Institute of Technology, Carlow

B.Sc. Hons. in Software Engineering

CW228

# Research Manual

|  |  |
|---|---|
| **Project Title:** | **Number Plate Recognition** |
| Name: | Dongfan Kuang |
| Login ID: | C00131031 |
| Supervisor: | Nigel Whyte |
| Date: | 1st December 2009 |

## Table of Contents

# 1. Introduction of Number Plate Recognition

## 1.1. What is NPR?

NPR (Number Plate Recognition) is an image-processing technology used to automatically identify vehicles by their number plate captured by a camera, allowing these details to be compared against database records. NPR systems are used in various security and traffic application, such as access-control, parking and so on.

For example in an access-control system, when the vehicle arrives the gate of the area it wants to enter into, the NPR unit will takes pictures of the vehicle and "reads" the plate number from the picture, compares it to a predefined list and opens the gate if there is a match.

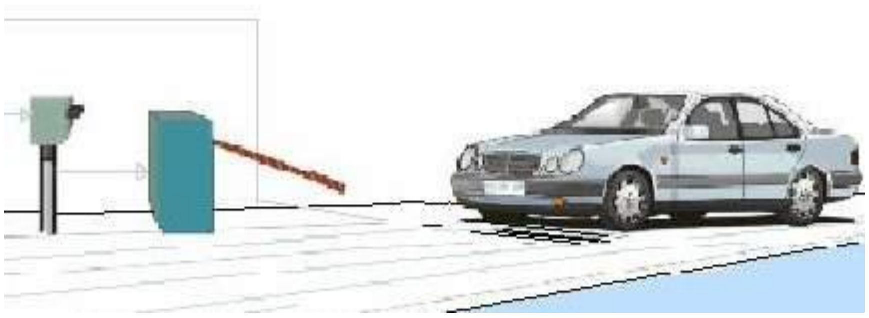


Figure 1-1 NPR Used in Access Control System [1]

## 1.2. NPR Process

The steps involved in NPR include plate localization which contains the recognition and isolation of the number plate in a vehicle image captured by the camera. Next step is called plate orientation and sizing adjusted the skew angle and the size of the plate image. After adjusted the size and the shape of the image, the contrast and brightness of the image are adjusted in a step called normalization. And then, the individual characters in the number plate are separated and identified through character segmentation and optical character recognition. The identified number is then used to compare with the records in a database to show the related information like the vehicle's owner, place of registration, address and so on.

## 1.3. Other Names of NPR

NPR also has some other names [1]:

- Automatic Vehicle Identification (AVI)
- Car Plate Recognition (CPR)
- Automatic Number Plate Recognition (ANPR)
- Car Plate Reader (CPR)
- Optical Character Recognition (OCR) for Cars

# 2. Precedent Similar Software

## 2.1. Autonomy Virage's Automatic Number Plate Recognition (ANPR) System

Autonomy Virage's ANPR system use neural network techniques in its character recognition which improves the efficiency and integrity of the system. Autonomy ANPR system is based on a proven digital video capture and transmission system and suitable for a wide range of applications, for example: car parks, traffic surveys, petrol filling stations and to detect stolen vehicles if used in conjunction with database searching.

Features of Autonomy Virage's ANPR system[2]:

- Neural network based character recognition
- Multiple cameras can be monitored using one channel via a sequencing facility
- Can be trained to recognize any international plate format
- Capture, storage & reviewing of number plates
- Ability to initiate automated responses such as opening of barriers, email, SMS messaging, or image transmission to central locations
- Real-time number plate searching against user defined databases

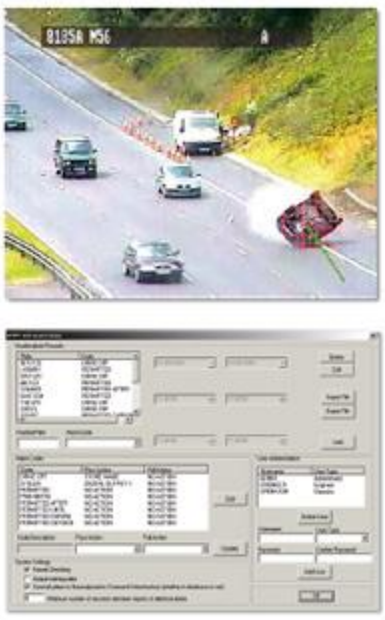


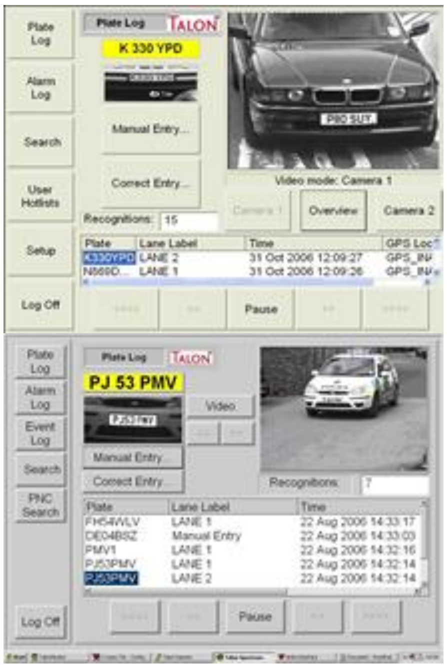
Figure 2-1 Screenshot of Autonomy Virage's ANPR System

## 2.2. Talon Automatic Number Plate Recognition Systems

The basic Talon ANPR system comprises of a PC, a suitable frame grabber and a video camera. Talon ANPR system is based on a neural network recognition engine called Talon which developed by NDI Recognition Systems Ltd.

Features of Talon ANPR system [3]:

- Trained Neural Network - Neural Networks have a large advantage over other OCR solutions as the correct read rate of number plates is significantly higher. The Talon Neural are not programmed in the traditional way but are trained by example on a large number of repetitions of character sets. So it is more able to recognize poorly defined, distorted and dirty characters, ensuring accuracy in the most demanding real world conditions.
- Multi lane Processing Capability - The Talon Neural Networks are capable of accurate high speed plate recognition from multiple lanes of high speed and high density traffic. The system is currently capable of processing 4 camera feeds on a single PC.
- Confidence Levels - Each individual character within a vehicle number plate is given a specific score, this score is the confidence of read accuracy. Multiple number plate images are processed for each passing vehicle. This allows the PC to post the best possible result for each vehicle.



Recognitions: 15

| Plate | Lane Label | Time | GPS Loc |
|---|---|---|---|
| K330YPD | LANE 2 | 31 Oct 2006 12:09:27 | GPS_IN |
| N669D... | LANE 1 | 31 Oct 2006 12:09:26 | GPS_IN |

Recognitions: 7

| Plate | Lane Label | Time |
|---|---|---|
| FH54WLV | LANE 1 | 22 Aug 2006 14:33:17 |
| DE04BSZ | Manual Entry | 22 Aug 2006 14:33:03 |
| PMV1 | LANE 1 | 22 Aug 2006 14:32:16 |
| PJ53PMV | LANE 1 | 22 Aug 2006 14:32:14 |
| PJ53PMV | LANE 2 | 22 Aug 2006 14:32:14 |

Figure 2-2 Screenshots of Talon ANPR system

## 2.3. Conclusion

A lot of ANPR systems are use neural network to recognize character, because neural network technology is superior to any template based Optical Character Recognition (OCR) ANPR system, offering significantly higher performance and accuracy.

# 3. Some Basic Knowledge

## 3.1. Vehicle Registration Plates of Ireland

Registration marks on number plates in Ireland issued since 1987 have the format YY-CC-SSSSSS where the components are:

YY — a 2-digit year (e.g. 87 for 1987; 05 for 2005)

CC — a 1 or 2 character county identifier (e.g. D for Dublin; CW for Carlow).

SSSSSS — a 1 to 6 digit sequence number, starting with the first vehicle registered in the county that year.

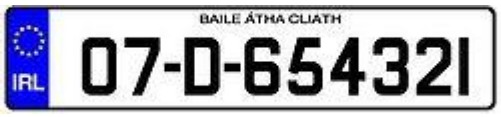


Figure 3-1 Example of Current Standard Irish Number Plate [4]

Table 3.1 Current index mark codes [4]

| Code | County or city | Code | County or city |
|---|---|---|---|
| C | Cork | LS | Laois |
| CE | Clare | MH | Meath |
| CN | Cavan | MN | Monaghan |
| CW | Carlow | MO | Mayo |
| D | Dublin | OY | Offaly |
| DL | Donegal | RN | Roscommon |
| G | Galway | SO | Sligo |

| | | | |
|---|---|---|---|
| KE | Kildare | TN | Tipperary North |
| KK | Kilkenny | TS | Tipperary South |
| KY | Kerry | W | Waterford City |
| L | Limerick City | WD | County Waterford |
| LD | Longford | WH | Westmeath |
| LH | Louth | WX | Wexford |
| LK | County Limerick | WW | Wicklow |
| LM | Leitrim | | |

## 3.2. Pixel

The word pixel is based on a contraction of pix (for "pictures") and el (for "element") [5]. In digital imaging, a pixel is the smallest item of information in an image. Pixels are normally arranged in a 2-dimensional grid, and are often represented using dots or squares. Each pixel is a sample of an original image, where more samples typically provide more-accurate representations of the original. The intensity of each pixel is variable; in color systems, each pixel has typically three components such as red, green, and blue.

## 3.3. RGB

The RGB color model is an additive color model in which red, green, and blue are added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three primary colors, red, green, and blue.

A color in the RGB color model is described by indicating how much of each of the red, green, and blue is included. The color is expressed as an RGB triplet (r, g, b), each component of which can vary from zero to a defined maximum value. If all the components are at zero the color is black and if all are at maximum, the color is white [6].

These ranges may be quantified in several different ways:

- From 0 to 1, with any fractional value in between. This representation is used in theoretical analyses, and in systems that use floating-point representations.
- Each color component value can also be written as a percentage, from 0% to 100%.
- In computing, the component values are often stored as integer numbers in the range 0 to 255, the range that a single 8-bit byte can offer (by encoding 256 distinct values).
- High-end digital image equipment can deal with the integer range 0 to 65,535 for each primary color, by employing 16-bit words instead of 8-bit bytes.

For example: (0, 0, 0) is black, (255, 255, 255) is white, (255, 0, 0) is red, (0, 255, 0) is green, (0, 0, 255) is blue, (255, 255, 0) is yellow, (0, 255, 255) is cyan, (255, 0, 255) is magenta.

## 3.4. Grayscale

In photography and computing, a grayscale or grayscale digital image is an image in which the value of each pixel is a single sample, that is, it carries only intensity information. Images of this sort, also known as black-and-white, are composed exclusively of shades of gray, varying from black at the weakest intensity to white at the strongest.

Grayscale images are distinct from one-bit black-and-white images, which in the context of computer imaging are images with only two colors: black and white (also called *bi-level* or *binary images*) [7]. Grayscale images have many levels of gray in between.

In computing, although the grayscale can be computed through rational numbers, image pixels are stored in binary form. Some early grayscale monitors can only show up to sixteen (4-bit) different shades, but today grayscale images intended for visual display (both on screen and printed) are commonly stored with 8 bits per pixel, which allows 256 different intensities to be recorded.

## 3.5. Thresholding

In many vision applications, it is useful to be able to separate out the regions of the image corresponding to objects in which we are interested, from the regions of the image that correspond to background. Thresholding often provides an easy and convenient way to perform this segmentation on the basis of the different intensities or colors in the foreground and background regions of an image.
In addition, it is often useful to be able to see what areas of an image consist of pixels whose values lie within a specified range, or band of intensities (or colors). Thresholding can be used for this as well.
The input to a thresholding operation is typically a grayscale or color image. In the simplest implementation, the output is a binary image representing the segmentation. Black pixels correspond to background and white pixels correspond to foreground (or vice versa). In simple implementations, the segmentation is determined by a single parameter known as the intensity threshold. In a single pass, each pixel in the image is compared with this threshold. If the pixel's intensity is higher than the threshold, the pixel is set to, say, white in the output. If it is less than the threshold, it is set to black [8].

$$B[i,j]=\begin{cases}1, F[i,j]\geq T\\ 0. otherwise\end{cases}\qquad T=100$$

In more sophisticated implementations, multiple thresholds can be specified, so that a band of intensity values can be set to white while everything else is set to black. For color or multi-spectral images, it may be possible to set different thresholds for each color channel, and so select just those pixels within specified cuboids in RGB space. Another common variant is to set to black all those pixels corresponding to background, but leave foreground pixels at their original color/intensity (as opposed to forcing them to white), so that that information is not lost.

# 4. Steps for Identifying a Number Plate

## 4.1. Pre-processing

### 4.1.1. Converting Color to Grayscale

To convert any color to a grayscale representation of its luminance, first one must obtain the values of its red, green, and blue (RGB) primaries in linear intensity encoding, by gamma expansion. Then, add together 30% of the red value, 59% of the green value, and 11% of the blue value (these weights depend on the exact choice of the RGB primaries, but are typical). Regardless of the scale employed (0.0 to 1.0, 0 to 255, 0% to 100%, etc.), the resultant number is the desired linear luminance value; it typically needs to be gamma compressed to get back to a conventional grayscale representation [7].

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

### 4.1.2. Gaussian Smoothing

The Gaussian smoothing operator is a 2-D convolution operator that is used remove detail and noise in images. In 2-D, an Gaussian has the form [9]:

$$G(x,y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$

The idea of Gaussian smoothing is to use this 2-D distribution as a “point-spread” function, and this is achieved by convolution. Since the image is stored as a collection of single pixels we need to produce a discrete approximation to the Gaussian function before we can perform the convolution. In theory, the Gaussian distribution is non-zero everywhere, which would require an infinitely large convolution kernel, but in practice it is effectively zero more than about three standard deviations from the mean, and so we can truncate the kernel at this point. Figure 4-1 shows a suitable integer-valued convolution kernel that approximates a Gaussian with $\sigma = 1.0$.

$$\frac{1}{273}\begin{array}{|c|c|c|c|c|}\hline 1 & 4 & 7 & 4 & 1 \\ \hline 4 & 16 & 26 & 16 & 4 \\ \hline 7 & 26 & 41 & 26 & 7 \\ \hline 4 & 16 & 26 & 16 & 4 \\ \hline 1 & 4 & 7 & 4 & 1 \\ \hline \end{array}$$

Figure 4-1 Gaussian with $\sigma = 1.0$ [9]

## 4.2. Edge Detection

### 4.2.1. Sobel Operator

Sobel operator calculates the gradient of the image intensity at each point, giving the direction of the largest possible increase from light to dark and the rate of change in that direction. The result therefore shows changes between points and therefore highlights the part of the image that represents an edge, as well as how that edge is likely to be oriented. In practice, the magnitude (likelihood of an edge) calculation is more reliable and easier to interpret than the direction calculation.

Mathematically, the gradient of a two-variable function is at each image point a 2D vector with the components given by the derivatives in the horizontal and vertical directions. At each image point, the gradient vector points in the direction of largest possible intensity increase, and the length of the gradient vector corresponds to the rate of change in that direction. This implies that the result of the Sobel operator at an image point which is in a region of constant image intensity is a zero vector and at a point on an edge is a vector which points across the edge, from darker to brighter values.

The operator uses two 3×3 kernels which are convolved with the original image to calculate approximations of the derivatives - one for horizontal changes, and one for vertical. If we define A as the source image, and $G_x$ and $G_y$ are two images which at each point contain the horizontal and vertical derivative approximations, the computations are as follows[10]:

$$\mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A}$$

where * here denotes the 2-dimensional convolution operation.

The *x*-coordinate is here defined as increasing in the "right"-direction, and the *y*-coordinate is defined as increasing in the "down"-direction. At each point in the image, the resulting gradient approximations can be combined to give the gradient magnitude, using [10]:

$$\mathbf{G} = \sqrt{\mathbf{G}_x^{\,2} + \mathbf{G}_y^{\,2}}$$

Using this information, we can also calculate the gradient's direction [10]:

$$\mathbf{\Theta} = \arctan\left(\frac{\mathbf{G}_y}{\mathbf{G}_x}\right)$$

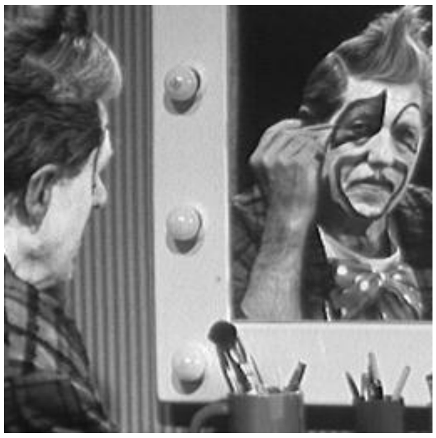


Figure 4-2 Image before Sobel edge detection

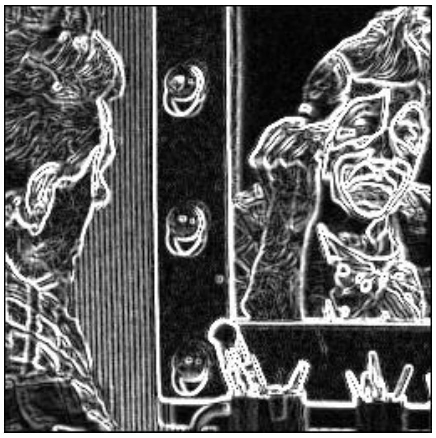


Figure 4-3 Image after Sobel edge detection

## 4.2.2. Canny Edge Detection

**Step 1: Image Smoothing**

In pre-processing step, the picture is already smoothed by using Gaussian Smoothing operator, so we can just skip this step.

**Step 2: Differentiation**

After smoothing the image and removing the noise, the next step is to find the edge strength by calculating the gradient of the image. The Sobel operator performs a 2-D spatial gradient measurement on an image. Then, the approximate absolute gradient magnitude (edge strength) at each point can be found. The Sobel operator uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows).

The edge strength of the gradient is then approximated using the formula:

$$|G| = |Gx| + |Gy|$$

**Step 3: Non-maximum Suppression**

After found the rate of intensity change at each point in the image, edges must now be placed at the points of maxima which means non-maxima must be suppressed. A local maximum occurs at a peak in the gradient function, or alternatively where the derivative of the gradient function is set to zero. However, in this case we wish to suppress non-maxima perpendicular to the edge direction, rather than parallel to the edge direction, since we expect continuity of edge strength along an extended contour.

Rather than perform an explicit differentiation perpendicular to each edge, another approximation is often used. Each pixel in turn forms the centre of a nine pixel neighborhood. By interpolation of the surrounding discrete grid values, the gradient magnitudes are calculated at the neighborhood boundary in both directions perpendicular to the centre pixel. If the pixel under consideration is not greater than these two values (non-maximum), it is suppressed.

**Step 4: Edge Thresholding**

Finally, hysteresis is used as a way of eliminating streaking. Streaking is the breaking up of an edge contour caused by the operator output fluctuating above and below the threshold. If a single threshold, T1 is applied to an image, and an edge has an average strength equal to T1, then due to noise, there will be possibility where the edge lowers than the threshold. Equally it will also higher than the threshold making an edge look like a dashed line. To avoid this, hysteresis uses 2 thresholds, a high and a low. Any pixel in the image that has a value greater than T1 is presumed to be an edge pixel, and is marked immediately. Then, any pixels that are connected to this edge pixel and that have a value greater than T2 are also marked as edge pixels [11].

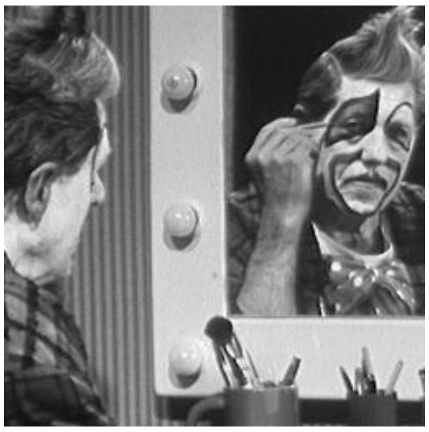


Figure 4-4 image before Canny edge detection

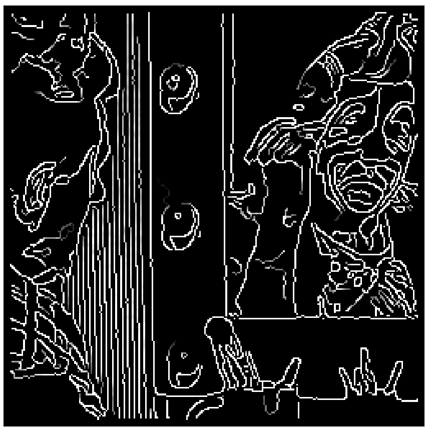


Figure 4-5 image after Canny edge detection

## 4.2.3. Prewitt Operator

The Prewitt edge detection masks are one of the oldest and best understood methods of detecting edges in images. Basically, there are two masks, one for detecting image derivatives in X and one for detecting image derivatives in Y. To find edges, a user convolves an image with both masks, producing two derivative images. The strength of the edge at any given image location is then the square root of the sum of the squares of these two derivatives.

In practice, usually one thresholds is used in Prewitt edge detection, in order to produce a discrete set of edges. Bellowing figures show a source image and the thresholded result of Prewitt edge detection:

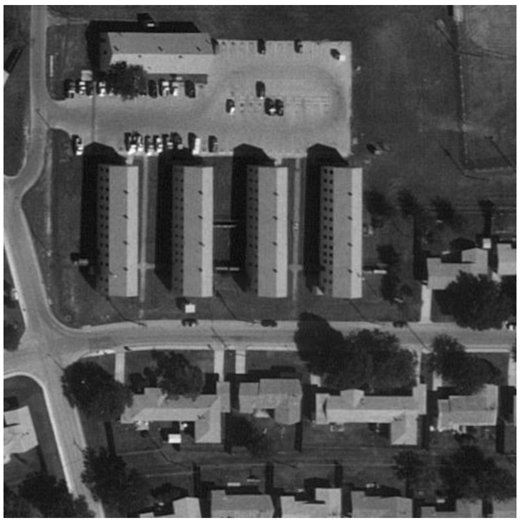


Figure 4-6 image before Prewitt edge detection

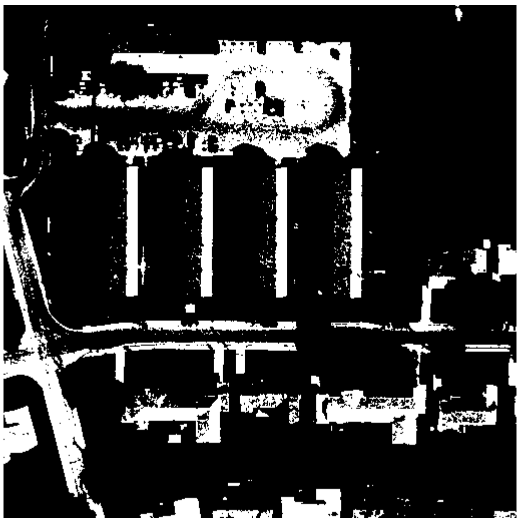


Figure 4-7 image after Prewitt edge detection

### 4.2.4. Kirsch Operator

The Kirsch Edge detection detects edges using eight compass filters. All eight filters are applied to the image with the maximum being retained for the final image. The eight filters are a rotation of a basic compass convolution filter. The operator is calculated as follows for directions with 45° difference [12]:

$$h_{n,m} = \max_{z=1,\ldots,8} \sum_{i=-1}^{1} \sum_{j=-1}^{1} g_{ij}^{(z)} \cdot f_{n+i,m+j}$$

where the direction kernels are [12]:

$$\mathbf{g}^{(1)} = \begin{bmatrix} +5 & +5 & +5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}, \mathbf{g}^{(2)} = \begin{bmatrix} +5 & +5 & -3 \\ +5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}, \mathbf{g}^{(3)} = \begin{bmatrix} +5 & -3 & -3 \\ +5 & 0 & -3 \\ +5 & -3 & -3 \end{bmatrix}, \mathbf{g}^{(4)} = \begin{bmatrix} -3 & -3 & -3 \\ +5 & 0 & -3 \\ +5 & +5 & -3 \end{bmatrix}$$

and so on.

## 4.2.5. Roberts' Cross Operator

The Roberts Cross operator performs a simple and quick way to compute 2-D spatial gradient measurement on an image. It thus highlights regions of high spatial frequency which often correspond to edges. In its most common usage, the input to the operator is a grayscale image, as is the output. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point.

In theory, the operator consists of a pair of 2×2 convolution kernels as shown in below. One kernel is simply the other rotated by 90°. This is very similar to the Sobel operator.

| | |
|---|---|
| +1 | 0 |
| 0 | -1 |

Gx

| | |
|---|---|
| 0 | +1 |
| -1 | 0 |

Gy

Figure 4-8 Roberts Cross masks [13]

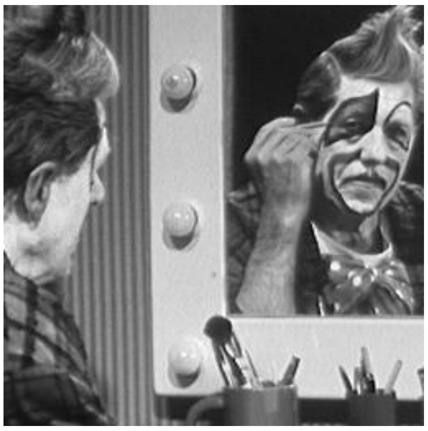


Figure 4-9 image before Roberts' edge detection

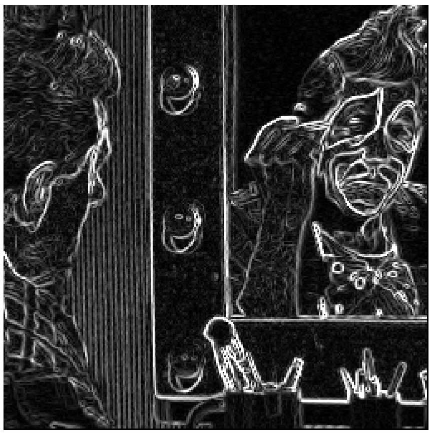


Figure 4-10 image after Roberts' edge detection

Roberts' Cross is still in use due to the speed of computation, but performance compared to the other method is poor, with noise sensitivity a significant problem.

### 4.2.6. Gauss-Laplace Operator

The Laplacian is a 2-D isotropic measure of the 2nd spatial derivative of an image. The Laplacian of an image highlights regions of rapid intensity change and is therefore often used for edge detection. The Laplacian is often applied to an image that has first been smoothed with something approximating a Gaussian smoothing filter in order to reduce its sensitivity to noise, and hence the two variants will be described together here. The operator normally takes a single gray level image as input and produces another gray level image as output.

The Laplacian *L(x,y)* of an image with pixel intensity values *I(x,y)* is given by [14]:

$$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

This can be calculated using a convolution filter.

Since the input image is represented as a set of discrete pixels, we have to find a discrete convolution kernel that can approximate the second derivatives in the definition of the Laplacian. Two commonly used small kernels are shown as follow:

| | | |
|---|---|---|
| 0 | −1 | 0 |
| −1 | 4 | −1 |
| 0 | −1 | 0 |

| | | |
|---|---|---|
| −1 | −1 | −1 |
| −1 | 8 | −1 |
| −1 | −1 | −1 |

Figure 4-11 Two commonly used Laplacian filters [14]

### 4.2.7. Hough Transform

The Hough transform is a technique which can be used to isolate features of a particular shape within an image. Because it requires that the desired features be specified in some parametric form, the classical Hough transform is most commonly used for the detection of regular curves such as lines, circles, ellipses, etc. A generalized Hough transform can be employed in applications where a simple analytic description of a feature(s) is not possible. Due to the computational complexity of the generalized Hough algorithm, we restrict the main focus of this discussion to the classical Hough transform. Despite its domain restrictions, the classical Hough transform (hereafter referred to without the classical prefix) retains many applications, as most manufactured parts (and many anatomical parts investigated in medical imagery) contain feature boundaries which can be described by regular curves. The main advantage of the Hough transform technique is that it is tolerant of gaps in feature boundary descriptions and is relatively unaffected by image noise.

### 4.2.8. Conclusion

After compared the above edge detection algorithm, I decide to use Canny edge detection, because this algorithm can effectively suppress noise, retaining continuous, clear edges.

## 4.3. Number Plate Localization

The Irish number plate is white background with black characters on it and surrounded by a black border having a stroke width of 5 millimeters. The external size of the plate shall be 520 millimeters in width and 110 millimeters in height. The flag of the European Communities with letters IRL is in the left side of plate in blue background. The characters shall have a height of 70 millimeters and a stroke width of 10 millimeters. The width of each character (other than the letter "I" or the number "1") shall be not less than 36millimetres and not more than 50 millimeters. The distances between adjoining letters and adjoining figures shall be not less than 8 millimeters [15].

### 4.3.1. Number Plate Localization Overview

Depending on the implementation method, the existing localization method can be categorized as following:

**Direct method:** this kind of methods analysis the features of the inputted image directly to find out the plate area. For example: locate the plate area by the higher frequency of gray scale or color change in this area; found the four corners of the plate by using a template matching method.

**Texture detection method**: locate the plate by analysis its texture information. But the bumpers and lights around the plate also have rich texture information mainly on horizontal direction. So the plate area can be found by analysis the vertical texture information.

**Neural network method:** use neural network to find the plate area is another common used method. The basic idea is first divide the inputted image into a number of sub-images, and then classify these sub-images through a neural network. And then analysis the result of classification to find the sub-image contains the plate.

**Localization based on color image:** the standard Irish number plates always have white background and black characters and a blue part on the left side, the plate area can be found by using this information.

### 4.3.2. Number Plate Isolation

After applying edge detection algorithm, the edge information of the area which contains the number plate is much richer than the other areas. This is because there are many characters which contains a lot of strokes internal the number plate. At the same time, number plate area is a relatively rectangle surrounded by straight lines. Therefore, we first define a window which has the same ratio of length and width of number plate but larger than it. And then move this window around in the image and count the number of white pixel within the window, because the edge information is most concentrated in the sub-image which contains the number plate, we can identify that the sub-image which has the maximum number of white pixel is the area contains number plate.

In the following figure, the red rectangular is a slide-window which moves around in the picture until find the sub-image contains the maximum number of white pixel and this sub-image usually contains the number plate.

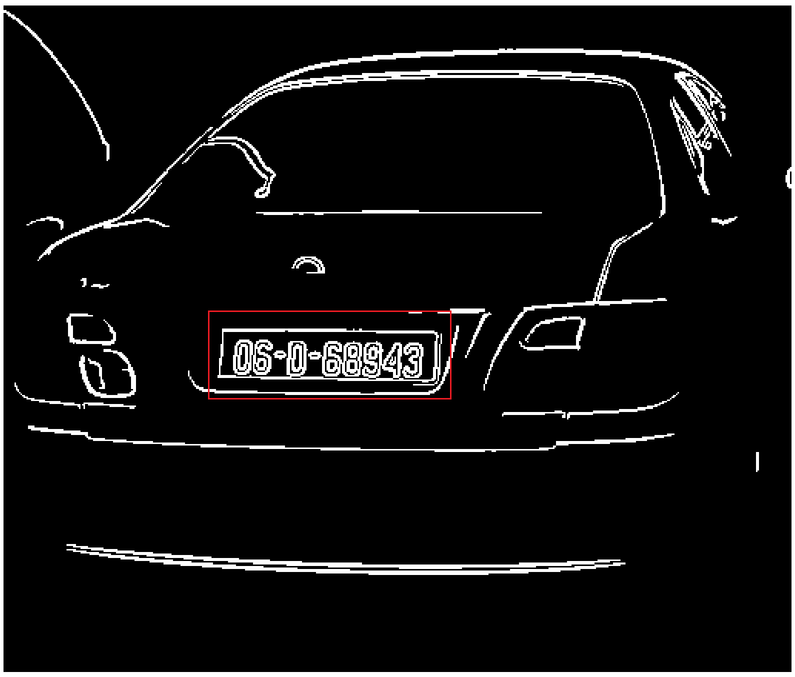


Figure 4-12 using slide-window to isolate plate

### 4.3.3. Plate Orientation and Sizing

**Step 1: identify the top and bottom boundaries of number plate.**

This algorithm scans line by line from top to bottom, count the times of pixel changes (from 0 to 1 or from 1 to 0) between adjacent pixels. When the time of changes in one certain line is greater than a threshold value and the same as the following n lines, we can assume this line is the top line of the number plate. And then continue scans line by line until find a certain line has the times of changes less than threshold value and assume this line is the bottom line of the number plate.

Algorithm:

1) Define an array with value 0 for its entire element to record the times of gray-scale changes in each horizon line, the length of the array equals to the height of the sub-image.
2) Scan the sub-image line by line from top to bottom, count the time of gray-scale changes in each line, if the value of time > threshold then stores 1 into the corresponding position of the array.
3) After the whole sub-image scanned, first check the array from its first element, if there are n continues elements have value 1 then we can assume the line related to the first element is the top line of the number plate. The bottom line can be found in same method by check the array from its last element.

In following figure, the number plate has 7 characters, each character at least has two times of pixel changes when a line scans cross it, so set the threshold value as 14. And we can identify from to line 6 to line 24 is the area contains character, figure 4-5 is the sub-image after locate top and bottom boundary, the top and bottom border of the plate has been removed.

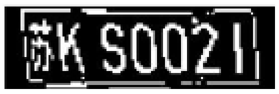


Figure 4-13 original image [17]

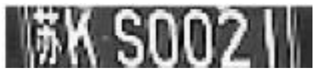


Figure 4-14 image after locate top and bottom boundary [17]

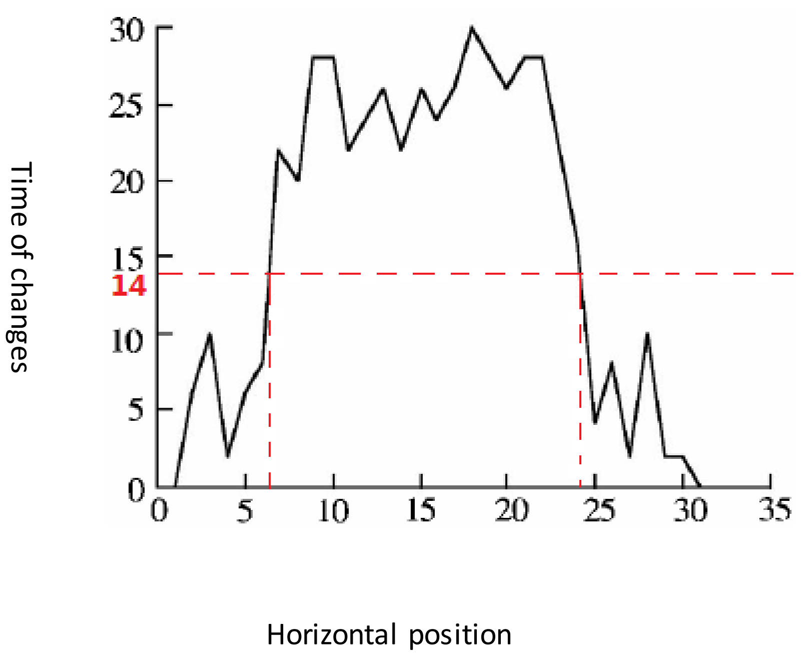


Figure 4-15 the value of threshold is 14 [17]

**Step 2: Identify the left and right boundaries of number plate.**

Because the method used to find the top and bottom line of number plate is not working very well when used to identify the left and right line, we use vertical projection method instead.

Algorithm:

1) Define an array with value 0 for its elements to store the projection information of each vertical line.
2) Scan the sub-image line by line in vertical direction. If the number of white pixel > n, then change the value of the corresponding element in the array into 1.
3) After the whole sub-image scanned, first check the array from its first element until finds the n continues elements have value 0, and assume the line related to the first element which has value 1 after these elements is the left boundary of the number plate. The right boundary can be found by in same method by check the array from its last element.

### 4.3.4. Number Plate Localization Based on HSV Color Space

There is always a blue rectangle part with the flag of the European Communities and letters "IRL" on the left side of the number plate. So, after isolate the number plate, we can confirm whether the sub-image contains the number plate by check whether there is a blue rectangle in the left side of the sub-image. Because the height of the blue rectangle is also the height of the number plate, so the top line and the bottom line is also the top and bottom boundary of the number plate. Also because the aspect ratio of number plate is fixed, so the length of the plate can be calculated after we know the height of the plate.

It is very complicate and time-consuming to identify colors in RGB color space, so we use HSV color space instead.

HSV can be thought of as describing colors as points in a cylinder (called a color solid) whose central axis ranges from black at the bottom to white at the top, with neutral colors between them. The angle around the axis corresponds to "hue", the distance from the axis corresponds to "saturation", and the distance along the axis corresponds to "lightness", "value" or "brightness" [16].

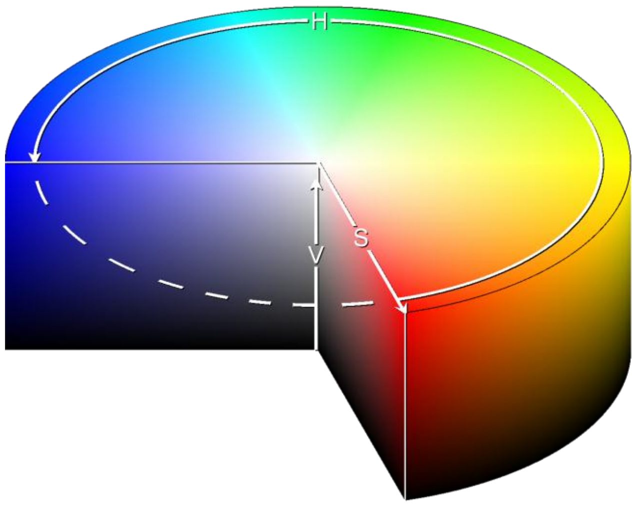


Figure 16 HSV arranged as a cylinder [16]

#### Conversion from RGB to HSV

First convert the value of R (red), G (green), B (blue) from 0-255 to 0.0-1.0, let *max* be the greatest of R, G, B, and *min* the least. The value of H (hue), S (saturation), and V (value) can be calculated as following [16]:

$$H = \begin{cases} \left(6 + \dfrac{G - B}{MAX - MIN}\right) \times 60°, if \quad R = MAX \\ \left(2 + \dfrac{B - R}{MAX - MIN}\right) \times 60°, if \quad G = MAX \\ \left(4 + \dfrac{R - G}{MAX - MIN}\right) \times 60°, if \quad B = MAX \end{cases}$$

$$S = \frac{MAX - MIN}{MAX}$$

$$V = MAX$$

**Find the blue rectangle area in the sub-image**

After convert RGB to HSV, scans the whole sub-image to find the blue rectangle area by check the value of H (hue) in each pixel.

Figure 4-1 hue region for different value

| Value of H (0°-360°) | Hue Region |
|---|---|
| 0°-60° | Red-Yellow |
| 60°-120° | Yellow-Green |
| 120°-180° | Green-Cyan |
| 180°-240° | Cyan-Blue |
| 240°-300° | Blue-Magenta |
| 300°-360° | Magenta-Red |

## 4.4. Character Segmentation and Recognition

### 4.4.1. Character Segmentation

The format of Irish number plate is YY-CC-SSSSSS where YY is a 2 digit year, CC a 1 or 2 letter county identifier and SSSSSS is a 1 to 6 digit sequence number. So the length of number plate is from 4 characters (2 numbers + 1 letter +1 number) to 10 characters (2 numbers + 2 letters + 6 numbers). There are always gaps between characters in number plate, so vertical projection method can also be used to separate characters.

### 4.4.2. Character Segmentation Based on Vertical Projection

First scans the number plate area line by line in vertical direction, count the number of white pixel in each line. Because the projection of numbers and letters are continues, then if the number of white pixel in n adjacent lines less than a threshold value, we can assume these lines is a gap area between characters.

Algorithm:

1) Define an array with value 0 for its elements to store the projection information of each vertical line.
2) Scan the sub-image line by line in vertical direction. If the number of white pixel > n, then change the value of the corresponding element in the array into 1.
3) After the whole sub-image scanned, check the array from its first element, if one element has value 0 and next element has value 1, we can assume these lines are the beginning position of a character, or vice versa.

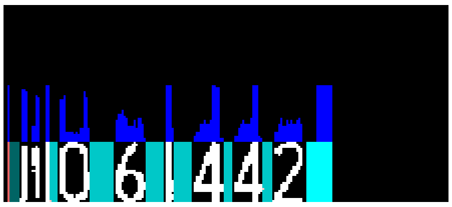
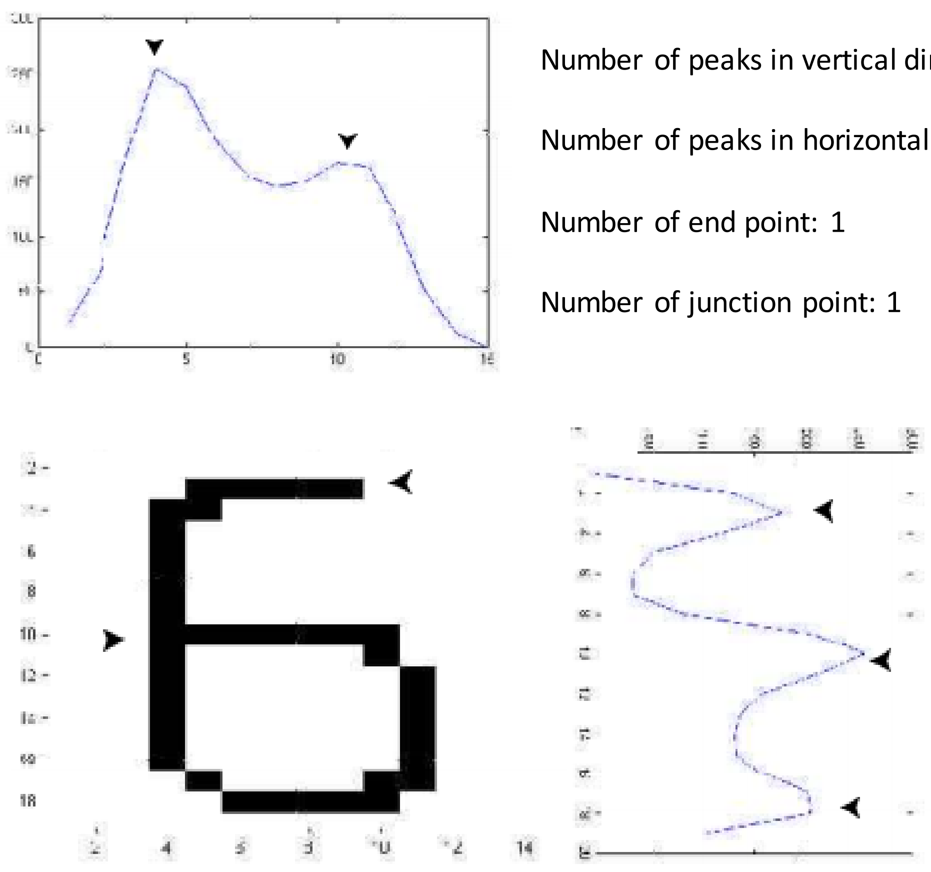


Figure 4-17 vertical projection of a number plate[18]

### 4.4.3. Optical Character Recognition

#### 4.4.3.1. Projection Method

This method first each character bitmap is projected in both vertical and horizontal direction, and then count number of peaks in vertical direction, the number of peaks in horizontal direction, the number of end points of the character and the number of junctions of the character.



Number of peaks in vertical direction: 2

Number of peaks in horizontal direction: 3

Number of end point: 1

Number of junction point: 1

Figure 4-18 vertical and horizontal projection for character “6” [19].

The algorithm used to identify character “6”:

1) The character bitmap might contain all characters. The algorithm starts with determining the number of peaks in the vertical direction and the number of peaks in horizontal direction.
2) The characters {5, 6, 8, 9, B, G} all have 2 vertical peaks and 3 horizontal peaks. The other characters cannot be contained in this bitmap. The second step of the recognition

algorithm is determined by the result of the first step. In this case, the algorithm determines the number of endpoints and junctions.

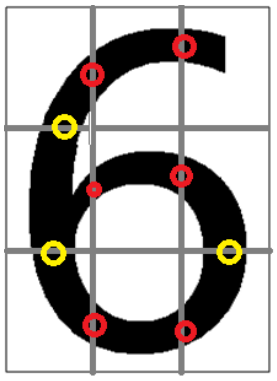
3) From the remaining characters {5, 6, 8, 9, B, G}, the characters {6, 9} are the only characters with 1 endpoint and 1 junction. The third step of the classification algorithm is dependent on the result of the second step. In this case, the algorithm determines the row position of the endpoint with respect to the junction.
4) From the remaining characters {6, 9} is {6} the only character that has an endpoint that is higher than its junction.

Conclusion: the character contained in this bitmap is a 6.

### 4.4.3.2. Grid Feature Method

In this method, two features are extracted. The first one grid feature is extracted by divide the bitmap into 3*3= 9 small grids and counts the white pixels in each grid to form a 9-dimentinal vector. Another one is cross-point feature which get by drawing two vertical lines onto the bitmap to divide it into three equal parts in vertical direction, and draw two horizontal lines onto the bitmap to divide it into three equal parts in horizontal direction, then count the number of cross-point for each line to form a 4-dimensional vector. Connect the two vectors together to form a 13-dementional vector. Use this method we first create vector templates for all the characters. So, when need to identify a character, first get the 13-dementional vector from the bitmap and compare it with the templates to find the most similar template [20].

The following figure gives an example of grid feature method for character “6”.



Apply gird feature method on character “6” to form a 13-demantional vector:

| Dimensional number | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| value | 24 | 36 | 10 | 52 | 32 | 40 | 31 |
| Dimensional number | 8 | 9 | 10 | 11 | 12 | 13 | |
| value | 28 | 38 | 3 | 3 | 1 | 2 | |

Figure 4-19 applying grid feature method on character “6”

### 4.4.3.3. Conclusion

the problems of projection method that are very much alike is the characters B and 8, 0 and D, and 2 and Z are often not distinguishable, but the characters S and 5, V and Y, and C and G are.

To avoid these problems we can first classify the character. The only letters in number plate is the 1 or 2 character county identifier, so we can first divide the plate to three parts: 2-digit year part, 1- or 2-letter part and 1- to 6-digit part, and then recognize them part by part, this can improve the recognition rate.

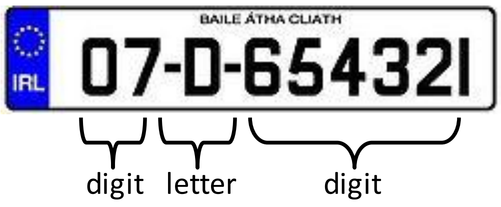


Figure 4-20 Irish number plate

Grid Feature Method makes full use of the distribution information of pixels in images. On the other hand, it does not require thinning the characters, thus avoiding the broken and adhesions of strokes caused by thinning, so that increased the rate of character recognition.

# 5. Programming Languages

## 5.1. Programming Language

- **C**

C language is more advanced than assembly language. It uses a process-oriented programming method. At the same time, it is kind of machine-oriented programming language and still very close to assembly language. It is one of the most efficient high-level programming languages. Its platform compatibility is very good, and almost all systems have the C language compiler. It is suitable for developing operating system and other programming which direct operate hardware and it is also suitable for all kinds of programmer, especially for beginners.

- **C++**

With the continuous expansion of the size of the software, people found that the traditional use of: “data structure + algorithm” structured programming model has been difficult to adapt to the development of the software. At this time, the object-oriented programming idea was attracted programmers’ attention. The first C + + languages was published by AT & T's Bell Labs in 1985. It is based on C language, and adds the object-oriented thinking. It is support three programming modes: process-oriented, object-oriented and template. It has become one of the most popular programming languages.

- **Pascal**

Pascal language was designed by Nicolas Wirth in the early 1970s. Pascal was originally strictly designed to be used in teaching, eventually, a large number of advocates to promote it into the commercial programming. Compared to the previous programming language, Pascal language is a structured language, it has a wealth of data types and control structure, and it is easy to understand, it is particularly suitable for teaching. Pascal language is also a self-compiled language, which greatly enhanced its reliability.

- **Java**

Java was launched by SUN company during SUN WORLD 95 MEETING. Though java is not an official sequel of C++, it uses a large member of grammar from C++. It threw away a lot of the complex function of C++ to form a compact and easy-learned language. Unlike C++, Java mandatory object-oriented programming. "Virtual machine" mechanism, garbage collection and no pointer make it easy to achieve a reliable application.

- **C#**

C# is an object-oriented language which works with Microsoft .NET framework, and it integrates the advantages of a lot of other language, like Java, C++, VB and Delphi. It not only can be used for the development of WEB services, but also the development of powerful system-level program.

## 5.2. Comparison of Common Programming Language

Table 5-1 advantage, drawback and portability of C

| Language Name | C |
|---|---|
| Advantage | 1) Process-oriented development and function-centred. Simple and effective.<br>2) Machine-oriented, so that users can manipulate the machines more efficient.<br>3) High-performance, widely used in all kinds of computer fields. Very effective for simple project. |
| Drawback | 1) The ability of package data is week, grammar is not strict restrictions, which make the C language a lot of weaknesses in data safety.<br>2) Control the machine too strong means depends on machine too strong. Because of too concentrate on efficiency, when we use C to programming, we should consider the machines mach more than the problem itself.<br>3) Unsuitable for develop large project. |
| Portability | The portability of C language limited to process control, memory management and simple document handling. Other things related to the relevant platform. |

Table 5-2 advantage, drawback and portability of C++

| Language Name | C++ |
|---|---|
| Advantage | 1) Much better than C when develop large project.<br>2) Has a good support for object-oriented mechanism.<br>3) Universal data structure. |
| Drawback | 1) Very large and complex.<br>2) Grammar is not strict restrictions as C language.<br>3) Slower than C. |

| | |
|---|---|
| Portability | Much better than the C language, but still not very optimistic. Because it has the same shortcomings with the C language. |

Table 5-3 advantage, drawback and portability of Pascal

| Language Name | Pascal |
|---|---|
| Advantage | 1) Easy to learn.<br>2) Has a very good platform (Delphi). |
| Drawback | The standard of language is not accepted by compiler developers. |
| Portability | Very poor. Language functions changes in different platform. |

Table 5-4 advantage, drawback and portability of Java

| Language Name | Java |
|---|---|
| Advantage | 1) Binary code can be ported to other platforms.<br>2) Application can run on the webpage.<br>3) Its class libraries are very standard and extremely strong.<br>4) Automatic allocation and garbage collection can avoid leakage of resources. |
| Drawback | Use a "virtual machine" to run portable byte code rather than the local machine code, its compile and running speed are much slower than other local application. |
| Portability | The best. Low-level code has a very high portability, but a lot of new features are unstable in some platform. |

Table 5-5 advantage, drawback and portability of C#

| Language Name | C# |
|---|---|
| Advantage | 1) Simple, easy to learn and understand.<br>2) No pointer.<br>3) Garbage collector can manage memory automatically. |
| Drawback | Only few components and libraries can be used now. |
| Portability | Not very good. Software developed by using C# requires .NET runtime as basis, and it only can work in Windows system. |

## 5.3. Conclusion

After compare the advantages, disadvantages and portability of the above language, I decide to use Java as my programming language this year. Because Java not only has extremely powerful class libraries and the best portability, but also very easy to learn and use. So, use Java as my programming language can reduce the difficult of my project and also enables me to learn a new popular language.

# 6. Bibliography

[1] http://www.licenseplaterecognition.com/#Technology

[2] http://www.virage.com/security-and-surveillance/products/anpr/index.htm

[3] http://appian-tech.com/products/anpr-software/talon

[4] http://en.wikipedia.org/wiki/Automatic_number_plate_recognition

[5] http://en.wikipedia.org/wiki/Pixel

[6] http://en.wikipedia.org/wiki/RGB#

[7] http://en.wikipedia.org/wiki/Grayscale

[8] http://en.wikipedia.org/wiki/Thresholding_(image_processing)

[9] http://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm

[10] http://en.wikipedia.org/wiki/Sobel_operator

[11] J. Canny, A computational approach to edge detection, IEEE Trans. on Pattern Analysis and Machine Intelligence, 1986, 679-698

[12] http://en.wikipedia.org/wiki/Kirsch-Operator

[13] http://en.wikipedia.org/wiki/Sobel_operator

[14] http://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm

[15] http://www.craigsplates.ie/irish-number-plate-law

[16] http://en.wikipedia.org/wiki/HSV_color_space

[17] YeChenZhou, JanJie, XuanGuorong. Number Plate Character Recognition. Shanghai JiaotongDaxue Xuebao, 2000, 672-675

[18] Haralick R.M，Shapiro L.G. Survey: image segmentation techniques. Computer Vision, Graphics & Image Processing, 1985, 100-132

[19] Lee J Nelson. Recent Advances in License Plate Recognition. Advanced Imaging, 2002, 18-21, 53

[20] Seong Whan Lee, Dong-June Lee, Hee-Seon Park. New methodology for ray-scale character segmentation and recognition. IEEE Transactions on pattern Analysis and Machine Intelligence, 1996, 1045-1050

[21] Barroso J, Rafael. A Number Plate Reading Using Computer Vision[J].IEEE