著相關的對比變異數來實作。Haar Cascade 可以搭配 OpenCV(或 EmguCV)來應用，在網路上有很多開放原始碼，可以免費提供下載使用，藉由訓練自己的分類器 XML 文件產生效果[13]

以本研究，應用在賽車遊戲上，擴增實境為虛擬裝備附加在玩家身上，它是一種結合虛擬化技術再來觀察世界的方法。在玩家的視覺上，彷彿穿了一件衣服或帽子。在攝影機的實體畫面中，擴增實境對象是以人臉做為媒介，可以看到這些虛擬資訊，如圖十三：面具、方向盤、等。以這些虛擬資訊，來輔助遊戲擴增實境的效果。

［圖十三］擴增實境的應用

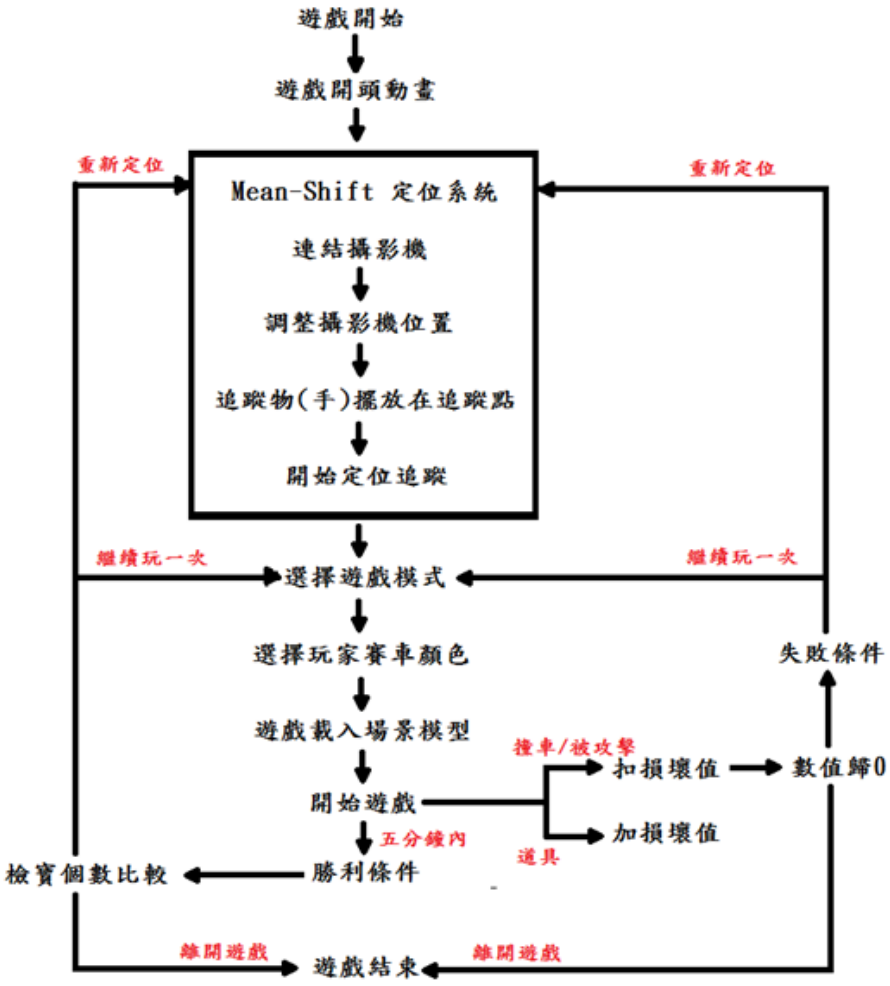
4 研究結果

4.1 使用的軟硬體設備

硬　　體	網路視訊攝影機乙個
	顯示卡支援 DirectX 9.0c 和 Shader Model 1.1 以上
軟　　體	Microsoft .Net framework 3.5 以上版本
	Microsoft XNA3.1 以上版本
	EmguCV2.1.0.793 以上版本

［表二］本系統軟硬體設備

4.2 遊戲流程圖



［圖十四］本系統遊戲的流程圖

4.3 系統介面

實作的系統介面，如圖十五。

［圖十五］遊戲介面

遊戲介面功能如下：

左上角：1. 道具顯示(紅格和藍格)兩種
2. 目前撿寶數判斷自己的名次

左中角：各個玩家(顏色) 所撿到的寶物數顯示，並名次排列

左下角：玩家在中華大學場景地圖中，賽車的位置與 Copyright 連絡方式。

中上角：時間顯示。

中中角：賽車於 3D 賽道中。

中下角：1. 攝影機畫面的右下角有一個框框，偵測追蹤物。如同真實開車一樣，轉動虛擬方向盤，右手的軌跡移動，可以產生不同的效果例如右轉、左轉，然而賽車的速度是由手與方向盤的距離控制，可快可慢。
2. 攝影機畫面的左下角有兩框（紅/藍），做為膚色判斷使用道具
3. 虛擬實境顯示於實際攝影機畫面當中。

右上角：1. 顯示目前撿寶數（Star 條）。
2. 玩家車子損壞值（GJ 條），撞車則減，補品則加，歸零則 GameOver。

右中角：玩家賽車速度顯示，遊戲 FPS 數值顯示。

右下角：擴增資訊顯示資訊。

4.4 完成之工作項目及具體成果

本研究，完成一個結合軌跡追蹤與擴增實境之 XNA 娛樂遊戲，可以依人體特定部位之特徵，準確且快速的進行追蹤並紀錄軌跡，藉由其軌跡啟動相對應之控制或是效果。成果中包含下列模組：

1. 人體特定部位追蹤系統。
2. 軌跡分析及效果控制系統。
3. 實境與虛擬資料融合系統。
4. 互動娛樂系統。

5 結論與未來展望

在研究過程中，主要針對均值位移演算法實作與擴增實境的 XNA 遊戲設計。結合這些技術，應用在一款 3D 的賽車遊戲上，玩家只需要安裝攝影機，開啟遊戲主程式，即可擺脫鍵盤、滑鼠等其他控制器，達到高互動且無負擔之娛樂性。玩家面對著攝影機做出移動軌跡，系統就可以自動判讀，而產生出不同的遊戲效果，以實現自然人體操作方式。

然而此系統，在未來可以擴增至更多應用，社會上一些行動不方便的殘疾人士(如：雙手萎縮的小兒麻痺症病患)，雙手無法握力，如果想要學習駕車，欲使用本系統影像追蹤功能，模擬車輛方向盤控制，只需要對著攝影機做軌跡移動，即可操控車輛！

本計劃之成果也可以延伸於遠端家電控制等系統上。因為網路即時通訊越來越發達，攝影機的使用已成大眾化，人人電腦前都有一台。藉由本計畫，可以讓攝影機不再是視訊系統的使用專利，而是把這項電子產品，擴充在人類娛樂生活上；繁瑣的實體操作介面已不再有，現在未來科技趨勢就是隔空控制人機介面！只要依靠影像追蹤技術，相信在未來絕對是一大商機。

參 考 文 獻

[1] XNA Developer Center,Microsoft, http://msdn.microsoft.com/zh-tw/xna/default.aspx

[2] XNA Creators Club Online, Microsoft, http://creators.xna.com/en-US/

[3] 葉思義, 李震宇, "XNA・PC XOBOX 360 C#遊戲程式設計", 碁峯出版社, 2009

[4] 鄧永傳, 何振揚, "2D/3D 遊戲程式設計入門-使用 XNA3.0 與 C#", 文魁出版社, 2009

[5] Aaron Reed, "Learning XNA 3.0 實戰手冊", O'REILLY, 2009

[6] D. Comaniciu and P. Meer, "Mean Shift Analysis and Applications", Proc. Seventh International Conference Computer Vision, pp 1197 – 1203, September 1999.

[7] D. Comaniciu, V. Ramesh and P. Meer, "Real-Time Tracking of Non-Rigid Objects Using Mean Shift", Proc. 2000 IEEE Conference Computer Vision and Pattern Recognition, vol. II, pp. 142 – 149, June 2000.

[8] D. Comaniciu, V. Ramesh, P. Meer , Kernel-based object tracking, Real-Time Vision & Modeling Dept., Siemens Corporate Res., Princeton, NJ, USA, 2003.

[9] Gary Bradski, "Computer Vision Face Tracking For Use in a Perceptual User Interface", Proc. IEEE Workshop Applications of Computer Vision, pp. 214 – 219, October 1998.

[10] Elad Ben-Israel, "Tracking of Humans Using Masked Histograms and Mean shift", The Interdisciplinary Center Herzliya, March 2007.

[11] C. Yang, R. Duraiswamim and L. Davis, "Efficient Mean-Shift Tracking via a New Similarity Measure", Proc. 2005 IEEE Conference Computer Vision and Pattern Recognition, 2005.

[12] Computer Vision Applications with C#, part1. Part 2. Part 3. http://www.codeproject.com/KB/GDI-plus/MeanshiftTracking.aspx

[13] Haar Cascode with openCV, XML http://www.alereimondo.com.ar/OpenCV