銘傳大學

電腦與通訊工程學系

專題研究總審報告

具人臉追蹤功能之網路監控系統

指導教授：蔡建戊 老師

專研組員：王翊航 鄭俊傑 陳佑杰 張雅淇 吳雋湧

中　華　民　國　九　十　六　年　十　月

銘　傳　大　學

電　腦　與　通　訊　工　程　學　系

專　案　實　作　審　定　書

本校 九六 學年度 電腦與通訊工程學系 四 年 乙 班

組員：＿＿＿王翊航＿＿＿、＿＿＿鄭俊傑＿＿＿

＿＿＿張雅淇＿＿＿、＿＿＿吳雋湧＿＿＿

＿＿＿陳佑杰＿＿＿

所提專案實作：＿＿＿＿具人臉追蹤功能之網路監控系統＿＿＿＿

合於及格水準，業經評審認可。

指 導 教 授 ：＿＿＿＿＿＿＿＿＿＿＿＿＿＿

電通系系主任：＿＿＿＿＿＿＿＿＿＿＿＿＿＿

中 華 民 國　　年　　月　　日

摘要

本專題研究我們提出了具有人臉追蹤的遠端監控系統，透過網路傳輸，將影像傳送至客戶端顯示。此專題研究我們結合了 JPEG 壓縮、人臉追蹤、網路傳輸及攝影機控制四個部份。目標流程架構為在遠端架設攝影機，當接收到影像後，做臉部的追蹤及影像的壓縮，並透過網路傳輸將人臉的座標位置(X,Y)及壓縮後的影像資訊傳遞給客戶端，接著在此端做解壓縮的動作，進而顯示在客戶端所建立的介面上。

Abstract

In this project we proposed has face to trace surveillance system, Used transmit by Internet, delivery the image from client terminal to server terminal, and server terminal can show it. In this project, we adopt the JPEG compression, face to trace, transmit by Internet and camera controlled. Our technological processes was set up the camera in server terminal, when received the image, use face to trace and image compression, to take the human face axis position(X,Y) and convey compression image’s information to server terminal, then in this side, we used decompression, and display on interface that builded by client terminal.

目錄

圖表目錄

第一章 研究動機與背景

現今網際網路的發達，會用網路的人也愈來愈多，由於網路的盛行使得人與人之間的距離可說是從有到無，也使得人們生活的節奏變得更快，更逐漸改變人們的生活習慣，而為了增加人與人之間的溝通的真實性，聲音、影像也可以透過網路來傳輸，因此視訊影音傳播的應用軟體也應運而生，像是在視訊遠距教學方面、家庭監控視訊、網路視訊會議等等，即使距離遙遠也可以通過網路視訊將現在的畫面傳遞給對方;遠距教學現在有很多學校已經開始使用此技術，老師可以利用視訊與聲音訊息教導給各地的學生，在視訊會議方面，許多企業公司開始沿用此方法，既方便又節省時間，在家庭監控視訊方面，人雖然不在家但是可以透過網際網路連結到家中裝設的視訊觀察家中狀況，可以關心年老的人又可以防範小偷。

在這些應用當中，所產生的資料量是相當的龐大，因此影音壓縮技術在現在與未來扮演著相當重要的角色。雖然網路頻寬增加，但隨著使用人數的增加與對視訊品質的要求，勢必仍會造成網路頻寬不足的問題，所以，我們需要數位影音壓縮技術來將這些龐大的資料做壓縮處理。數位影音壓縮的目的是多媒體的減少資料量，增加較多的存量，也可以縮短通訊網路上的傳輸時間。

也因為視訊技術的成長，發展出視覺追蹤監控辨識系統，其應用非常廣泛，例如:自動化安全監控(工廠、社區、大樓)、門禁系統(辨公室、住家)可以辨識出陌生人及可疑人物，身份鑑定(ATM)、電腦使用者開機(Login)驗證，視訊會議、互動式遊戲軟體、手機、IA 產品安全設定以及智慧型機器寵物玩具等都據有辨認主人的功能。

無論經濟景氣與否，犯罪事件依然存在，所以保全機制隨著科技的進步，也迅速發展；日常生活中，我們常以輸入密碼、磁卡感應等作身份確認使用，但那些確認方式有一些缺點，例如密碼會被破解甚至被竄改密碼或是忘記密碼、遺失卡片等風險；在監控技術方面一直不斷的被研發、改進，例如，使用身上的特徵作為保全機制確認使用者身份的方式，如人臉、指紋、語音、虹膜等，比較完善、可靠、人性化。

所以，未來資訊在傳遞和交換上勢必更為快速，而即時通訊技術也會更為純熟，更為好用。如此的省時間，也容易掌握資訊，非常便利。

第二章 研究目的

從網路發展的初期到現在，短短幾十年的時間裡，網路頻寬的變大、電信費用漸便宜以及個人電腦處理能力的增加，藉由電腦網路來進行溝通的比率大幅度地成長，而進行溝通時的品質也越來越好，顯示出網路視訊的發展正漸漸的成長。所以視訊與網路的結合是波趨勢，隨著這波趨勢，在日常生活上，視訊在網路上的使用也越來越廣，例如：公司和公司之間的視訊會議、朋友之間的視訊聊天，還有現在手機主打的 3G 影像通話，都可以立即地讓對方接收到最新的訊息跟狀態，這種即時性與便利性是傳統的電話或是郵件所不能及的。所以我們這次的研究是朝著網路視訊這方面去著手，並且想達到遠端控制還有追蹤的功能。

傳統上的攝影機不是固定不動就是左右來回移動，不但移動的範圍有限且使用者不能控制其方向；而攝影方面只負責影像拍攝，沒有鎖定一個明確的目標在接收端的畫面上，因此我們嘗試從遠端去控制攝影機的方向並加上人臉追蹤的功能，讓使用者得以控制方向並對目標作有效的找尋，如此可以讓目標在螢幕中顯得更為明確，讓使用者找尋目標更有效率。

在網路傳輸方面，由於即時性的影像傳輸，長時間資料量比傳輸一般資料大很多，所以在有限頻寬下，影像傳輸的效率顯得極為重要。因此我們利用壓縮的方法讓資料能夠迅速、省時地遞送。於是，我們使用 JEPG 壓縮來減少我們的資料大小，再經由 UDP 協定來傳輸，以減少網路與主機的負擔。而我們選用 JPEG 壓縮是因為它是現在很普遍的壓縮技術，壓縮率跟壓縮後的品質也被大家接受，而且在學習上也比較容易。

第三章 相關系統研究

這次的專題研究我們需要網路的傳輸，影像的壓縮，以及人臉的追蹤等等。而在研究的同時，也參考了許多不同的方法，以下是將所碰到的方法加以整理，而在第四章可以更加了解我們所使用的方法及其步驟。

當我們研究影像壓縮時，我們會看到許多不同的壓縮方式，而這些影像壓縮的方式大致可分為二種: (1)無失真的影像壓縮格式 (2)失真的影像壓縮格式

●在無失真的影像壓縮格式中，是利用傳統檔案的壓縮原理及技術來處理影像壓縮，所以壓縮前的原始影像與壓縮後還原的結果絲毫不差。其常用的壓縮格式有以下所列: (1) PCX壓縮 (2) TGA壓縮 (3) TIFF壓縮 (4) GIF壓縮
●而在失真的影像壓縮當中，常見的壓縮格式有: (1) JPEG 壓縮 (2) MPEG 壓縮

而我們在網路方面必須先探討 TCP 及 UDP 兩功能的比較，加以選擇適合我們傳輸的功能。以下為 Microsoft Winsock Control 控制元件簡介。

Microsoft Winsock Control 控制元件簡介:

Microsoft Winsock Control 控制元件支援 UDP(User Datagram Protocol)與 TCP(Transmission Control Protocol)兩種通訊協定。UDP 是非連結式通訊協定，不須建立特定的網路連結，只要設定電腦間的 IP 位址與使用相同的 Port，即可互相傳遞訊息。TCP 是連結式通訊協定，用戶端電腦與伺服端電腦必須先建立網路連結，便可在此連結下，互相傳遞資料與訊息。

UDP 可以說是發送出去就不去理會了，TCP 則會在時限之內，若沒有回音就不斷嘗試重發，因此在網路吵雜的環境裡，TCP 成功完成傳信任務的機率較高，但效率也較低且耗時，且成功完成傳信任務並不保證信中的內容無誤。

協定	優點	缺點
TCP	傳送可靠，程式可省略可靠機制。	速度比較慢。
UDP	傳輸量大，迅速。	不可靠，程式或需自行提供可靠機制。

圖 3-1 TCP 與 UDP 的比較圖

(1) Stream socket

Stream socket 提供了雙向的、有序的、無重複以及無記錄邊界的資料流程服務，它非常適合於處理大量資料。Stream socket 是連線導向的，在進行資料交換之前，要先建立資料傳鏈路，這樣就為以後繼資料的傳輸確定了可以確保有序到達的路徑，同時為了確保資料的正確性，可能還會執行額外的計算來驗證正確性，所以相對於 datagram socket，它的系統開銷較大。

(2) Datagram socket

Datagram socket 支援雙向的資料流程，但在傳輸過程中並不保證資料的傳輸可靠性、有序性和無重複性。任何資料被發送出去，都不能保證該資料能夠完好無損，正確無誤的被對方接受。除此之外，datagram socket 還有一個重要特點，就是它保記錄邊界。由於 datagram socket 是無連接的，所以在資料傳輸時，它並不能保證接收端是否正在偵聽。因此 datagram 並不十分可靠，需要編碼來實現資料的排序和傳輸的可靠性，但由於它的傳輸效率非常高，所以至今還得到較為廣泛的應用。

關於人臉追蹤的技術研究，普遍被使用的方法約略分為 Template matching methods 、 Color-based approach 、 Neural network 、 Feature-based approaches 、 Motion-based approach 、 Knowledge-based methods 等

1. Template matching（樣板比對法）：用臉部特徵建立為一個或多個樣板，以“搜尋視窗”的方式實現偵測的動作，缺點：易受臉部的旋轉角度、比例等條件影響追蹤結果，增加運算的時間複雜度[8]。

2. Neural network (類神經網路) ：利用大量人臉及非人臉的影像資料（Training Data）[7]，讓 Neural network 架構去作訓練， Neural network 會自動區別影像中哪一部分為人臉、哪一部分為非人臉，這樣的追蹤方式對於正面且不旋轉的人臉影像作追蹤有極佳的成效，缺點：當人臉的傾斜與旋轉時，效果會差一點。

3. Knowledge-based methods ：根據臉部之間的相關性來搜尋人臉，缺點：容易因為取的數據不夠完整而產生誤差，若運算式條件定得過度嚴苛，會造成存在人臉的區域被忽略；若過於寬鬆，則會將非人臉的區域誤判為人臉[6] ，較適合背景單純及正面的臉部影像。

4. Color-based approach ：將輸入影像在 HSV 中分離出 hue 儲存成直方圖建立色彩模型，計算出膚色的色彩概率分布，缺點：需要避開與膚色相近的的背景，其為我們採用的方法。

5. Motion-based approach ：利用兩張時間點不同但背景一致的影像相減，將影像中變動的物體作定位, 再和臉部的輪廓作比對追蹤人臉，較適合用於動態影像中。

6. Feature-based approaches ：利用影像中背景與人臉的邊緣線，對照臉部特徵之間的關係尋找人臉，缺點：當臉部特徵受光線影響，會降低追蹤效果。

第四章 研究方法

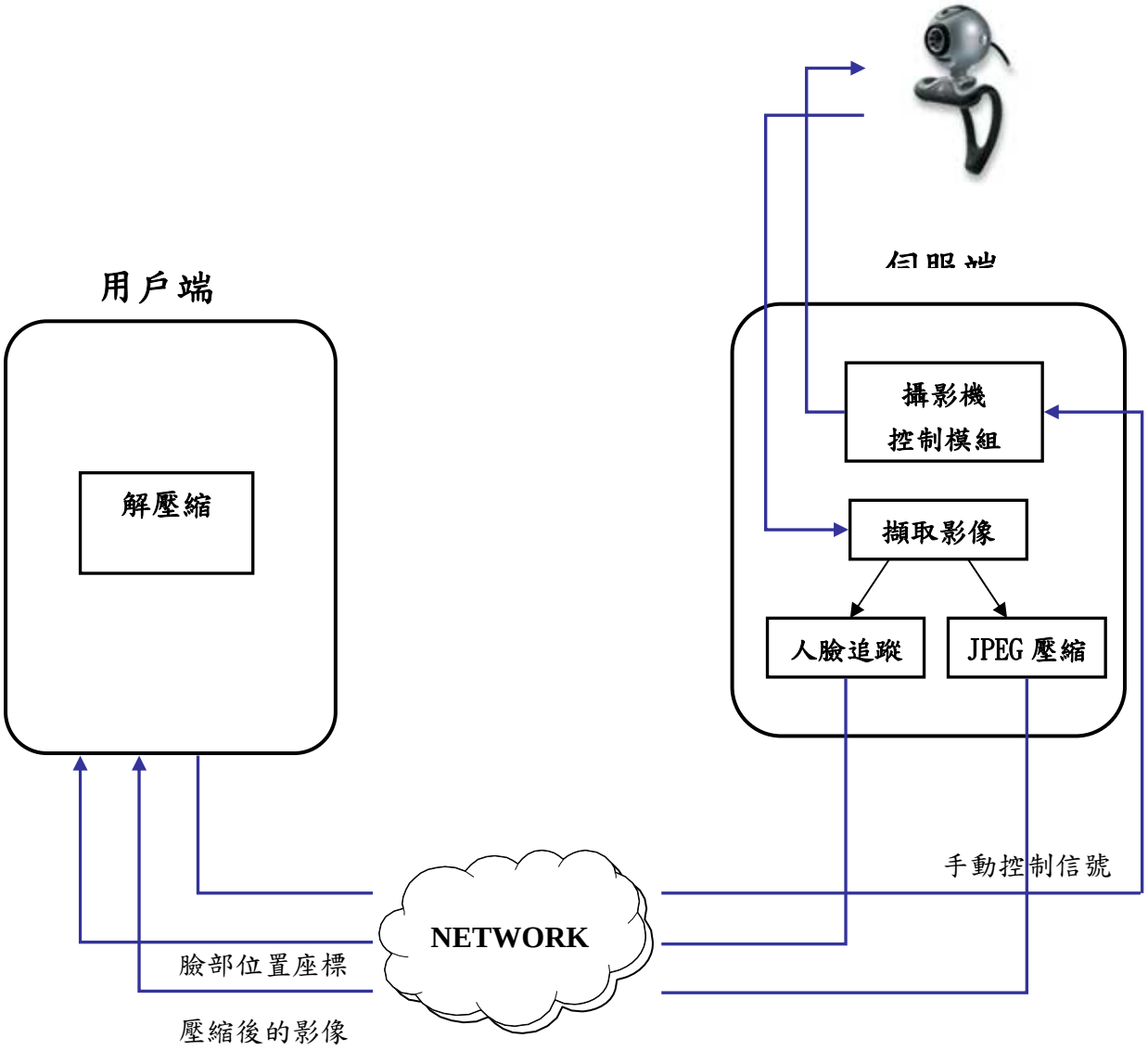

圖 4-1 系統結構圖

大致的系統運作程序是：先在用戶端和伺服端建立 SOCKET 連線，完成網路連線，在遠端伺服器擷取影像，進行人臉追蹤和 JPEG 壓縮動作，將計算好的座標點和壓縮後的影像傳送給用戶端，用戶端則負責解壓縮和傳達手動信號給伺服端的攝影機模組操作攝影機。以下會依依說明結構圖中每一方塊所要執行的工作。

而攝影機控制模組是在使用者端與遠端連線後，由於攝影機是可以移動的，使用者端只要經由鍵盤發出控制訊號，經由網路傳送到遠端，遠端接收到指令後並命令攝影機隨著我們發出的訊號而有所動作。在我們規劃中，此模組是用來控

制攝影機的鏡頭方向，只要使用熱鍵，就可以輕鬆將攝影機做水平、垂直、傾斜的移動，更可以 360 度旋轉，以及將影像放大縮小，讓使用者看的更清楚。

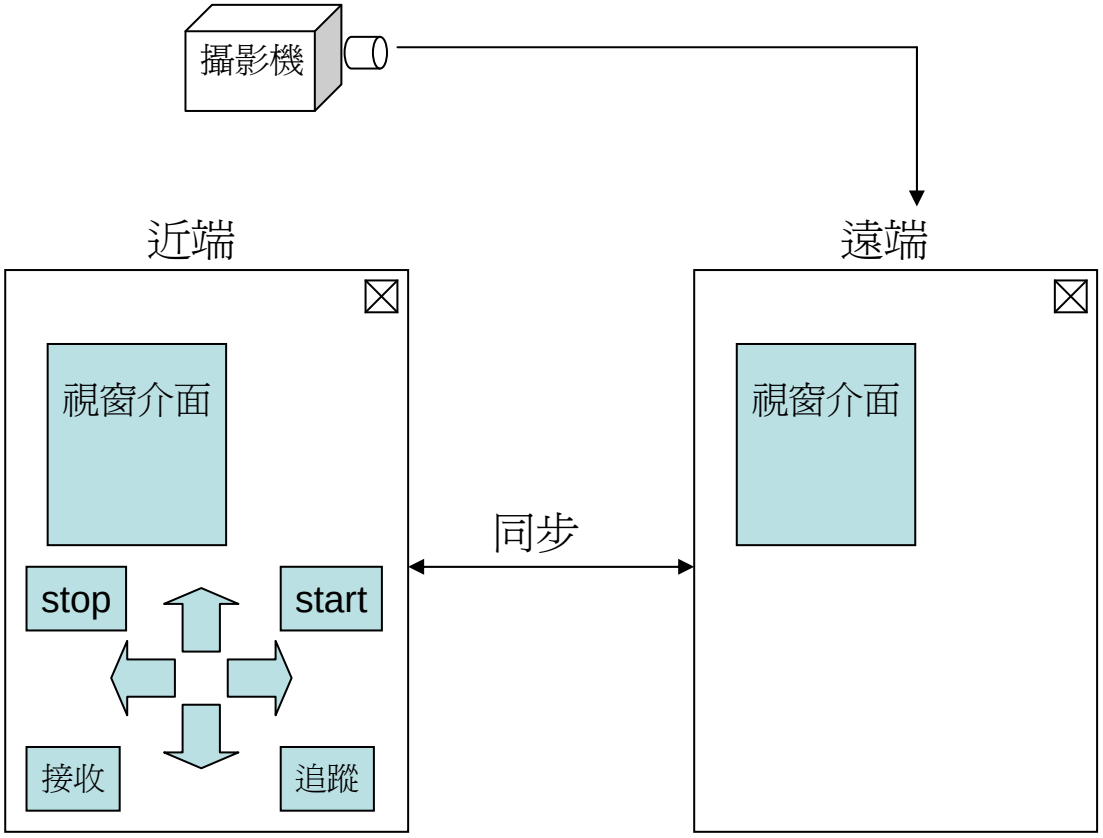

圖 4-2 使用者控制介面

在使用者介面中，我們經由攝影機來攝取影像，並顯示在遠端的視窗介面上，然後按下近端的接收鈕時，遠端則開始做壓縮跟傳送資料到近端的動作，當資料到達時，近端則開始做解壓縮的動作並將影像顯示在視窗介面上，而近端介面上有按鈕可以讓使用者去控制攝影機跟遠端的動作。

以下會一一說明結構圖中每一方塊所要執行的工作。

4.1 JPEG 壓縮

JPEG全名稱為Joint Photographic Experts Group，由國際標準組織(International Organization for Standardization，簡稱ISO)和國際電話電報諮詢委員會(International Telegraph and Telephone Consultative Committee，簡稱CCITT) 所建立的一個數位影像壓縮標準，對於靜態的全彩和灰階影像是一種很有效的壓縮方法，已經是一個國際標準的影像壓縮法。它採用可失真Lossy編碼法的概念，利用離散餘弦轉換法（Discrete Cosine Transform），簡稱DCT將影像資料中較不重要的部份去除，僅保留重要的資訊，以達到高壓縮率的目的，適用於一般連續色調、多級灰階、彩色或黑白靜止圖像壓縮。雖然被處理成JPEG後的影像會有失真的現象，但JPEG的失真比例可以利用參數來加以控制。一般而言，當壓縮率（即壓縮過後的體積除以原有資料量的結果）在5% ～15%之間，JPEG依然能保證其適當的影像品質。

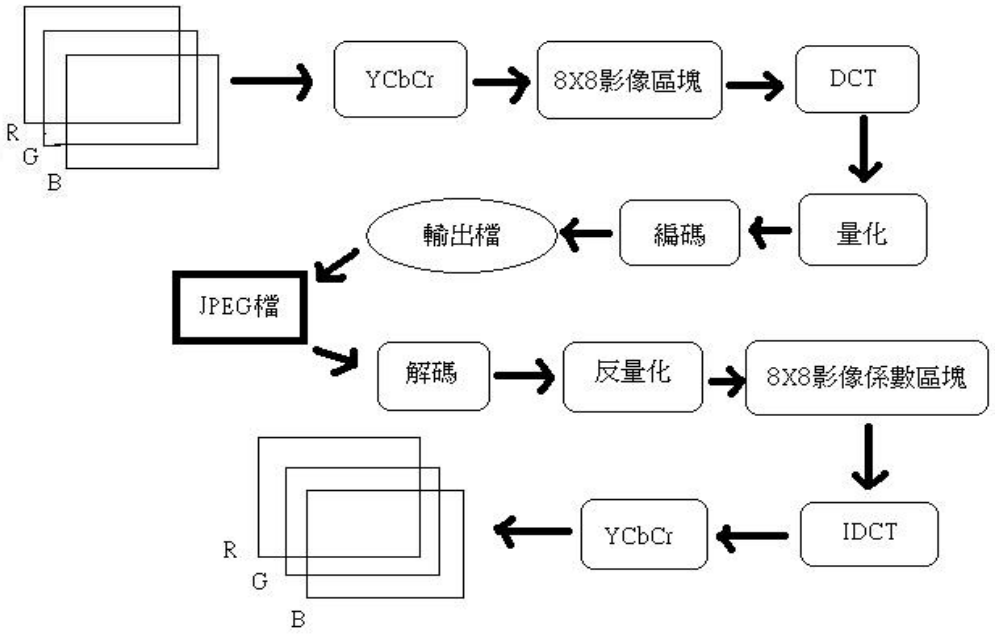

圖 4-3　JPEG 系統結構圖

資料壓縮的過程，主要包含兩步驟：編碼(coding)及模式化(modeling)。所謂模式化是指萃取出(extract)資料中的累贅，並且選擇一個適當的模式來描述這些累贅的動作過程，也就是說減少且移除這些累贅，就能達到資料壓縮的目的。而編碼則是將所選擇的模式之描述(model description)及資料與模式間的差異（或稱誤差），加以編碼（通常編成二進制碼）的動作過程。將資料量減少的技術就叫資料壓縮。

以下是 JPEG 系統的架構順序:
1.輸入一張完整的 RGB 彩色影像
2.將 RGB 色彩轉換成 YCbCr 色彩
3.取樣 8X8 區塊
4.DCT 轉換
5.量化
6.預測編碼
7.霍夫曼編碼
8.壓縮後字串
此逆過程就形成 JPEG 的解碼系統

以下我們將逐一的介紹，當輸入一張完整的 RGB 影像時，在各部份的工作情形以及工作原理。

4.1.1 RGB轉成YCbCr

RGB就是紅綠藍三原色。彩色系統另有好幾種，任兩種彩色系統之間都存在聯繫彼此的數學關係。因為JPEG是採用Y,Cb,Cr模式，所以我們必須先把影像或圖檔轉換成亮度與彩度分開的形式。利用以下的公式即可將RGB色彩模式轉換成YCbCr的色彩模式。

RGB和YCbCr有以下的關係:

$$Y = 0.3R + 0.6G + 0.1B$$
$$Cb = -0.168R - 0.331G + 0.499B$$
$$Cr = 0.5R - 0.419G - 0.081B$$

其中YCbCr的Y代表灰階亮度，而Cb和Cr代表彩度，因為人類對於較低頻的訊號比高頻訊號來的敏感，亮度的改變也比色彩的改變來的敏感，所以亮度的成份較為重要，而捨棄了較多的彩度成份。

4.1.2 取樣8X8區塊

因為人對於亮度的改變比彩度的改變來的敏感，所以在取樣的過程中我們保留了較多的 Y 成份，而彩度資料部份的捨去。由圖 4-4 可以得知，在取樣時，會先將畫面分割成許多互不重疊的 16×16 的巨方塊，在 JPEG 壓縮標準，它會對亮度與彩度做 4:1:1 的取樣格式。所以保留 Y 的全部，而彩度(Cb 與 Cr)資料依照 4:1:1 的取樣格式將其取樣出來，原本亮度及彩度都為 16×16 的巨方塊，取樣時彩度只取 8×8 的方塊。由圖中也可以知道在比例上 Y 所佔的比例比較大，因為人眼對亮度的敏感度高於彩度，所以彩度顯得比較不為重要。

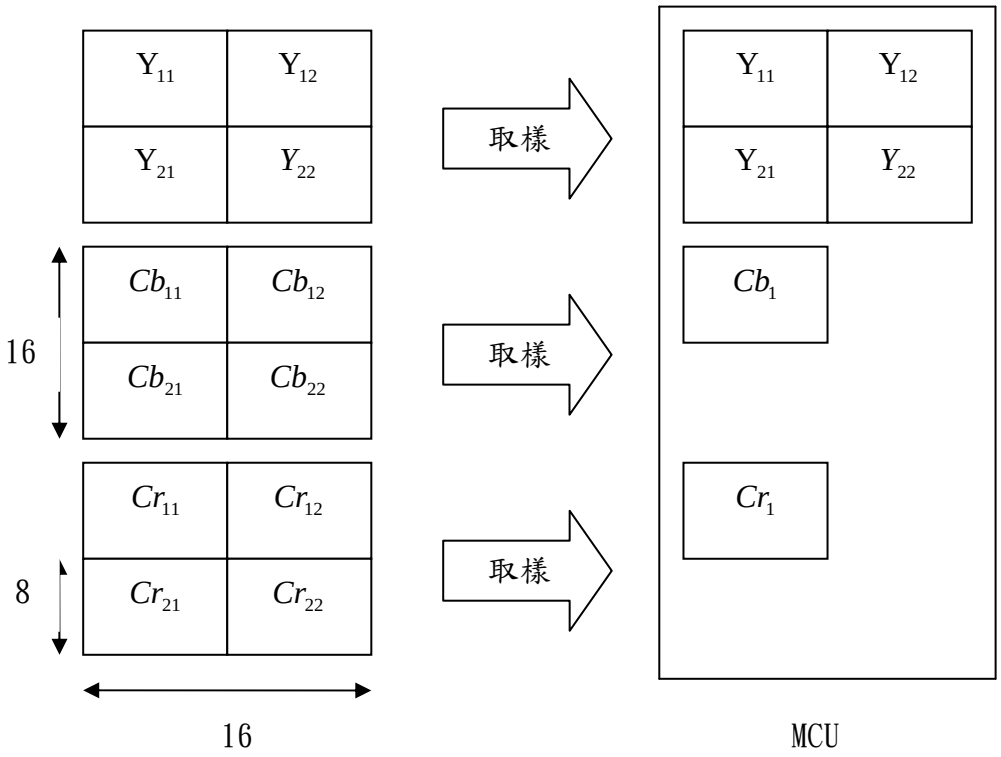

圖 4-4　區塊取樣圖

舉例說明: (以4:2:2的取樣格式)

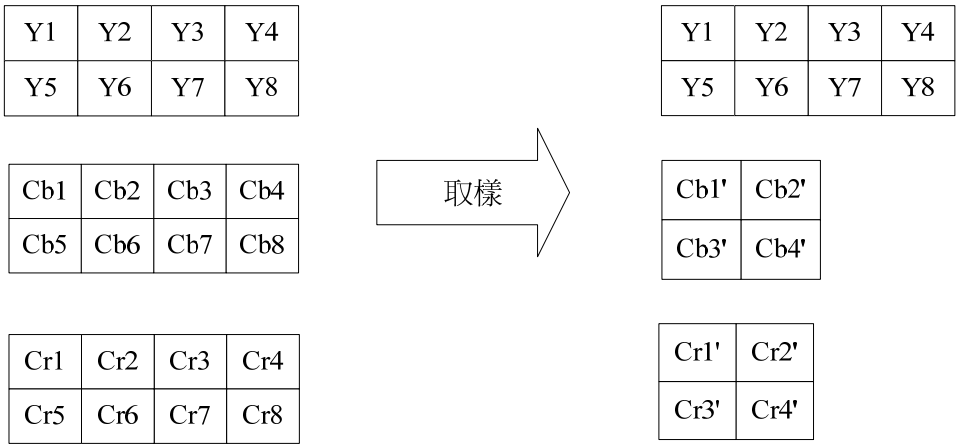

圖4-5　區塊取樣圖(2)

給一個16×32的子影像，假設八個8×8亮度區塊，八個8×8 Cb彩度區塊和八個8×8 Cr彩度區塊，我們保留Y的所有資料量，對8×8 Cb1和8×8 Cb2進行取樣，得到一個8×8 Cb1’，同理對8×8 Cb3和8×8 Cb4進行取樣，得到一個8×8 Cb2’， 對8×8 Cb5和8×8 Cb6進行取樣，得到一個8×8 Cb3’， 對8×8 Cb7和8×8 Cb8進行

取樣，得到一個8×8 Cb4’，對16×32大小的八個Cr區塊，進行類似於得到Cb’的取樣動作，我們可以得到圖4-5的右方所示。因為採用抽樣的關係，所以會造成彩度的失真現象。

4.1.3 DCT轉換(Discrete Cosine Transform 離散餘弦轉換)

輸入是8×8個點，輸出則是8×8個係數。把影像由空間定義域 (space domain) 轉換到頻率定義域 (frequency domain)，每個8×8小方塊裡面係數的位置愈靠近左上角，它代表的頻率愈低，愈靠近右下角，則它代表的頻率愈高。大部份的影像能量會集中在低頻部份，也就是轉換之後的輸出係數在低頻部份的值較大，而輸出係數在高頻部份的值很小。當輸出係數經過量化之後，高頻部份的值大部份都會變為 0，其主要是將能量集中。現在有許多壓縮方法使用此離散餘弦轉換。

離散餘弦轉換公式如下：

$$F(u,v)=\frac{1}{4}c(u)c(v)\sum_{i=0}^{7}\sum_{j=0}^{7}f(i,j)\cos\frac{(2i+1)u\pi}{16}\cos\frac{(2j+1)v\pi}{16}$$

其中 $F(u,v)$ 表示輸出係數， $f(i,j)$ 表示輸入點的值。

舉例說明:

我們將影像劃分成8×8像素大小的區塊且每一個區塊並不重疊，在對8×8的 Y 進行 DCT 轉換前，先將 Y 的每一像素皆減去 128，因為 DCT 轉換的公式所接受的數值範圍是在-128 到+127 之間。令減後的像素灰階值為 $f(i,j)$，將 DCT 作用於其上並利用離散餘弦轉換公式計算。

當 u = 0 時，則 $C(u)=\frac{1}{\sqrt{2}}$ ； 當 u≠0 時，則 C(u) = 1； 當 v = 0 時，則 $C(v)=\frac{1}{\sqrt{2}}$；當 v≠0 時，則 C(v) = 1。

假如給一個8×8子影像如圖 4-6 所示，經 DCT 轉換作用後得到圖的 DCT 係數矩陣。

212	208	212	215	215	219	227	227
209	211	209	215	216	219	218	227
205	208	200	211	213	211	207	215
207	209	208	208	219	213	214	212
206	203	208	200	211	211	212	218
202	196	201	202	208	211	215	213

209	209	204	204	200	212	213	216
205	210	211	202	208	208	211	211

圖 4-6　8×8子影像 Y

Y 經過 DCT 作用後的結果，如圖 4-7

659.62	-29.97	8.62	1.91	1.62	-30.92	0.89	1.52
23.23	-7.18	-4.31	-0.42	7.35	0.01	-2.25	-3.17
11.8	-0.26	5.2	-4.77	-3.57	4.16	-0.26	-3.49
2.3	-10.74	5.49	0.79	-1.01	7.6	3.79	2.82
6.37	2.51	-1.53	-1.07	-3.61	-0.78	0.51	8.72
0.74	2.61	0.72	2.53	-0.91	3.21	-2.94	2.79
-9.08	-1.66	-4.51	1.74	2.16	1.55	-1.68	2.06
-3.61	2.24	5.35	-1.96	0.9	-1.37	1.83	-3.31

圖 4-7　DCT 作用後係數值

由此可得知，影像經過 DCT 轉換後有能量聚集的現象，也就是說低頻率的 DCT 係數會集中在左上角，畢竟在一張影像上，低頻的紋理佔較多。

4.1.4　量化(Quantization)

我們在DCT係數矩陣上進行量化時，需要一個量化表，量化表的功能旨在達到壓縮的效果。量化的目的是將經DCT轉換後出來之區塊內各係數值變小，以利後續之編碼能取得更好的壓縮效果，將區塊內各係數值除以量化表(quantization table)該位置之值，只取整數，小數以下忽略不計。假設如圖:

498.46	35.54	-4.14	1.91	1.33	-12.95	-0.92	1.48
-224.33	-9.45	-5.66	14.24	7.21	-0.53	-2.57	-3.26
13.42	4.47	3.42	-4.42	-3.26	3.79	-0.49	-3.23
4.95	-4.53	6.43	0.88	-1.23	6.71	3.6	2.48
5.72	2.51	-1.47	-0.93	-2.57	-0.67	0.44	8.61
1.32	2.61	0.47	2.23	-0.89	2.54	-2.73	-2.21
-2.45	-1.66	-4.63	2.14	2.57	-4.6	-1.2	1.89
4.36	3.21	4.82	-1.31	0.54	1.36	0.87	-2.54

圖4-8 8×8DCT係數矩陣

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

圖4-9 8×8量化表

31	3	0	0	0	0	0	0
-19	-1	0	1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

圖4-10 量化後DCT係數矩陣

量化後 DCT 係數矩陣左上角值稱為 DC 值，其餘 63 個值稱為 AC 值。如圖 4-8，我們是將8×8的 DCT 係數矩陣放置在8×8的量化表上，將 DCT 的係數除以對應到的量化值，在予以四捨五入，以便得到整數值。量化表的值越大表示壓縮比較好，AC 的值越小代表其係數越不重要。

量化的目的是希望將 DCT 轉換後的每個係數值變小，使得後來的編碼能取得更好的壓縮效果。而量化表的數值並不是一定，可自行設計，因為設計好的量化表往往能兼顧壓縮與品質的雙重考量。

舉例說明:

659.62	-29.97	8.62	1.91	1.62	-30.92	0.89	1.52
23.23	-7.18	-4.31	-0.42	7.35	0.01	-2.25	-3.17
11.8	-0.26	5.2	-4.77	-3.57	4.16	-0.26	-3.49
2.3	-10.74	5.49	0.79	-1.01	7.6	3.79	2.82
6.37	2.51	-1.53	-1.07	-3.61	-0.78	0.51	8.72
0.74	2.61	0.72	2.53	-0.91	3.21	-2.94	2.79
-9.08	-1.66	-4.51	1.74	2.16	1.55	-1.68	2.06
-3.61	2.24	5.35	-1.96	0.9	-1.37	1.83	-3.31

圖 4-11 DCT 作用後係數值圖(2)

此為 DCT 轉化後的係數值，我們利用圖 4-9 量化表進行量化，得到圖 4-12 的量化結果。在圖的左上角 41 稱為 DC 值，其餘的 63 個值稱為 AC 值。在圖 4-9 中的量化表為事先建好的，將左上角 659.62 除以量化表左上角 16 得到 41.22，經過四捨五入後，得到整數值 41。其他數值也直接仿照此方法得到。

41	-3	1	0	0	0	0	0
2	-1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

圖 4-12 量化後的數值圖

4.1.5 霍夫曼編碼

編碼時，每一個8×8矩陣資料的 DC 值與 63 個 AC 值，將分別使用不同的 Huffman 編碼表，而亮度與色度也需要不同的 Huffman 編碼表，所以一共需要四個編碼表，才能順利地完成 JPEG 編碼工作。

由此可知，在編碼過程當中，我們可以細分成 DC 編碼和 AC 編碼:

4.1.5.1 DC 編碼

DC 編碼是採用差值脈衝編碼的差值編碼法，也就是在同一個圖像分量中取得每個 DC 值與前一個 DC 值的差值來編碼，我們用圖 4-13 來表示，同一個圖像取兩個子影像做解釋，第一個子影像的 DC 值稱為 DCx-1，第二個子影像的 DC 值為 DCx，將 DCx 減去 DCx-1 就是所得的差值。

其主要原因是由於在連續色調的圖像中，其差值多半比原值小，對差值進行編碼所需的位數，會比對原值進行編碼所需的位數少。例如差值為 5，它的二進制表示值為 101，如果差值為-5，則先改為正整數 5，再將其二進制轉換成 1 的補數即可。

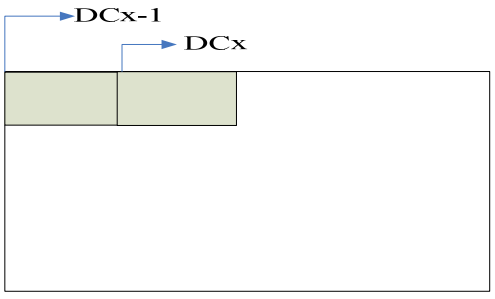

圖 4-13 差值表示圖

差值Bits數	DC差值內容
0	0
1	-1,1
2	-3,-2,2,3
3	-7,⋯,-4,4,⋯,7
4	-15,⋯,-8,8,⋯,15
5	-31,⋯,-16,16,⋯,31
6	-63,⋯,-32,32,⋯,63
7	-127,⋯,-64,64,...127
8	-255,⋯,-128,128,⋯,255
9	-511,⋯,-256,256,⋯,511
10	-1023,⋯,-512,512,⋯,1023
11	-2047,⋯,-1024,1024,⋯,2047

表 4-14 差值對照表

差值bits數	編碼bits數	編碼內容
0	2	00
1	3	010
2	3	011
3	3	100
4	3	101
5	3	110
6	4	1110
7	5	11110
8	6	111110
9	7	1111110
10	8	11111110
11	9	111111110

表 4-15 亮度 DC 差值的霍夫曼編碼表

差值bits數	編碼bits數	編碼內容
0	2	00
1	2	01
2	2	10
3	3	110
4	4	1110
5	5	11110
6	6	111110
7	7	1111110
8	8	11111110
9	9	111111110
10	10	1111111110
11	11	11111111110

表 4-16 色度 DC 差值的霍夫曼編碼表

在差值前端另外加入一些差值的霍夫曼碼值，從圖可以得知，亮度值差為 5 的位元數等於 3，則獲夫曼碼值應該是 100，將兩者連接在一起即為 100101，完成 DC 的編碼工作。

4.1.5.2 AC 編碼

而 AC 編碼方式與 DC 不同，是在 AC 編碼之前，先將 63 個 AC 值按 Zig-zag 排序。Zig-Zag 的方法為斜向掃描，沿著空間頻率大小增加的方向做掃描，從左上至右下，藉此可提高壓縮的效率，如圖 4-17:

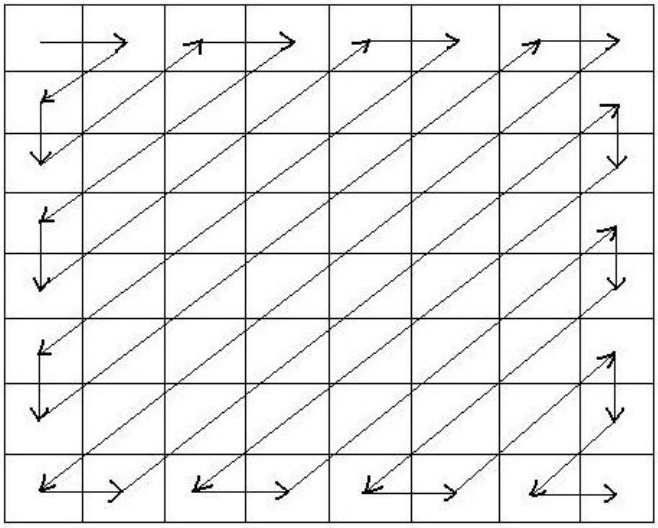

圖 4-17 Zig-Zag 掃描次序

63 個 AC 值排列好，其表示符號為 R/S(Runlength，Size)，R 是指第非零的 AC 之前，其值為 0 的 AC 個數，S 是指 AC 值所需的位數，AC 系數的範圍與 S 的對應關係則與 DC 差值 Bits 數與差值內容對照表類似。如果連續為 0 的 AC 個數大於 15，則用 15/0 來表示連續的 16 個 0，15/0 稱為 ZRL（ZeroRumLength），而（0/0）稱為 EOB（EnelofBlock）用來表示其後所剩餘的 AC 係數皆為 0，以符號值作為索引值，從相應的 AC 編碼表中找出適當的霍夫曼碼值，再與 AC 值相連即可。

舉例說明:

41	-3	1	0	0	0	0	0
2	-1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

圖 4-18　量化後的數值圖(2)

利用 Zig-Zag 掃描,得到 DC 值等於 41,而 AC 值依序排列得到向量形式(41, -3, 2, 1, -1, 1, 0, 0, 0, 0, 0, -1,……,0)。下一步驟對於上序排列進行 Run-Length 編碼，在這裡，我們只針對 AC 值來編碼。以(X，Y)的格式紀錄，X 代表 Y 之前 0 的個數，Y 代表 ACs 中非 0 值，而 EOB 用來表示其後所剩餘的 AC 係數皆為 0。

上述的向量形式可編碼為(0，-3)(0，2)(0，1)(0，-1)(0，1)(5，-1)EOB，此處(0，2)中的 0 代表 AC 值 2 和前一個非零值-3 中間沒有零的 AC 值。(5，-1)中的 5 代表 AC 值-1 和前一個非零的 AC 值 1 的中間有連續 5 個零。EOB 只是代表結束的符號。在 Run-Length 編碼的格式(X，Y)中，X 通常採用固定長度編碼，而 Y 必須事先建好的圖進行變動長度編碼。

位元數	Y的範圍
0	0
1	-1,1
2	-3,-2,2,3
3	-7,⋯,-4,4,⋯,7
4	-15,⋯,-8,8,⋯,15
5	-31,⋯,-16,16,⋯31

表 4-19　Y 的編碼對照表

依照 X 的固定長度編碼和 Y 的變動編碼，上述的向量型式進一步編成(0，2)(00)(0，2)(10)(0，1)(1)(0，1)(0)(0，1)(1)(5，1)(0)EOB，此處(0，2)(00)中的 2 代表位元數而 00 代表-3 的位元編碼。Y 若本身大於 0 ，直接以二進位表示，若小於 0，則以二進位表示其絕對值後，取其 1 的補數。

(x,y)	碼
EOB	1010
(0,1)	00
(0,2)	01
(0,3)	100
(0,4)	1011
(4,1)	111011
(4,2)	1111111000
(5,1)	1111010
(6,1)	1111011
(6,2)	111111110110

表 4-20　霍夫曼 AC 亮度碼表

接下來，我們利用 JPEG 中的霍夫曼 AC 亮度表來將上述碼編成二位元字串。我們只針對(X，Y)的部份來進行霍夫曼編碼，根據查表結果可知(0，2)對應的碼為 01 ; (0，1)對應的碼為 00 ; (5，1)對應的碼為 1111010 ; EOB 經查表得到碼 1010。最終，我們將圖編成 0100 0110 001 000 001 11110100 1010。由此可得知。

4.2 人臉追蹤

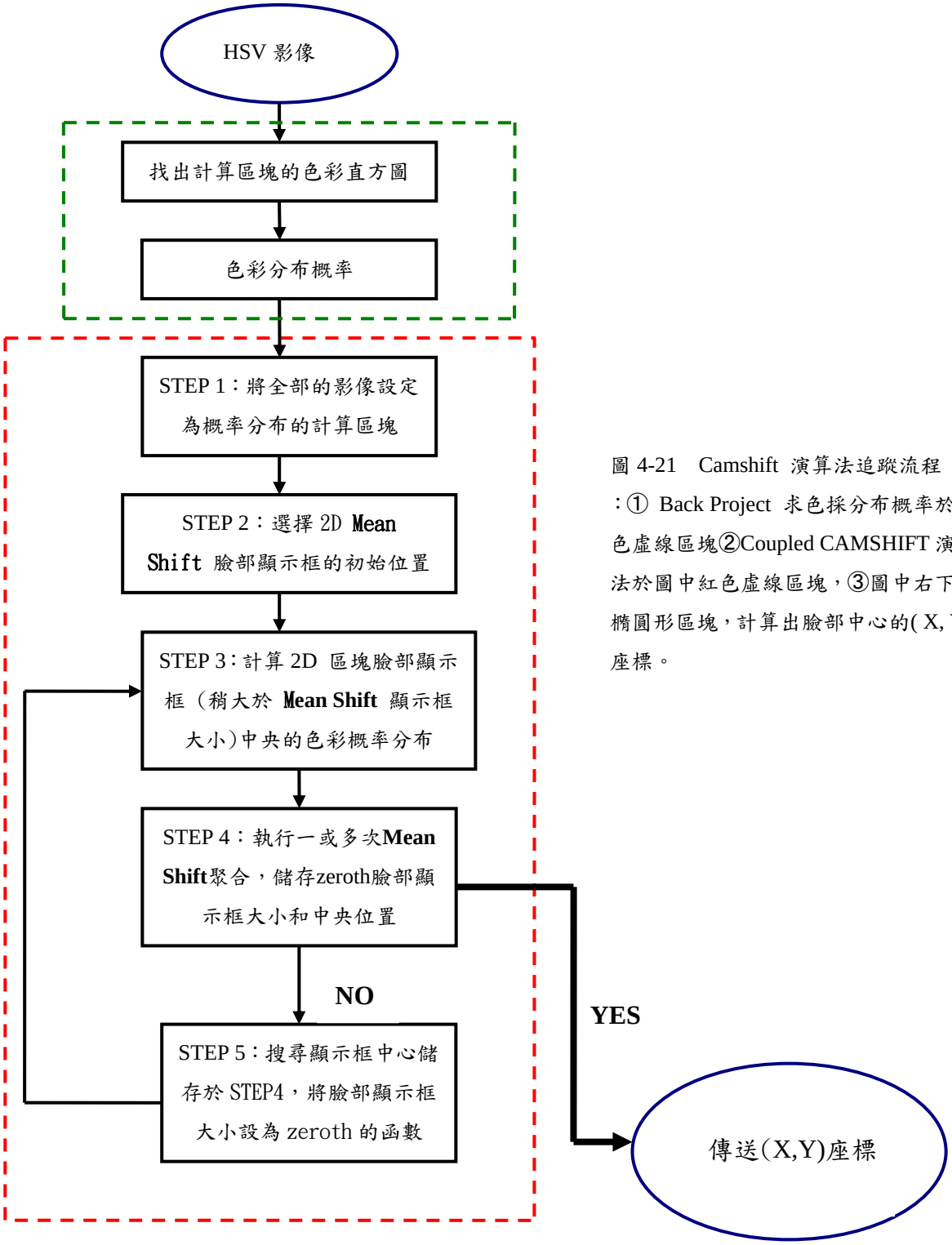

圖 4-21　Camshift 演算法追蹤流程：① Back Project 求色採分布概率於綠色虛線區塊②Coupled CAMSHIFT 演算法於圖中紅色虛線區塊，③圖中右下方橢圓形區塊，計算出臉部中心的(X, Y) 座標。

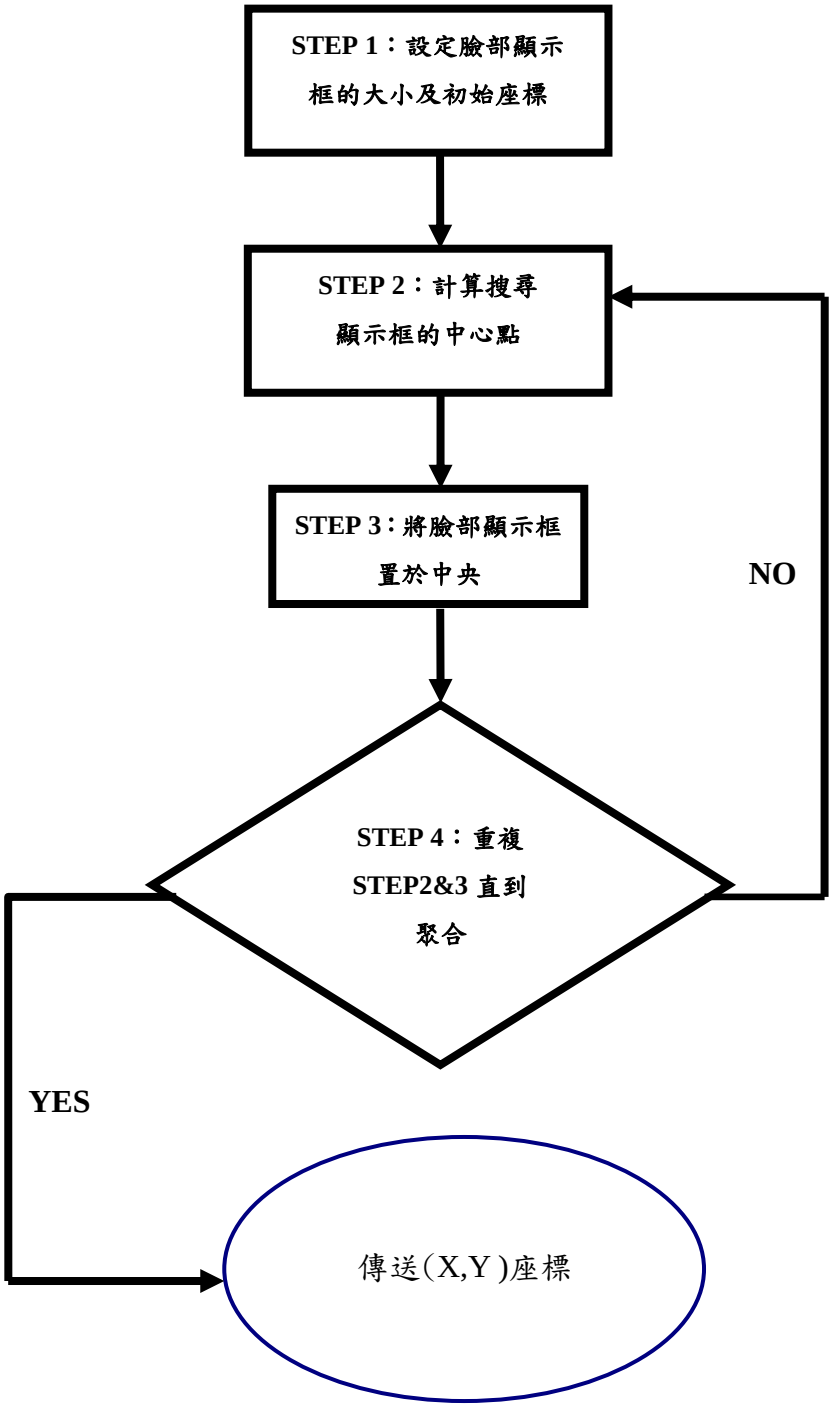

圖 4-22　Mean Shift 演算法

大部分偵測臉部的常見方法，有依臉部的膚色、人臉移動的位置、臉部特

徵等作判斷、追蹤。

影像處理臉部追蹤這部分，我們用 OpenCV 電腦視覺庫撰寫程式追蹤人臉，在影像的膚色區塊裡，依照臉部的膚色 (色彩概率分布) 作判別，將鏡頭裡臉部位置顯現出來， OpenCV 是 Intel® 開放原始碼的程式庫，由一系列 C 函數和少量 C++ 類別構成，提供許多有關圖像處理、模式識別、三維重建、物體跟蹤、機器學習和線性代數等的通用演算法。此外我們應用其中的 CAMSHIFT（Continuously Adaptive Mean Shift） 演算法來找出臉部位置的座標，以下將依依介紹。

Mean Shift 演算法會影響色彩概率分布，在追蹤視訊連續鏡頭裡的色彩物件時，色彩影像資料被視為一個色彩概率分布；概率分布可利用色彩直方圖得到，從連續擷取視訊影像中取得色彩概率，由於動態影像中，追蹤物體的大小及位置不斷改變，但 Mean Shift 的搜尋視窗大小是固定的，所以 Mean Shift 演算法需要再作修正，修改為不斷調整正在追蹤的物件之概率分布，具備此條件的新演算法稱為 CAMSHIFT（Continuously Adaptive Mean Shift）演算法。 Mean Shift 演算法是對靜態概率分布設計的； CAMSHIFT 演算法是對動態調整概率分布設計的。

人臉追蹤動態影像必須考慮：背景顏色 (膚色相近) 、人臉大小、亮度變化等因素。

將三維的 RGB 空間與二維的空間作對映(正規化)：

$$r = R/(R+G+B) \qquad \text{(式子 a)}$$

$$g = G/(R+G+B) \qquad \text{(式子 b)}$$

$$b = B/(R+G+B) \qquad \text{(式子 c)}$$

Hue（色相）：色彩的相位，範圍在0度到359度之間，如0度時為紅色、120度為綠色、240度為藍色…

$$H = \cos^{-1}\left\{\frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\sqrt{(R-G)^{20}+(R-B)(G-B)}}\right\} \qquad \text{（式子d）}$$

我們會先在 HSV 空間 (圖) 中分離出 hue (式子d） ，影像中的每一個像素都分別以光的三原色值 (RGB) 來表示，色彩會因亮度的不同，使得 RGB 值會有極大的差異，無法正確判斷出色彩的分布，所以忽略亮度部份，將 R、G、B 作正規化 (式子 a & b & c) ，可使 RG B 減少對光 (亮度) 的依賴，得到 hue 存入 1D 直方圖中,藉由 1D 直方圖建立色彩模型，再算出臉部的色系分布概率。

(a) HSV 色彩系統　　(b) RGB 色彩立方體

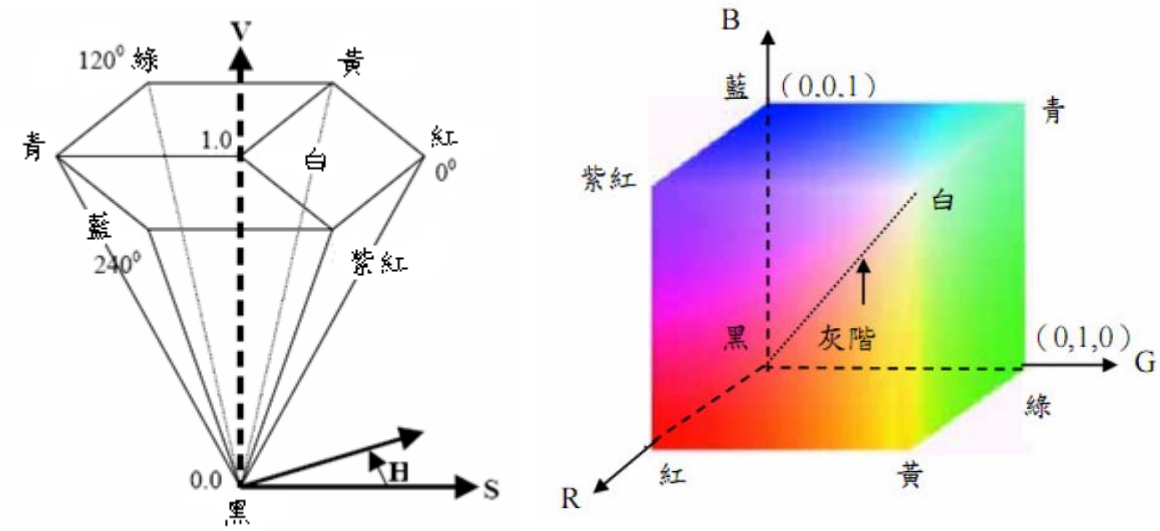

圖 4-23　HSV 色彩系統

(a)相當於投射 RGB 色彩立方體(b)，遵循的原則從白到

1) 色彩概率分布：

為了使用 Camshift 演算法追蹤鏡頭裡的色彩物件，必須找出色彩概率分布。 圖 4-21 的綠色虛線區塊：如何從將 HSV 影像的色彩直方圖轉換成色彩概率分布，此運作為"Back Projection"。

1.在色彩空間中，只有色彩直方圖的 hue 能顯現出顏色信息，所以須先計算出直方圖，再轉換成色彩概率分布。

2. hue 是從 H channel 被取樣影像中的膚色像素，將 hue 儲入 1D 直方圖中,藉由 1D 直方圖建立色彩模型。

3.在操作期間被儲存的膚色直方圖會被視作一個模型或對照表，比照後增加的影像像素將轉換成相對應的膚色影像概率，如圖4-25。離散概率範圍從 0 (probability 0.0) 到最大概率像素值 (probability 1.0) ，對 8-bit hue ,這範圍是界於 0 和 255 之間,然後在膚色影像概率使用 CAMSHIFT 演算法追蹤色彩物件。

圖 4-24　追蹤到的人臉影像

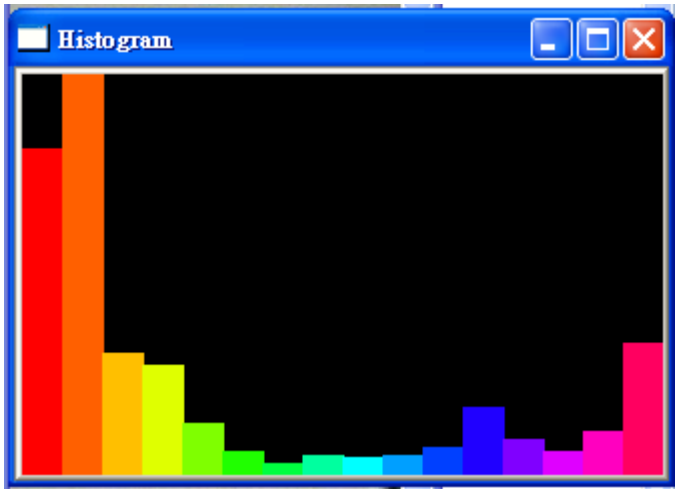

圖 4-25　視訊影像膚色直方圖

2）Mean Shift 演算法：於圖 4-22，運作程序如下：

步驟 1 ：設定臉部顯示框的大小及初始座標。

步驟 2 ：在初始點，計算全部鏡頭色彩概率分布及用第0個時的資訊去設定臉部顯示框大小和用質量中心設定顯示框中心點。

步驟 3 ：將步驟2的搜尋顯示框中心點置於中央。

步驟 4：重複步驟2和3直到聚合(直到中央位置移動小於事先裝置的初始點)。

對於離散 2D 影像概率分布,搜尋顯示框 (步驟3和4)的中央位置(質量中心)如下：

找到第 0 片刻時

$$M_{00} = \sum_x \sum_y I(x,y)$$

找到第一個 x 和 y 時

$$M_{10} = \sum_x \sum_y xI(x,y) \qquad M_{01} = \sum_x \sum_y yI(x,y)$$

中央搜尋顯示框位置(質量中心)為

$$x_c = \frac{M_{10}}{M_{00}} \qquad y_c = \frac{M_{01}}{M_{00}}$$

I(x,y) 是在影像裡 **(x,y)** 位置的像素(概率)值， x 和 y 涵蓋搜尋臉部的顯示框範圍。搜尋顯示框中央座標設定為 **(x,y)** ,用 $2*area^{1/2}$ 設定大小。

3）Coupled CAMSHIFT 演算法： CAMSHIFT 演算法會持續調整運作中搜尋臉部顯示框的大小於圖 4-21 紅色虛線區塊。

步驟 1 ：先設定全部的影像為概率分布的計算區塊。

步驟 2 ：選擇 2D 中央位移搜尋顯示框的初始位置。

步驟 3 ：計算在 2D 區塊搜尋顯示框中心位置的色彩概率分布（稍大於中央位移顯示框大小）。

步驟 4 ：聚合後，選擇重新設定 (NO) 或傳送 X, Y, Z 座標和轉動角度給下一個方塊 (YES) 。儲存第 0 片刻時的大小和中央位置。

步驟 5 ：下一個視訊影像中，用步驟 4 儲存的值，初始化搜尋顯示框的位置

和大小。再回到步驟 3 。

- **frame 的大小和初始定位**

臉部顯示框設定固定的大小(寬度和長度)及初始座標，接著尋找色彩直方圖計算全部鏡頭色彩概率分布。

- **設定適合的顯示框大小函數**

以決定第 0 片刻時的函數設定搜尋顯示框大小為 CAMSHIFT 演算法的步驟 3 （稍大於中央位移顯示框大小）：視一個想要追蹤的分布範圍和一個想要達成的目標而定。首先考慮是轉換第 0 片刻時的資訊將顯示框大小合理化。圖 4-26 中，每一離散細胞最大分布值為 206 ，用 206 分割第 0 片刻時的資訊，將搜尋顯示框下的計算區域轉換為細胞數單位。我們的目標是追蹤全部的色彩物件，所以需要一個遼闊的顯示框。持續包圍下一個最大的不固定搜尋顯示框以致顯示框有一個中心點。對於 2D 色彩概率分布最大像素值為 255 ，設定顯示框大小 s 為

$$s = 2 * \sqrt{\frac{M_{00}}{256}}$$

上面的式子同樣地也用 256 分割，但轉換結果： 2D 區塊為 1D 長度，還要用到方根。對於臉部追蹤範圍，設定顯示框寬度為s和長度為 1.2s ，使臉部呈橢圓形。

- 合併(將 OPENCV library 轉換為 BCB 可用的 library)：

由於我們的介面計設是採用 Borland C++ Builder 軟體來撰寫程式，OPENCV 的 library 是源自 Visual C++ 型式，要使用在 BCB 環境須要再轉成 BCB 可用的 library ，執行才不會產生連結上的問題，再結合攝影機影像擷取卡，將擷取的影像作臉部追蹤。以下是VC library 轉換成 BCB library 的詳細過程：

開一個 txt 檔，如要將原本 OPENCV library 中的 cv.lib 轉成新的 BCB 型式 cv2.lib 之指令內容： coff2omf -lib:ms cv.lib cv2.lib ，將要用到的 library 都寫進轉換指令後存成 .dat 檔，建一個 BCB Project 資料夾，把 opencv/bin 的 dll 和 OPENCV 的 lib 複製到此資料夾，和，執行 dat 檔後便產生新的 library 可用於 BCB ，如 cv2.lib ...在 Project Manager 中加入須用到的 .h 檔、原本的 lib 及轉換過後產生的 library ，然後設定路徑：在 Option 選項裡的 Directories/Conditionals 設定 Include path 和 Library path ，這轉換 library 的程序就完成了。

4.3 WINSOCK

我們專題利用 2 台電腦，一台作為有裝設視訊主機的 Server 端，一台則為可放置於任何區域的電腦作為 Client 端，我們就由 Client 端來啟動連線建立的 Socket，當 Client 端與等待接受連線的 Server 端透過 Internet 連接視訊影像時，便可透過 Server 端把對方視訊影像擷取下來，壓縮並傳回 Client 端。

在網路的連線方面，為了要使兩台電腦連通之後，時間能夠同步，我們應用到 Socket 的技術，開發在 Server 端與Client 端上分別撰寫應用程式相互呼叫控制，並使用到UDP網路協定，因為UDP協定在傳輸過程中並不保證資料的傳輸可靠性、有序性和無重複性，資料遺失不會重傳 。

而我們要傳的資料只是一張擷取下來的影像，遺失了不需要重傳，只須在重新擷取一次影像就可以，而且它的傳輸效率非常高，因此我們選擇使用此協定。而相對下，TCP則比較麻煩，必須雙方要先建立資料傳輸鏈路，才能開始傳資料，所以它的系統開銷較大。

● 以下介紹WINSOCK的作法介紹

研究方法:

1、研讀BC++Builder程式、winsock 程式。

2、開發Client-Server 架構程式，可以透過網路連線。

3、以 BC++Builder 撰寫Socket 應用程式。

Socket程式流程函數解釋如下:

Socket():建立一個 socket，傳回一個 socket 的描述子

Bind():伺服器使用 bind()來指定本身的端點位址

Send():用戶端或伺服器在連線 UDP socket 上送資料

SendBuf ():將一個資料段(datagram)送到一個指定的端點

Recv():用戶端或伺服器在 TCP 連線或 UDP socket 上收資料

ReceiveBuf ():收取下一個進來的資料段，並紀錄傳送端的位址

● Socket架構如下:

舉例說明 ：client端傳圖檔給sever端

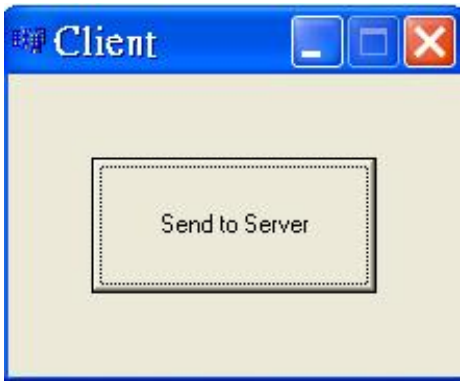

圖4-26　客戶端介面

圖4-27　伺服器端介面

1.在client端開啟一個socket()通訊端點,設定通訊端點內的各項基本資料，如IP 位址, 通訊埠號,通訊協定並給予此通訊端點一個名稱。

```
{
    ClientSocket1->Address="127.0.0.1";
    ClientSocket1->Port=4000;
    ClientSocket1->Open();
    gmsFile=new TMemoryStream;
}
```

2.與其他的通訊端點建立連線關係如圖4-30。

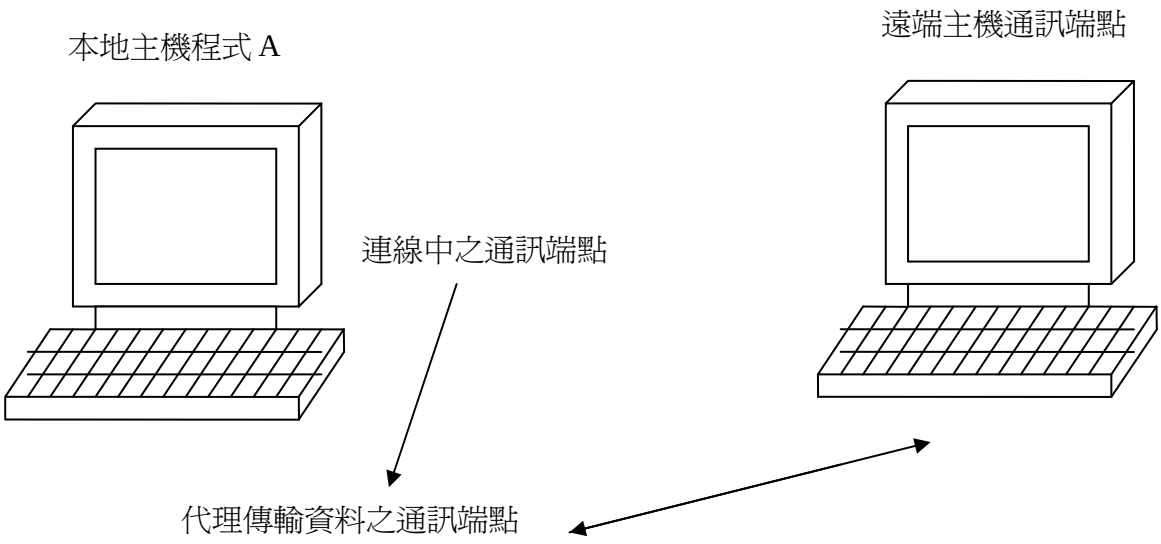

圖 4-28　遠端向本地請求通訊端點連線圖

3. 從client端建立一個文件夾，把圖檔存在此文件夾裡面，
開一個TMemoryStream出來並把我們要傳送的檔案”abc1.jpg”
經由該TMemoryStream傳送一個Stream到sever端

```
    TMemoryStream *msFile=new TMemoryStream;
    msFile->LoadFromFile("abc1.jpg");
    MYPACK mp;
    int i=0;
    int block=(msFile->Size%MAX_DATASIZE==0)?
        msFile->Size/MAX_DATASIZE: msFile->Size/MAX_DATASIZE+1;

    for(i=0; i<block; i++)
    {
        if(i==0)
        {
            lstrcpy(mp.szHeader, "FILE-START");
        }
        else if(i==block-1)
        {
            lstrcpy(mp.szHeader, "FILE-END");
        }
        else
        {
            lstrcpy(mp.szHeader, "FILE-CONTINUE");
        }
        lstrcpy(mp.szFileName, "abc1.jpg");
        mp.dwPackTotal=block;
        mp.dwPackCount=i;
        mp.dwTotalDataSize=msFile->Size;
        mp.dwThisDataSize=((i+1)*MAX_DATASIZE>msFile->Size)?
            msFile->Size-i*MAX_DATASIZE: MAX_DATASIZE;
        msFile->Position=i*MAX_DATASIZE;
        msFile->Read(mp.pbThisData, mp.dwThisDataSize);

        ClientSocket1->Socket->SendBuf(&mp, sizeof(MYPACK));
        Sleep(10);
    }
    delete msFile;
```

4.sever端接收client端送過來的Stream並把裡面的abc1.jpg檔存到
  sever端建立的文件夾裡。

```
{
    MYPACK mp;
```

```
    Socket->ReceiveBuf(&mp, sizeof(MYPACK));

    AnsiString sz=(AnsiString)mp.szHeader;
    if(sz.SubString(1, 4)=="FILE")
    {
        if(sz=="FILE-START")
        {
            gmsFile->Size=mp.dwTotalDataSize;
            gmsFile->Position=0;
        }
        gmsFile->Write(mp.pbThisData, mp.dwThisDataSize);
        if(sz=="FILE-END")
        {
            gmsFile->Position=0;
            gmsFile->SaveToFile(mp.szFileName);
            gmsFile->Clear();
        }
    }
}
```

4.4 介面製作

通訊：

通訊是兩個以上設備之間的溝通，就像設備間、電腦與電腦間或是電腦與設備間經由線路互相傳送資料(交換資料)，其主要的目的為資料的交換，由傳送端將資料經由一定的程序與線路傳送出去，接收端則依其協定將資料收集、儲存或顯示在畫面上，而一個完整的通訊系統包括傳送端、接收端、轉換資料的介面及傳送資料的實際通道(Channel)或媒體(Medium)。

通訊的形式可以區分為兩種，一種為並列傳輸式通訊(Serial Communication)，另一種則為串列傳輸式通訊(Parallel Communication)。

並列傳輸式通訊：

並列傳輸 1 次傳輸 8 個位元，傳輸速率快、效益高，但因資料電壓傳送的過程中，容易因線路的因素而使得電壓準位發生變化，最常發生的是電壓衰減問題及訊號間互相干擾問題(CrossTalk)，而使資料傳輸發生錯誤。如果傳輸距離長的話，這兩個問題會更加明顯，資料的錯誤也就會比較容易發生。另一個缺點是線

路數較多，長距離通訊會增加線路的維護費用，使成本較高。因此並列傳輸只適用於短距離、高速度的環境，例如電腦內部、印表機。

串列傳輸式通訊：

串列傳輸 1 次傳輸 1 個位元，傳輸速率較並列傳輸慢，但因處理的資料電壓只有一個準位，所以比較不容易造成資料的漏失，且因為只使用一條線路，使得維護與線路成本都降低，因此串列傳輸適用於長距離、低速度的環境。

如圖 4-29：

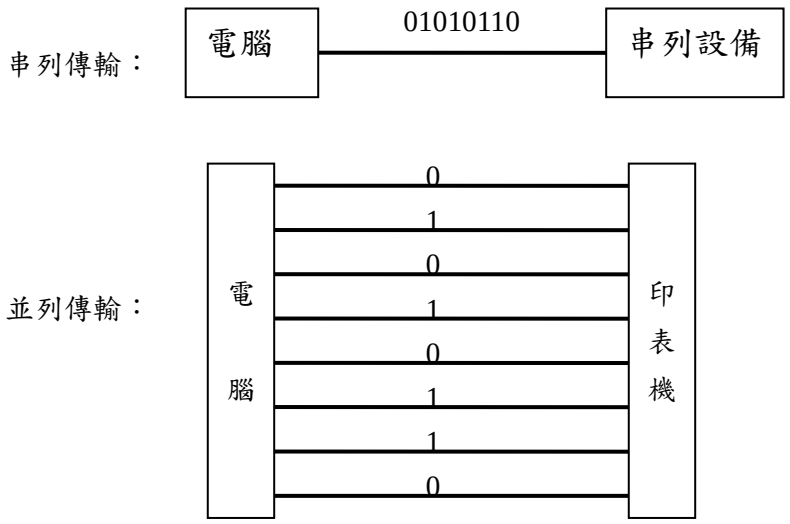

圖 4-29　串列傳輸圖

單工、半雙工與全雙工：

資料傳輸方向可分為單工、半雙工與全雙工傳輸方式。電腦在進行資料傳送與接收過程中，如果資料傳輸只有單一方向時，稱為「單工」，例如：CPU 傳送至列表機。如果資料傳輸可做雙向傳送時，稱為「雙工」，可分為半雙工與全雙工。能夠「同時」做雙向資料的傳送稱為「全雙工」，例如 RS-232。如果資料允許雙向傳送但不能同時進行，而是以傳送端與接收端兩端輪流進行時，稱為「半雙工」，例如：RS-485。

如圖 4-30：

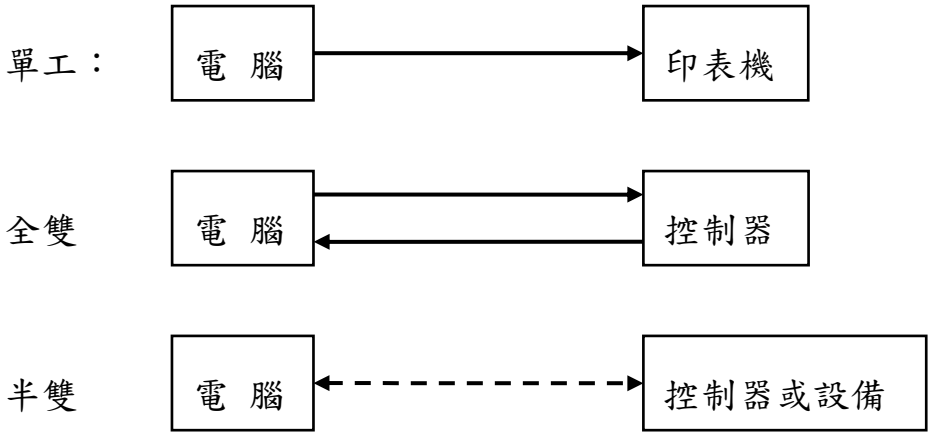

圖 4-30　資料傳輸方式圖

RS - 232：

RS-232 採用串列通訊，其資料傳輸方向為全雙工，最長的傳送距離約 15 公尺，最大傳輸速率為 20kBps(Bit Per Second)。

RS - 485：

RS-485 的訊號傳送時，分成正負兩條線路，當到達接收端後，再將訊號相減還原成原來的訊號，這種作法可有效防止雜訊的干擾。RS-485 也是採用串列傳輸，其資料傳輸方向為半雙工，最長的傳送距離約 1200 公尺，最大傳輸速率為 10Mbps，抗雜訊的能力比 RS-232 強。

RS - 485 轉 RS - 232 :

我們攝影機之間的連結所使用的通訊協定為 RS-485，而攝影機與電腦之間的連結則為 RS-232，因此我們使用了 RS-232/RS-485 轉換器進行通訊協定之間的轉換，讓攝影機可以與電腦溝通並建立連線。以下為意示圖:

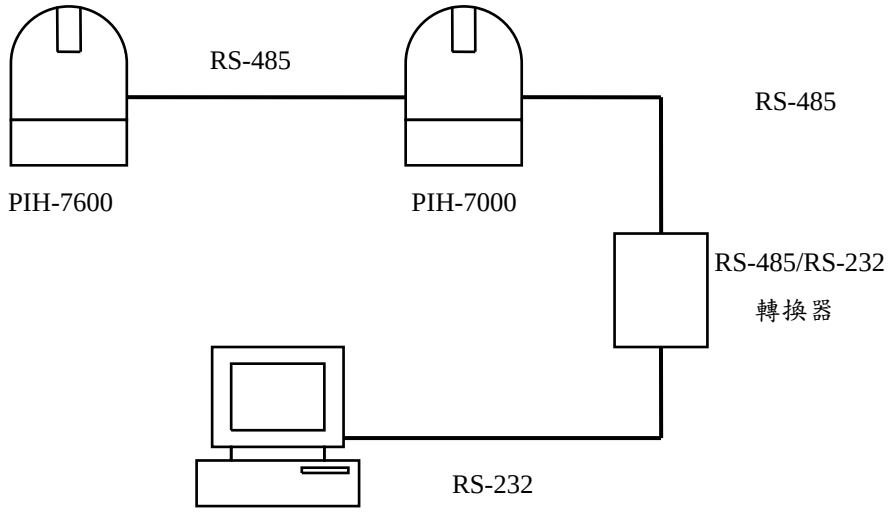

圖 4-31　RS-485 轉 RS-232 意示圖

攝影機控制方式：

攝影機與電腦建立連線後，我們使用攝影機的 protocol 去控制其動作。Protocol 裡包括了許多的 command，讓我們可以使攝影機執行不同的動作，以下將舉一個例子來說明一個動作的開始到結束過程。

First command			Second command		
Byte1	Byte2	Byte3	Byte1	Byte2	Byte3
01h	01h	81h	01h	00h	0FFh
Panning to the right 2 ^{o}			Stop panning status		

當下達一個 command 給攝影機時，它會執行該 command 的動作然後傳回一個 Byte3 為 0FFh 的 command 回來，幫系統收到這個 command 的時候，它就會停止執行該動作。

以下用我們的 Button 來補充其控制方式，Up、Down、Left、Right、Zoom In、Zoom Out，這六個 Button 由我們下 command 使其動作。
控制指令：

Up：
```
ByteSend.Length=3;
ByteSend[0]=0x01;
ByteSend[1]=0x04;
ByteSend[2]=0x98;
```

Down：
```
ByteSend.Length=3;
ByteSend[0]=0x01;
ByteSend[1]=0x08;
ByteSend[2]=0x98;
```

Left：
```
ByteSend.Length=3;
ByteSend[0]=0x01;
ByteSend[1]=0x02;
ByteSend[2]=0x81;
```

Right：
```
ByteSend.Length=3;
ByteSend[0]=0x01;
ByteSend[1]=0x01;
ByteSend[2]=0x81;
```

ZoomIn：
```
ByteSend.Length=3;
ByteSend[0]=0x01;
ByteSend[1]=0x10;
ByteSend[2]=0x0FF;
```

ZoomOut：
```
ByteSend.Length=3;
ByteSend[0]=0x01;
ByteSend[1]=0x20;
ByteSend[2]=0x0FF;
```

NO-KEY command：要停止某個動作的話，在 Byte 3 輸入 0FF。

Byte 1	Receiver Address	001h – 040h
Byte 2	Control Byte 1	Bit 0 = PAN Right Bit 1 = PAN Left Bit 2 = TILT Up Bit 3 = TILT Down Bit 4 = ZOOM Tele Bit 5 = ZOOM Wide Bit 6 = FOCUS Far Bit 7 = FOCUS Near
Byte 3	Control Byte 2	Explain as below (Bit 7 ~ Bit 0)

表 4-32　控制指令表

如表所示，控制 Byte2 的值就可以控制攝影機執行你想要的動作，如果要讓攝影機執行 PAN Right，那就讓 Byte2 的 Bit0 = 1，其他 Bit 為 0，變成 00000001 b，即 01h。又如果要讓攝影機執行 TILT Down，那就讓 Byte2 的 Bit3 = 1，其他 Bit 為 0，變成 00001000 b，即 08h。其他動作以此類推即可。

下表為控制動作速度的指令參數表，我們也可藉由 Byte3 去控制其動作的快慢。

Bit 7 = 1 ; Bit 6 = 0 ; (10XXXXXX)

Bit 5	Bit 4	Bit 3	TILT Speed	PIH-7000/7600
0	0	0	0	0.2 ^{o} /sec
0	0	1	1	2 ^{o} /sec
0	1	0	2	8 ^{o} /sec
0	1	1	3	20 ^{o} /sec
1	0	0	4	50 ^{o} /sec
1	0	1	5	90 ^{o} /sec
1	1	0	6	130 ^{o} /sec
1	1	1	7	180 ^{o} /sec
Bit 2	Bit 1	Bit 0	PAN Speed	PIH-7000/7600
0	0	0	0	0.2 ^{o} /sec
0	0	1	1	2 ^{o} /sec
0	1	0	2	8 ^{o} /sec
0	1	1	3	20 ^{o} /sec
1	0	0	4	50 ^{o} /sec
1	0	1	5	90 ^{o} /sec

1	1	0	6	130 ^{o} /sec
1	1	1	7	180 ^{o} /sec

表 4-33　控制指令表(Speed Change for PAN & TILT)

首先 Bit 7 = 1 , Bit 6 = 0 是固定的，即(10XXXXXX)，其他的六個 Bit 就由我們自己去設定了，可以依自己的想法去給參數，接下來舉例說明如何給。

當攝影機在執行 PAN 的動作時，如果想要其速度為 2 ^{o} /sec，查表的下半部(上半部 TILT 的部份給 0)，可以得到 10000001 b，即 81h。

當攝影機在執行 TILT 的動作時，如果想要其速度為 20 ^{o} /sec，查表的上半部(下半部 PAN 的部份給 0)，可以得到 10011000 b，即 98h。

介面說明：

近端：

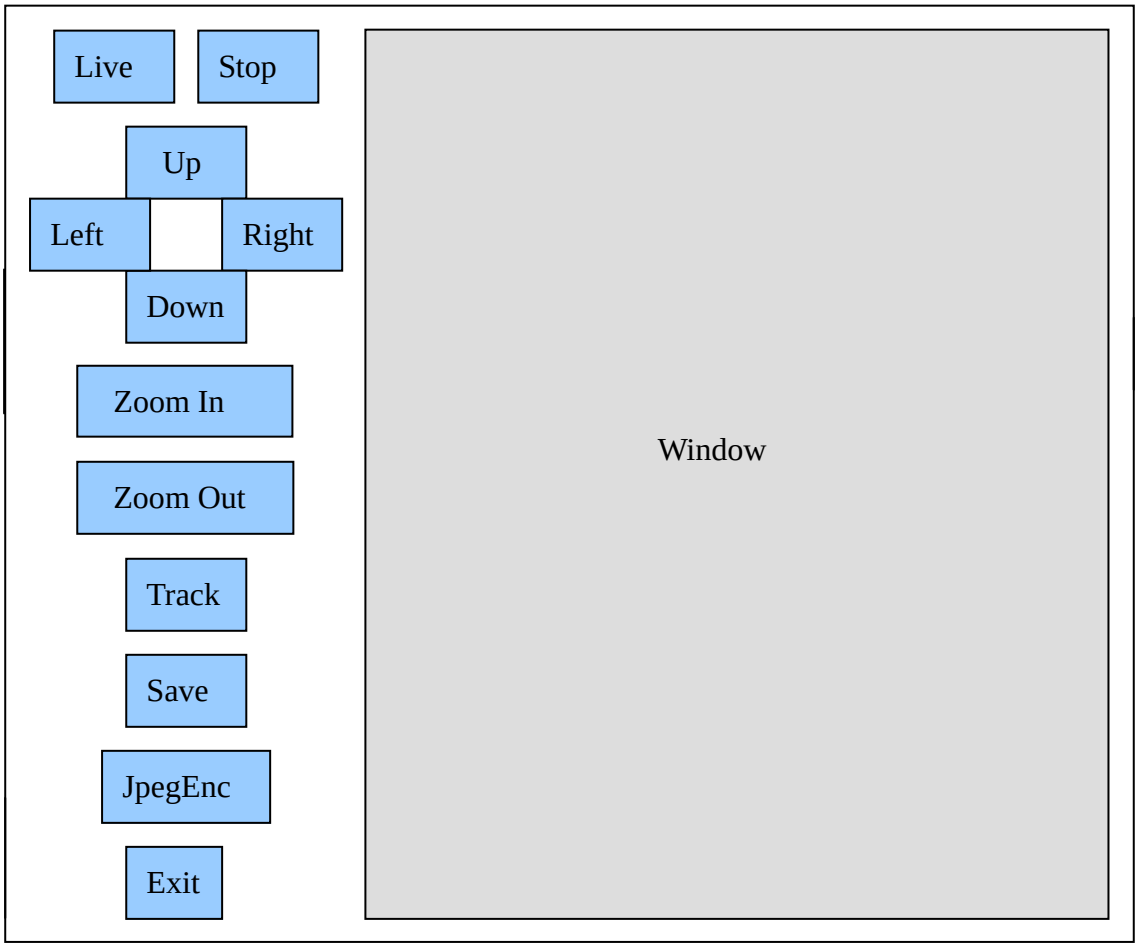

圖 4-34　伺服端介面圖

Button 說明：
Window：呈現影像的視窗。
Live & Stop：影像的抓取跟停止。
方向鍵：控制攝影機的鏡頭方向。
Zoom In & Zoom Out：調整鏡頭控制影像的放大與縮小。
Track：搜尋影像中人臉的位置並框出。
Save：存一張影像。
JpegEnc：把存下來的影像做 Jpeg 壓縮。
Exit：離開介面程式。

BMP 圖檔的結構:

BMP 圖片檔是一種與裝置無關的圖檔 (DIB，device-independent bitmap)，意思是在檔案內包含了顏色的資訊，所以在不同顯示器之下，所顯示的結果仍然相同。BMP 位元圖有兩種格式，一種是 OS/2 格式(註一)，另一種是 WINDOWS 格式。

現在說明的是 WINDOWS 格式，一個 BMP 檔案包含了四部份：BITMAPFILEHEADER、BITMAPINFOHEADER、RGBQUAD 以及位元資料。也有人把 BITMAPINFOHEADER、RGBQUAD 這兩部份合稱為 BITMAPINFO。

(1) BITMAPFILEHEADER:

為一個長 14 位元組的結構體，其定義是：

BITMAPFILEHEADER		STRUC
bfType	DW	4D42H
bfSize	DD	?
bfReserved1	DW	0
bfReserved2	DW	0
bfOffBits	DD	?
BITMAPFILEHEADER		ENDS

表 4-35 BITMAPFILEHEADER 表

BITMAPFILEHEADER 的第一個欄位是長兩個位元組，其意義為 ASCII 碼的『BM』，它代表 BMP 檔的識別字，假如某個檔案的前兩個位元組是『BM』的話，那就有可能是 BMP 位元圖：如果不是『BM』，那就不是 BMP 位元圖。bfSize 是這個位元圖的檔案長度，以位元組為單位。接下來的兩個欄位是保留欄位，必定為 0。最後一個欄位 bfOffBits 是位元資料相對於檔案頭的偏移量，以位元組為單位。

(2) BITMAPINFOHEADER 結構體，其組成欄位是：

BITMAPINFOHEADER	STRUC
biSize DD	?
biWidth DD	?
biHeight DD	?
biPlanes DW	1
biBitCount DW	?
biCompression DD	?
biSizeImage DD	?
biXPelsPerMeter DD	?
biYPelsPerMeter DD	?
biClrUsed DD	?
biClrImportant DD	?
BITMA	
PINFOHEADER	

表 4-36 BITMAPINFOHEADER 表

biSize 是這個結構體的長度，以位元組為單位，對 Windows 格式而言，其大小為 28h。 biWidth、biHeight 分別表示位元圖的寬度和長度，以點為單位。biHeight 需要加以說明一下，biHeight 如果為正值，表示這張位元圖的資料由左下方開始向右，一點一點排列，到最右邊時，再往上一列排；biHeight 如果負值，表示這張位元圖的資料由左上方開始向右，一點一點排列，到最右邊時，再往下一列排。biPlanes 是顏色平面數，這是早期顯示卡硬體尚未發達時遺留下來的，現在已廢棄不用，必須設為 1。

biBitCount 表示每一點需要用多少位元，只可以是 1、4、8、16、24 或 32 這些數值(註二)。biCompression 是壓縮方式，這種壓縮方式是不失真的，和 JPEG 失真壓縮不同。biCompression 可以是 BI_RGB、BI_RLE8、BI_RLE4、BI_BITFIELDS 值，其中 BI_RGB 是完全無壓縮的，也是最常用的。biSizeImage 是 BMP 檔的長度，以位元組為單位。

biXPelsPerMeter、biYPelsPerMeter 是表示這張圖的解析度，以每公尺有多少點，前者是寬的解析度，後者是指高的解析度 (這裏的解析度和一般螢幕的解析度意義稍微不同)。

最後兩個欄位 biClrUsed、biClrImportant 是當 biBitCount 不是 1、4、8、16、24 或 32 這些標準的數時使用的。

(3) 色彩對照表

它是由許多數目不定的 RGBQUAD 結構體組成的，RGBQUAD 結構體各欄位如下：

RGBQUAD		STRUC
rgbBlue	db	?
rgbGreen	db	?
rgbRed	db	?
rgbReserved	db	0
RGBQUAD	ENDS	

表 4-37　色彩對照表

RGBQUAD 的功用與 OS/2 格式的 RGBTRIPLE 相同，請參考(註三)，但是多了一個保留位元組，rgbReserved，這個位元組均設為零，有了這個位元組，使得色彩對照表上的每個 RGBQUAD 結構體長度均為 32 位元，可以使 32 位元的 CPU 存取時得到較好的效率。

(4) 位元資料，也是存有整張圖片的圖案資料存放處。

註一：OS/2 是 IBM 出品的一種作業系統，早期與微軟共同開發，以期能取代 DOS，成為第二代作業系統，這可以由其名稱，Operating System/2 看出。後來由於 Windows 3.0 的成功，微軟決定開發下一代 Windows 95 而退出 OS/2 的開發，最後 OS/2 由 IBM 獨自開發並銷售。

註二：Windows 的 BMP 檔案結構與 OS/2 的 BMP 檔案結構類似，因為 Windows 的 BMP 檔結構是由 OS/2 擴充而來的。OS/2 的 BMP 檔案結構的第一部份是由 14 個位元組組成的 BITMAPFILEHEADER 結構體，其意義與 Windows 格式的相同。在 BITMAPFILEHEADER 結構體之後的第二部份是由 12 個位元組組成的 BITMAPCOREHEADER 結構體：

BITMAPCOREHEADER		STRUC
bcSize	dd	0ch
bcWidth	dw	?
bcHeight	dw	?
bcPlanes	dw	1
bcBitCount	dw	?
BITMAPCOREHEADER		ENDS

表 4-38　BITMAPCOREHEADER 表

第一個欄位，bcSize，指的是這個結構體的大小，以位元組為單位，因此其數值是 0ch，它可以用來分辨是 OS/2 格式或是 Windwos 格式的 BMP 檔。第二、三個欄位，bcWidth、bcHeight 是這個位元圖的寬度和長度，以點為單位。bcPlanes 是顏色平面數，也和 Windows 格式一樣，應該設為一。

最後一個欄位，bcBitCount，是每個點所佔用的位元數，可以是 1、4、8、24。如果是 1，表示每點僅佔用一個位元，一個位元可以是 0 或 1，因此可以表示兩種顏色；如果是 4，那麼會有下面 16 種情形：

0000	0001	0010	0011	0100	0101	0110	0111
1000	1001	1010	1011	1100	1101	1110	1111

表 4-39　bcBitCount 欄位表

可以用 $2^4=16$ 計算，如果每一種數值代表一種顏色，那就能表示 16 種顏色。其餘顏色如下表：

bcBitCount	計算	顏色數	RGBTRIPLE 結構體數目
1	2^1	2	2
4	2^4	16	16
8	2^8	256	256
24 (True-Color)	2^{24}	16777216	0

表 4-40　bcBitCount 欄位表(2)

註三：第三部份是一個由許多 RGBTRIPLE 結構體所組成的色彩對照表，顧名思義，它是記錄色彩用的，它的長度不固定，與顏色數有關。先看它的組成單位，RGBTRIPLE 結構體，由其名字就可猜想得到，RGBTRIPLE 是一個由紅綠藍三原色光所組成代表各種顏色強弱的結構體：

RGBTRIPLE		STRUC
rgbBlue	db	?　　;藍色光強度
rgbGreen	db	?　　;綠色光強度
rgbRed	db	?　　;紅色光強度
RGBTRIPLE		ENDS

表 4-41　RGBTRIPLE 結構表

參考上表，如果 bcBitCount 為 1 時，色彩對照表有兩個 RGBTRIPLE 結

構體；如果 bcBitCount 為 4 時，色彩對照表有 16 個 RGBTRIPLE 結構體……，這應不難理解。舉例來說， bcBitCount 為 4 時，可以表示 16 種顏色，但是是那 16 種顏色呢？就由色彩對照表記錄，每個 RGBTRIPLE 結構體表示一種顏色，這種顏色是由三原色光強弱不同而產生，因此 16 種顏色就得有 16 個 RGBTRIPLE 結構體記錄。

擷取卡控制方式：

在 Windows NT/98/2000/XP 這 4 種作業系統下，這張擷取卡可以接受的語言有 VC/BCB/VB/Delphi，而我們選擇用 BCB 來控制它，因為我們在介面的部份是使用 BCB，使用同樣的語言在整合比較不容易有衝突。在這 4 種語言下，可以利用 Function Library 來發展程式，而我們使用到的 Function Library 有：

Configuration、Image Grabbing、Callback 和 Frame Buffer。這張擷取卡共有 4 個 Port (Port ID = 0~3)，我們使用 Port 0 當我們的輸入；在 Video Format，設定為 NTSC；在 Video Size，我們設定為 640x480 for NTSC 的畫面；在 Color Format，設定為 RGB24。

第五章 開發環境及設備介紹

我們在人臉追蹤、影像壓縮的執行使用 Microsoft Visual C++ 6.0，再將整合的程式轉變成 Borland C++ Builder 所使用的規格，而 Borland C++ Builder 簡稱 BCB，是內建許多視覺化元件(Visual Component Library，VCL)的整合開發環境(Integrated Development Environment , IDE)，所謂整合性開發環境是說在同一個視窗即擁有程式的編寫、編譯、連結與執行所需要的功能選單或按鈕，其優點是容易學習且容易開發。

將各種程式運用所需的功能，簡化成元件的方式，資料庫連線，建立成按鈕的型式。在我們專題中，我們建立了許多按鍵，例如:控制攝影機的上下左右，放大縮小等等，非常便利。

我們使用攝影機進行人臉追蹤，從已建立好的介面控制攝影機的方向及追蹤，而攝影機我們使用寬動態快速球型攝影機，它提供了更新的功能及更方便的操控，且能持續作 360 度高速旋轉，運轉速度從每秒 0.18 度到每秒 360 度，故能確保直接且正確的捕捉監控目標位置。

第六章 問題與檢討

在這次的專題研究當中，遇到了程式知識缺乏及整合的問題，藉由這次的專題研究，慢慢的理解程式的運用及整合的方法。

將所使用的人臉追蹤，影像壓縮在 Microsoft Visual C++ 6.0 底下作用，寫出我們所需要的型式，並且把此程式運用到 Borland C++ Builder，出現了許多問題，因為 Microsoft Visual C++，所使用的 library 不能直接在 Borland C++ Builder 編譯完成，我們必須先將 library 轉變成 Borland C++ Builder 所可以使用的 library。

在影像控制及人臉追蹤方面，產生臉部顯示框重複的寫入影像，以及 CPU 使用過大的問題，因為原本人臉追蹤是獨立顯示在另外一個視窗，使得 CPU 使用過大，產生當機。所以我們改用直接將顯示框劃入影像當中而不另外開啟一個視窗。

整合的部份發生了互不相容的問題，我們查閱了許多有關 Borland C++ Builder 的書，了解 BCB 指令以及設定，將其程式整合完成。

第七章 參考文獻

[1] 鐘國亮，資料壓縮的原理與應用，第二版.全華科技圖書出版 2004 年 10 月

[2] 黃嘉輝，2002，Visual Basic 網際網路程式設計-TCP/IP 與 Internet Programming 篇最新版，文魁資訊股份有限公司。

[3] K. Konstantinides， V. Bhaskaran，and G.. Beretta，"Imange sharpening in the JPEG domain，" IEEE Trans. On Image Processing，8(6)，1999，PP. 874-878

[4] 鐘國亮，離散數學，全華科技圖書，台北，2003

[5] Winsock2 網路程式設計實用教程 (<松崗> 李凌 編著)

[6] 李振輝、李仁各編著，《探索圖像文件的奧秘》，清華大學出版社，1996 年

[7] 黎洪松、成實譯《JPEG 靜止資料壓縮標準》，學苑出版社，1996 年

[8] Tom Lookabaugh. Dave Lindbergh.著 李煜暉等譯 《多媒體數位壓縮原理與標準》 電子工業 出版社 2000 年 8 月

[9] 黃賢武 王加俊 李家華 《數位圖像處理與壓縮編碼技術》 電子科技大學出版社 2000 年 12 月

[10] 戴顯權著，「資料壓縮」，紳藍出版社，台北市，2003 年。

[11] 施威銘研究室(1992) ：PC 影像處理技術<一> 圖檔壓縮續篇旗

[12] 施威銘研究室(1992) ：PC 影像處理技術<二> 圖檔壓縮續篇

[13] 私立中原大學電子工程學系碩士論文：具備人臉追蹤辨識功能的一個智慧型數位監視系統(A Smart Digital Surveillance with Face Tracking and Recognition Capability)，指導教授：繆紹綱博士，研究生：黃泰祥。

[14] 研究者：李姿慧，人臉追蹤應用於監控系統之研究(The study of Monitoring and Control System Using Human Face Tracking Methods)

[15] 研究者：Gary R. Bradski, Microcomputer Research Lab, Santa Clara, CA, Intel Corporation，電腦視覺之人臉追蹤應用於感知使用者介面(Computer Vision Face Tracking For Use in a Perceptual User Interface)

[16] Gary R. Bradski, Microcomputer Research Lab, Santa Clara, CA, Intel Corporation" Computer Vision Face Tracking For Use in a Perceptual User Interface",Intel

[17] 研究生：林煌山 指導教授：范國清 教授國立中央大學資訊工程研究所碩士論文，利用膚色及區域極小值作人臉特徵是否遮蔽之偵測判斷，Detection of Facial Occlusions by Skin-Color based and Local-Minimum based Feature Extractor

[18] 研究生：熊昭岳，指導教授：蘇木春 博士行車安全偵測系統 (An Attention Detection System for Vehicles) 部分人臉偵測內容

[19] 計畫主持人：蔡建戊 教授 計畫參與人員：周士揚 教授、陳奐彰 教授 使用臉控式人機介面之遠端即時監控系統之設計與實作

[20] 研究生：鄭凱方，指導教授：范國清 教授 國立中央大學資訊工程研究所碩士論文，人臉可辨識度計算用於監控系統中人臉正面最佳影像判定，Face Recognizability － best face shotcandidate for surveillance system

[21] Computer Vision Face Tracking For Use in a Perceptual User Interface Gary R. Bradski, Microcomputer Research Lab, Santa Clara, CA, Intel Corporation

[22] 專題學生： 康煥堯，大頭照的影像辨識，大同大學，資訊工程學系專題報告，中華民國九十六年六月
[23] 陳必衷，Four Approaches to Face Detection
[24] Lilin-FastDome_ver7.pdf
[25] Visual Basic 與 RS-232 串列通訊控制最新版 <文魁資訊>
[26] Pocket PC無線網路與RS-232 <文魁資訊>