License Plate Recognition at Close Distances

Report for BTech 451

by

Muhammad Saad Malik

Supervisor: Professor Reinhard Klette

Tamaki Campus
Department of Computer Science
The University of Auckland
New Zealand

October 2014

For my mum and sister

Abstract

This mid-year report summarises my work on license plate recognition in the first semester of 2014.

Keywords: License plate detection, computer vision, character recognition.

Acknowledgments

I thank Professor Reinhard Klette, Dr. S. Manoharan, and Terry Sun for support while doing this project.

Saad Malik
Auckland
August 14, 2014

Contents

Chapter 1
Localisation Basics

This chapter briefly recalls some basics of image processing and analysis as used in this report, it reviews the license plate detection and segmentation (i.e. into characters) subjects, and outlines the approach followed in this project.

1.1 Image Processing Basics

In this section I will briefly recall basics of image processing as of relevance for my project.

Histogram. A histogram of an image shows tabulated frequencies of image pixel values. See Figure 1.1 for an example. A histogram gives useful information about the tonal distribution in an image. This information can be used to adjust an image accordingly. Contrast and exposure adjustment are some examples of using a histogram. In computer vision a histogram is a compelling tool for image analysis. The tabulated frequencies can be studied in several ways to help with certain problems. For example, a histogram can be studied for pattern recognition and image segmentation. For pattern recognition a histogram helps because patterns can be easily identified in the tabulated frequencies. For image segmentation an image can be segmented by finding high or low areas of contrast through the image histogram. A histogram has several other applications and in section 1.2 we will see a few of them in use.

Horizontal and Vertical Projection Histograms. The horizontal and vertical projections are simple histograms concerned with calculating the number of pixels (of some intensity) in each row and column of an image.This technique is useful for image segmentation. Please note that horizontal and vertical projections are different to normal image histograms.

Connected Component Analysis (CCA). CCA also known as blob extraction is a very useful and an essential technique used to find objects in an image. The method

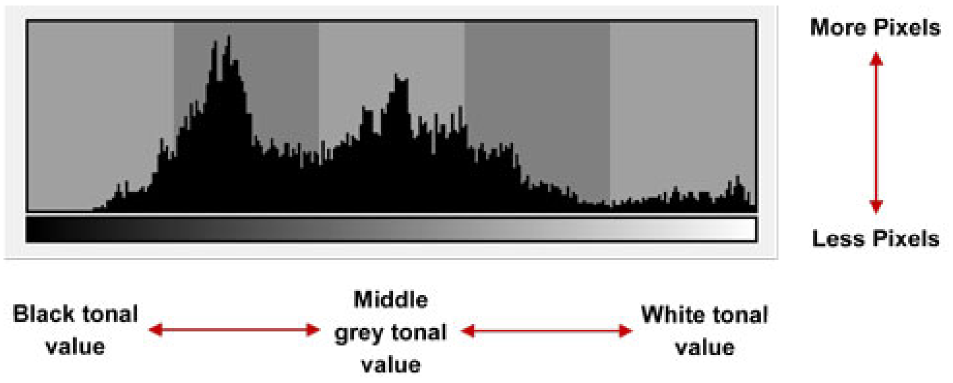

Figure 1.1: Example of an image histogram [3].

simply scans an image looking for regions of pixels that have similar intensities [2]. The detected regions are known as connected components or blobs. The blobs have several useful properties such as area, shape and location. Due to these properties the blobs can be processed and recognised as a certain object and then be easily extracted out of the image. However, the downside of CCA is that it can not detect specific objects on its own. This is because CCA is only able to detect objects that have some form of pixel connectivity (such as similar pixel intensities). Therefore, using the CCA method requires major pre-processing as well as post-processing to be able to recognise and extract objects. In the case of license plate detection, CCA is a popular technique. This is because a license plate forms into a simple rectilinear shape which CCA can detect easily if pre-processing is done correctly. In the next section we will see examples of CCA in use.

Kernel. A kernel is typically a small matrix of numbers. It is used in image convolutions [4]. An image convolution is a simple process of multiplying two matrices and producing an output matrix. The two matrices are the input image and the kernel. The kernel can be used to have different set of values to produce different effects for the input image. Kernels are fundemental in image processing and several filters and operators make use of them[4]. See Fig. 1.3 for an example of a kernel.

Basic Edge Detection. Edges are important features in an image, which are used for image analysis purposes. Edge detection applies mathematical methods to identify pixels at brightness discontinuities. Such pixels define edges. Edge detection is an example of feature detection. There are a number of edge detectors in image processing but I will only briefly explain the Sobel operator and the Canny edge

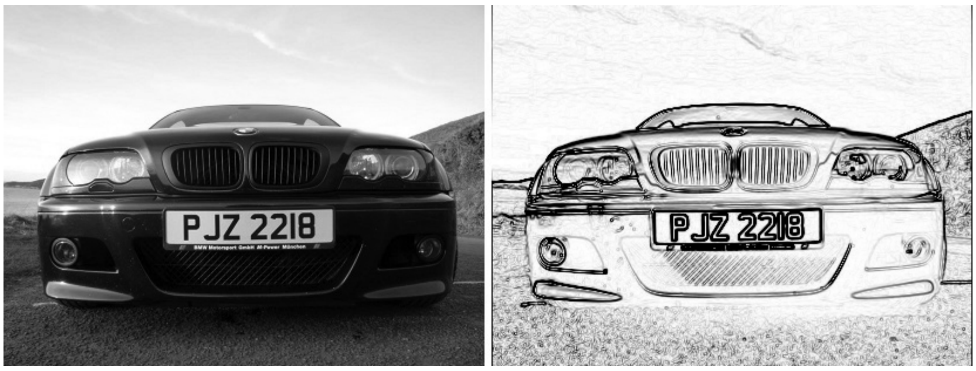

Figure 1.2: *Left:* Grey-level input image. *Right:* Result after applying the Sobel operator.

detector.

Sobel Operator. The *Sobel operator* is a discrete differentiation operator [5]. It detects edges by finding the approximate absolute gradient magnitude at each point of an input grayscale image. The sobel operator is applied by using the two kernels shown in Fig. 1.3. The two kernels are applied seperately at each pixel and their results are combined to compute the approximate absolute gradient magnitude. Applying the kernels seperately also means that seperate gradient measurements for vertical and horizontal edges can also be produced [6].

The value of the Sobel operator at pixel location (x, y) is simply a first-order partial derivative calculation, and therefore equals [7]:

$$S(x, y) = |S_x(x, y)| + |S_y(x, y)| \tag{1.1}$$

Canny Operator. The *Canny operator* detects edges and produces thin edge segments as the final output. The thin edge segments only have a width of one pixel[CCV]. The operator goes through multiple stages to process an image. An image is firstly converted into grayscale, then it is smoothed by Gaussian smoothing. An operator such as *Roberts Cross* or *Sobel* is then applied to the image to provide estimates of the edges. The next step known as *non-maxima suppression* tracks along the ridges of the estimated edges and thins them by setting pixels that are not part of the ridge to 0. The tracking process makes use of two thresholds T_1 and T_2 where T_1 is greater than T_2. The tracking begins on a ridge at a pixel with a value greater than T_1. Tracking stops when a pixel with a value less than T_2 is found.This process applies hysteresis with the use of the two thresholds. This means the outcome of a pixel is not only de-

$$\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}\ /1 \qquad \begin{array}{|c|c|c|}\hline -1 & -2 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 2 & 1 \\ \hline \end{array}\ /1$$

Figure 1.3: Kernels of the Sobel operator [7]. *Left:* S_x . *Right:* S_y.

cided by the previous tracked pixel value, but pixels before that are also considered [8] [9].

Hough Line Transform. *Hough line transform* is a technique which is used to detect straight lines. The *standard Hough line transform* expresses lines in the polar system. These lines are given by the following equation

$$r = x \cos\theta + y \sin\theta \tag{1.2}$$

In the equation 1.2 r is the length (distance to origin) of the line in the interval [0, r_{max}] with r_{max} as given in 1.3 [10]. The θ is the angle of the line with respect to the x-axis, in the interval [0, 2π). Using some point from an image with the equation 1.2 produces sinusoid curves. Lines can be detected when the curves for all points in an image are produced and analyzed for intersections. This is because the intersections between the curves suggest the points of the possible lines. Therefore, in general standard Hough line transform detects lines by counting the number of intersections in the curves produced by each point in the image[10] [11] [12].

$$r_{max} = \sqrt{N_{cols}^2 + N_{rows}^2} \tag{1.3}$$

Spatial Domain. The *spatial domain* is the normal image space. If an image is projected to some scene and changes are made to the image, the changes will directly project to the scene. This means that distances in the image correspond to real distances in the scene [13].

Discrete Fourier Transform. A *Fourier transform* decomposes an image into sine and cosine components. It is used to convert images in the spatial domain into the *Fourier domain*. Hence, the Fourier domain consists of the output transformed image. Points in the Fourier domain image represent certain frequencies in the spatial domain image. A *discrete Fourier transform* is a sampled Fourier transform. Therefore, not all frequencies from the spatial domain image are taken for the transformation. The sample taken is just large enough to represent the entire spatial domain image. This type of transform has several applications in image processing. One main reason for using this transform is to be able to study the geometric characteristics of a spatial domain image [14].

Discrete Wavelet Transform. A *discrete wavelet transform* is an alternative to the Fourier transform.

Gaussian Smoothing Operator. *Gaussian smoothing operator* is an essential and widely used operator in image processing. It is used to blur images. The purpose of blurring is to reduce extra details and noise. The Gaussian smoothing operator works by applying a special kernel filter. This kernel is defined by the samples of the 2D *Gauss function* which is the product of two 1D Gauss functions. The Gauss function is defined as

$$G_{\sigma,\mu_x,\mu_y}(x,y) = \frac{1}{2\pi\sigma^2} \times e^{\frac{(x-\mu_x)^2+(y-\mu_y)^2}{2\sigma^2}} \tag{1.4}$$

where σ is the standard deviation and μ_x and μ_y are the expected values for x and y [15].

Structuring Element. A structuring element is very similar to a kernel. It is basically a matrix of binary values. The difference between a structuring element and a kernel is that structuring elements are used for morphological operations, not for image convolution. The structuring element has the properties of size, shape and origin. The size of the structuring element is given by the dimension of the matrix, the shape is given by the pattern of ones and zeros in the matrix and the origin is usually a pixel within the matrix such as the center pixel [16]. When the structuring element is applied to an image it compares its pixel values with the corresponding pixel values of the image. The result depends on the type of operation being applied. A structuring element is said to be *fitting* when all its values are exactly the same as the underlying image pixels. It is said to be *hitting* if at least one of its values are the same [16].

Morphological Operations. Morphological operations are used to process an image based on its shapes or morphology of its features. The operations are especially suited to work with binary images. All morphological operations are applied by using a structuring element. Following will be an overview of some of the types of morphological operations.

Erosion. Erosion is a very basic morphological operation that erodes away the foreground pixels. The effect of erosion shrinks or reduces the regions (i.e. objects) in the foreground. This means the background grows, holes and gaps between regions become larger. Small details and noise is also eliminated. In erosion, for a pixel to be eroded the structuring element must fit the underlying image pixels.

Dilation. Dilation is another very basic morphological operation which works completely opposite to erosion. Dilation increase the foreground pixels. Hence, the effect of dilation grows or expands the regions (i.e. objects) in the foreground. The background reduces, holes and gaps between regions become smaller. In dilation for a pixel to be dilated the structuring element only has to hit the underlying image pixels [16].

Opening. The opening operation is a composite function. It is simply an erosion operation followed by a dilation operation. It helps open up gaps between regions by eliminating thin pixel connections [16].

Closing. Similar to the opening operation, the closing operation is also a composite function. Closing however is a dilation operation followed by an erosion operation. It helps close up gaps between regions by adding pixels in small gaps [16].

Thinning. Thinning in image processing is a morphological operation that simplifies images by producing very thin curves of the input image. The operation works on binary images and produces binary images as output. The thinning operation is also applied to an image by using a structuring element.

Thresholding. In image processing the simplest method of image segmentation is thresholding. Thresholding on grayscale images produces binary images. There are several methods to thresholding an image, here I will give an overview of basic binary thresholding and also otsu thresholding performed on grayscale images.

Binary thresholding: Sets the pixels to zero if the value of the current pixel is less than a given threshold. If the value of the current pixel is greater than the selected threshold the pixel's value is set to a maximum value such as 255.

Otsu's Binarization. Otsu binarization assumes the image consists of two classes of pixels, the foreground and the background. Otsu's method calculates the optimal threshold of the image by separating the two classes of pixels and minimizing the intra-class variance the method works by the formula [17]. See Fig. 1.4

1.2 Methods for License Plate Detection

The first step in the license plate reading system is license plate detection. In this section I will give an overview of different methods of license plate detection.

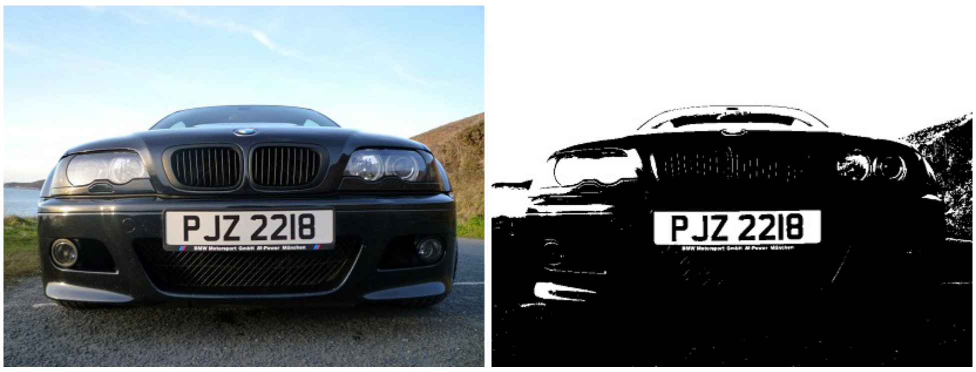

Figure 1.4: Example of Otsu binarization. *Left:* Original image [24]. *Right:* Result of Otsu's binarization.

Method 1. [18] employs two different methods using CCA to detect license plates. The first method works by using prior knowledge of license plates. Detecting of a white frame is carried out by using CCA and some candidate frames are collected. Then a few steps are taken to validate if a certain region is a license plate. The second method is used when the first method is not able to detect a license plate. The second method works by searching for large numerals or characters in the image seqeunce. This search is also carried out by using CCA. Candidate regions are collected and are classified as characters based on prior knowledge. This includes the properties of the characters such as their average area and average distance between each character on a license plate. The average success rate of this algorithm reported in [18] is 97.16%.

Method 2. The approach taken by [19] to detect license plates involves thresholding and analysis of black to white pixel transitions. [19]'s method firstly applies a threshold of 170 to the input image. This threshold value is predetermined from experimentations with their particular scenario. Then some additional thresholding is again performed to remove further unwanted data. This additional thresholding is done through an analysis of black to white pixel transitions. The focus is to count the number of transitions in each row of the remaining data. If the count is less than 14 for any row, the pixels in the row are all set to 0 or black. The count of 14 transitions is calculated through the assumption that each license plate has at least 7 characters. The additional thresholding results in the license plate being somewhere in between these thresholded black rows. To complete the localisation the remaining data is searched for the longest vertical line of white pixels. This means that when searching the image from left to right, the first longest vertical line of white pixels is the left edge of the license plate. Similarly the last longest vertical line of white pixels is the right edge of the license plate. By finding the vertical lines there is enough information gathered to extract out the license plate. Some checks regarding the aspect ratio are also performed for more accuracy.

Method 3. [20] detects license plates by studying the variance of the input image. A few pre-processing steps are taken to reduce noise and increase contrast before localisation. Sobel edge detection is performed on the image and the image is split into 25 blocks each of the same size. The size ofcourse is large enough to fit a license plate. Variance in each block is calculated and a threshold variance value is obtained by the simple formula $V_{th} = (V_{max} + V_{min})/2$, where V is variance. Each block is then compared with the obtained threshold variance value. Any block below the threshold variance is removed. Hough line transform is performed on the remaining blocks to remove any unnesscary long lines. The last step consists of some morphological operations and CCA to extract out the license plate from the remaining blocks.

Method 4. In [21] the approach to detecting license plates is mainly concerned with the sobel filter and CCA. After some pre-processing of the input image a vertical sobel filter is applied to the image. The gradient of the filtered image is obtained for the purpose of validating the possible license plate regions. The filtered image is then thresholded using otsu's binarization and analyzed using CCA. Hence the candidate regions are extracted and validated through the formula $S_G = w_1 \times S_{Grad} + w_2 \times S_{Ratio} + w_3 \times S_{Shape} + w_4 \times S_{GTrack}$. Where S_G is the sum of priors the gradient, aspect ratio, shape and tracking criterion for the candidate. The candidate which has the best value of S_G is processed and the rest are stored for future considerations.

Method 5. [22] uses discrete wavelet transform with a sliding window to detect license plates. Firstly, a grayscale image is transformed by using a 1-level 5/3 discrete wavelet transform (DWT). The HL and LH subbands are used to locate the license plate as they provide vertical and horizontal information about the image. Using the HL subband a histogram of heights of all vertical lines is calculated. This is followed by calculating the average of the heights. Then noise in the HL subband is reduced by removing any vertical lines that are too long or below the average line height. The next step reduces noise in the LH subband. This is done by calculating the average of the coefficients in the LH subband and then removing all coefficients below this average. Then a transition analysis is carried out in the HL subband to give an approximate location of the license plate. Along with this step the LH subband is also scanned for horizontal lines to provide a more accurate approximation.

These steps result in a detected candidate region. Since the candidate region is only an approximation further processing is required. A block around the candidate region is created to be sure that the entire license plate is inside this area (i.e. the block's area). Then to finalize the detection a sliding window is applied within the block. The sliding window is created to have an average size of a license plate. The last step of this process is to convert the HL frequency domain back to the spatial domain. [22] has reported this method to have a success rate of 97.33%. The success rate is obtained from testing the algorithm on 292 images.

1.3 Methods for Segmenting License Plates

After a license plate has been successfully detected, the next step is segmentation. Segmentation is performed to seperate and extract each character out from the license. This step is crucial because character recognition algorithms depend on the quality of license plate segmentation. The segmentation process can be a little challenging in most cases. This is because most real world situations result in license plate images that are either tilted or out of perspective. The different types of illumination and noise also needs to be considered. Therefore, for license plates to be segmented correctly the process must first deal with these problems. Following are different proposed methods of license plate segmentation.

Method 1. [18]'s segmentation process starts by resizing all detected plates to 100 x 200 pixels. The next steps are taken to correct the horizontal tilt. They use CCA to find large numerals in the license plate and find the center point of each detected numeral. Then calculating of the tilt angle between two central points is done and the average tilt angle is obtained. Using the average tilt angle a 2D rotation method is applied to horizontally correct the license plate. To fix the vertical slant of the license plate the area of the large numerals is extracted. The width of the horizontal projection is calculated from the extraction. Then the extracted area is rotated from $-45°$ to $45°$ while the projection value and rotation angle is recorded. The projection minimum is found from the projection data and it is used to correct the vertical slant. The correction method used is the horizontal pixel movement method. This method basically shifts pixels horizontally according to the formula

$$d_x = (y - \frac{1}{2}H)/\tan(\zeta) \tag{1.5}$$

where d is the distance each pixel moves horizontally, x and y are the pixel coordinates, H is the height of the image and ζ is the vertical slant angle. After the license plate has been put into correct perspective, it is processed to enhance its contrast. To do this [18] uses a modified version of the gray statistical approach (GSA).

Details of this GSA are omitted here. Once the enhancement has been complete, the plate is binarized using an improved bersen algorithm proposed by [ ]. Lastly, the simple projection technique is used to segment the license plate into seperate characters.

Method 2. [21]'s approach to segmentation is CCA. CCA is applied to the license plate and prior knowledge is used to extract the character. The main priors include linearity, parallelism and the distance between characters. The tilt of the license plate is corrected by using metric rectification. The details of the algorithm are not provided in [21] but the basic idea provides sufficient information on how it works.

Method 3. [23] uses vertical and horizontal projections to segment the license plate. The difference in their approach from other papers that use projections is in the pre-processing of the license plate. They use an object enhancement algorithm to enhance the license plate characters. The effect of the object enhancement algorithm weakens background pixels while strenghening the foreground pixels. [23] estimate that 20% of the license plate image are character pixels. The goal is to enhance these pixels which in their object enhancement algorithm takes two steps. The first step is to scale the gray level of all pixels into the range of 0 to 100. Then second step sorts all the pixels by their gray level in a descending order and multiplies the top 20% of the pixels by 2.55. This way the object pixels are enhanced whereas background pixels remain weak. From this enhancement the characters can be more accurately segmented using the projection method.

1.4 Selected Approach

In this section I will discuss my selected method of license plate detection and how it works.

License Plate Localisation. My approach to license plate detection is somewhat similar to the approach in [21]. My approach however involves mathematical morphology. In the following I will go over the steps taken in my approach to detect license plates.

Step 1. My algorithm starts off with an input image which is then converted to grayscale. Since all images consist of some form of noise, the focus of this step is to reduce noise and smoothen the image. This is necessary as it improves the results of further processing on the image. Therefore, to reduce noise I apply a 5×5 gaussian blur to the image.

Step 2. In this step I process the image to detect edges. To do this I use the Sobel operator. License plate characters have more dominant and meaningful vertical edges in comparsion to horizontal edges. Hence, I apply the Sobel operator in the vertical direction. This means d_x is set to 1 whereas d_y is set to 0 (i.e. kept as a constant). The kernel size I used is 3×3. Note that the gaussian blur applied in step 1 helps this step of edge detection.

Step 3. After edge detection I apply a closing morphological operation to the image. The closing operation helps dilate white regions as well as erode away small unwanted details. The structuring element used to apply this operation is given a similar aspect ratio to a license plate. This is so the resulting image produces rectilinear shapes (i.e. like the shape of a license plate). Once this is done one can realise grayscale information is no longer required. Therefore to simplify the image I process it using Otsu's binarization. When the binary image is obtained I apply two more morphological operations. These are an erosion operation followed by a dilation operation. One could simply apply an opening operation but the structuring element will then have to be the same for both operations. For my approach I require two different structuring elements so the opening operation on its own would not work. For erosion I use the same structuring element as for the closing operation. Using erosion again further helps in removing shapes that are too small to be a license plate. It also helps break apart large shapes which may contain the license plate (i.e. due to the application of these morphological operations the license plate may sometimes join into other objects). Then for dilation I use a larger and more square shaped structuring element. There are two reasons I applied dilation after the erosion operation. First reason is to simply bring back important removed information. For example, a license plate may already be in perfect shape but by applying erosion it may lose its shape. This is where dilation can be used to recover the shape. The second reason is because I needed the remaining shapes to gain more area around them. This is because if some shape is eventually a license plate it should cover as much of the license plate as possible. This reason also explains why I used a larger structuring element for dilation.

Step 4. By the end of step 3 an image of several regions which may consist of a license plate is produced. The focus in this step is to extract out regions which are most likely to be license plates. To do this I have used connected component analysis (CCA). Firstly I calculate the minimum rectangular area for each region from which I obtain rectangles that represent each region. Then I compare the properties of the rectangles to the properties of a license plate. Based on the comparison I eliminate regions which are unlikely to be a license plate. The properties I use to determine if a region is a license plate or not include the area, the angle and the aspect ratio of an average license plate. If the area of a rectangle is either too big or too small the corresponding region is ignored. If the angle of a rectangle suggests the corresponding region is tilted too much (e.g. $75°$), then that region is also ignored. Lastly, if the aspect ratio of a rectangle is quite different from a license plate's aspect ratio then that region is also ignored. In the end regions that correctly meet the properties of a license plate are extracted from the original grayscale image using CCA. The candidate regions are then pre-processed before they are validated as a license plate.

Candidate Region Pre-processing. Pre-processing the candidate regions is required to get a better result when validating them. Candidates not pre-processed will have noise and other undesired conditions that may produce incorrect validations. The main focus of this pre-processing is to cleanly convert the candidate regions into a binary image. The first step is to reduce noise. For this I apply a 3×3 gaussian blur as I did previously. Then I use a simple contrast enhancement method to give more dominance to the darker pixels in the region. I do this because if a candidate region is in fact a license plate then the contrast enhancement will make the characters more dominant. The contrast enhancement I apply is given the formula

$$g_{(x,y)} = \alpha \cdot f(x,y) + \beta \tag{1.6}$$

where g is the output and f is the input and α and β are the gain and bias which are known to control the contrast and brightness. I have predefined α as 2 and β as -150. I obtained these values through experimentations with different license plates. Finally to convert the candidate regions into a binary image I use Otsu's binarization as I did in the localisation process.

License Plate Validation. License plate validation is a process of validating images as a license plate. In this case these images are the pre-processed candidate regions extracted in the localisation process. My approach to validate a region as a license plate requires two tests. The candidate region must pass both tests in order to be classified as a license plate.

Test 1. A candidate region must contain an internal rectangular shape with an aspect ratio of a license plate. I use this condition for validation because every license plate has an outlined border. So if a region has an internal rectangle I can be sure that a border exists around this rectangle. Also please note that here by *internal* I mean a rectangle that is not completely at the boundaries of the candidate region. To perform this test I use CCA on the candidate region looking for a rectangular shape that matches the test condition. If no rectangle is found the test is re-applied to the inverse of the region. This is because sometimes parts of the rectangle are at the boundary of the region and because of this they are not included in the connected component labeling process. It is very unlikely for a candidate region that is not a license plate to have such a rectangle. Therefore, in most cases this test is sufficient enough to validate a candidate region as a license plate. However, a second test is done to ensure correct validation.

Test 2. A candidate region must contain at least two same or similar sized rectangles apart from the rectangle in test 1. I use this condition for validation because all characters on a license plate have the same width and height. If two similar sized rectangles are found then they are most likely to be license plate characters. On average license plates have at least five characters which means there is a good chance two characters are detected if the candidate region is a license plate. This test is performed concurrently with test 1 as both these tests are done through CCA.

Chapter 2

Conclusions

This report outlined my work done so far. I have also tested methods extensively on test data.

In the second semester I will focus on character recognition for reading the detected license plates accurately.

I will also implement an Android app for users that the detection program can be used flexibly at various locations.

Bibliography

[1] Guan, S., Klette, R.: Belief-propagation on edge images for stereo analysis of image sequences. In Proc. *Robot Vision*, LNCS ????, pages 291–302, Springer, 2008

[2] HIPR2. homepages.inf.ed.ac.uk/rbf/HIPR2/label.htm, last visit: 11 August 2014

[3] Go Digital. godigitalslr.com/wp-content/uploads/2012/02/HistogramDiagram.jpg, last visit: 7 August 2014.

[4] HIPR2. homepages.inf.ed.ac.uk/rbf/HIPR2/kernel.htm, last visit: 11 August 2014

[5] Open CV 2.4.9.0 Documentation. docs.opencv.org/doc/tutorials/imgproc/imgtrans/sobel_derivatives/sobel_derivatives.html, last visit: 11 August 2014

[6] HIPR2. homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm, last visit: 11 August 2014

[7] Klette, R.: Concise Computer Vision. Page 63, Springer, 2014

[8] Klette, R.: Concise Computer Vision. Page 64, Springer, 2014

[9] HIPR2. homepages.inf.ed.ac.uk/rbf/HIPR2/canny.htm, last visit: 11 August 2014

[10] Klette, R.: Concise Computer Vision. Page 123, Springer, 2014

[11] HIPR2. homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm, last visit: 11 August 2014

[12] Open CV 2.4.9.0 Documentation. docs.opencv.org/doc/tutorials/imgproc/imgtrans/hough_lines/hough_lines.html, last visit: 11 August 2014

[13] HIPR2. homepages.inf.ed.ac.uk/rbf/HIPR2/spatdom.htm, last visit: 11 August 2014

[14] HIPR2. homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm, last visit: 11 August 2014

[15] Klette, R.: Concise Computer Vision. Page 57, Springer, 2014

[16] COMPSCI 373 Computer Graphics and Image Processing. cs.auckland.ac.nz/courses/compsci373s1c/PatricesLectures/2013/CS373-IP-02.pdf, last visit: 11 August 2014

[17] Klette, R.: Concise Computer Vision. Page 170, Springer, 2014

[18] Wen, Y., Lu, Y., Yan, J., Zhou, Z., Deneen, K., Shi, P.:An Algorithm for License Plate Recognition Applied to Intelligent Transportation System. IEEE Transactions, 2011

[19] Romic, K., Galic, I., Baumgartner, A.: Character Recognition Based On Region Pixel Concentration For License Plate Identification. 2012

[20] Kodwani, L., Meher, S.: Automatic License Plate Recognition in Real Time Videos using Visual Surveillance Techniques.

[21] Thome, N., Vacavant, A., Robinault, L., Miguet, S.: A Cognitive and Video-based Approach for Multinational License Plate Recognition. Machine Vision and Applications, 2011

[22] Wang, Y., Lin, W., Horng, S.:A Sliding Window Technique for Efficient License Plate Localization Based on Discrete Wavelet Transform. Expert Systems with Applications, 2011

[23] Kranthi, S., Pranathi, K., Srisaila, A.: Automatic Number Plate Recognition. International Journal of Advancements in Technology, 2011

[24] Image Library: i126.photobucket.com/albums/p105/DoZZa1/P1010414.jpg, last visit: last visit: 11 August 2014