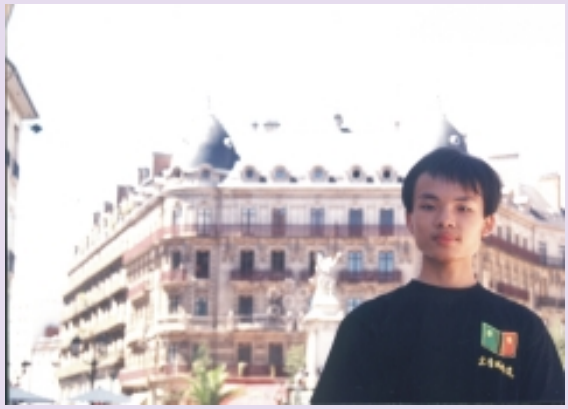
ARGUS — A Real-time Gaze-tracking Utilizing System 即時視焦自動追蹤系統

作者：鄭劭宇
就讀學校：建國中學
指導老師：林祁堂



參加科展，除了要冒著沒有大學可念的生命危險之外，還要特別小心有一天會眾叛親離，更可怕的是社會以及教育政策方面的歧視。儘管如此，這兩年作了兩次國際科展，我深信在這之中比別人學到了更多，而且不是書本上所能得來的，這即是我一直認為科展活動是應該不斷推行的原因。

這次科展的成功，必須感謝的有一直大力支持的林祁堂老師、台大電機的詹國禎教授、台大資工的歐陽明教授、台大資工的陳鼎勻學長、好友兼老師 William Morgan 的幾次無價討論、好友 Horris Tse 在我和程式碼奮戰時陪我度過無數個夜晚，最後要謝謝許多國內電腦大廠提供硬體贊助，讓我能夠順利赴法國參展。

研究動機

人類眼球肌肉極為精密，可以作極細微的運動，使兩眼注視同一目標物。仔細觀察雙眼的動態，可從中獲知許多有用的資訊。

隨著人類每日平均使用電腦的時數日益增長，手部關節在長時間使用下易於疲勞，因操作鍵盤與滑鼠過久併發各種症狀的案例比比皆是。人們在操作滑鼠時，雙眼必會注視著游標。電腦是否能夠藉著捕捉視焦，而計算出相對應的位置，自動將游標移動過去，減少因操作過度所引起的傷害？

現有的類似產品大多為頭戴形的裝置，造價昂貴且對使用者造成很大的不便及侵犯性；傳統使用紅外線的測距裝置亦對雙眼有極嚴重的傷害。是否可以用廉價的視訊硬體來達到非侵犯式且有效的視焦追蹤目的？

本研究擬以市面上一般視訊會議使用之 CCD 攝影鏡頭，配合類神經網路及相關數學方法，設計一「非侵犯式即時視焦自動追蹤系統」，並發展其理論及在各方面的應用。

研究目的

（一）設計並製作視焦追蹤硬體架構。

（二）發展並改進視焦追蹤系統所需數學模型、類神經網路方面的理論及設計相關的新理論，觀察之間的成效。

（三）發展一種以雙影像輸入的方式來測出物體在空間中的座標位置。

（四）將視焦追蹤系統應用在實際操作上。

研究設備器材

（一） Intel x86-compatible Processors

（二）Video Camera

（三）120 R 7.5 度之步進馬達

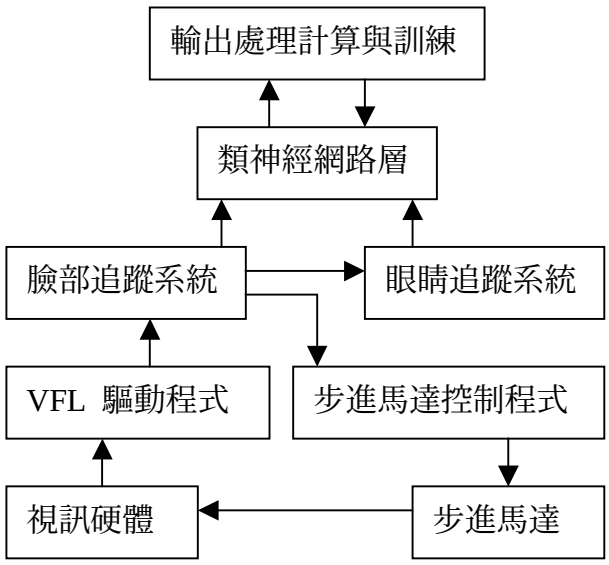
（四）PCI frame grabber

（五）Linux Kernel 2.2.19 with Video4Linux Interface and XFree86 4.0.3

（六）GNU C/C++ Compiler 2.95.3 and NASM 0.98

研究過程與方式

（一）系統架構圖



（二）臉部影像分析

在找出眼睛之前，必須先找出整個使用者的頭部位置，一方面可以縮小眼睛尋找範圍，另一方面則可以獲得更多關於頭部傾斜、偏移角度的相關數據。

1. Color Heuristic Model

臉部由於特色因人而異，很難以傳統的物體辨認方法作為尋找的方式。我們又要求在保持一定的精確度下，計算所需時間愈少愈好，以保持即時追蹤的功能。以特徵辨認的方式運算雖然可以做到很精確，但相對地執行時間也長，用在即時追蹤上面並不合適。

我們發現可以利用人類臉部顏色的分佈來快速將臉部定位。當影像擷取裝置擷取到一張影像時，該影像為 320x240 個像素的集合。每一個像素都分別以光的三原色值 RGB 來表示。若直接以 RGB 來將影像加以分析，會因為每個人的膚色、當時的燈光明暗及種

臉部由於特色因人而異，很難以傳統的物體辨認方法作為尋找的方式。我們又要求在保持一定的精確度下，計算所需時間愈少愈好，以保持即時追蹤的功能。以特徵辨認的方式運算雖然可以做到很精確，但相對地執行時間也長，用在即時追蹤上面並不合適。

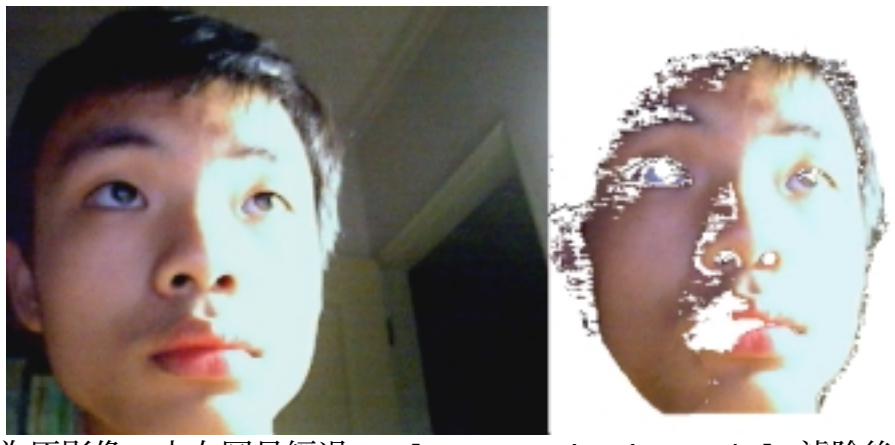
我們發現可以利用人類臉部顏色的分佈來快速將臉部定位。當影像擷取裝置擷取到一張影像時，該影像為 320x240 個像素的集合。每一個像素都分別以光的三原色值 RGB 來表示。若直接以 RGB 來將影像加以分析，會因為每個人的膚色、當時的燈光明暗及種種因素而有不同的 RGB 值，故直接使用 RGB 值不適合。若能夠計算出臉部的色系分佈情況而不看明暗，則可以輕易在這個 searching space 中找出臉部的範圍。因此我們採用以下的式子將三維的 RGB 空間與二維的空間作對映：

$$r = R / (R + G + B) \quad (1)$$

$$g = G / (R + G + B) \quad (2)$$

$$b = B / (R + G + B) \quad (3)$$

因 $r + g + b = 1$，所以實際使用上可將 b 忽略。在此正規化之後，計算出的 rgb 值將不帶有亮度資訊，而只有顏色的種類。故使用此方法可以將許多燈光的問題解決。



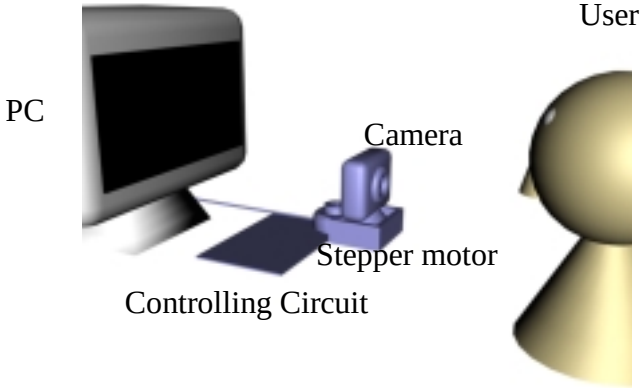
上左圖為原影像，上右圖是經過 Color Heuristic Model 濾除後所得的圖形。可以發現程式只抽取有皮膚色系的部分，而不受亮度及周圍影像的影響。

我們分析各種臉部，並觀察其中 rgb 分佈的特性。

2.動作偵測與追蹤

(1)步進馬達自動追蹤控制的設計

人在操作電腦時，頭部經常會有左右的偏移。為了能夠掌握這種偏移及使影像不致於移出拍攝範圍，我們在攝影鏡頭下自行裝設一步進馬達，並撰寫控制程式，使其與先前的臉部搜尋程式配合，控制鏡頭的左右旋轉，期使能夠擷取到最清晰及精準的影像。



(2) 影像差分

使用者在使用電腦時，臉部多少會有移動現象。我們利用這個特性來作影像差分，即可得出臉部的輪廓位置。對兩個影像 $f(x, y)$、$g(x, y)$ 的差分影像 $h(x, y)$，可以下式定義：

h(x, y) = g(x, y) f(x, y)

因對於所有 $f(x, y)$, $g(x, y)$, $0 \leqq f(x, y) \leqq 255$, $0 \leqq g(x, y) \leqq 255$,

故 $-255 \leqq h(x, y) \leqq 255$

可用絕對值將之作 0 ~ 255 的正規化：

$$h(x, y) = | g(x, y) f(x, y) | \quad (4)$$

在影像序列 s 中連續取三張動態影像 $f_{i-1}(x, y)$, $fi(x, y)$, $fi_{+1}(x, y)$, 取 $f_{i-1}(x, y)$ 與 $f_i(x, y)$ 的差分影像 $d_i(x, y)$ 以及 $f_i(x, y)$ 與 $f_{i+1}(x, y)$ 的差分影像 $d_{i+1}(x, y)$。分別對 d_i 與 d_{i+1} 作二值化處理，得 m_i 與 m_{i+1}。再作 $m_i(x, y)$ 與 $m_{i+1}(x, y)$ 的 AND 邏輯運算，得到最終的二值影像 m_{i+1} 即為我們所要的臉部輪廓。此方法快速且簡潔，配合 Color Heuristic Model 可以準確追蹤臉部的動態。

3.眼部追蹤

(1)Position Heuristic Model

由於人類的臉部五官所屬的位置都有一定的比例，因此可以把搜尋範圍由整個臉縮小為上半個臉部。在整個臉上亮度較低的部分有瞳孔、眉毛及鼻孔。在找到具有低亮度的顏色特徵區域後，即可判斷出眼睛的位置。

(2) 座標鎖定方法

使用者在坐定之後，便少再有大幅度的劇烈移動。我們利用這個特性，將第二張影像以後的 Frames 的搜尋範圍縮小，可以減少搜尋的空間與時間。

在影像分析程式將人類臉部的雙眼找出之後，可使用兩種方式來進行視焦計算。其一為運用類神經網路訓練，另一則為直接分析眼部影像而使用數學方法計算得知。

（三）類神經網路架構

我們使用一個 Back-Propagation 網路來處理輸入及相對應的輸出。我們將兩隻眼睛的範圍鎖定在 25*10 像素的維度，以 (1)、(2)、(3) 式來計算得知每個元素浮點數的值，即對於所有的輸入值 I, $0<I<1$。類神經網路的輸出為 40 個單元，前 20 個代表螢幕上眼睛正在注視的 x 座標，後 20 個則代表 y 座標。電腦先採集 400 個輸入範例，接著做數次的訓練，俾使系統正確分辨眼睛所注視的位置。

在網路的輸出方面，我們則設計了下列數種可行的方法，並將其輸出之精確度及執行效率作比較。

1.Direct Representation 的應用

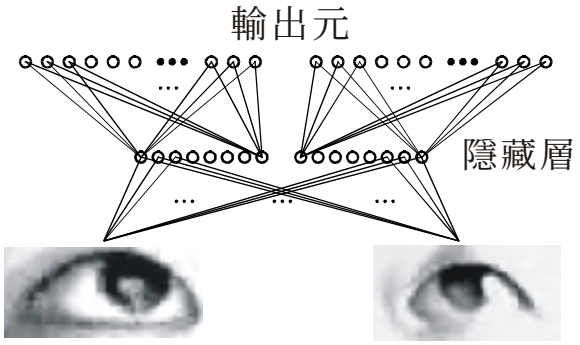
Direct Representation 係將類神經網路的兩個輸出元直接與螢幕的高與寬相乘以計算出座標。在 Back-Propagation Network 訓練時，也使用同樣的方法將訓練資料轉換為介於 0~1 之間的浮點數，以用作訓練時所需的誤差計算。

$X = N_x * S_w$

$Y = N_y * S_h$

其中 X, Y 為輸出座標，N_x 與 N_y 為網路輸出值，S_w 與 S_h 分別為螢幕寬度與高度（像素）。

圖　：類神經網路架構圖

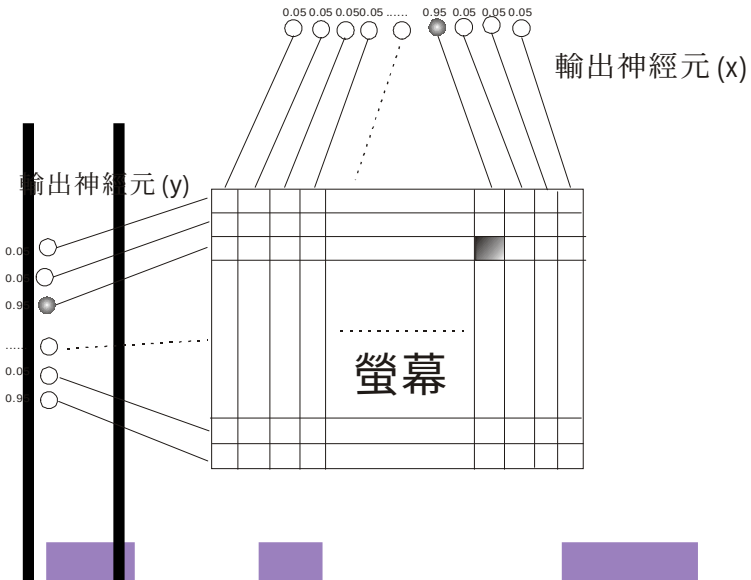


2.Binary Output 與 Gray Code 的應用

我們也想探討當輸出的神經元數量增加時，對於精確度及網路收斂速度是否有影響。為了使多個輸出元能夠充分被利用，我們將輸出元的數量定為 20，分別轉換為二進位形式，亦轉換為較普遍的 Gray 形式，並比較之間的效能差異。

3.Geometric Partitioning 及 Winner Take All Strategy 的應用

我們使用此方法來將螢幕的每一個部分作分割，對應到不同的輸出神經元。我們使用 10、20、30、40 個輸出元來作比較。訓練時，使用者的眼睛目前注視的區域編號，其對應的輸出元的 Activation Value 將被加強，而其餘的則減弱。網路誤差值的計算，則採用 0.95 為最高 Activation Value，0.05 為最低 Activation Value。對於網路實際的輸出，僅取最大的單元作為其對映的螢幕區塊。



4.Gaussian Output Representation 的應用

為增加原始的Geometric Partitioning所能表示的精確度，我們使用Gaussian Distribution來計算出更精確的輸出座標值。在輸出時，先取輸出值中的最大值，然後將座標以 Gaussian function 的值與網路的輸出值之積加以修正。我們將 bell curve 的中心與具有最大的輸出單元對齊，而其餘的輸出值則各自乘上其對應的 Gaussian 值。

Gaussian Distribution 式：

$$N(x)=\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

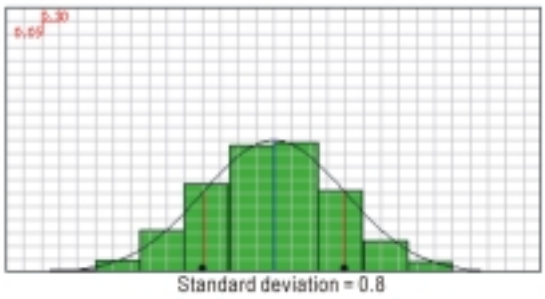
給定一個 $W*H$ 大小的螢幕，且螢幕被分為 P_X*P_Y 個區塊。我們將網路的 X 輸出集合定為 $[A_1\ A_2\ \dots\ A_{P_X}]$，其中 S 為具有最大輸出值的索引。我們將最後的 X 座標以下式求出：

$$X=\frac{1}{A_S}\sum_{i=1}^{P_X}N(i-S)A_i i\frac{W}{P_X}-\frac{W}{2P_X}$$

關於Y座標的求得，則只要將W取代為H，配合Y方向的輸出值，用同樣的方法得出。

因為Gaussian函數值不會改變且左右對稱，我們可用查表的方法，減少上述演算法的計算時間。

訓練網路時，以此式求出訓練的範例，取目前的區域索引為 μ 值，接著計算鄰近區域相對應的 Gaussian Distribution 函數值。如此即可求出對應的 x, y 座標，而不只是對應區塊。



上圖為典型的 Gaussian Distribution 分布圖，其中的標準差為 0.8。對於網路的輸入與輸出值的訓練及處理，採用此分布便可以達到良好效果。

（四）視焦追蹤系統在各方面的應用

我們嘗試將開發出的視焦追蹤系統應用在各種方面，期使能夠達到預期的結果。

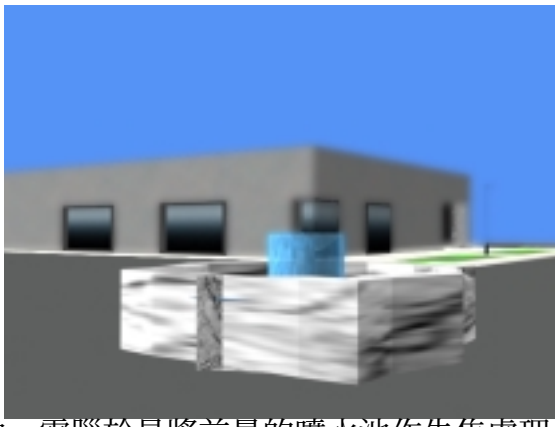
1.目控滑鼠游標

以本系統所得出的結果，可以直接應用在滑鼠游標的自動操控方面，唯需在滑鼠鍵按下的控制方面加以討論。模擬手指按下滑鼠鍵，可以採用連續眨兩下眼睛以及配合學習功能的決定方式。

當使用者注視螢幕上某一物體達一定時間後，系統便檢查該物件是否可被按下，而在系統內部模擬出滑鼠鍵按下的訊號。另外我們也設計學習功能，讓電腦能夠依照當時的使用情況判斷是否應該按下滑鼠鍵，以達到完全模擬滑鼠的功能，大幅減少手部因長期操作的疲勞。

2. 虛擬實境

除了在二維空間的模擬，本系統亦可以再度延伸到三度空間。我們設計了一個專門的遊覽程式，在電腦中呈現一個三度空間的場景，讓使用者直接以眼睛操控遊覽。當使用者觀看遠處物體時，雙眼視線的交點將會往後移，電腦可以藉此自動將場景向前拉，也可以模擬出真實視線裡失焦的景像，使電腦的虛擬實境展現更為逼真，更重要的是使原本二度空間的滑鼠游標無法辦到的事在本系統下經由模擬實現。當使用者注視某物件達一定時間後，系統便自動將 Viewpoint 緩緩移至該物件，如此可以很方便地隨時調整行走的速度及方向。



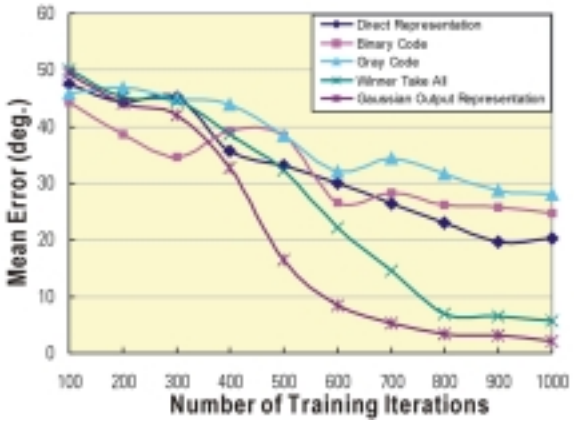
左圖為雙眼注視背景的建築物所得的影像，電腦於是將前景的噴水池作失焦處理。右圖則是雙眼注視噴水池時所得的影像，電腦則將背景的建築物作失焦處理。

結果

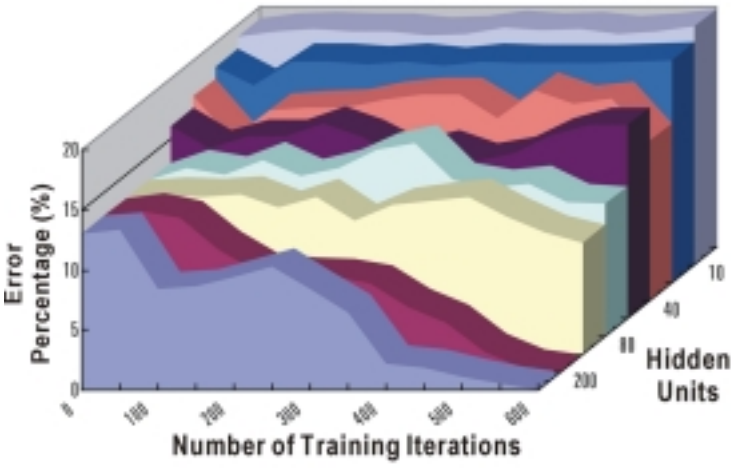
以下列出各種實驗的成果以及分析出來的數據圖表。

（一）類神經網路的訓練成效：

右圖為使用各種不同輸出方法時，類神經網路的訓練成效。平均誤差角度為眼球的轉動角度大小變化量與系統最小感知量的差異值。在訓練次數達 800 次之後，Winner Take All Strategy 和 Gaussian Output Representation 逐漸展現良好的收斂成果。



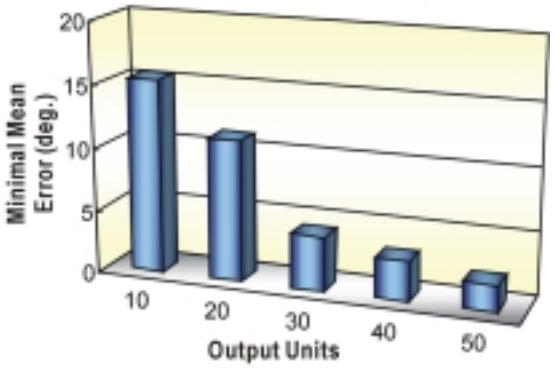
右圖則為改變不同的隱藏神經元數量，對於整個網路訓練成效所造成的影響趨勢。可以發現，在隱藏神經元大於 100 時，網路可以呈現良好的收斂成果。但過大的隱藏元數量則會造成運算速度緩慢，反而使整體效能降低。



（二）各種方法的比較

此為各種方法的效率及速度比較表。每秒影像張數為使用 Video4Linux 介面在 640x480 的影像大小下之每秒接收的影像張數（含運算時間）。

採用的方法	每秒影像張數(FPS)	平均誤差角度（度）
Direct Representation	20.8	21.1
Binary Code	20.6	25.3
Gray Code	20.6	28.2
Winner Take All	20.7	5.5
Gaussian Output Representation	19.5	2.1
Bi-Camera Calculation	24.1	1.9



上圖顯示輸出神經元數量對於 Gaussian Output Representation 的最後輸出精確度的變化關係。

討論及應用

（一）臉部及瞳孔追蹤的討論

由於一般的方法若要到達準確性，需要龐大的運算量，故不適於用在即時追蹤方面。使用 Color Distribution Heuristic Model 在尋找臉部可提供快速且準確的搜尋結果。在背景雜訊較多時，亦可配合 Position Heuristic Model 來確保面部的搜尋正確性。

（二）步進馬達的成效

步進馬達可以確保臉部位置隨時保持在擷取影像的中間，便於影像的追蹤，以適應使用者平時在操作中身體以及頭部的移動。對於類神經網路在訓練時，需要考慮較多的輸入元，但可以在較高的解析度與較大的影像下運作。

（三）Direct Representation

使用 Direct Representation 時，由於 Back-Propagation Network 中的 Sigmoid Function無法將輸出數值很快收斂到精確的值，因此用兩個輸出元在800 次訓練中均無法將正確的結果輸出。此外，以兩個輸出元表示含有10^6個不同元素的集合中的一個元素，網路的 Capacity 不足以做到令人滿意的結果。

（四）Binary & Gray Code Representation

使用 Binary 或 Gray Code 的方式輸出時，需要將 1 轉為 0.95， 0 轉為 0.05，收斂速度才較快，因為 sigmoid function 無法達到 0 或 1。

但Binary Code中的MSB(Most Significant Bit)及LSB (Least Significant Bit)不能夠平衡，如此一來將會造成每一個神經元的輸出於結果所佔的比重不相同，這是造成結果誤差的最主要原因。而Gray Code或具有相似性質的碼也不能改善這樣的缺點。

（表：Binary Code 與 Gray Code 碼表）

Decimal Value	Binary Code	Gray Code	Decimal Value	Binary Code	Gray Code
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

如上表所示，當最左邊的一位由 0 改變為 1 時，將會大幅影響整個輸出。意即代表最左邊一位的神經元不能有些許誤差，否則將影響輸出的準確性。

（五）Geometric Partitioning & Winner Take All Strategy

使用Geometric Partitioning將螢幕分割為數個區塊，每一個區塊以一個輸出元代表，可以使每一個神經元的輸出所佔比重均等，大幅降低了因為比重不同而產生的誤差，克服了Binary Code及Gray Code在這方面的缺點。而Winner Take All Strategy的使用也能夠確保在每一個訓練及輸出範例中網路的輸出與螢幕上的每個區塊都能夠有一對一的相互對應關係，為甚佳的編碼方式。

（六）Gaussian Output Representation

Gaussian Output Representation主要係針對Geometric Partitioning作改進。由於Geometric Partitioning的準確度最多只能到螢幕寬度與分割的區塊數的商，因此使用Gaussian Output Representation能夠使網路輸出的值突破區域的限制，藉由Normal Distribution Formula可以計算出更為精確的值。由實驗結果得知，此種方法配合Geometric Partitioning為作為類神經網路最佳的編碼與表示方式。

（七）視焦追蹤裝置的應用

本研究中，利用視焦追蹤裝置所做的應用皆有相當不錯的成效。

1.目控滑鼠游標

以視焦追蹤裝置來實作目控滑鼠游標所遭遇到最大的問題即是該如何決定滑鼠鍵的按下動作。我們也計畫設計一個針對每一個使用者的使用習慣而有不同適應能力的判斷程式。由於我們的裝置為非侵犯性，故無法做到頭戴式裝置的精確程度，但是仍然能夠應付一般的滑鼠操作。當攝影機的解析度或程式的判斷系統有更加的改進時，電腦必能有更佳的判斷能力，可容許眼睛操控螢幕上更小的物件。

2.虛擬實境

本研究在虛擬實境方面的應用，將來可在汽車的自動駕駛或娛樂方面有所發展。透過眼睛的操作，系統變得更具有親和性，且操作更加方便。若能夠配合已有的立體眼鏡甚至全像技術，也可以做到真正的視覺遠近偵測，使原本平面的二度空間操作轉變為真正的三度空間視覺追蹤，其應用將無可限量。

3.未來的研究展望

目前仍有許多理論可加以實作並改善本系統，如Fuzzy Logic及Bayesian Approach。我們預期加入這些理論，將能改善現有理論在預測未知量以及容忍度上的缺點。而現有的類神經網路架構及參數也還有許多最佳化的空間。

我們計畫將現有的研究方法再加以改進，修正細部的缺陷，使之更加容易適合每個人使用，亦即增加系統對於不同臉部、不同外觀、不同環境的適應能力，並繼續研究在同樣的設備及較佳的設備下增加追蹤精確度的方法；當然，亦希望本系統能有更多且方便的用途。

結論

本研究驗證了使用一般視訊會議用攝影機配合類神經網路及數學方法可以準確追蹤視焦的假設，並實作了此系統，也成功將其應用在目控滑鼠游標、虛擬實境方面，獲得令人滿意的結果。

我們已成功找出方法克服了傳統數字型態在類神經網路上的表達方式所造成的極大誤差，也找出Gaussian Output Representation為目前實驗數據中顯示出整體表現最佳的編碼及解碼方法。

相信配合著更好的視訊器材及理論，將可使本系統的效能提升到更完美的境界，而其廣泛的應用領域也將一一展現。

參考資料

(一) Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall Inc, 1995.

(二) Andr Redert, Emile Hendriks, and Jan Biemond, Correspondence Estimation in Image Pairs ,IEEE Signal Processing, vol.16, no.3, pp. 29-46, May 1999.

(三) Jie Yang and Alex Waibel, A Real-Time Face Tracker, School of Computer Science, Carnegie Mellon University, January 1998.

(四) 楊武智，影像處理與辨認，全華科技圖書股份有限公司，1998 年 4 月。

(五) Neil Matthew and Richard Stones, Beginning Linux Programming, Wrox Press Ltd., 1999.

(六) Valluru B. Rao and Hayagriva V. Rao, C++ Neural Networks and Fuzzy Logic 2nd Edition, MIS: Press, 1995.

ARGUS — A Real-time Gaze-tracking Utilizing System

Abstract

As mouse and keyboard use becomes more frequent and extensive so the potential risk of users developing symptoms such as carpal tunnel syndrome becomes increasingly likely. Therefore, the development of applications that reduce over-usage of these peripherals has become essential. Gaze tracking is one of the commonly used methods; however, most gaze tracking systems are either intrusive to human body or require costly equipment. The purpose of this project was to develop a non-intrusive and efficient gaze tracking system with cost-effective equipment.

We acquired the user's facial images using a video camera through the Video4Linux interface. Some efficient algorithms were designed to automatically locate the position of the eyes. An image differentiating algorithm was used to detect random movements and create a silhouette of the user's face from the image series. A stepper motor bound to the camera was connected to the parallel port, enabling the camera to rotate according to the user's face movement. Then we designed a position and color distribution heuristic model to determine the position of the eyes. We developed a back-propagation neural network to process the image data. The input consisted of the color values of the eyes in binary format and the output represented the geometric partition where the user is currently gazing. Then we wrote a 3D browser to let the user navigate through the scene just with eyes.

Various factors regarding the network sizes and encoding methods were examined. Using Winner Take All and Gaussian Output Representation with 1100 input units, 100 hidden units and 50 output units, the system yielded only 2.1 degree of mean error whilst maintaining 19.5 FPS of grabbing frame rate after 1000 training iterations. The system possesses the potential for widespread applications; as an alternative to using a mouse, as a means of providing more realistic entertainment and as an aid for the disabled.

Introduction

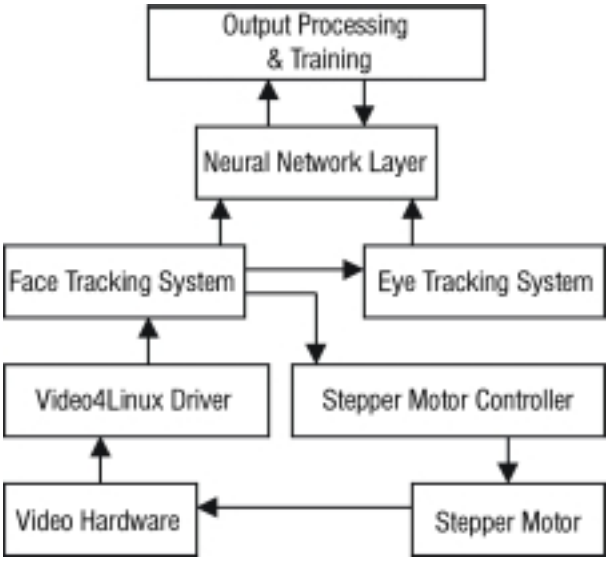
As mouse and keyboard use becomes more frequent and extensive so the potential risk of users developing symptoms such as carpal tunnel syndrome becomes increasingly likely. Therefore, the development of applications that reduce over-usage of these peripherals has become essential. Gaze tracking is one of the commonly used methods; however, most gaze tracking systems are either intrusive to human body or require costly equipment. The purpose of this project was to develop a non-intrusive and efficient gaze tracking system with cost-effective equipment.

Equipment

(1) Intel x86-compatible Processors
(2) Video Camera
(3) Stepper Motor
(4) PCI frame grabber

Methods

(1) System Architecture



(2) Facial Image Analysis

1. Color Heuristic Model

Facial features differ from person to person so it is difficult to locate the face this way. We also hope to keep the tracking process efficient while maintaining accuracy. Although identifying with specific characteristics yields quite precise results, the consumption of calculation time is relatively noticeable so it is not feasible to apply this method to real-time gaze tracking.

Hence, we locate the human face by its color distribution. Every acquired pixel is representable in RGB values. However, it is not practical to determine the face range directly from RGB values because those values may vary due to illumination and different skin colors. Therefore, we eliminate this problem by mapping the 3-dimensional RGB values to 2-dimensional values as follows:

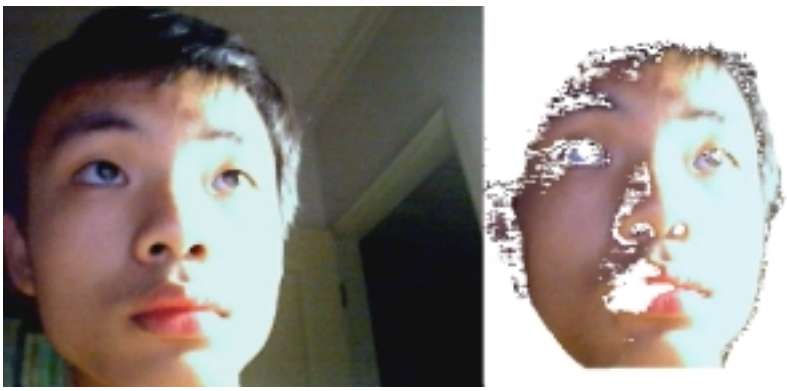
$$r = R / (R + G + B) \quad (1)$$

$$g = G / (R + G + B) \quad (2)$$

$$b = B / (R + G + B) \quad (3)$$

Obviously, b = 1 - r - g. Since b is redundant and can be determined from a pair of r and g values, we only need to consider (r,g) as the new color value.

After this normalization, the new rgb values do not contain brightness information and thus the illumination problem can be resolved.

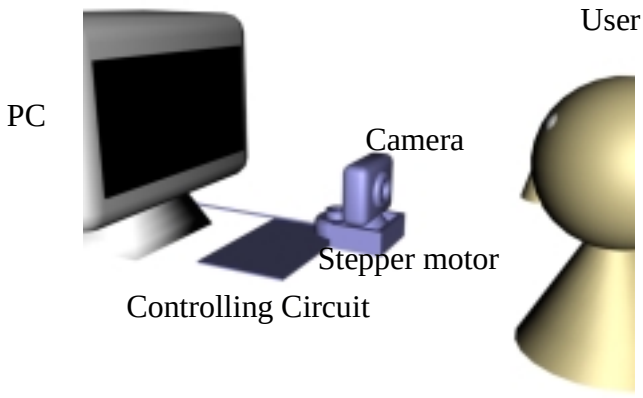


The upper left picture is the original image and the upper right picture is the image filtered with color heuristic model. The Program only extracts areas containing skin color and is not affected by the illumination and the surrounding objects.

2. Motion Detection and Tracking

(1) Stepper Motor

In order to deal with movements of the human face and keep the image in a visible range, we set up a stepper motor and bound it to the video camera. The stepper motor was connected to the parallel port, enabling the camera to rotate according to the user's face movement.



(2) Image Differentiating

The user's face moves frequently when using the computer. Therefore, we can locate the face position by differentiating algorithms. We define h(x,y), the differentiated image of two images f(x,y) and g(x,y), as follows:

$h(x, y) = g(x, y) - f(x, y)$

For every $f(x, y)$ and $g(x, y)$, we have

$0 \leqq f(x, y) \leqq 255$

$0 \leqq g(x, y) \leqq 255,$

$\text{Therefore, } 55 \leqq h(x, y) \leqq 255$

We then define

$h(x, y) = | g(x, y) - f(x, y) |$

We collect three consecutive images from image series s: $f_{i-1}(x, y)$, $f_i(x, y)$ and $f_{i+1}(x, y)$, then we calculate the differentiated images $d_i(x, y)$ of $f_{i-1}(x, y) - f_i(x, y)$, and $d_{i+1}(x, y)$ of $f_i(x, y) - f_{i+1}(x, y)$. We convert d_i and d_{i+1} into binary values, that is, m_i and m_{i+1}. Finally, we calculate the result of $m_i(x, y)$ and $m_{i+1}(x, y)$ with AND logic gate and get the final binary image m_{i+1}. This final result can be recognized as the contour of the moving object. This method is fast enough to track the face movement with color heuristic model.

3. Eye Tracking

(1) Position Heuristic Model

Human eyes are located at a certain portion in the face so we can narrow down the searching space. After confining an area consisting low-brightness pixels with high activation, we can spot the eyes.

(2) Locking Mechanism

The user seldom moves after settled down in front of the computer. Therefore, we can narrow down the searching space after the second frame and reduce the computation time.

(3) Neural Network Structure

We implemented a back-propagation neural network to process the input and output data. The system acquires some input examples first and then starts training the network to recognize where the eyes are gazing. A few encoding methods were examined and are explained in details as follows.

1. Direct Representation

We multiply the two output value of the neural network with the width and height of the screen directly and calculate the coordinates on the screen respectively. When doing back-propagation training, we use the same method to convert the training data into floating values between 0 and 1:

$X = N_x * S_w$

$Y = N_y * S_h$

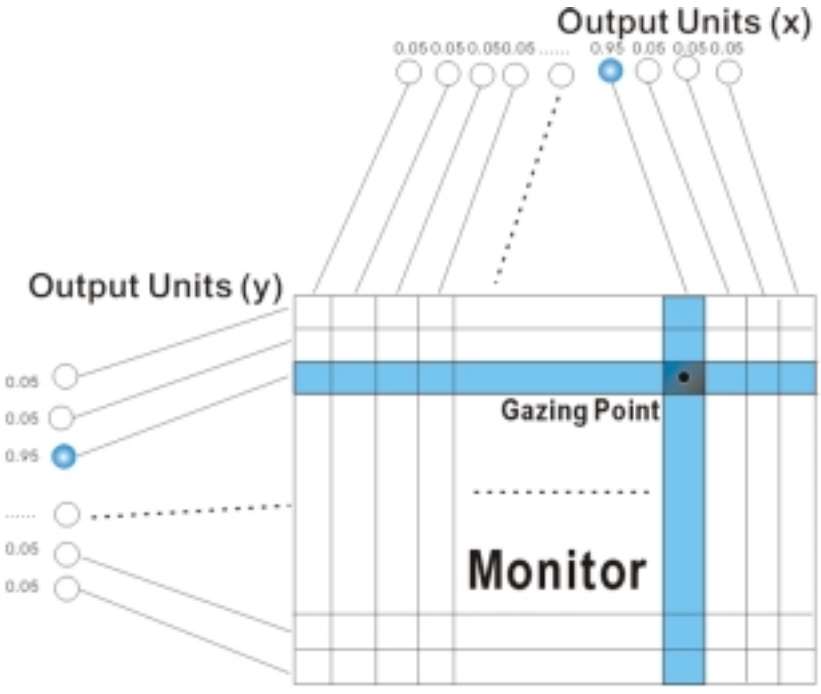
where X and Y are the actual coordinates on the screen, N_x and N_y are output values of the neurons, and S_w and S_h represents the width and height of the screen measured by the pixel.

2. Binary Output and Gray Code

We also examined the learning performance of the neural network when we increased the number of the input and output units. We converted the input values to binary values and Gray code and tested the performance under different input and output units.

3. Geometric Partitioning and Winner Take All Strategy

We separated the screen into a few parts and mapped them to different output units. While the network was being trained, the activation value of the partition that the user is gazing at would be increased. We used 0.95 for the highest activation value and 0.05 as the lowest when producing training data for the network. For the network output, we only selected unit with the highest output value as the actual output partition.



4. Gaussian Output Representation

In order to increase the accuracy of the geometric partitioning algorithm, we employed Gaussian variation, as known as normal distribution, and expect more accurate output values. While processing the output values, we take the unit which has the maximum output value and modify the coordinate by multiplying the value of the Gaussian function at selected intervals by the value of the network's output.

The Gaussian function is shown as follow:

$$N(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

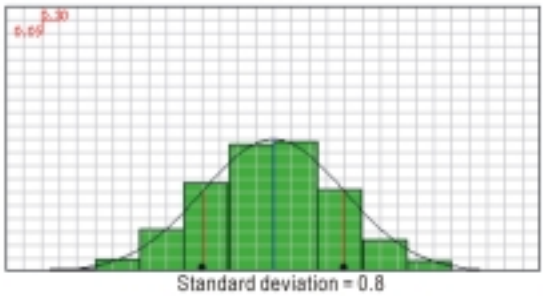
The unit with the most significant activation value was aligned to the center of the bell curve. The other output values are multiplied with their respective Gaussian values.

Given a W*H size of screen, and we divide the screen into P_X*P_Y partitions. Let the network output for the X-axis be $[A_1\ A_2\ \ldots\ A_{P_X}]$ and S be the index of the most significant activation unit. The X coordinate is then calculated with the following formula:

$$X = \frac{1}{A_S} \sum_{i=1}^{P_X} N(i-S) A_i i \frac{W}{P_X} - \frac{W}{2P_X}$$

The Y coordinate can also be calculated by substituting W with H and using Y values with the same formula.

Because the Gaussian function is symmetric, the calculation above can be made fast by lookup tables and reduce the computation time of this algorithm,



The figure above is an ordinary Gaussian distribution with a standard deviation of 0.8. The distribution can yield good results when applied to the input and output of the neural network.

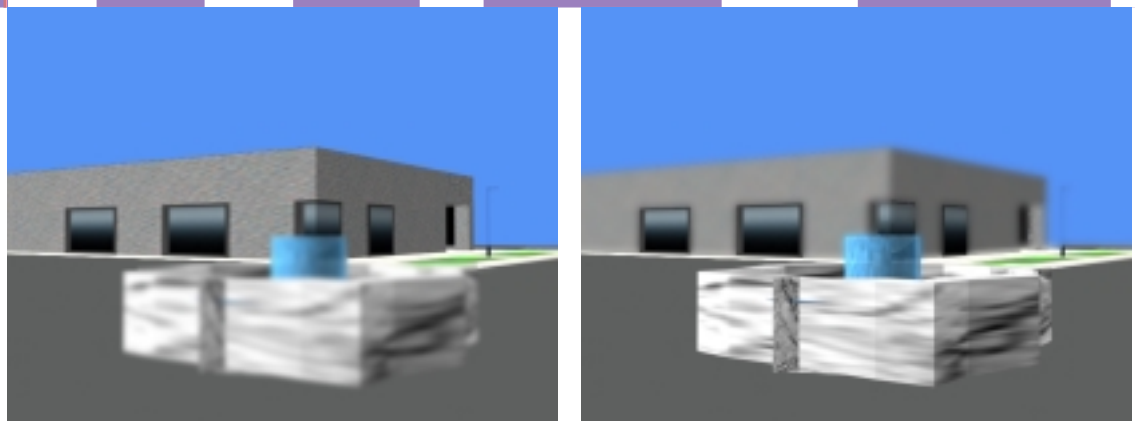
(4) Applications of the Gaze-tracking System

1. Eye-driven Mouse Cursor

We can apply this gaze-tracking system to mouse cursor controlling since the operation is quite intuitive to normal users. However, how to trigger mouse clicks will be another interesting subject for further researching. After the user has gazed at an object on the screen for a certain period of time, the system can then emulate the clicking message internally.

2. Virtual Reality

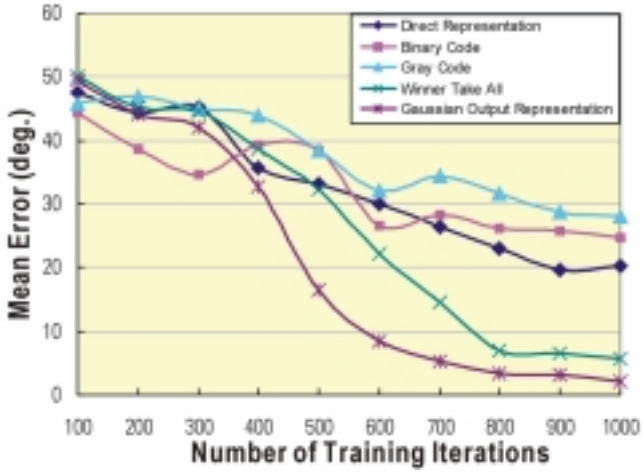
In addition to operations in 2-dimensional planes such as the mouse cursor, our system is also applicable to 3-dimensional spaces. We developed a 3-D browser for the user to navigate in the scene by eyesight. When the user is gazing at a building, the system moves the current viewpoint to that building. Besides, the system can also determine the currently focused position and blur the area that is out of focus at the same time. Navigating through 3-D scenes is no longer only accessible with mice and joysticks and is much more realistic.



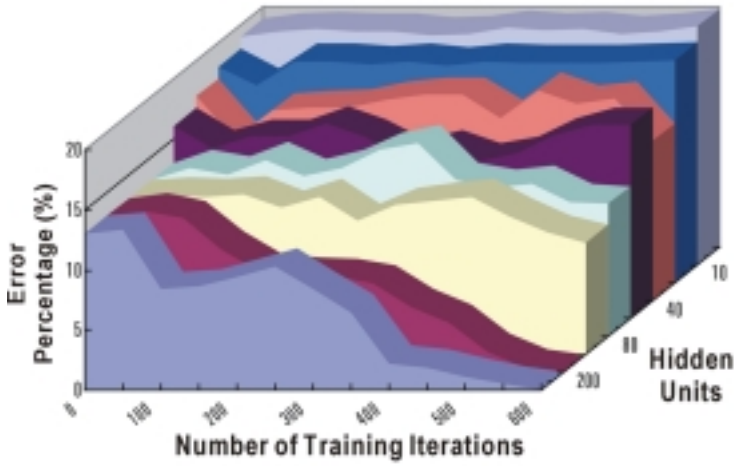
The left picture is the image generated while the user is gazing at the building. To simulate the actual vision, the fountain is blurred. The building in the right picture is blurred because the user is gazing at the fountain.

Results

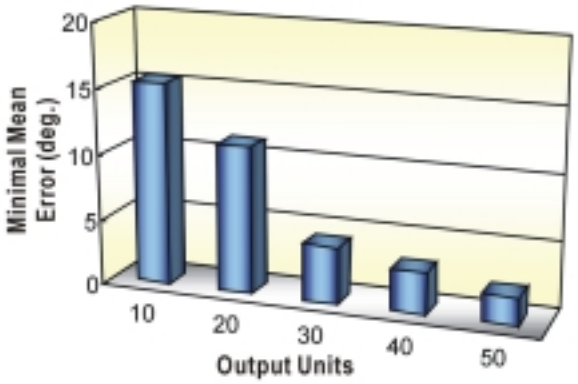
Various experiments were done and the performances of different methods were examined. Due to lacking of output units, the network would not converge if direct representation was used; the accuracy was not satisfactory after 800 training iterations. (As shown below) The MSB and LSB problem in binary code and Gray code could result in unpredictable output and influence the whole outcome.



Separating the screen into a few partitions, geometric partitioning could balance the importance of each unit and resolve the MSB and LSB problem caused by the use of binary code and Gray code. The number of hidden units of the neural network was one of the most significant factors that affected the training performance in our research. (As shown below)



Gaussian output representation has increased the accuracy of geometric partitioning and overcome the partition limitation while representing output values. (As shown below)



The table below summarizes the overall performance of each algorithm.

Method Applied	Frames per second	Mean errors (deg.)
Direct Representation	20.8	21.1
Binary Code	20.6	25.3
Gray Code	20.6	28.2
Winner Take All	20.7	5.5
Gaussian Output Representation	19.5	2.1
Bi-Camera Calculation	24.1	1.9

(Tested under the Video4Linux interface. The image dimension is 640x480 pixels and 16-bit colors)

Discussions and Applications

(1) Face and Gaze Tracking

To attain high accuracy, a huge amount of computations are required using traditional methods. Thus, they are not suitable for application in tracking. However, by using color distribution heuristic model, the face can be located rapidly and accurately. When the background noise becomes a concern, position heuristic model can be introduced to maintain the accuracy in locating the face.

(2) The Advantages of the Stepper Motor

The stepper motor ensures the face to be always kept in the middle of the grabbed image. This can not only benefit the tracking of the images but also the movements of the human body and face. However, it takes more time to train the neural network when the stepper motor is used.

(3) Direct Representation

In Direct Representation, due to the Sigmoid Function's disability in quickly converging the output to an accurate value in Back-Propagation Network, it is unable to generate acceptable results in the 800-iteration training process using 2 output neurons. Besides, the network capacity would be unable to yield satisfactory results by representing an element in a set of 10^6 elements.

(4) Binary & Gray Code Representation

Since the sigmoid function can never attain zero or one, it is necessary to convert 0 to 0.05 and 1 to 0.95 to speed up the convergence when we use Binary or Gray Code for output representation. However, the MSB (Most Significant Bit) and the LSB (Least Significant Bit) will not balance, resulting in a biased weighing in the neuron output, which is the main reason for the error. Unfortunately, the problem cannot be solved using either Gray Code or other codes with similar characteristics.

(Table: Binary Code and Gray Code)

Decimal Value	Binary Code	Gray Code	Decimal Value	Binary Code	Gray Code
0	0000	0000	8	1000	1100
1	0001	0001	9	1001	1101
2	0010	0011	10	1010	1111
3	0011	0010	11	1011	1110
4	0100	0110	12	1100	1010
5	0101	0111	13	1101	1011
6	0110	0101	14	1110	1001
7	0111	0100	15	1111	1000

As shown above, the output will be greatly altered when the value of the leftmost neuron changes from 0 to 1. This implies that even little error in the leftmost neuron is unbearable because the whole output would be seriously affected.

(5) Geometric Partitioning & Winner Take All Strategy

Using Geometric Partitioning, the screen is divided into several partitions, each represented by an output neuron. This equalizes the weight of every neural output, Agreatly reducing the error due to unequal weighing and conquering what Binary Code and Gray Code are deficient in this. On the other hand, the implementation of "Winner Take All Strategy" guarantees the one-to-one relationship between each output and partition on the screen, in every training and output demonstration. All this shows that it is distinguished coding method.

(6) Gaussian Output Representation

Gaussian Output Representation is an advancement targeted at Geometric Partitioning. As Geometric Partitioning is only accurate to the quotient of screen width and the number of partitions, by using Gaussian Output Representation, the network output can transcend the partition limitation with the aid of Normal Distribution Formula, to calculate a more accurate value. The research reveals that acting in concert with Geometric Partitioning, this is the best coding and representation method for neural networks.

(7) Applications of the Gaze-tracking System

In this research project, the eye-focus tracing system has quite satisfactory applications.

1. Eye-driven Mouse Cursor

The greatest problem encountered in implementing the eye-driven mouse cursor is how to determine whether to emulate the clicking action. In this light, we plan to design an adaptable program to determine actions according to users' habits. As a non-intrusive device, our system cannot attain the accuracy of head-wearing devices, but is still capable of general mouse operations. When the cameras are of better resolution or when the determination program is advanced, the computer will be able to have better accuracy, enabling the eyes to control more delicate elements.

2. Virtual Reality

The application of this project in virtual reality has good prospectus in car-driving automation and entertainment enterprise. Controlling through the eyes, the system becomes more user-friendly and convenient to use. Incorporating 3D spectacles or even holographic technologies, real eye-focus detection can be achieved, transforming the 2D operating environment into a real 3D visualization, breaking the application boundaries.

3. The Future

At present, there are still lots of theories that can be implemented to improve this system, for instance, Fuzzy Logic and Bayesian Approach. By applying such theories, the disadvantages in indeterminate value anticipation and tolerance using the current approach will be greatly improved. In addition, there are possibilities for further optimization for the structure and parameters of the neural networks.

We are now planning to make advancements based on the current researching method and to patch the defects, to cater for everyone's need, i.e. to enhance the adaptability of the system to different facial characteristics, appearances and environments. We will also continue to find out how to increase tracking accuracy using both current and better equipment. Last but not least, we hope this system would have a wider application.

Conclusion

Our studies has proved that gaze-tracking is achievable using a normal video camera and neural network by implementing the system and has also successfully applied it to eye-driven cursor and virtual reality applications.

We have found the alternative of encoding decimal values with binary values and reduced the errors of this representation method, and also found out that Gaussian output representation is the optimal encoding/decoding method according to the results we have so far.

We believe that our system can be improved with better video equipment and theories, and we can discover its vast application domain.

References

1. Stuart Russell and Peter Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall Inc, 1995.
2. André Redert, Emile Hendriks, and Jan Biemond, "Correspondence Estimation in Image Pairs", IEEE Signal Processing, vol.16, no.3, pp. 29-46, May 1999.
3. Jie Yang and Alex Waibel, A Real-Time Face Tracker, School of Computer Science, Carnegie Mellon University, January 1998.
4. Wu-Ji Yang, Image Processing and Recognition, Chuan Hwa Technology Books Inc., April 1998.5. Neil Matthew and Richard Stones, Beginning Linux Programming, Wrox Press Ltd., 1999.
6. Valluru B. Rao and Hayagriva V. Rao, C++ Neural Networks and Fuzzy Logic 2nd Edition, MIS: Press, 1995.