美和學校財團法人美和科技大學

102 年度教師產學合作計畫

結案報告書

計畫名稱：利用 kinect 辨識人體姿勢之研究

計畫編號：102-FI-DIT-IAC-R-003

計畫期間：民國 102 年 6 月 1 日起至

民國 103 年 3 月 31 日止。

計畫主持人：游義地

共同主持人：李鐘孝

研究助理：

經費總額：50,000 元

經費來源：碁峰資訊股份有限公司

摘要

在影像感測方面，微軟公司所發展的 Kinect 感測器不僅可利用影像與聲音作為訊號輸入，並可與環境產生互動，增加深度量測。Kinect 具有彩色、深度及聲音三種感應器，可結合 3D 影像偵測、人體骨架追蹤與聲音訊號處理等技術，提高影像辨識應用之功能。

本計畫採用 C#程式語言，利用 Kinect 所提供的骨架資訊，結合類神經網路達到姿勢辨識之目的。所開發之軟體可提高 Kinect 感測器的應用價值，利用所建立之 Kinect 感測器判讀技術，可作為姿勢判別、物理復健或健康照護之輔助工具。

關鍵字：kinect、類神經網路

第一章　　前言

隨著科技的進步，訊息輸入方式不再侷限於滑鼠、鍵盤或觸控方式。近年來，利用眼動訊號、腦波訊息及影像的追蹤，已被充分應用於資訊設備的控制。在影像感測方面，微軟公司所發展的 Kinect 感測器[1]不僅可利用影像與聲音作為訊號輸入，並可與環境產生互動，增加深度量測，結合 3D 影像偵測、人體骨架追蹤與聲音訊號處理等技術，提高影像辨識應用之功能。

影像辨識已廣泛應用於自動化產品檢驗上，近年來有多位學者將影像感測技術應用於機器人導控方面[2~4]，以人體移動軌跡達到操控目的。周敬敏[5]利用 kinect 偵測的人體特徵點，結合幾何運算建立人體移動軌跡。

本計畫以開發 Kinect 人體骨架追蹤軟體為目的，利用 C#語言[6]建立人體骨架模式與偵測，並利用類神經網路提高姿態判讀之準確性。計畫執行後，本團隊對 kinect 感測器的實務操作更加瞭解，並建立 kinect 感測器的研究能量，成果如下：

1. 協助合作廠商開發 kinect 感測器應用程式，提高 kinect 感測器的應用價值。
2. 利用 kinect 感測器判讀人體姿態，可作為物理復健或健康照護的輔助工具。

第二章　研究方法

2.1 硬體與開發環境

由於 C#已廣泛運用在資訊平台應用軟體及遊戲軟體開發上，且程式設計相關資料十分完備，本系統程式採用 C# 語言，結合 ACCESS 資料庫，達到資料儲存之目的。開發本系統所用之軟體如表 1 所示。

表 1：軟體開發環境

開發環境	優點
開發語言 C# 2008	選擇 C#為開發工具的好處，是可以以極短的時間與成本製作出 Windows 各式應用軟體、資料庫軟體等。
資料庫 ACCESS	1. 有效共用資訊，設計可在 ACCESS 中儲存並開啟的表單和報告，使共用資訊更為輕鬆容易。 2. 協助消除錯誤： 錯誤檢查功能能以旗標標出在表單和報告中一般錯誤，使得測試和修正錯誤更為迅速。
執行環境 Windows XP	使用較普及

Kinect感測器

Kinect感測器如圖1所示，配合應用程式可以取得影像三種資訊：

1. 彩色影像 (透過中間那顆 RGB 鏡頭)
2. 3D 深度影像 (透過左右兩顆鏡頭)

   紅外線發射器和紅外線 CMOS 攝影機

3. 聲音 (透過陣列式麥克風)

圖 1：Kinect 感測器

Kinect 也支援追焦功能，底座馬達會隨著焦點人物而轉動 Kinect 方向 (左右各 28 度)，詳細規格如表 2：

表 2：Kinect 規格表

感應項目	有效範圍
顏色與深度	1.2 ~ 3.6 公尺

感應項目	有效範圍
骨架追蹤	1.2 ~ 3.6 公尺
視野角度	水平 57 度、垂直 43 度
底座馬達旋轉	左右各 28 度
每秒畫格	30 FPS
深度解析度	QVGA (320 x 240)
顏色解析度	VGA (640 x 480)
聲音格式	16KHz, 16 位元 mono pulse code modulation (PCM)
聲音輸入	四麥克風陣列、24 位元類比數位轉換 (ADC)、雜音消除

2.2 類神經網路

2.2.1 倒傳遞類神經網路

倒傳遞類神經網路(Back-propagation Neural Network)[7]是一種具有學習能力的多層前授型網路。此種神經網路，是由 Rumelhart、Mcclelland 在 1985 年所提出的，當初他們之所以會提出此種網路，是希望提出一種平行分布訊息的處理方法來探索人類認知的微結構。

神經元(Neuron)結構如圖 2-1 所示，可分為:

- 輸入神經樹(Dendrites)：可將訊號輸入細胞體的路徑。
- 神經軸(Axon)：訊號由細胞體輸出的路徑。
- 神經節(Synapse)：影響傳遞訊號的強度。

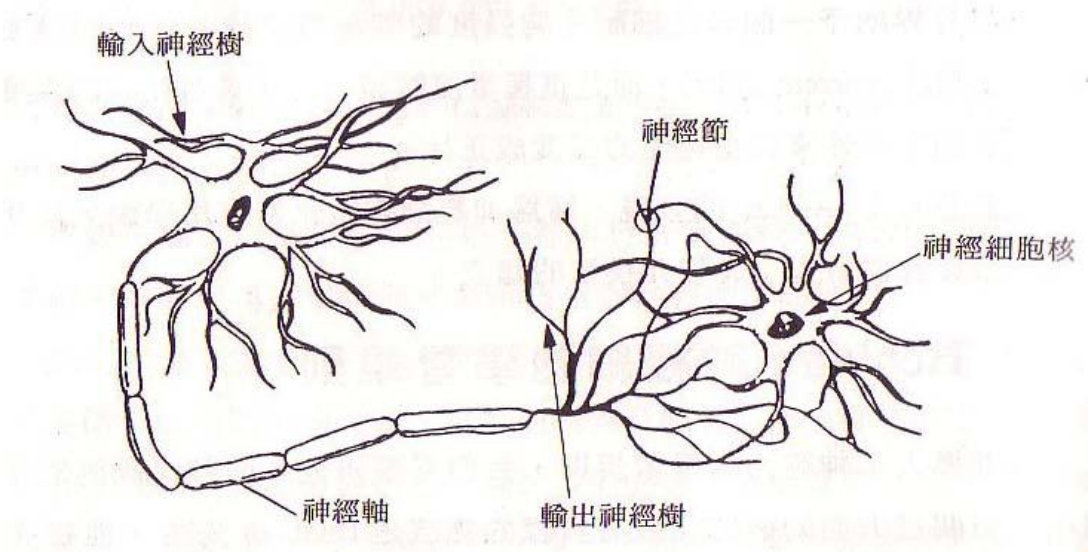

圖 2-1 神經細胞結構圖

資料來源：http://neuron.csie.ntust.edu.tw/homework/93/nn/homework2/m9304302/12.htm

2.2.2 網路架構

倒傳遞網路的網路架構如圖 2-2 所示，包含了輸入層、隱藏層、輸出層；而以隱藏層可以不只一層。基本上，其網路架構於前授型多層感知機的網路架構類似，且每一層皆由一些神經元建構而成。請注意：在圖 2-2 中，同一層中的神經元彼此並不相連，而不同層間的神經元則彼此相連，且信號的流向是由輸入層向輸出層單向傳播。

1. 輸入層：用以表現網路之輸入變數，其處理單元數目依問題而定。

2. 隱藏層：用以表現網路之輸入處理單元間的交互影響，其處理單元數目並無標準方式可以決定，需經常以試驗方式決定其最佳數目。

3. 輸出層：用以表現網路之輸出變數，其處理單元數目依問題而定。

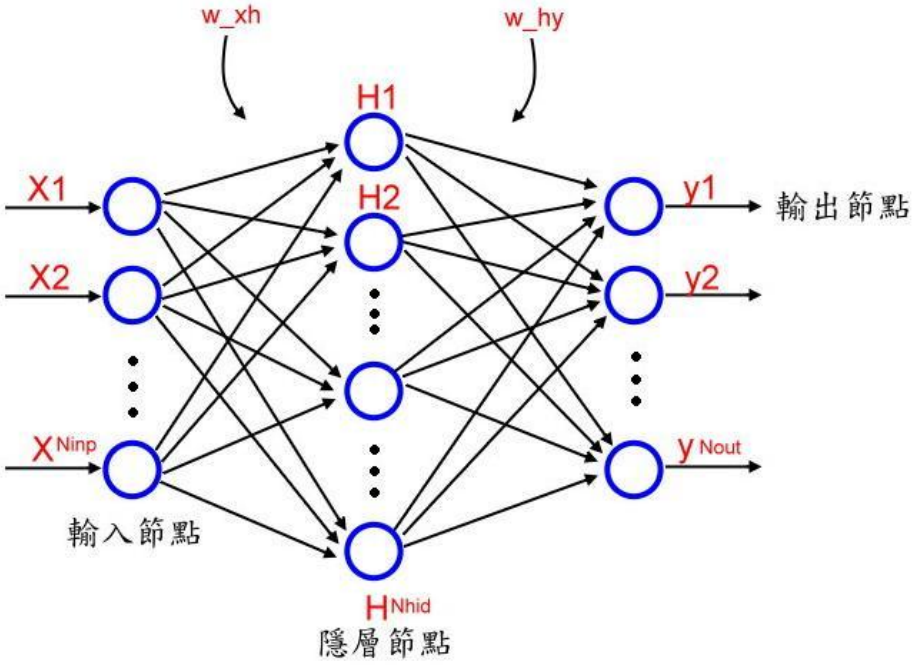

圖 2-2 倒傳遞神經網路

倒傳遞網路中的神經元，其最常用的非線性轉換函數為雙彎曲函數(sigmoide funtion)，如圖 2-3 所示，其中

$$y(t) = f(\sum_{i=1}^{n} w_i \cdot x_i(t) - \theta)$$

w_i：模仿生物神經細胞的神經節加權值

θ：模仿生物神經細胞的細胞核偏權值(biss)，即輸入訊號的加權乘積和必須要大於偏權值後，才能被傳輸至其它人工神經元中

f：模仿生物神經細胞的細胞核非線性轉換函數

$$f(z)=\frac{1}{1+e^{-z}}$$

t：時間

n：人工神經元輸入數目

這種函數有一種特性，即當 z 趨近於正負無窮大時，f(z)趨近於 0 或 1，而 f(z)的值則介於(0,1)之間。

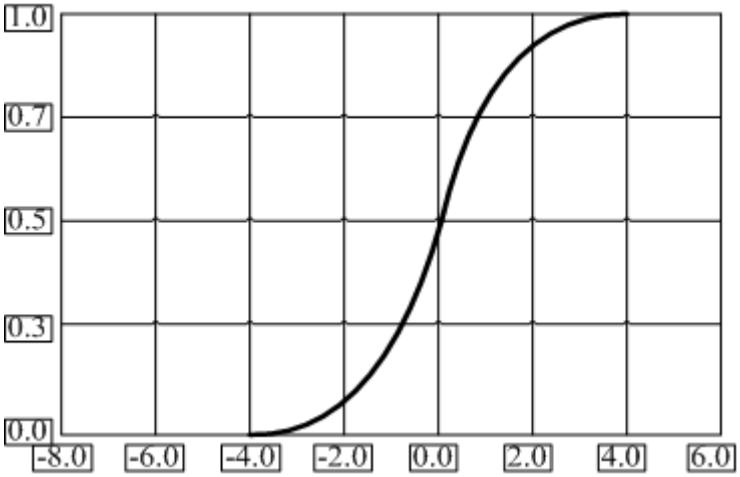

圖 2-3 雙彎曲函數

2.2.3 倒傳遞網路演算法

倒傳遞網路演算法，簡稱為 BP 演算法，分為學習及回想兩部分，將這 BP 演算法的學習步驟及回想步驟詳細說明如下。

學習演算法

BP 學習演算法共分為 8 個步驟，說明如下：

步驟一：決定網路的層數及各層間神經元數目。

說明：為了讓讀者容易明瞭 BP 演算法，在此假設網路的架構為輸入層、一層隱藏層及輸出層，即假設網路是三層網路架構，且假設輸入層的神經元數目有 Ninp 個、隱藏層的神經元數目有 Nhid 個，而輸出層的神經元數目有 Nout 個。

步驟二：以均佈隨機亂數設定網路的初始加權值及初始偏權值。

說明：因為不同層的神經元彼此相連，如果我們令 W_xh[i][h]為輸入層第 i 個神經元與隱藏層第 h 個神經元的加權值，由於有 Ninp 個輸入神經元與 Nhid 的隱藏神經元，所以我們可以用一個雙層迴圈來設定所有輸入層與隱藏層間的初始加權值，方法如下：

$$for\ \ i = 1\ to\ \ Ninp$$

$$for\ \ h = 1\ to\ \ Nhid$$

W_xh[i][h] = 均佈隨機亂數

同理，如果我們令 W_hy[h][y]為隱藏層第 h 個神經元與輸出層第 j 個神經元間的加權值，則我們設定所有隱藏層與輸出層間的初始加權值的方法如下：

for $h = 1$ *to* *Nhid*

for $j = 1$ *to* *N*out

W_hy[h][j]= 均佈隨機亂數

在來我們要設定網路中的初始偏權值，要注意的是只有隱藏層及輸出層才有偏權值，輸入層是沒有的。事實上，輸入層是沒有運算能力的，他只是將一個神經元接收到的訊號平行輸出至隱藏層各個神經元中。若令 θ_h[h]為隱藏層第 h 個神經元的偏權值，θ_y[j]為輸出層第 j 個神經元的偏權值，則設定初始偏權值的方法如下：

for $h = 1$ *to* *N*hid

θ_h[h]= 均佈隨機亂數

for $j = 1$ *to* *N*out

θ_y[j] = 均佈隨機亂數

步驟三：輸入訓練樣本 x[1],x[2],…,x[Ninp]及目標輸出值 T[1],T[2],…,T[Nout] 。

說明：輸入值 x[1],x[2],…,x[Ninp]可為任意的實數值，但是由於倒傳遞網路採用雙彎曲函數(sigmoid function)當做神經元的非線性轉換函數，網路的推論輸出值的值域也必須落在[0,1]之間，所以目標輸出值 T[1],T[2],…,T[Nout]其值域也必須落在[0,1]之間。

步驟四：計算網路的推論輸出值 Y[1],Y[2],…,Y[Nout]。

說明：計算的方法是先算出隱藏層的輸出值，方法如下：

$$for \;\; h = 1 \; to \;\; Nhid$$

$$net\_h[h] = \sum_{i=1}^{Ninp} W\_xh[i][h] \cdot x[i] - \theta\_h[h] \tag{2-1}$$

$$for \;\; h = 1 \; to \;\; Nhid$$

$$H[h] = \frac{1}{1 + e^{-net\_h[h]}} \tag{2-2}$$

其中 net_h[h]為隱藏層第 h 個神經元的加權乘積和，而 H[h]為隱藏層第 h 個神經元的輸出值，他將收集到的加權乘積和 net_h[h]再做一次非線性轉換。

由於輸出層的輸入訊號來自隱藏層的輸出值，所以其推論輸出值可計算如下：

$$for \;\; j = 1 \; to \;\; Nout$$

$$net\_y[j]=\sum_{h=1}^{Nhid} W\_hy[h][j]\cdot H[h]-\theta\_y[j] \tag{2-3}$$

$$for\ \ j=1\ to\ \ Nout$$

$$Y[j]=\frac{1}{1+e^{-net\_y[j]}} \tag{2-4}$$

其中net_y[j]及Y[j]分別是輸出層第j個神經元的加權乘積和及推論輸出值。

步驟五：計算輸出層與隱藏層的差距量。

說明：計算輸出層差距量的公式如下：

$$for\ \ j=1\ to\ \ Nout$$

$$\delta\_y[j]=Y[j]\cdot(1-Y[j])\cdot(T[j]-Y[j]) \tag{2-5}$$

其中δ_y[j]是輸出層第j個神經元的差距量。在公式(2-5)(T[j]-Y[j])表示目標輸出值與網路推論輸出直間的誤差，所以δ_y[j]表示T[j]與Y[j]之間的誤差量度。

而計算隱藏層差距量的公式如下：

$$for\ \ h=1\ to\ \ Nhid$$

$$\delta\_h[h]=H[h]\cdot(1-H[h])\cdot\sum_{j=1}^{Nout} W\_hy[h][j]\cdot\delta\_y[j] \tag{2-6}$$

其中 δ_h[h]表示隱藏層第 h 個神經元的差距量。請注意，在(2-6)式中包含了一項子式：

$$\sum_{j=1}^{Nout} W\_hy[h][j]\cdot\delta\_y[j]$$

此式表示輸出層差距量的加權乘積和。所以 δ_h[h]的計算與輸出層的差距量有關，這意味我們將輸出層的誤差倒傳至隱藏層來計算其差距量，這是此網路之所以有“倒傳遞”名稱的由來。

步驟六：計算各層間的加權值修正量及偏權值修正量。

說明：若令Δ W_hy[h][j]表示隱藏層第 h 個神經元與輸出層第 j 個神經元間的加權值修正量，且令Δ θ_y[j]表示輸出第 j 個神經元的偏權值修正量，則計算其間所有加權值及偏權值修正量的方法如下：

$$\begin{aligned}
&for\ \ h=1\ to\ \ Nhid\\
&\quad for\ \ j=1\ to\ \ Nout\\
&\qquad \Delta W_hy[h][j]=\eta\cdot\delta_y[j]\cdot H[h] \qquad (2\text{-}7)\\
&\qquad \Delta\theta_y[j]=-\eta\cdot\delta_y[j] \qquad (2\text{-}8)
\end{aligned}$$

其中 η 為學習速率，一般取值為 0.1~1.0。寫時為了加速網路的收斂速度，可將公式(2-7)與(2-8)改寫成

$$\Delta W\_hy[h][j]=\eta\cdot\delta\_y[j]\cdot H[h]+\alpha\cdot\Delta W\_hy[h][j] \quad (2\text{-}9)$$

$$\Delta\theta\_y[j]=-\eta\cdot\delta\_y[j]+\alpha\cdot\Delta\theta\_y[j] \quad (2\text{-}10)$$

其中 α 為慣性因子，一般取值為 0.0~0.9。

同理，若令Δ W_xh[i][h]為輸入層第 i 個神經元與隱藏層第 h 個神經元間的加權值修正量，且令Δ θ_h[h]為隱藏層第 h 個神經元的偏權值修正量，則計算其間所有加權值及偏權值修正量的公式如下：

$$\begin{aligned}&for\ \ i=1\ to\ \ Ninp\\&\quad for\ \ h=1\ to\ \ Nhid\\&\quad \Delta W\_xh[i][h]=\eta\cdot\delta\_h[h]\cdot x[i]+\alpha\cdot\Delta W\_xh[i][h] \quad (2\text{-}11)\\&for\ \ h=1\ to\ \ Nhid\\&\quad \Delta\theta\_h[h]=-\eta\cdot\delta\_h[h]+\alpha\cdot\Delta\theta\_h[h] \quad (2\text{-}12)\end{aligned}$$

步驟七：更新各層間的加權值及偏權值。

說明：更新隱藏層與輸出層間的加權值及輸出層偏權值的方法如下：

$$\begin{aligned}&for\ \ h=1\ to\ \ Nhid\\&\quad for\ \ j=1\ to\ \ Nout\\&\quad W\_hy[h][j]=W\_hy[h][j]+\Delta W\_hy[h][j] \quad (2\text{-}14)\end{aligned}$$

$$for \;\; j = 1 \; to \; Nout$$

$$\theta\_y[j] = \theta\_y[j] + \Delta\theta\_y[j] \qquad (2\text{-}15)$$

同理，更新輸入層與隱藏層間的加權值及隱藏層偏權值的方法如下：

$$for \;\; i = 1 \; to \; Ninp$$

$$for \;\; h = 1 \; to \; Nhid$$

$$W\_xh[i][h] = W\_xh[i][h] + \Delta W\_xh[i][h] \qquad (2\text{-}16)$$

$$for \;\; h = 1 \; to \; Nhid$$

$$\theta\_h[h] = \theta\_h[h] + \Delta\theta\_h[h] \qquad (2\text{-}17)$$

步驟八：重複步驟 3 至步驟 7，直到網路收斂。

說明：學習過程通常以一次一個訓練樣本進行，直到網路學習完所有的訓練樣本，稱為一個學習循環(learning circle)，我們可以讓網路重複學習個學習循環，直到網路收斂為止。為了測試網路是否收斂，我們定義下列誤差函數來表示網路的學習品質：

$$E = (\frac{1}{2})\sum_{j}(T[j] - Y[j])^2 \qquad (2\text{-}18)$$

此式表示輸出層各個神經元的平方誤差和。因為在學習過程中，我們希望網路的推論輸出值 Y[j]與目標輸出值 T[j]越接近越好，所以(2-18)式的計算值應小於一個合理的範圍才行。

回想演算法

BP 回想演算法，其步驟說明如下：

步驟一：設定網路的層數及各層間神經元的數目。

步驟二：讀入以訓練好的網路加權值及偏權值。

步驟三：輸入一個測試範圍 x[1]，x[2]，，x[Ninp]。

步驟四：計算網路的推論值出值 Y[1]，Y[2]，，Y[Nout]。

說明：此步驟與學習演算法的步驟四相同。

第三章　研究結果與討論

為了熟悉 Kinect 的操作性，本研究先測試空間軌跡功能。以手指滑動即可建立移動軌跡(如圖 3-1)，可利用於人員移動軌跡建立。

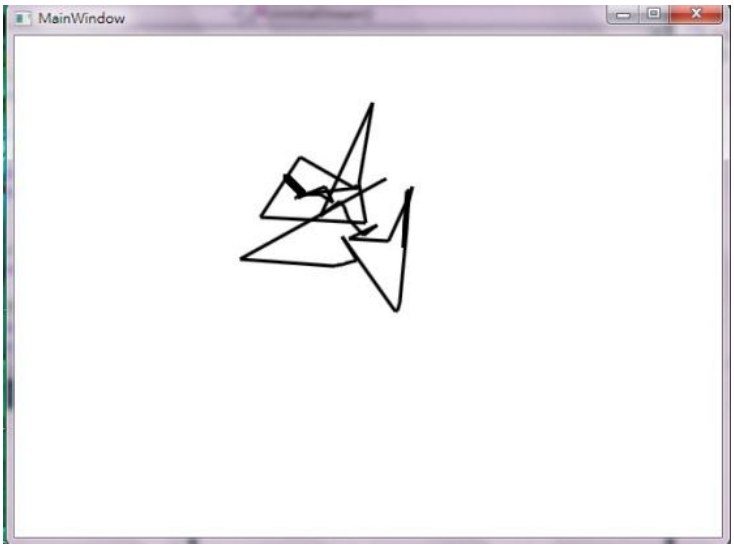

圖 3-1：用雙手手指畫圖功能

Kinect 感測器可測量人體與感測器之距離，當玩家進入 Kinect 感應器深度感應範圍(0.8~4 公尺)顯示影像中的玩家會以不同顏色出現，如果玩家距離感應器小於 1.5 公尺(目前設定的門檻)，進入 1.5 公尺範圍的部分會以紅色表示；每次進出感應器深度感應範圍，也會被賦予不同顏色(如圖 3-2)。

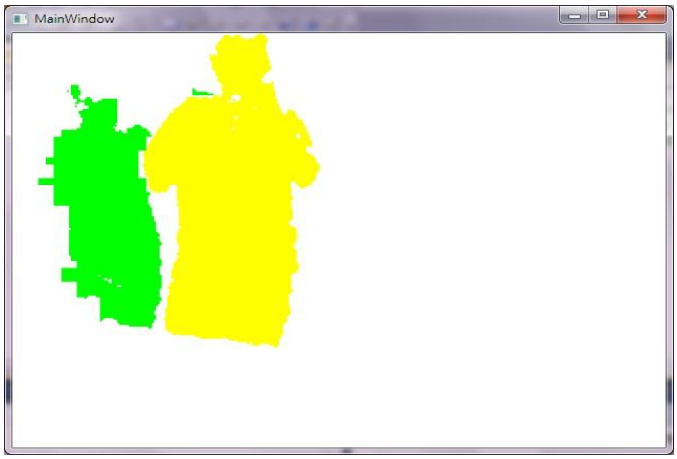

圖 3-2：距離測試顯示圖

Kinect SDK 開放人體的 20 個特徵點(如圖 3-3)，這些節點都是人體的重要關節，將各姿勢節點位置輸入類神經網路學習，可達到姿勢判別之功能。本研究選取 6 種骨架特徵圖做為學習範例，包括舉右手之骨架(如圖 3-4)、籃球防守卡位之骨架(如圖 3-5)、拉弓動作之骨架(如圖 3-6)、拉筋動作之骨架(如圖 3-7)及舉起雙手之骨架(如圖 3-8)。經學習完成後之神經元參數，都能藉由回想過程正確判斷姿勢。

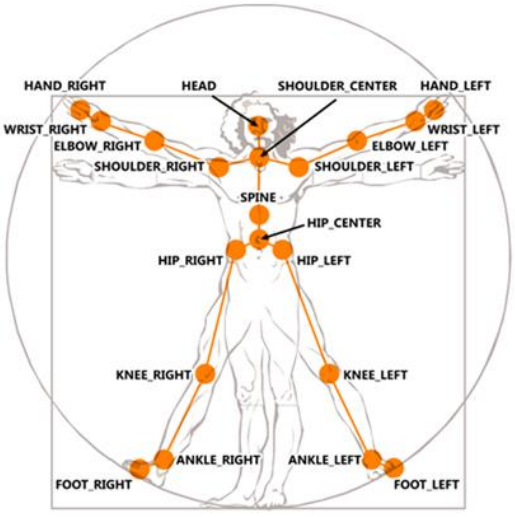

圖 3-3：Kinect 特徵點標示圖[1]

圖 3-4：舉右手之骨架圖形

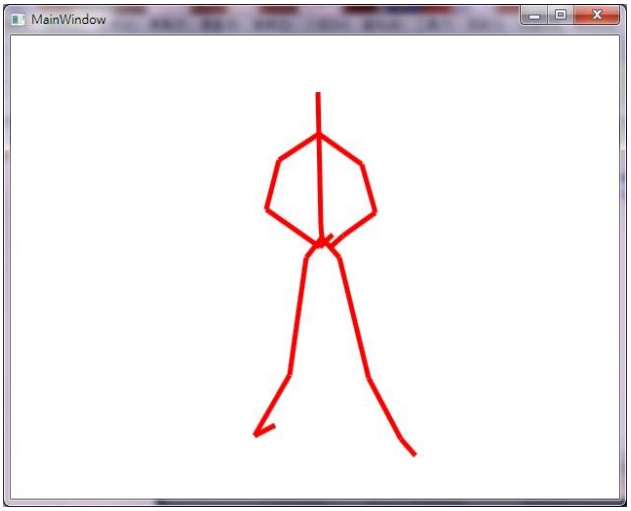

圖 3-5：籃球防守卡位之骨架圖形

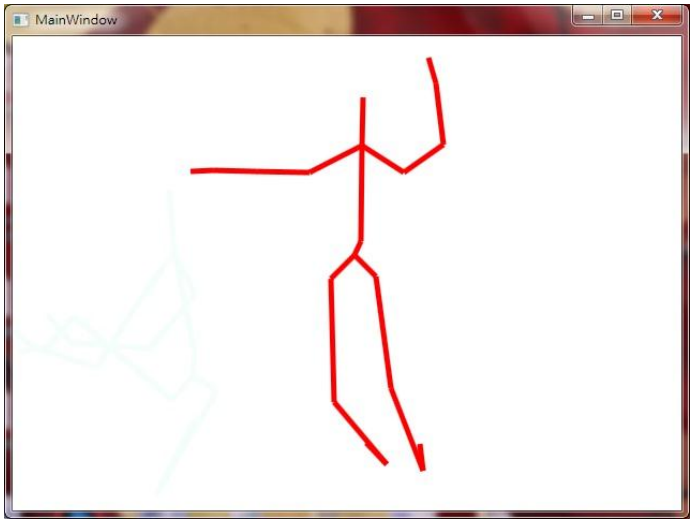

圖 3-6：拉弓動作之骨架圖形

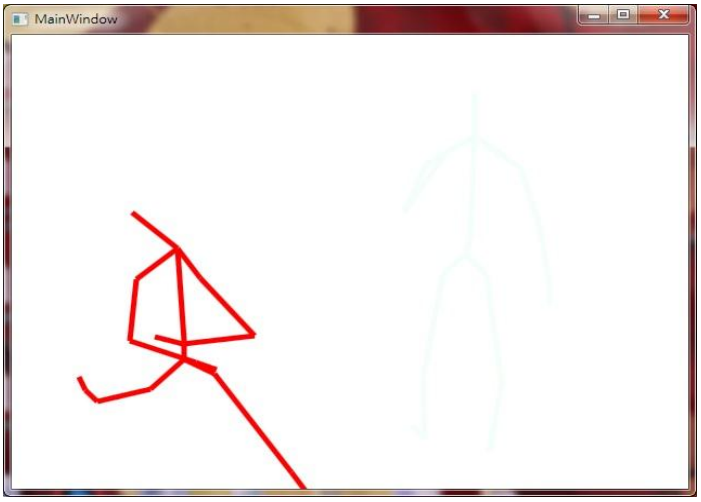

圖 3-7：拉筋動作之骨架圖形

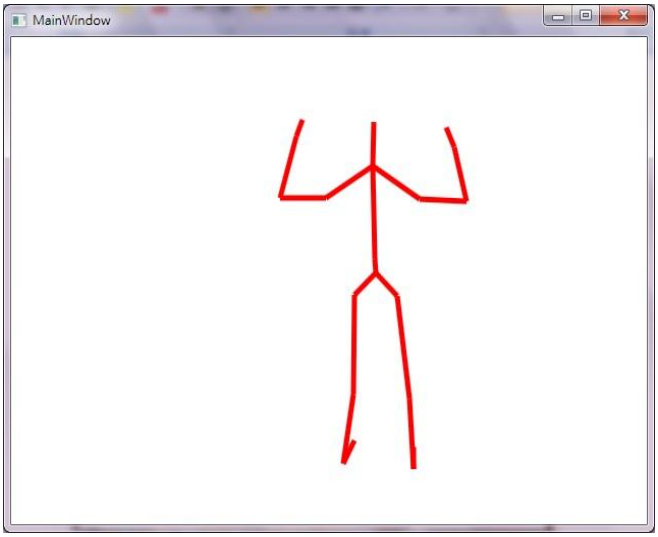

圖 3-8：舉起雙手之骨架圖形

第四章　結論

本計畫已開發完成 Kinect 人體骨架追蹤軟體，利用 Kinect 人體特徵點技術結合類神經網路建立人體骨架模式，提高姿態判讀之準確性，其結果可作為物理復健或健康照護的輔助工具。未來可將判讀技術應用於機器人引導，以提高機器人的機動性。

參考文獻

[1] 微軟 Kinect 官方網站，http://www.xbox.com/en-US/KINECT

[2] Li T.-H.S., Shih-Jie and Wei Tong, ”Fuzzy Target Tracking Control of Autonomous Mobile Robote by Using Intrared Sensor”, IEEE Transaction on Fuzzy System, Vol.12, No.4, 2004.

[3] 何治鋒，深度感測氣應用於追蹤機器人之研究，國立勤益科技大學機械工程研究所，碩士論文，中華民國 101 年。

[4] 周敬敏，基於軌跡辨識技術之人體姿勢自定與分辨研究，國立台灣師範大學資訊工程研究所，碩士論文，中華民國 101 年。

[5] 周建佑，基於 L-K 演算法及 Kinect 的動態目標追蹤系統之研究，國立中央大學電機工程研究所，碩士論文，中華民國 101 年。

[6] 劉超群，Kinect 體感程式探索-使用 C#，松崗圖書。

[7] 葉怡成，類神經網路模式應用與實作，儒林，2009