# 即時車牌辨識與通報系統之研究

## A Study On The Real-Time License Plate Recognition And Notification System


| 劉鎮城 | 俞博文 | 張凱程 | 莊緒聰 | 陳建全 |
|---|---|---|---|---|
| 台中技術學院 | 台中技術學院 | 台中技術學院 | 台中技術學院 | 台中技術學院 |
| 資訊管理系 | 資訊管理系 | 資訊管理系 | 資訊管理系 | 資訊管理系 |
| ccliu@ | s1494B031@ | s1494B034@ | s1494B038@ | s1494B039@ |
| ntit.edu.tw | ntit.edu.tw | ntit.edu.tw | ntit.edu.tw | ntit.edu.tw |



## 摘要

國內交通環境日益惡化，違規停車的事件頻繁，因此需要一個具有機動性的即時通報系統，所以本研究提出讓使用者透過 3G 照相手機取得車輛照片，並藉由 3G 行動網路傳到伺服器做車牌辨識，取得違規車輛的車號，透過此車號到資料庫中尋找車主的聯絡資料，接著伺服器會替使用者轉送簡訊及 e-mail 給車主，以達到通報的目的。在伺服端車牌辨識的部分，可分為車牌定位、車牌矯正、字元切割、字元正規化、字元辨識五部分。

研究以 455 張地下停車場與室外所擷取的車牌影像，做車牌辨識之實驗，經實驗結果顯示，本系統的辨識成功率為 83%，辨識時間每張平均為 1.5912 秒，而本系統使用 Java 的 byte stream 技術來降低影像傳輸時間，系統的平均反應時間為 5.94 秒。


**關鍵詞**： 車牌辨識、第三代行動網路


## Abstract

In Taiwan, the transportation environment is getting worse and worse. Illegally parked cars are frequent. Consequently, the real-time notification system with the function of mobility is needed. In this paper we propose a scheme where users may take pictures of license plates with a 3G cell phone and send the pictures to the server through 3G Mobile-Internet. The pictures are then transmitted to the server through 3G Mobile-Internet. The server then performs license plate recognition and uses the license plate number to locate the car owner, relevant phone number and e-mail address from the database. A message is then delivered and emailed to the car owner. The license plate recognition system is divided into five parts: license plate localization, license plate calibration, character segmentation, character normalization and character recognition.

To perform the experiments we took 445 license plate pictures from both outdoor and underground parking lots. Experiment results showed that the average recognition rate is 83%, the average recognition time of each image required 1.5912 seconds. We also used the Java byte stream technique to reduce system response time which came out to be 5.94 seconds.


**Keywords** : license plate recognition, 3G Mobile-Internet

## 1.前言

今日社會裡，汽機車已成為國人的主要交通工具，不過汽機車帶給大眾方便的同時，也造成相當多的問題，例如超速、闖紅燈，政府為了解決這類問題，在路口設置捕捉違規影像的攝影裝置，再把違規影像交由人來核對車號，最後經由核對出的車號來查出違規的駕駛人，這整個流程不僅費力且費時。

為了節省人力和時間，研究學者開始開發自動辨識車牌系統，初始階段，因軟硬體設備的限制，研究進度受阻，不過近來軟硬體設備的效能大幅提升，使車牌辨識的技術日漸成熟，於是企業及政府將車牌辨識系統應用於停車場或是十字路口，此外還應用於高速公路的電子收費站，它儼然已成為智慧型運輸系統(ITS)的一環[1]。

根據內政部警政署統計，從民國 97 年 1 月到 10 月底，台灣違規停車事件已經超過 151 萬件[2]。這些違規停車的駕駛人因貪圖一時便利，而把車停放在別人的進出通道上，影響附近居民的進出。因此要如何儘快通知車主，將是解決問題的關鍵。

傳統的車牌辨識系統常應用在停車場入口管制、高速公路收費等，但這些應用大多是定點式拍攝，無法應用於隨機發生的違規停車事件上，不過隨著資訊技術的日益精進，手機基本上都擁有照相、上網功能，甚至使用者還可以透過 Java 在手機上開發小程式，上網方面，根據資策會網站的資料顯示，2008 年第二季台灣的行動上網總用戶數約為 1338 萬戶，占行動通信用戶比例提升至 54.2%，比較各行動上網技術佔有率，目前 3G 數據用戶占總體 63.9%，已大幅領先 GPRS 的 24.8%，3G 數據服務已成為行動上網的主流技術[17]，而且近期電信業者在推廣 3G 手機，3G 網路的傳輸費率越來越便宜。有鑑於此，本研究提出以照相手機透過 3G 網路來結合車牌辨識，如此將可以使車牌辨識系統具有機動性，擴展車牌辨識的應用範圍。

車牌辨識系統與行動裝置整合，提高了車牌辨識系統的機動性，不管違規停車事件發生在何處，只要隨身攜帶 3G 照相手機，使用者皆能使用它來擷取違規車輛影像，並透過行動 3G 網路來傳送此影像到伺服器端，到了伺服端則做車牌辨識處理產生車號，接著伺服端再利用辨識出來的車號去資料庫中查詢車主的聯絡資料，因此使用者可透過伺服器來轉送簡訊和 e-mail 給車主，不但快速又方便，而且能即時通報駕駛人，以此點看來，本系統已達到了即時性及機動性兩大目標。

本研究提出利用第三代行動網路(third generation, 3G)來傳送違規車輛之車牌影像，而影像來源則是透過手機照相擷取，這對車牌辨識系統是一種新型態的應用。即時傳送車牌影像，並在電腦伺服端接收影像進行車牌辨識，並把辨識結果與資料庫進行比對，查出與車號相符的車主聯絡資料，進而對車主發出簡訊及 e-mail。另外本研究針對目前車牌辨識方法加以分析，並實際利用這些方法加以改良來完成本系統。

本篇論文的架構，第二章是介紹車牌辨識的相關研究及探討各行動上網技術的速度與價格，第三章則介紹本系統車牌辨識的研究方法，第四章介紹使用的實驗設備及系統的反應時間，並把車牌辨識部分單獨測試，列出它的辨識時間及辨識率，第五章則是評估實驗結果及探討系統未來發展。

## 2.文獻探討

近年來有越來越多的人研究車牌辨識系統，在國內外都可以找到相關的研究論文，車牌辨識系統大體上可分為三部分—車牌定位、字元切割與字元辨識。

### 2.1 車牌定位之相關研究

車牌定位分為連通物件法與空間頻域分析法，連通物件法是先將影像依灰階變化做區分，將相鄰像素的物件標示起來，再做條件判定，如車牌長寬比，挑出最相近於車牌的物件。由於車牌是由許多字元組合而成，所以車牌的灰階值分佈密集且變化急遽，空間頻域分析法藉由車牌的這個特性，找出車牌的位置，再利用條件判斷，如車牌長寬比來擷取車牌。

陳翔傑[3]先利用中值濾波器來消除雜訊干擾，透過 Prewitt 與 Sobel 遮罩來偵測影像中的邊緣點，再定義規則將邊緣點串接成線段，挑選出可能為車牌的區塊，最後利用 4 鄰點或 8 鄰點的連通物件法將車牌的可能區塊擷取出來。王精忠[4]利用 Sobel 遮罩偵測車牌的邊緣，再利用垂直投影法與水平投影法投射出影像波峰(peak)圖，再比較波峰區塊的長寬比，將最接近車牌比例的區塊擷取出來。王振興[5]主要是利用邊緣偵測，將邊緣與邊緣之間的縫隙用型態學(morphology)中的膨脹(dilation)填補起來，透過連通物件法將連接物件標記出來，再利用條件篩選，如車牌的長寬比例，挑出有可能為車牌的物件。

至於背景複雜度的影像定位，余忠潔[6]利用 H(影像色調)S(飽和度)I(強度)，並配合邊緣偵測，再對影像做模糊處理，使車牌定位不受背景複雜度的影響。另外利用影像直方圖(Histogram)統計資料，建立可調式二值化門檻值，防止光線強弱問題造成定位失敗。上述的方法主要是對標準車牌做處理，如果遇到歪斜的車牌，則必須先對車牌做正規化、利用數學函數做角度轉移，再做處理。

Ondrej Martinsky[13]將霍夫轉換(Hough transform)演算法應用在偵測車牌歪斜角度，再將歪斜角度套用到旋轉的數學函式，矯正歪斜的車牌。

本論文車牌定位的方法為空間頻域分析法，本論文的重點在於只針對影像做垂直邊緣偵測，使用的遮罩為 Sobel 遮罩，由於 Sobel 水平邊緣雜訊太多，會增加車牌定位上的困難，因此本論文不做水平邊緣偵測。車牌矯正方面，本論文使用霍夫轉換演算法將傾斜的車牌矯正。

### 2.2 字元切割與辨識之相關研究

常見的字元切割方法為連通物件法[7,15]與投影法，投影法就是預設字元的寬度，先利用垂直投影找出字元的邊界，再用水平投影將字元切割出來[4,9]。投影法較容易受到車牌歪斜的影響而造成切割失敗；連通物件法的優點為不受車牌歪斜的影響，還是能將字元當作獨立物件切割出來，缺點為只要字元有筆劃斷掉，連通的物件不完整，切割出來的字元也會不完整，不過只要利用型態學中的膨脹技巧，將斷裂的部分補齊即可。

由於連通物件法對於傾斜的車牌有良好的切割效果，因此本論文字元切割使用連通物件法，本論文使用四連通將物件標記出來，再使用 Bounding box 法找出相同標記的最小及最大的 x、y 座標，配合車牌字元特徵，如字元長寬比，將車牌字元擷取出來。

字元辨識方面，有類神經網路法、樣板比對法、統計法、結構法等，由於類神經網路具有抗雜訊、高容忍性等優點，因此被提出後，越來越多人採用類神經網路法做字元辨識。

張簡子介[8]將類神經網路法中的小腦模型演算法配合樣板比對法，藉以彌補樣板比對法缺乏適應性的缺點，但對於一些相近字元，如 0、D，1、I 仍然會辨識失敗，因此必須對這兩組字元做再辨識。蔡銘鑫[10]將倒傳遞類神經網路與樣板比對法整合，互相彌補兩者的優缺點。呂炎州[11]以字元的筆劃當作特徵值，建立 35 個標準字的筆劃特徵表，再與特徵表中的特徵值逐一比對。

本論文字元辨識方法使用樣本比對法[3]比對欲辨識字元與樣本字元相同座標上的二元值，如果相同，則權重值加 1，比對下來權重值最大的字元即是辨識結果。由於有些字元特徵相似，如 0、D，6、8、9、S 等，權重值也會相似，因此必須對這些字元做再辨識，本論文再辨識的方法是根據字元局部性的差異來做偵測。

### 2.3 行動通訊網路比較

#### 2.3.1 GPRS 和 3G 價格之比較

目前市面上手機上網費率的計算都是以封包為單位。以下介紹如何以電腦單位來計算封包：

每一封包為 128bytes，不足則以一封包計。本研究以傳一張大小約 80KB 的 JPG 圖檔為例，而 80KB 傳到 Server 端需要傳 640 封包，以下分別比較 GPRS 和 3G 手機行動上網的費率。

以下方案假設成本為 250 元，費率以中華電信為例，本研究以月租費 0 元的 GPRS 與月租費 183 元的 3G 方案做比較。

**表 2.1　GPRS 和 3G 費率之比較**

| 選擇方案 | 通信費 (元/封包) | 可傳 封包數 | 可傳張數 |
|---|---|---|---|
| GPRS | 0.03 元 | 8333 | 13 張 |
| 3G 183 | 0.005 | 36600 | 57 張 |

依表 2.1，本研究選擇價錢較便宜，且可傳送更多張圖的 3G 183 方案。

#### 2.3.2 GPRS 和 3G 速度比較

表 2.2 提供以 GPRS、3G 及 3.5G 來傳送一張檔案大小為 80KB 的圖檔，分別列出所需的時間。

**表 2.2　GPRS、3G 及 3.5G 的速度比較**

| 型態 | 速率(單位：bit) | 時間 |
|---|---|---|
| GPRS | 170 Kbps | 3.76 秒 |
| 3G | 384Kbps | 1.6 秒 |
| 3.5G | 3.6Mbps | 0.1736 秒 |

依表 2.2，可看出傳輸速率最快的是 3.5G，其次是 3G，因 3.5G 尚未普及，所以本研究選擇 3G 來傳送擷取的影像。

## 3.研究方法

### 3.1 即時車牌辨識與通報系統架構設計

本研究之即時車牌辨識與通報系統主要是撰寫 JavaME 應用程式在手機上執行，使用者使用此應用程式即可透過手機進行拍照，與伺服器進行連線將車牌影像傳給伺服器，接著伺服器將辨識結果回傳使用者，使用者確認無誤後，伺服器將違規訊息透過台灣簡訊 SMS Server[16]與 Mail Server 傳送給車主，系統運作流程如圖 3.1，系統架構如圖 3.2 所示：

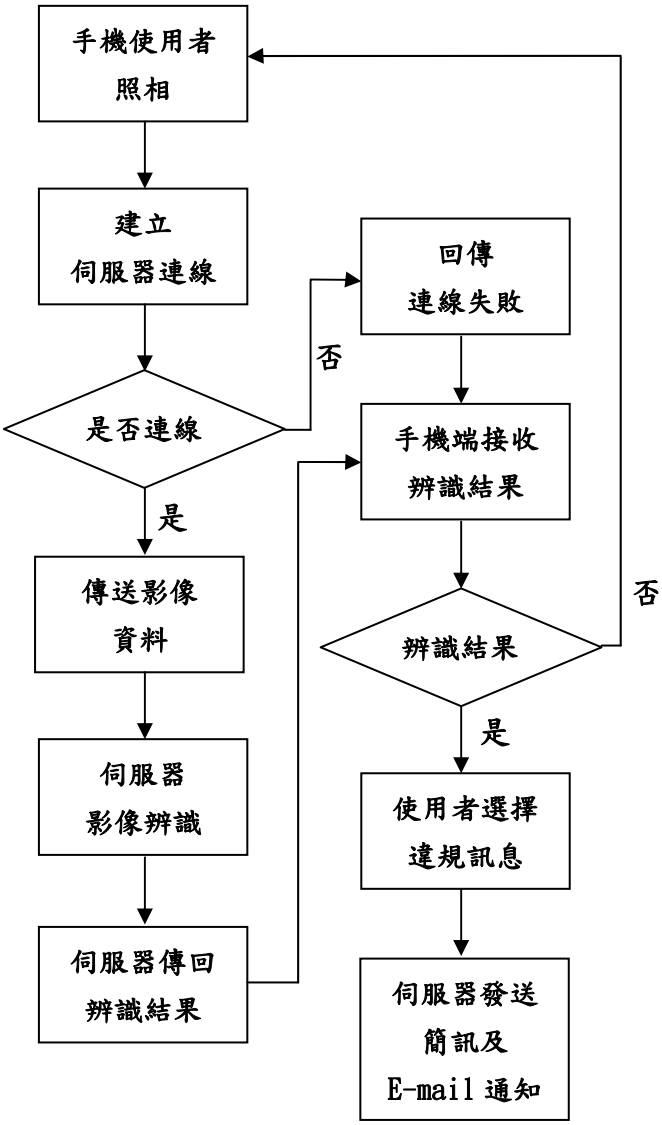



**圖 3.1　系統流程圖**

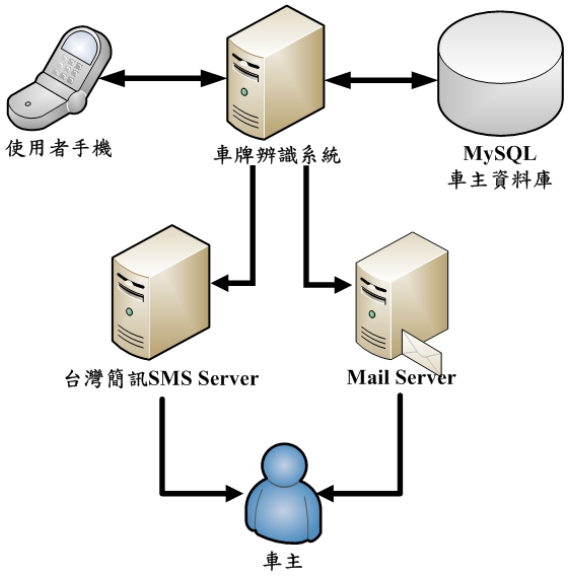



**圖 3.2　系統架構圖**

### 3.1.1 手機與系統連線

手機的照相功能我們參考 SonyEricsson 所開放的照相程式碼撰寫而成。為了確保影像傳輸的正確性，在手機與伺服器的連線上，本系統採用 TCP 通訊協定進行連線與影像傳輸；而為了加快系統的反應時間，本研究不把拍照取得的影像做存檔處理，直接以串流的方式傳送給伺服器，一來加快系統反應速度，二來使用者不需再手動將車牌影像刪除，增加系統使用的方便性。

影像的傳輸以串流為主，為了可以不用存檔，本系統使用 Java 的 ImageIO.read 這個物件方法，來達成將影像串流以不存檔、讀檔的方式直接轉換成 BufferedImage 物件供車牌辨識部分操作。本系統與手機端連線時以 DataInputStream 物件接收串流，假如直接讓 ImageIO.read 接收 DataInputStream 的影像串流，將花費較長的時間，以 45KB 的影像來為例，平均花費時間為 0.7671 秒；為了縮減時間，本研究使用 ByteArrayInputStream 來加快傳輸時間，先把 DataInputStream 的影像串流資訊轉成 byte 陣列，然後把 byte 陣列當成 ByteArrayInputStream 的緩衝陣列 (buffer array) ，讓 ImageIO.read 接收此串流，經過系統測試結果，45KB 的影像平均花費時間為 0.0687 秒，比直接使用 DataInputStream 處理時間快 11 倍。

## 3.2 車牌辨識

車牌辨識系統的流程分為五個部分，如圖 3.3 所示：

1. 車牌定位：從影像中將可能是車牌的區域定位出來。
2. 車牌矯正：針對已定位的車牌進行車牌是否歪斜的判定與矯正。
3. 字元切割：進行擷取車牌字元。
4. 字元正規化：針對字元的大小做統一正規化。
5. 字元辨識：針對車牌字元進行辨識。

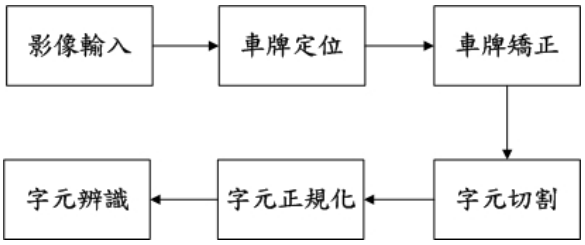



**圖 3.3 車牌辨識流程圖**

### 3.2.1 車牌定位

#### 3.2.1.1 前置處理

步驟一：先將輸入的車牌如圖 3.6(A)進行灰階化處理如圖 3.6(B)，公式如下：

$$Y(x,y) = 0.299 \times R + 0.587 \times G + 0.114 \times B$$

**公式 3.1 灰階化公式**

步驟二：對灰階化後的影像作 3 次平滑化處理如圖 3.6(C)，使用遮罩如圖 3.4。

| | | |
|---|---|---|
| 0.11 | 0.11 | 0.11 |
| 0.11 | 0.12 | 0.11 |
| 0.11 | 0.11 | 0.11 |

**圖 3.4 平滑化遮罩**

步驟三：針對平滑化後的影像做一次銳化處理如圖 3.6(D)，使用遮罩如圖 3.5。

| | | |
|---|---|---|
| -0.11 | -0.11 | -0.11 |
| -0.11 | 1.88 | -0.11 |
| -0.11 | -0.11 | -0.11 |

**圖 3.5 銳化遮罩**

(A) 原圖

(B) 灰階圖

(C) 平滑三次

(D)銳化

**圖 3.6 定位前置處理**

### 3.2.1.2 邊緣偵測

影像前置處理完成後，接下來將對影像做邊緣偵測處理，本系統所使用的邊緣偵測遮罩為 Sobel 遮罩[12,14]如圖 3.7：

左邊界遮罩

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

右邊界遮罩

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

**圖 3.7 Sobel 遮罩**

由於水平邊緣在影像中出現太多次如圖 3.8(C)，不利於車牌在影像中位置的判定，故本系統只針對影像做垂直邊的邊緣偵測。

影像分別經過上述遮罩處理後，會找出影像的左邊界如圖 3.8(A)和右邊界如圖 3.8(B)，再取這兩張圖相同位置像素值的最大值來跟固定門檻值 160 做比較，若比固定門檻值大，則設像素值為白色(255)，反之則設像素值為黑色(0)如圖 3.8(D)。

(A) 左邊界

(B) 右邊界

(C) Sobel 水平邊

(D) Sobel 垂直邊

**圖 3.8 邊緣偵測**

### 3.2.1.3 車牌垂直位置探索

做完垂直邊緣偵測後，使用垂直投影法來計算出影像在 Y 軸第 j 列的所有像素值總和 $P_y(j)$，公式如下：

$$P_y(j) = \sum_{i=0}^{w-1} f(i,j)$$

**公式 3.2 垂直投影**

定義：$P_y(j)$為一陣列，$\{j \mid 0 \le j \le (h-1)\}$，h 為影像的高度。

然而$P_y(j)$所統計出來的值可能分佈太過離散，故針對$P_y(j)$採用中值濾波(median filter)[12,14]的原理來降低離散的程度，本系統所使用的視窗大小是 9。

經過處理後，$P_y(j)$的值會變成$P_y'(j)$，透過$P_y'(j)$來探索影像中車牌的垂直位置(y 座標)。從$P_y'(j)$中尋找波峰，波峰可能就是車牌所在的區域，找出$P_y'(j)$中的最大值。

尋找車牌垂直位置的演算法如下：

步驟一： 複製$P_y'(j)$，令它為$cP_y'(j)$。

步驟二： 在陣列$cP_y'(j)$中找出最大值 maxY，其索引值為$y_{pm}$。

步驟三： 利用步驟二的$y_{pm}$和 maxY 找出車牌的最高點$y_{top}$和最低點$y_{bottom}$，計算如下：令$r = y_{pm}$為起始值，開始讓$r = r-1$，直到$P_y'(r) < maxY \times C$條件或$r = 0$條件成立為止，就讓$y_{top} = r$，C 為求波谷的常數，本研究預設為 0.38；相同地，$r = y_{pm}$為起始值，使$r = r+1$，直到$P_y'(r) < maxY \times C$條件或$r = h-1$條件成立為止，令$y_{bottom} = r$。

步驟四： 找出 $y_{top}$ 和 $y_{bottom}$ 後，則把$(y_{top}, y_{bottom})$所對應的值皆設為 0，若$y_{bottom} - y_{top} > 20$時，則找出一組車牌 y 座標的候選區，重複步驟二，直到$cP_y'(j)$的最大值為 0 或有 3 組$y_{top}$和$y_{bottom}$時結束。

經過上述演算法可找到一組以上車牌可能位於影像中的 y 座標位置區間，實作結果如圖 3.9(A)，圖 3.9(B)是從第一組 y 區間切出來的圖，圖 3.9(C)是第二組，圖 3.9(D)是第三組。


**(A)三組$y_{top}$和$y_{bottom}$**


**(B)第一組$y_{top}$和$y_{bottom}$**


**(C)第二組$y_{top}$和$y_{bottom}$**

**(D)第三組$y_{top}$和$y_{bottom}$**

**圖 3.9 垂直位置探索**

### 3.2.1.4 車牌水平位置探索

當垂直位置的候選區產生後，使用水平投影法來計算出影像在 X 軸第 i 行的所有像素值總和為$P_x(i)$，公式如下：

$$P_x(i) = \sum_{j=y_{top}}^{y_{bottom}} f(i,j)$$

**公式 3.3 水平投影**

定義：$P_x(i)$為一陣列，$\{i \mid 0 \le i \le (w-1)\}$

w 為影像的寬度。

$P_x(i)$所統計出來的值可能分佈太過離散，故針對$P_x(i)$採用均值濾波(average filter)[12][14]處理來降低離散的程度，本系統所使用的視窗大小是$y_{bottom} - y_{top}$。

經過處理後，$P_x(i)$的值會變成$P_x'(i)$，透過$P_x'(i)$來探索影像中車牌的水平位置(x 座標)。從$P_x'(x)$中尋找波峰，波峰可能就是車牌所在的區域，找出$P_x'(i)$中的最大值。

尋找車牌水平位置的演算法如下：

步驟一： 複製$P_x'(i)$，令它為$cP_x'(i)$。

步驟二： 在陣列$cP_x'(i)$中找出最大值 maxX，其索引值為$x_{pm}$。

步驟三： 利用步驟二的$x_{pm}$和 maxX 找出車牌的左邊界$x_{left}$和右邊界$x_{right}$，計算如下：以$r = x_{pm}$為起始值，使$r = r - 1$直到$P_x'(r) < maxX \times C$條件或$r = 0$成立為止，就讓$x_{left} = r$，C 是求波谷的常數，本研究預設為 0.5；相同地，以$r = x_{pm}$為起始值，讓$r = r + 1$直到$P_x'(r) < maxX \times C$條件或$r = w - 1$條件成立為止，令$x_{right} = r$。

步驟四： 找出$x_{left}$和$x_{right}$後，$(x_{right} - x_{left})$是車牌的寬，$(y_{bottom} - y_{top})$是車牌的高。而車牌的寬高比例必須同時符合下列二個條件：

1. $1.5 \times$ 高 $<$ 寬 $< 5 \times$ 高
2. 寬 $> 30$ pixel

符合條件就找出一組$x_{left}$和$x_{right}$與$y_{top}$和$y_{bottom}$產生出一個車牌後選區；若不符合條件，$(x_{left}, x_{right})$所對應的值皆設為 0，重複步驟二，直到$cP_x'(i)$的最大值為 0 時，結束水平位置探索。

經過尋找車牌 X 座標後，這三組垂直位置探索候選區會如圖 3.10(A)、圖 3.10(B)、圖 3.10(C)，紅線所指的部份就是車牌可能所在的位置。


圖(A)第一組$x_{left}$和$x_{right}$候選區


圖(B)第二組$x_{left}$和$x_{right}$候選區

圖(C)第三組$x_{left}$和$x_{right}$候選區

**圖 3.10 水平位置探索**

### 3.2.1.5 選擇車牌候選區

經過車牌位置探索後，會產生最多三個候選區，第一候選區不一定是車牌，必須增加一個判定的條件，將這些候選區做垂直 Sobel 遮罩，產生垂直邊後，再從圖一半高的位置做水平掃描，當掃描的過程中遇到黑白變化時，就計數一次，如圖 3.11。

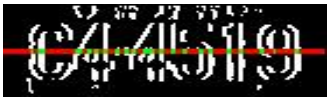


**圖 3.11 篩選候選區**

在圖 3.11 中綠色的點是指遇到黑白變化時，若超過 16 個綠色點，就可能是車牌的所在位置，因為在車牌沒有歪斜、汙點等正常情況下，黑白變化次數一定會大於 16 次。最後截取出來的車牌如圖 3.12。


**圖 3.12 擷取出的車牌**

### 3.2.2 車牌矯正

正常情況下拍攝車牌影像，在沒有傾斜的情況，上下邊界應該是平行於影像的上下邊界，可是拍照時往往無法這麼理想，經常出現車牌影像發生傾斜的狀況。發生傾斜的原因很多，其中以拍攝車牌的角度是最常發生，其次車牌本身因螺絲鬆動或受過外力撞擊而發生傾斜，自然所拍攝的車牌也會跟著傾斜。

當車牌影像發生傾斜時，對於車牌字串切割與辨識造成極大的困擾，如圖 3.13，車牌向右傾斜，在進行字元切割時，如果使用縱向分割，無論分割線在哪個位置都無法得到很好的分割效果，造成字元的破壞；如果使用連通物件法，在分割後字元的篩選條件上將不容易設定，可能影響字元在篩選時被捨棄。不管哪種狀況，都會大幅降低後續的字元辨識成功率。

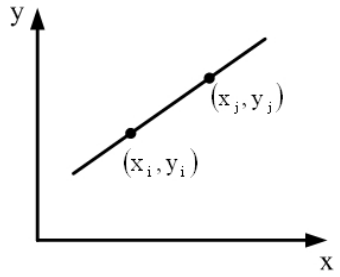

**圖 3.13 傾斜車牌**

所以當車牌傾斜時，系統必須對車牌進行調整，在字元切割前將車牌字串矯正回來，以利之後的字元切割與字元辨識的處理。而本系統以霍夫轉換來進行車牌矯正。

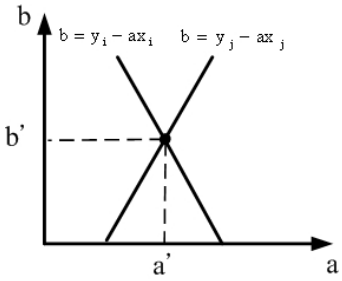
#### 3.2.2.1 霍夫轉換偵測矯正法

在座標空間中有一點$(x_i, y_i)$，通過此點的直線方程式可以用斜截式表示$y_i = ax_i + b$。通過點$(x_i, y_i)$有無窮多條直線，對於各種a ,b都會滿足直線方程式$y_i = ax_i + b$。

把等式寫成$b = y_i - ax_i$，並考慮 ab 平面(又稱參數空間(parameter space))，會產生對於固定的$(x_i, y_i)$座標有單一條直線的方程式。另外，在座標空間中有第二個點$(x_j, y_j)$在 ab 平面中也有一條相對應的直線。$(x_i, y_i)$與$(x_j, y_j)$在參數空間所對應的直線會相交於$(a', b')$，$a'$和$b'$是座標空間中含有$(x_i, y_i)$和$(x_j, y_j)$兩點之直線方程式的斜率與截距。這條直線方程式上所有的點在參數空間中都會有對應的直線並且相交於$(a', b')$。圖 3.14 說明了霍夫轉換的概念。




(a)座標空間




(b)參數空間

**圖 3.14**

透過上述霍夫轉換的概念，本研究使用下列演算法完成車牌矯正：

步驟一： 將車牌影像做 Sobel 水平邊緣偵測找出水平邊，遮罩如下，並將邊緣偵測結果以門檻值 160 做二值化處理。若邊緣偵測後的影像寬大於 300px，將影像等比縮小到寬為 300px。

水平邊遮罩

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

**圖 3.15 Sobel 遮罩**

步驟二： 產生一個與影像陣列$A_p$大小相同的陣列$A_c$，陣列$A_c$為一累計陣列，寬為影像的寬 w，高為影像的高 h，陣列內所有值皆為 0。

步驟三： 將陣列$A_p$的像素值P(x, y)由 RGB 色彩模型轉成 HSV 色彩模型取得亮度 b，其中$\{x \mid 0 \le x < w\}$、$\{y \mid 0 \le y < h\}$。

步驟四： 將影像P(x, y)座標空間正規化到參數空間，其範圍在$(-1, 1)$間，產生新的座標P′(xf, yf)，公式如下：

$$xf = \frac{2 \times x}{w} - 1 \text{；} yf = \frac{2 \times y}{h} - 1$$

**公式 3.4**

步驟五： 在參數空間中，找出斜率a與截距b，作法如下：

$$\{a \mid 0 \le a < w\}$$

$$af = \frac{2 \times a}{w} - 1$$

$$bf = yf - af \times xf$$

$$b = \frac{(bf + 1) \times h}{2}$$

步驟六： 若步驟五所求的截距 b 符合條件$0 < b < h - 1$，則在陣列$A_c$中P(a, b)位置累加陣列$A_p$中P(x, y)的亮度 b，重複步驟五直到所有斜率 a 都做完。

步驟七： 重複步驟四直到所有影像中的像素都做完。

步驟八： 在陣列$A_c$中亮度總和最大的座標$P_{max}$(maxX, maxY)透過下列方法找出垂直推移量$d_y$：

$$a = \frac{2 \times maxX}{w} - 1$$

$$b = \frac{2 \times maxY}{h} - 1$$

$$x0f = -1 \text{；} y0f = a \times x0f + b$$

$$x1f = 1 \text{；} y1f = a \times x1f + b$$

$$y0 = \frac{(y0f + 1) \times h}{2} \text{；} y1 = \frac{(y1f + 1) \times h}{2}$$

$$d_y = y1 - y0$$

步驟九： 以$d_y$透過推移公式將影像矯正，公式如下，推移後新的點為P(x′, y′)：

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ d_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

**公式 3.5 推移矩陣**

### 3.2.3 字元切割

車牌在經過初步定位及矯正後，大致已決定出車牌位置，為了減少定位所造成的車牌截斷，採取的方法是將定位的候選區擴大，以減少定位所造成車牌截斷，因此包含了非車牌字元的區域，如車牌四周的車身、台灣省、螺絲孔等，而這些皆為不需要的區域，其主要流程如圖 3.16 所示：

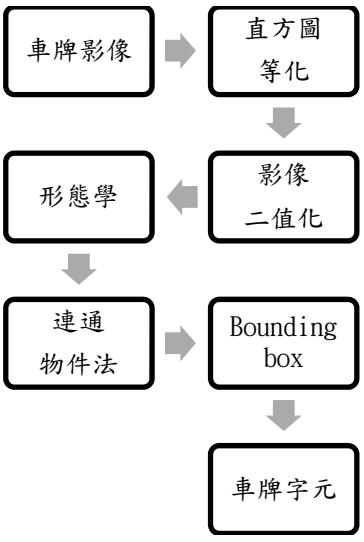



**圖 3.16 字元切割流程圖**

### 3.2.3.1 直方圖等化

影像可能會因為環境的不同，光線陰影所造成對比上的不明顯，很可能字元與背景無法分離。本研究使用直方圖等化，將所有像素統計 0~255 灰階值的數量，若是發生對比不足，則會發現統計出來的數值，會集中在某些區域，而這些區域，極有可能是含有背景及車牌字元的部分，我們想辦法將它們距離平均分散在 0~255 灰階，這樣就可以將影像的對比更加明顯，車牌字元更加清晰，如圖 3.17。

**圖 3.17 直方圖等化**

### 3.2.3.2 影像二值化

影像二值化的主要目的在尋找一個最佳的門檻值，使影像的元件與背景分離，而門檻值的取得方式有很多，我們採用統計式二值化法。統計式二值化對陰影、灰階度不均的影像有著很大的改善，我們採用 ISODATA 二值化法，從直方圖隨機取得一值做為初階門檻，計算門檻左右二邊的平均灰階值，再將二數做平均，做為下一次的門檻值，直到找到前後二次的結果相差小於 1 為止，此值為最佳門檻值，這種演算法比 Otsu 來得快，其結果如圖 3.18。

**圖 3.18 影像二值化**

### 3.2.3.3 字元修補及雜訊濾除

拍攝的影像成像不好，會導致二值化產生許多零碎的區塊，而這些區塊可能是因為車牌字元破損所造成的，也有一些太接近門檻值的像素造成雜訊。此情況可利用型態學的閉合及斷開運算來解決。

我們將破損的字元做修補，首先利用型態學的閉合運算，閉合運算是經過膨脹以及侵蝕運算所合成的，其作用是將破損字元做擴展，使破損字元能夠相黏在一起，之後再利用侵蝕將膨脹的部份侵蝕回來，而若是相黏過緊的區域就不會分離，其效果將圖 3.18 的字元破損部分修補成圖 3.19(A)，使得影像較為完整。反之，若我們想要刪除雜訊，則可利用型態學的斷開運算，先做侵蝕再做膨脹，將雜訊給清除掉，再將其他非雜訊的物件給修補回來，其效果如圖 3.19(B)，明顯的將非車牌字元的雜訊清除。

(A)閉合　　(B)斷開

**圖 3.19 型態學運算**

### 3.2.3.4 尋找相鄰的區塊

在雜訊的濾除後，剩下的較大區塊極可能是車牌字元。此時若採用投影法來做字元的切割，很可能會因為車牌的傾斜及定位的結果，無法正確取出車牌字元的間隔。因此我們採用連通物件標記法 (Connected Component Labeling)將相鄰的黑色像素做群組，每群組代表一個個體，計算出各群組的屬性來判斷是否為車牌字元，而我們這邊採取較嚴謹的四連通法作為區塊標記的準則，以避免字元相黏情形。

### 3.2.3.5 擷取區塊

經過連通物件標記法將相鄰像素群組化後，接著我們取得各區塊的特徵，作為車牌字元判斷的依據。我們採用 Bounding box 將影像利用最小矩形來包圍區塊，其做法可以簡易計

算出該字元的寬、高屬性，作為判斷是否為車牌字元的條件。尋得最小矩形的方式如下：

如圖 3.20，取得相同標記的 x 座標最大值與最小值以及 y 座標最大值與最小值，再將 x 的最大值與最小值相減即可算出區塊的寬，同理將 y 座標的最大值與最小值相減即可得出區塊的高，而這些屬性作為判斷是否為車牌字元的條件。

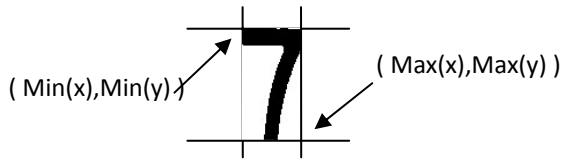



**圖 3.20 Bounding box 示意圖**

**3.2.3.6 篩選字元**

在小客車及摩托車的車牌字元數目都在六個之內，若是尋得的區塊大於六，很可能含入了雜訊、非車牌字元所致，因此我們設定一些條件來車牌字元的屬性做進一步的判定，規則如下：

1. 區塊黑色比例(黑色像素/全部的像素)必在 25%和 75%之間。
2. 區塊上、下半部的黑色比例必須少於 80%，否則捨棄。
3. 區塊的寬乘上 7 倍必定大於高。

上述屬性的判定主要統計車牌字元像素黑白比例做篩選條件，而這個方式難免還是會包含與車牌字元比例相近的雜訊。因此我們尋得一個方式，依循著車牌字元高度相近的特性，提出一個利用統計方法取得相近的車牌字元，此方法能有效抑制車牌外圍的車身雜訊，並能確切濾除掉比字元還小的雜訊其方法步驟如下：

步驟一： 將所有區塊的高屬性做排序，並保留區塊原本標記的位置。

步驟二： 將排序後的相鄰六個區塊編列為同一組，分別計算各組的變異數。

步驟三： 取得最小變異數做為車牌字元的判定，並依照各自標記的位置重新排列，取出車牌字元。

**3.2.4 字元正規化**

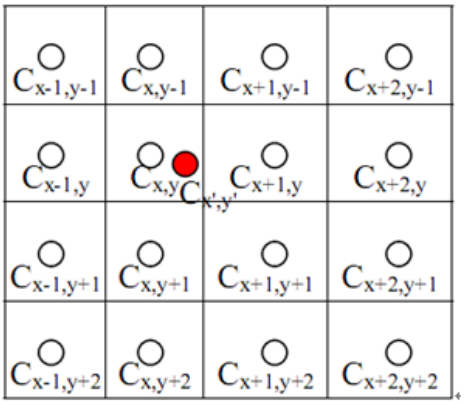
字元切割完畢後，接下來就是字元辨識的處理，在字元辨識之前，要把所有切割完的字元都做一次正規化的處理，拍攝的距離會影響字元的大小，由於字元辨識是用樣板比對的方法，所以必須將字元控制在固定的大小，以利車牌字元的辨識。

**3.2.4.1 字元大小轉換**

本系統考慮到辨識的正確率，將採用 27x 27 做為標準字元的大小，在字元大小轉換之前，必須先找出字元的範圍，利用尋找出字元最高、最低、最左及最右的位置來確定字元的範圍，避免把非字元範圍內的多餘白色背景也跟著縮放。在範圍確定後，藉由雙三次內插值演算法(Bicubic Interpolation)進行字元放大縮小的處理，也藉此避免字元在放大後產生字元空洞與鋸齒化現象的問題。

**3.2.4.2 雙三次內插值演算法**

雙三次內插值演算法是由雙線性內插法(Bilinear Interpolation)改進的演算法，此演算法中，縮放過後影像的像素顏色為原影像中與其最臨近的 16 個像素顏色$(4 \times 4)$，依其距離施以不同比重運算而得的結果。其演算法如下：




雙三次內插值演算法：

令$\Delta x = x' - x$；$\Delta y = y' - y$

$$t1 = -\Delta x(1-\Delta x)(1-\Delta x)C_{x-1,y-1} + (1-2\Delta x^2+\Delta x^3)C_{x,y-1} + \Delta x(1+\Delta x-\Delta x^2)C_{x+1,y-1} - \Delta x^2(1-\Delta x)C_{x+2,y-1}$$

$$t2 = -\Delta x(1-\Delta x)(1-\Delta x)C_{x-1,y} + (1-2\Delta x^2+\Delta x^3)C_{x,y} + \Delta x(1+\Delta x-\Delta x^2)C_{x+1,y} - \Delta x^2(1-\Delta x)C_{x+2,y}$$

$$t3 = -\Delta x(1-\Delta x)(1-\Delta x)C_{x-1,y+1} + (1-2\Delta x^2+\Delta x^3)C_{x,y+1} + \Delta x(1+\Delta x-\Delta x^2)C_{x+1,y+1} - \Delta x^2(1-\Delta x)C_{x+2,y+1}$$

$$t4 = -\Delta x(1-\Delta x)(1-\Delta x)C_{x-1,y+2} + (1-2\Delta x^2+\Delta x^3)C_{x,y+2} + \Delta x(1+\Delta x-\Delta x^2)C_{x+1,y+2} - \Delta x^2(1-\Delta x)C_{x+2,y+2}$$

$$C_{x',y'} = -\Delta y(1-\Delta y)(1-\Delta y)t1 + (1-2\Delta y^2+\Delta y^3)t2 + \Delta y(1+\Delta y-\Delta y^2)t3 - \Delta y^2(1-\Delta y)t4$$

**圖 3.21 雙三次內插值演算法**

### 3.2.5 字元辨識

#### 3.2.5.1 產生字元比對樣本

建立尺寸 27x27 的比對樣本，其中有英文字母 A-Z 與數字 0-9，英文字母 O 與數字 0 視為同一字元，有下列 35 種標準樣本，如表 3.1。

**表 3.1 字元樣本**

| 代號 | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 圖形 |  |  |  |  |  |
| 代號 | 5 | 6 | 7 | 8 | 9 |
| 圖形 |  |  |  |  |  |
| 代號 | A | B | C | D | E |
| 圖形 |  |  |  |  |  |
| 代號 | F | G | H | I | J |
| 圖形 |  |  |  |  |  |
| 代號 | K | L | M | N | P |
| 圖形 |  |  |  |  |  |
| 代號 | Q | R | S | T | U |
| 圖形 |  |  |  |  |  |
| 代號 | V | W | X | Y | Z |
| 圖形 |  |  |  |  |  |

#### 3.2.5.2 樣本比對法

把正規化的字元與資料庫樣本逐一比較，步驟如下：

步驟一： 切割後字元與比對樣本比較，字元與樣本像素點一樣的話，則相似值(Count) 加 1，若不同時則(Count)加 0。

步驟二： 重覆步驟直到字元與 35 種比對樣本比對完後，產生 35 個相似值進行步驟三。

步驟三： 找尋 35 個相似值中最大值，與此最大值相對應的比對樣本就是切割後的字元的辨識結果。

#### 3.2.5.3 字元的再辨認

字元會造成誤判的原因有很多種，有些字元外型相似，如(0、D)，而有些字元是因為歪斜的關係，如(8、B)、(6、8、9、S)，所以會誤判成另一組字元。本論文再辨識的方法是根據字元局部性的差異來做偵測，以下有更詳盡的介紹。

**3.2.5.3.1　6、8、9、S 的再辨識：**

6、8、9、S 這一組字元雖然外型差異大，但會因為字元歪斜而造成辨識誤判。它們之間的差異，在於 8 有兩個環形區塊，9 跟 6 只有一個環形區塊，S 則是沒有環形區塊。假設將字元從中心點切開，分成上下兩部分，可以發現 9 的環形區塊在上半部，而 6 的環形區塊則在下半部，根據這幾個地方做局部性的掃描，取出特徵值，就可以輕易的辨識出來，以下有更進一步的說明。

**圖 3.22　6、8、9、S 字元組的再辨識**

圖 3.22 是根據 6、8、9、S 字元的環形區塊所做的局部性掃描，將字元從中心點切開，分為上、下兩部分，分別做掃描。而掃描起點的定義為，以字元的中心點（x，y）為基準，X 軸不變，y 軸向上跟向下 2 至 3 個像素值，也就是以座標（x，y+2）、座標（x，y-3）為起點，開始向左右做水平掃描，上下做垂直掃描，圖中的紅線代表掃描線，當紅線碰到黑色像素時，則停止掃描，將權重值加 1。做完上述步驟後，可以得到 6、8、9、S 字元的權重表，如表 3.2 所示。接著根據表中的權重值做比對，就能夠將字元辨識出來。

**表 3.2　6、8、9、S 再辨識權重表**

| 字元 | 權重值 |
|---|---|
| 6 | 34 |
| 8 | 44 |
| 9 | 43 |
| S | 33 |

**3.2.5.3.2　0、D 的再辨識：**

0 與 D 外形相似，容易造成誤判。如圖 3.23 中，可以看出 0 與 D 的不同處在於紅色框起來的部分，0 有弧度而 D 則沒有，所以再辨識時，針對紅色框起來的部分做垂直掃描與水平掃描，當碰到黑色像素時則跳離，接著分別記錄碰到的第一個座標值，由於 0 有弧度，所以 0 的 Y 座標值會呈遞減的情形，D 沒有弧度，所以 D 的 Y 座標值不會有變化。

**圖 3.23　0、D 字元組的再辨識**

**3.2.5.3.3　8、B 的再辨識：**

8 與 B 的外形雖然差異頗大，由於在做辨識時，得到的字元不可能是完美的，而這些差異會造成辨識上的難度，所以為了提高辨識成功的機率，必須為 8 與 B 做再辨識的處理。而 8、B 再辨識的方法與 0、D 相同，都是針對紅色框起來的部分做垂直掃描與水平掃描（如圖 3.24），紀錄第一個碰到的黑色像素的座標值，而 8 有弧度 B 則沒有，所以 8 的 Y 座標值會呈遞減的情形，B 的 Y 座標則不會有變化，藉著這樣的差異，就能將 8 與 B 成功的辨識出來。

**圖 3.24　8、B 字元組的再辨識**

## 3.3 違規事項通報

本研究在通報違規事項採取兩個部分：手機簡訊與 e-mail，在手機使用者回傳辨識結果正確與違規事項時，系統會從資料庫尋找車主資訊(手機號碼與電子信箱)，並將違規訊息傳

送給車主，達到通報的即時性，手機簡訊與 e-mail 的作法如下：

1. 手機簡訊部分：透過台灣簡訊提供的簡訊 API 撰寫而成。系統可以在資料庫搜尋到已經建置的車主手機號碼，藉由台灣簡訊的 SMS Server 將違規事項以手機簡訊的方式傳送至車主的手機。
2. e-mail 部分：使用 JavaMail 來開發 e-mail 發信功能，由於本研究的作業環境是 Linux，所以採用 Postfix Mail Server 將違規訊息以 e-mail 的方式寄給車主。

## 4.實驗結果

### 4.1 系統需求

**伺服器環境：**

**表 4.1 伺服端裝置**

| | |
|---|---|
| 中央處理器 | Pentium 4 2.0G |
| 記憶體 | 512M |
| 作業系統 | Ubuntu 8.04 LTS |
| 網頁伺服器 | Apache 2.2<br>Tomcat 6<br>MySQL 5 |
| 郵件伺服器 | Postfix |

**硬體需求：**

**表 4.2 用戶端裝置**

| | |
|---|---|
| 手機 | Sony Ericsson K660i、K530i、W660i |
| 功能 | 拍照、上網 |
| 必須支援 JSR | MIDP 2.0、CLDC 1.1、MMAPI(JSR-135)、File/PIM(JSR-75) |

### 4.3 車牌辨識實驗結果

本研究所需車牌影像，是以手機或是數位相機在室外、室內停車場，拍攝明暗度不同、拍攝距離不同、環境複雜度不同的車牌影像，實驗車牌樣本共有 455 張 640x480 車牌影像。


**明暗度不同的車牌影像**


**拍攝距離不同的車牌影像**


**環境複雜的車牌影像**

**圖 4.1 車牌影像**

成功的車牌辨識案例：


**原始影像**


**車牌定位-垂直位置探索**


車牌定位-水平位置探索


字元切割

圖 4.2 實驗結果圖

### 4.4 使用者介面設計

當使用者啟動本系統時，就進入圖 4.3 的拍攝模式，將手機對準該違規車輛的車牌點選 snapshot 選項，系統會立即將影像傳送至伺服器做車牌號碼的辨識，隨後將結果回傳給使用者做進一步確認，如圖 4.4。


圖 4.3 拍攝介面


圖 4.4 確認車牌號碼

若使用者按下傳送按鈕時，伺服器會回傳違規事項選單供使用者做選取，如圖 4.5 所示，當選項被選取後，即完成通報步驟，最後車主便會收到簡訊與 e-mail 如圖 4.6 與 4.7。

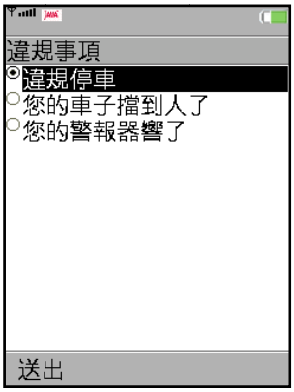



圖 4.5 違規通報選項

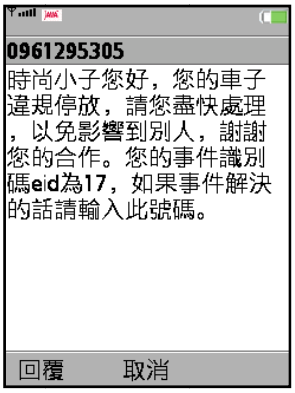



圖 4.6 收到簡訊畫面


圖 4.7 信箱收信畫面

### 4.5 實驗結果

本研究測試車牌辨識率的車牌影像共有 455 張，包含用數位相機與手機拍攝在主機測試，與使用開發完成的手機系統實地在路旁或停車場拍攝測試，實驗結果如表 4.3，本實驗車牌辨識成功的定義為 6 個車牌字元必須完全辨識出來，缺一個字即算失敗，總共成功張數為 380 張，其辨識率為 83%，平均辨識時間為 2.93(秒/張)；至於在手機系統部分，從拍照到系統回傳車牌號碼的反應時間平均為 5.96(秒/張)。

**表 4.3 實驗數據**

| | |
|---|---|
| 全部車牌影像 | 455 張 |
| 辨識成功影像 | 380 張 |
| 辨識成功率 | 83% |
| 辨識時間 | 2.93 秒 |
| 系統反應時間 | 5.96 秒 |

車牌辨識失敗的原因如下：

1. 拍攝過暗或過亮、距離太遠或太近、背景複雜與車體顏色較淺(如白色)的車牌影像，導致邊緣不易判定以致於車牌定位失敗。
2. 由於水平邊緣過多，造成車牌傾斜角度判定錯誤，以至於車牌矯正失敗。
3. 車牌有髒汙、或是車牌字元有缺損而導致字元切割與辨識的失敗。

## 5 結論

「即時車牌辨識與通報系統」是車牌辨識系統的加值應用。主要目的是使用者能夠即時通知違規停車的車主，提醒車主能夠快速遷移車輛，改善交通問題。本研究提出的方法是結合手機照相與行動網路的功能，使系統具有即時性與機動性。為了達到機動性，衍生了不少問題，如拍攝的角度造成車牌的傾斜、日照不均造成光線陰影等。為了改善上述問題，針對車牌傾斜採用霍夫轉換做矯正，車牌陰影利用直方圖等化使影像對比更明顯。本研究使用 455 張不同環境及大小的車牌做實驗，經本系統的定位與辨識，辨識結果能高達 83%。

本研究最大的貢獻，是使用行動裝置經由行動網路傳送資訊，以達到即時通知的效果，除此之外，還可應用至許多方面，如贓車查緝、違規停車等。在未來的發展上，可利用手機的輔助全球衛星定位系統(Assisted Global Positioning System, AGPS)功能取得拍攝現場的經緯度，找出違規車輛的地點，並與拍攝照片一併送至伺服器做處理。倘若使用者有車輛追蹤查詢的需求，可利用 Google Maps 結合網頁程式，將拍攝者傳來的經緯度標示在電子地圖，以利查詢違規車輛。

## 參考文獻


[1] **內政部警政署**，http://www.npa.gov.tw/。

[2] **中華智慧型運輸系統協會 ITS**，http://www.its-taiwan.org.tw/。

[3] 陳翔傑，**自動化車牌辨識系統設計**，國立中央大學電機工程研究所，碩士論文，民國九十四年。

[4] 王精忠，**車牌辨識之研究**，大同大學，通訊工程研究所，碩士論文，民國 94 年 1 月。

[5] 王振興，**多標的汽機車車牌辨識系統之研究**，元智大學，資訊管理學系碩士班，碩士論文，民國 92 年。

[6] 余忠潔，**新的車牌定位方法**，靜宜大學資訊管理研究所碩士論文，民國 92 年。

[7] 魏銷志，**動態多標的車牌辨識系統之研究**，元智大學，資訊研究所碩士論文，2000。

[8] 張簡子介，**用小腦模型在 FPGA 上作車牌辨識**，國立臺灣師範大學工業教育研究所碩士論文，2002。

[9] 李正裕，**車牌辨識系統之研究**，靜宜大學，資訊管理研究所論文, 2003。

[10] 蔡銘鑫，**小波轉換和類神經網路應用於車牌辨識**，朝陽科技大學，資訊工程系，碩士論文，民國 93 年 5 月。

[11] 呂炎州，**不需字元切割的車牌辨識法**，私立靜宜大學資訊管理學系碩士論文，2004。

[12] 八木伸行、林正樹、三谷公二、奥井誠人、

吳上立、林宏墩，**C 語言數位影像處理**，全華，2005。

[13] **Algorithmic and mathematical principles of automatic number plate recognition systems**, Ondrej Martinsky, Brno University of Technology, 2007.

[14] 繆紹綱，**數位影像處理 Digital Image Processing 2/e**，普林斯頓，2002。

[15] 陳鴻文，**車輛辨識系統應用於車輛進出管制**，國立高雄第一科技大學，電腦與通訊工程研究所，碩士論文，民國 96 年 6 月。

[16] **台灣簡訊**，http://www.twsms.com。

[17] **資策會 FIND，http://www.find.org.tw。**