

VEHICLE LICENSE PLATE DETECTION AND RECOGNITION

**A Thesis presented to
the Faculty of the Graduate School
at the University of Missouri**

**In Partial Fulfillment of the Requirements for the Degree
Master of Science**

**by
GUANGHAN NING**

Dr. Zhihai He, Thesis Supervisor

DECEMBER 2013

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

VEHICLE LICENSE PLATE DETECTION AND RECOGNITION

presented by Guanghan Ning,

a candidate for the degree of Master of Science,

and hereby certify that, in their opinion, it is worthy of acceptance.

Zhihai He, Ph.D., Department of Electrical and Computer Engineering

Tony Xu Han, Ph.D., Department of Electrical and Computer Engineering

Ye Duan, Ph.D., Department of Computer Science

ACKNOWLEDGEMENTS

First of all, I am heartily thankful to my advisor, Dr. Zihai He, whose encouragement, supervision and support enabled me to learn more about video processing knowledge and apply them to this project. Dr. He provides me lots of advice and inspiration on my research and always encourages me to become an independent researcher. Dr. He is very kind to us students and helps us to be better prepared for our future career. Not only have I learned a lot from him how to become a successful researcher but also how to become a better man.

Besides my advisor, I take this opportunity to convey my heartfelt gratitude to my committee member Dr. Tony Xu Han and Dr. Ye Duan. They donated their precious time reviewing my master thesis, and helped me a lot with anything they could.

As a member of Video Processing and Communication Lab, I am grateful to Xiaobo Ren, Chen Huang, York Chung, and Jay Eggert. They are my senior colleagues in the lab and they gave me a lot of help and support. Thanks also go to Zhi Zhang, Yizhe Zhu, Di Wu. They are my most precious peers and I enjoyed our cooperation of any kind, where we always encourage and help each other.

Last but not least, I would like to thank my fiancé and my parents for their unconditional love and support, without whom none of these will ever be possible.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	II
LIST OF FIGURES	V
ABSTRACT	VII
Chapter	
1 INTRODUCTION AND BACKGROUND	1
1.1 Research Topic and Objectives	1
1.2 Challenges	2
1.3 Background and Related Work	3
1.3.1 License Plate Detection	4
1.3.2 Character Segmentation	13
1.3.3 License Plate Recognition	15
1.4 Overview of Our Methods	16
2 LICENSE PLATE DETECTION	18
2.1 Scanning Window	18
2.2 HOG Features	20
2.2.1 Feature Extraction Procedure (for Dense-HOG)	21
2.2.2 Implementation	24
2.3 Support Vector Machine	24
2.4 Non-maximum Suppression	26
2.5 Refinement of the Algorithm.....	29
2.5.1 Edge Information.....	29
2.5.2 Scale Adaption.....	30
2.6 Results and Discussion	30
2.7 Summary	31
3 LICENSE PLATE RECOGNITION	32

3.1 License Plate Alignment Using Color Information.....	34
3.1.1 License Plate Alignment Without Angles.....	34
3.1.2 License Plate Alignment With Angles.....	38
3.2 Plate Binarization Using K-means Clustering.....	40
3.2.1 K-means Clustering.....	41
3.3 Character Segmentation Using an Innovative Histogram-based Model	43
3.4 Digit Characters and Capital Letters Recognition Using Simple Robust Features	48
3.4.1 Using Bag-of-words Model: Voting Schemes.....	49
3.4.2 Using SVM Classifier.....	53
3.5 Dataset and Results	54
3.6 Summary.....	57
4 REAL TIME EMBEDDED SYSTEM	58
4.1 Hardware Part.....	58
4.2 Software Part.....	59
4.2.1 The Main Board Side.....	59
4.2.2 The Child Board Side.....	60
4.3 Implementation.....	62
4.3.1 Kernel Module.....	62
4.3.2 Main Board Program.....	63
4.3.3 Socket and TCP.....	63
4.3.4 Child Board Program.....	64
4.4 Results and Discussion.....	64
5 CONCLUSIONS AND FUTURE WORK	66
REFERENCES	67

LIST OF FIGURES

Figure	Page
Fig. 1. 1 License plate samples from several different provinces of China.	2
Fig. 1. 2 Overview of our proposed method.....	17
Fig. 2. 1 An overview of license plate detection procedure.	18
Fig. 2. 2 Scanning-window and image pyramid.	19
Fig. 2. 3 Flowchart of the Scene Text Recognition framework in performance evaluation.....	20
Fig. 2. 4 Variants of HOG descriptors. (a) Rectangular HOG(R-HOG) with 3 by 3 blocks; (b) Circular HOG (C-HOG) descriptor with central cell divided into angular sections as in shape contexts.....	21
Fig. 2. 5 An overview of HOG feature extraction.	21
Fig. 2. 6 A sketch of local normalization.	23
Fig. 2. 7 The training images of the SVM classifier. Positive.....	25
Fig. 2. 8 The training images of the SVM classifier. Negative.....	25
Fig. 2. 9 Using edge information. (a) The canny edge image; (b) The scanning regions that will be classified by SVM.....	29
Fig. 3. 1 Aligned binary license plate.....	32
Fig. 3. 2 Segmented license plate.....	32
Fig. 3. 3 Segmentation Model.....	33
Fig. 3. 4 The analysis of the best fit model.....	33
Fig. 3. 5 The overview of the license plate recognition process.....	34
Fig. 3. 6 The detected license plate before and after alignment.....	34
Fig. 3. 7 The alignment model. L1-L4 are lines that will slide within a certain range. Region A and Region B are two regions that are symmetrical to each line. Region C is a region in the inner-middle of the detected license plate rectangle.....	35
Fig. 3. 8 The Histograms and their corresponding matrix for two regions.....	36

Fig. 3. 9 Quantize each pixel value into 32 intervals. Pixels whose value of a channel is within the range of the corresponding interval will vote that bin, which forms the 3 by 32 histogram matrix.....	37
Fig. 3. 10 The modes of the angles.....	39
Fig. 3.11 The alignment process: the lines searching for position and angle.....	40
Fig. 3. 12 Results of K-means clustering on aligned license plates.....	41
Fig. 3. 13 The general K-means clustering algorithm.....	43
Fig. 3. 14 The layout of a Chinese license plate.....	44
Fig. 3. 15 The pre-processing of segmentation: (a) The green rectangle shows the alignment result of the license plate. The rectangle will be extended to the red rectangle to ensure some scan range. (b) The pixels of extended region will be assigned 255 as they are considered to be background pixels.....	44
Fig. 3. 16 The histogram of the license plate derived from the vertical scan. (a) The histogram $h(x)$; (b) The corresponding model.....	45
Fig. 3. 17 The variables that will be explored: (scale, x, space). The green segment is character segment, while the yellow segment symbolizes dot segment. The white segments are spaces.....	46
Fig. 3. 18 The process of license plate segmentation: (a)the histogram after thresholding; (b)the raw histogram; (c) the segmented license plate.....	47
Fig. 3. 19 The results of segmentation.....	48
Fig. 3. 20 Result comparison of four voting schemes.....	50
Fig. 3. 21 Flowchart of the 2-level voting scheme.....	53
Fig. 3.22 Sampled results of global alignment, k-means binarization, segmentation and character recognition. The recognition part ignores Chinese characters and output a "*" instead.....	56
Fig. 4. 1 Diagram of the hardware part.....	59
Fig. 4. 2 Software part: main board side. The part within orange rectangle belongs to child board program.....	61
Fig. 4. 3 Software part: child board side. The part within orange rectangle belongs to the main board.....	62

Vehicle License Plate Detection and Recognition

Abstract

In this work, we develop a license plate detection method using a SVM (Support Vector Machine) classifier with HOG (Histogram of Oriented Gradients) features. The system performs window searching at different scales and analyzes the HOG feature using a SVM and locates their bounding boxes using a Mean Shift method. Edge information is used to accelerate the time consuming scanning process.

Our license plate detection results show that this method is relatively insensitive to variations in illumination, license plate patterns, camera perspective and background variations. We tested our method on 200 real life images, captured on Chinese highways under different weather conditions and lighting conditions. And we achieved a detection rate of 100%.

After detecting license plates, alignment is then performed on the plate candidates. Conceptually, this alignment method searches neighbors of the bounding box detected, and finds the optimum edge position where the outside regions are very different from the inside regions of the license plate, from color's perspective in RGB space. This method accurately aligns the bounding box to the edges of the plate so that the subsequent license plate segmentation and recognition can be performed accurately and reliably.

The system performs license plate segmentation using global alignment on the binary license plate. A global model depending on the layout of license plates is proposed to segment the plates. This model searches for the optimum position where the characters are all segmented but not chopped into pieces. At last, the characters are recognized by another SVM classifier, with a feature size of 576, including raw features, vertical and horizontal scanning features.

Our character recognition results show that 99% of the digits are successfully recognized, while the letters achieve an recognition rate of 95%.

The license plate recognition system was then incorporated into an embedded system for parallel computing. Several TS7250 and an auxiliary board are used to simulate the process of vehicle retrieval.

CHAPTER 1

INTRODUCTION AND BACKGROUND

1.1 Research Topic and Objectives

License Plate Recognition (LPR) is a problem aimed at identifying vehicles by detecting and recognizing its license plate. It has been broadly used in real life applications such as traffic monitoring systems which include unattended parking lots, automatic toll collection, and criminal pursuit [5].

The target of this thesis is to implement a vehicle retrieval system for a Chinese surveillance camera, by detecting and recognizing Chinese license plates. It will be useful for vehicle registration and identification, and therefore may further contribute to the possibility of vehicle tracking and vehicle activity analysis. The proposed method includes two main steps:

1) License Plate Detection: Using SVM classifier with HOG features based on a sliding window scheme, scan possible regions detected by edge information, and obtain license plate candidates. Then apply Non-Maximum Suppression (NMS) to finalize the plate locations.

2) License Plate Recognition: The detected license plate will be aligned first, after which its pixels can be successfully clustered by k-means into two classes: background pixels, and the foreground pixels, e.g., the pixels of the characters. The plate is segmented afterwards, into character patches that will be recognized using SVM classifier individually.

1.2 Challenges

The first challenge is plate variation. The plate can be various in location, quantity, size, color, font, occlusion, inclination, and plates may contain frames or screws [1]. The second challenge is environmental variation which includes change in illumination and background. Weather conditions, lighting conditions, and even camera conditions may contribute to the difficulty of this problem.

For Chinese license plates, the font is fixed except vanity plates designed by individuals, which is very rare. Fig.1.1 shows some Chinese license plates from several provinces. The challenge from the font is relieved while all the other challenges like variation in plate location, plate quantity, its size, color, occlusion, and inclination still remains. The weather condition where the images in the dataset are captured is various too. Images are captured in the daytime as well as at night; the weather can be sunny or it can be rainy.



Fig. 1. 1. License Plate samples from several different provinces of China

1.3 Background and Related Work

License plate recognition (LPR) system contribute to applications such as traffic surveillance, traffic law enforcement, automatic toll collection, vehicle parking identification, and vehicle access control in a restricted area. Typically, an LPR system is composed of License plate detection (LPD), license plate character segmentation (LPS) and License plate character recognition (LPR).

License plate detection is commonly the first procedure in a LPR system. It aims at locating the license plate, which provides the following LPR procedure with accurate region information. Instead of processing every pixel in the input image of the system, which is very time consuming, license plate detection is a necessary process before license plate recognition. Methods on LPD can be classified into such categories based on the color level: 1) Binary Image processing; 2) Gray-Level Processing; 3) Color Processing; 4) Classifiers [3]. While based on the information used to detect the license plate, the LPD methods can be categorized into six classes: 1) using edge information; 2) using global image information; 3) using texture features; 4) using color features; 5) using character features; 6) combing two or more features [1].

1.3.1 License Plate Detection

A. Using Edge or Boundary Information.

a. Filters

In [6]-[9], Sobel filter is used to detect edges, by which the boundaries of license plates are represented due to color transition between the license plate and the car body. Two horizontal lines are located when performing horizontal edge detection, and two vertical lines are located when performing vertical edge detection. The rectangle is fixed when two set of lines are located both at the same time. While in 2003, F. Kahraman [2] *et al.* applied Gabor filters to detect license plate regions which achieved a good performance when images are of a fixed angle. Gabor filters are helpful in analyzing textures as they are sensitive to textures with different scales and directions.

b. Edge Detection

Based on the intuitive that the license plate is of some shape, most likely rectangular, whose aspect ratio is known, methods are commonly used to extract plates from all possible rectangles. Edge Detection methods are such ones to find the rectangles [10]-[13]. Edges can be detected only vertically or horizontally, and can be statistically analyzed to determine license plate candidate regions [12]. A fast vertical edge detection algorithm (VEDA) was proposed in [14] for license plate detection, which is declared to be faster than Sobel operator by around seven to nine times.

c. Geometric Attributes

Geometric features may include: rectangle height, width, area, aspect ratio, the number of rectangles included in another rectangle, etc. License plate rectangles are detected by using efficient geometric attributes to locate lines that form the rectangles in [15]. In [16] the simple geometric feature, the aspect ratio, is used to filter out unlikely candidates. They claim a result of 96.2% on images under various illumination conditions.

d. Block-based

Block-based methods consider blocks to be possible license plate regions. High edge magnitude is a cue for such blocks in [17]. The advantage of their method is that it works well with license plates whose boundaries are not clear because block processing is independent of the edges of license plate boundary. The accuracy of their method is 92.5% with 180 image pairs.

e. Transforms

Image transformation methods based on Hough transform (HT), wavelet transform (WT) and other transforms like Generalized symmetry transform (GST) have been applied in license plate detection. Boundary-based extraction using Hough Transform has been described in [18]. Hough transform detects straight lines in the image to locate license plates, the advantage of it being that it is capable of detecting lines with up to 30 degree inclination [19]. However, the disadvantage of Hough Transform is that it is both time-consuming and memory-consuming. Hough transform is combined with contour algorithm in a boundary line-based method, which is described in [20]. They achieved 98.8% accuracy in license plate extraction. Another transform, GST is used to extract license plates in [21]. The image is scanned after getting edges, in the

selective directions to detect corners, where GST is used to detect similarity between these corners and to form license plate regions.

In summary, methods using edge information to detect license plates are advantageous as they are simple and fast. However, they require the continuity of the edges [22]. If edge information is not sound and complete, license plates may not be accurately detected.

B. Using Global Image Information

a. Connected Component Analysis

Connected component Analysis (CCA) is an important technique applied in binary image processing. It is widely used in Scene Text Recognition problems, and specifically, it is also used in license plate recognition problems [23]-[25]. CCA scans a binary image and gives each pixel a label based on pixel connectivity, each label symbolizing a component. Pixels that share the same label make a connected component. Spatial measurements, such as area and aspect ratio, are commonly used for license plate detection [26], [27]. CCA on low resolution is applied by [26] and they achieved the extraction rate of 96.62% with 1.77% false alarms on a 4h video.

Contour information can be incorporated to help connecting components. A contour detection algorithm is applied in [28] on the binary image to detect connected components, of whom those have the same geometrical features as the plate are chosen to be candidates. This method may fail if the image is of poor quality, which results in distorted contours.

b. 2-D Correlation With Pre-stored License

2-D cross correlation is used in [29] to detect license plates. There is a pre-stored license plate template used in this method, with which the 2-D cross correlation is performed through the whole image to locate regions that most likely contain a license plate. It may extract the plate wherever the plate is located. However, the 2-D cross correlation is very time-consuming.

C. Using Texture Features

The methods using texture features is based on the intuition that license plates contain characters, which results in significant change in the grey-level between characters and license plate background, or a high edge density area due to color transition. It is reasonable to assume characters are present in license plates and different kinds of techniques are used in [30]- [38].

a. Vector Quantization

Vector Quantization (VQ) representation gives hints about the contents of image regions. Higher contrast and more details are often mapped by those smaller blocks. In [39], VQ is applied to detect license plates and the results show that the detection rate is 98% and the processing time is around 200ms.

b. Sliding Concentric Window

In [40], a sliding concentric window method is proposed where license plates are regarded as irregularities in the texture of the image and therefore abrupt changes in local characteristics indicate the existence of potential license plates. In Reference [41], their proposed method is also based on sliding concentric window, but histograms are used as well.

c. Image Transformations Majorly For Feature Analysis

Image transformations are widely-used license plate extraction techniques. Gabor filters are one of the major tools in texture analysis, the reason being that it is capable of analyzing feature in unlimited scales and orientations [42]. It is used in [43] which achieved a result of 98% when applied to images of a specifically fixed angle, with the disadvantage however, to be time-consuming.

In [32], Discrete Fourier Transform (DFT) is applied to identify spatial frequency for spectrum analysis. It is usually used row-wise to detect the horizontal position of the plate and column-wise to detect the vertical position.

While in [36], another transform, the wavelet transform (WT) is applied to detect license plates. There are four sub-bands in WT and the sub-images HL and LH describe the vertical edge information and horizontal information separately. The maximum change in horizontal edges is decided via the scan of LH image and identified by a reference line, below which vertical edges are projected horizontally, to determine the position based on this maximum projection. In [44], the HL sub-band is used to search the features of license plate and then to verify the features by checking whether the LH sub-band include a horizontal line around the feature. The method is fast and less than 200ms; its accuracy is 97.33%.

d. AdaBoost

In [45], [46], [47], adaptive boosting (AdaBoost) is applied to detect license plates. It is incorporated with Haar-like features to obtain cascade classifiers, which are invariant to brightness, color, size, and position of the license plates. In [45], the cascade classifiers use gradient density in the first layer and then Haar-like features. The accuracy is 93.5% in license

plate detection. In [48] AdaBoost is also used but without Haar-like features. They achieved an accuracy of 99% in detecting license plates that are from images with different formats, sizes, various lighting conditions.

In summary, the methods that are using texture features are advantageous in detecting license plates even if the boundary is unclear or deformed. The disadvantage is, however, that they computationally expensive.

D. Using Color features

License plates from some countries have specific colors for their license plates, which can be made use of while extracting license plates. For such license plates extraction scenarios, assumption is that the color combination of a plate and characters is unique and that this combination only occurs in a plate region [49]. In [49], the author proposed according to the specific formats of Chinese license plates that all the pixels in the input image can be classified into 13 categories in the hue, lightness and saturation (HLS) color space.

a. Neural Network to Classify the Color of Each Pixel

Neural Network is used in [50] to classify the color of each pixel in HLS color space. The neural network output classification index, which symbolizes the color of green, red, and white. These colors are license plate colors in Korea. In [51], only four colors (white, black, red, green) are utilized in the license plates and therefore only three kinds of edges (i.e., black-white, red-white, and green-white) are focused on by their color-edge detector. They experimented with 1088 images captured from a variety of scenes and under diverse conditions and achieved a detection rate of 97.9%.

b. Genetic Algorithm to Search License Plate Color

Genetic algorithm (GA) is a computational intelligence algorithm for fitness function optimization and have been used in [52]-[53] to search for license plate color. In [53], a GA is used to determine thresholds for the plate color while training pictures with diverse lighting conditions. The thresholds are related to the average brightness via a function, and for each input image, the average brightness is computed and therefore the upper and lower thresholds are known through this function. Each pixel will get labeled if its intensity lies between this pair of thresholds and if the connectivity of these thresholds form a rectangle that is of an eligible aspect ratio, the region is regarded as a plate candidate.

c. Gaussian Weighted Histogram Intersection

Gaussian function is used in [54] to modify the conventional histogram intersection while detecting license plates by matching the color, in order to adapt to various illumination conditions, which affects the color level of images. These weights modified by Gaussian function are of several similar colors that contribute to matching a pre-defined color.

In [55]-[56], mean shift algorithm is used to segment color images into candidate regions, which are subsequently classified to decide if it is a license plate. A detection rate of 97.6% is achieved. A fast mean shift algorithm is afterwards proposed in [57] to accelerate the processing time.

d. Fuzzy Logic Based to Deal With Illumination Variance

A fuzzy logic based method is proposed in [58] in order to deal with the variety of illumination conditions. This method is associated with color information in hue, saturation,

value (HSV) color space. Three components of the HSV are mapped to fuzzy sets first, according to different membership functions. The degree to which the three membership functions are weighted in their fusion describes the fuzzy classification function. And then each region can be decided by this classifier where it is a license plate region or not.

In summary, the methods using color information holds the advantage of the capability in detecting inclined or deformed plates, while it has some difficulties too. Defining the pixel color using RGB value is hard under different illumination conditions. And its alternative color model HLS is quite sensitive to noise. Methods using color projection may encounter problems when the license plate color is not unique in plate regions but can be found elsewhere, such as the car body or the environment.

E. Using Character Features

License plate detection methods based on locating characters or recognizing character features have also been proposed. These methods aim at finding the characters, whose region is then extracted as the license plate region. In [59], stroke width transform is proposed to locate text in natural scenes, based on the intuition that text should have the same stroke width. This CVPR paper introduces a method that is more general a technique since it extracts text whether or not they are in a license plate.

a. Repeating Contrast Changes

The method adopted in [60] is looking for repeating contrast changes when scanning the input image horizontally, on a scale of 15 or more pixels. Assumption is made that the contrast between characters and the background is abundant and that there are at least three to four

characters whose minimum vertical size is 15 pixels. This method is combined with differential edge detection and achieved 99% accuracy.

b. Same Aspect Ratio As Characters

In [61], binary objects are labeled according to two characteristics: the aspect ratio, the number of pixels. The connected components are character candidates which will be applied Hough Transform on both the upper side and lower side, if two parallel straight lines are detected and the number of connected components is similar to that of a license plate, the area will be regarded as a license plate region.

c. SVM-trained SIFT Descriptors

In [62], the first stage of their proposed method is to classify the regions of the input image into possible character regions and highly-unlike character regions with thirty-six AdaBoost classifiers. The second stage is a further classification on the possible character regions obtained from the first stage, using a support vector machine (SVM) trained on scale-invariant feature transform (SIFT) descriptors. The positively-labeled regions after all the stages will be the license plate region detected.

d. MSER

In [63], maximally stable extremal region (MSER) is used to detect text or character regions in the license plates. Among the large amount of candidate regions detected by MSER, highly-unlike regions will be filtered by a simplistic heuristic-based filter. The remaining regions will be further filtered by checking the number of positively classified SIFT points, and those regions will be considered license plate regions if their number is greater than a threshold.

F. Combining Two or More Features.

In [64], the rectangle shape features, the texture feature, along with the color feature are all combined to detect license plates. The detection rate is 97.3% on 1176 images captured from various scenes. In [65], a method is proposed to detect license plates combining wavelet analysis, improved version of HLS color decomposition and Hough line detection.

License Plate segmentation is a pre-processing procedure before LPR and it is often necessary as most LPR algorithms require single character inputs. Based on the information used in segmentation, several methods have been applied: using pixel connectivity, using projection profiles, using prior knowledge of characters, using character contours [1].

1.3.2 License Plate Segmentation

A. Using Pixel Connectivity.

In [66]-[68], segmentation is performed upon the binary image derived from the input image, by labeling the connected pixels into connected components, which will be analyzed in aspect such as size and aspect ratio, to determine if they belong to license plate characters. The disadvantage however, is that it will fail on characters that are joined or broken.

B. Using Projection Profiles.

Since characters and plates are of different colors for human beings to distinguish, they should be of different values in the binary image. Some methods as in [69]-[73] are proposed to project the extracted binary license plate vertically to determine the starting and ending positions of the characters and horizontally to determine the position of each one. In [70], character color information replaced binary information in the projection.

The advantage of this kind of methods is that the segmentation is independent of the positions of characters, however, the disadvantage is that it depends on the quality of the input image because any noise could affect the projection value. Besides, prior knowledge about the number of characters is required by this kind of methods.

C. Using Prior Knowledge of Characters.

The prior knowledge of characters can be useful for plates from some countries because their plates are standardized and therefore not versatile, such as Chinese license plates. The layout is fixed except for some special plates such as military vehicle plates. In [74], an approach is proposed which provides a solution in detecting vehicle license plates that are severely degraded. The plate is first located by the using color collocation, then dimensions of each character are used for segmentation. The layout of the Chinese license plates provides information for the classifier to recognize characters afterwards. The advantage of such methods is its simplicity. Nevertheless, the extracted license plate must not be of any shift of the ground-truth license plate location, otherwise extraction results may be in background instead of characters.

D. Using Character Contours.

Contour information is also used for license plate segmentation. In [75], the author established a shape-driven active contour model that utilizes a variational fast marching algorithm. The rough location of the license plates is found in the first stage by an ordinary fast marching technique in combination with a gradient-dependent and curvature-dependent speed function. In the second stage, the exact boundaries of the license plates are derived by a special fast marching method.

E. Using Combined Features

Two or more features of the characters might be used in order to segment the license plate more efficiently. The method in [76] segments the main numeric characters on a license plate using dynamic programming (DP). It is very rapid by applying the bottom-up approach of the DP algorithm and it is robust by minimizing the use of environmental-dependent features such as color and edges.

1.3.3 License Plate Recognition

A. Using Raw Data.

Template matching is simple and straightforward in license plate recognition. It measures the distance or similarity between the template and the character. Whichever template the character is similar to, the corresponding index will be the recognition result. Binary image is commonly used in template matching because otherwise the grey-scale will change if the lighting is different. This kind of method is useful in recognizing characters that are single-font, non-rotated, non-broken, and fixed-size. Even though this method is simple, it is fairly limited.

B. Using Extracted Features.

This is the most common method for character recognition. It reduces the time of processing since it does not involve all the pixels as template matching method does. And it outperforms template matching method when characters are distorted. The extracted features form a feature vector which is compared with the pre-stored feature vectors to measure similarity. Or, classifiers can be used to recognize characters, like ANN, SVM, HMM. Some researchers

integrate multiple kinds of classification schemes in parallel [77], some perform multi-stage classification instead [78].

1.5 Overview of Our Methods

Specifically, the proposed procedure includes LPD, License Plate alignment (LPA), License Plate Binarization (LPB), LPS, and LPR. With a multi-scale sliding window scanned over the images, HOG features are extracted to be used by an SVM classifier. 3000 License plates from 50 states in the US are trained as positive class for this classifier. And 1200 background patches collected by hand from other traffic surveillance images are used as negative class during the training stage. Edge information is used as a preprocessing stage prior to sliding window so that the scanning region and therefore the computation time is minimized. Once candidate detection windows are obtained, the mean-shift algorithm is then applied to merge possible candidates into a final decision.

In order to make the detection window accurately aligned to the license plate, LPA will be performed. With the detection window ensured to be accurate, k-means clustering will be possible to binarize the license plate. e.g. cluster the plate pixels into two classes, based on the color difference between the text and the background. A model with the prior knowledge of the specific layout will segment the plate into patches, where each patch containing a character will be recognized via another SVM classifier, trained on single characters obtained from hand-made license plate characters. Fig. 1. 2 shows the overview of the proposed method.



Fig. 1. 2. Overview of our proposed method. (a) Shows the procedure of license plate detection; (b) shows the procedure of license plate recognition

CHAPTER 2

LICENSE PLATE DETECTION

Overview of the Extraction Process:

Preparation: 1) Extract HOG features from the training images of license plates. 2) Use SVM to train the model. 3) We trained 3378 license plates.

Apply edge detection on the input image first. Scan the whole image with a scanning window, and the regions with edges above a certain ratio will perform SVM classification in the window to decide if this window's score is above the threshold and therefore positive, meaning it contains a plate. So the scanning is not over the whole image, which saves time for detection. The HOG features for the whole image are computed beforehand, instead of calculating that with each scan, so it saves time too. The image will be resized to 909*523, and the scanning window is multiple scales by setting the resized image multiple scales. So for each scale the HOG features will be re-calculated. The nearby positive windows will merge to get the final location using k-means.

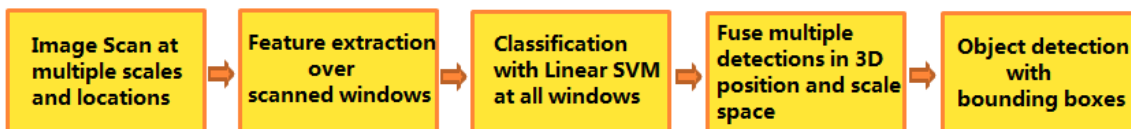


Fig. 2. 1. An overview of license plate detection procedure.

2.1 Scanning Window

The resolution of the images is 1920 by 1104, and is normalized to 903 by 523 pixels. The normalized image is scanned once for each scale. The scanning window size is fixed, while the image itself scales from 0.61, 0.72, 0.85 to 1.00 of its normalized size. Rather than scaling

the size of the scanning window, this method will save computation time by calculating the HOG features used by detection windows beforehand. And the scanning window only need a look-up to obtain the features indexed by the position being scanned.

Since the system is intended for a camera located on the highway, where the position of the camera is stable, the system speeds up the scanning process by fix the scale after several frames. The scale can be fixed at which level the license plates are detected and recognized in previous frames. Besides, the system does not perform SVM with HOG features to every scanning window, but only to those windows that contain enough edge pixels. This also accelerates the whole scanning process.

Scanned regions that contain enough edge pixels, and further with large positive response from SVM classifier, will be potential candidates of a detected license plate. These candidates will fuse into a final decision with non-maximum suppression algorithms.

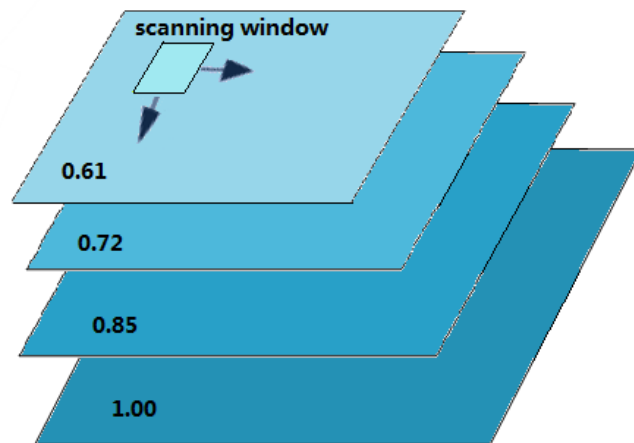


Fig. 2. 2. Scanning-window and image pyramid

2.2 HOG Features

Many features have been invented and proposed to describe text. Based on the most recent research [84], among feature descriptors such as HOG, SIFT, SURF, BRIEF, ORB and DAISY, the HOG features with SVM achieve the best result for recognizing scene texts, of whom license plate text captured from a real life camera on the highway is one scenario.

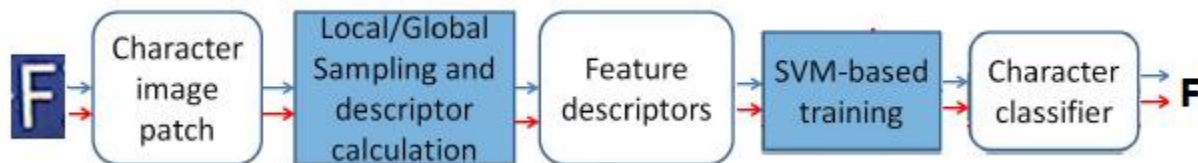


Fig. 2. 3. Flowchart of the Scene Text Recognition framework in performance evaluation

SIFT descriptors, edge orientation histograms, and shape contexts resemble HOG descriptors in some ways. Computed on a dense grid of cells that are uniformly spaced, HOG use overlapping local contrast normalizations to improve performance. Compared to SIFT, which uses scale-invariant key image points and rotated to align orientation, HOG is single scale without orientation alignment.

HOG descriptors hold some advantages in detecting a license plate though:

- 1) It extracts gradient or edge information and comparatively, this information appropriately represents text on a license plate because its characteristics describe well the local shapes.
- 2) It is also invariant to local geometric and photometric transformations, which is to a controllable degree.

HOG descriptors have some variants such as Rectangular HOG (R-HOG), Circular HOG (C-HOG), Bar HOG, Centre-Surround HOG [79]. The Co-occurrence HOG (Co-HOG) proposed in [82] is claimed to have the state-of-the-art result in Scene Text Recognition on ICDAR [85]

and SVT dataset [86]. R-HOG is adopted and discussed in this thesis as it is convenient to implement among all variants of HOG. R-HOG use rectangular blocks as shown in Fig. 2.4:

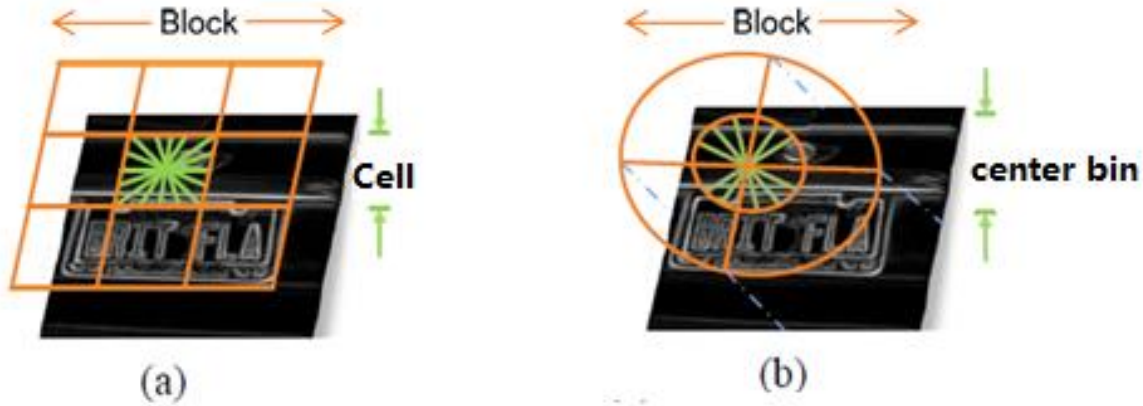


Fig. 2. 4. Variants of HOG descriptors. (a) Rectangular HOG(R-HOG) with 3 by 3 blocks (b) Circular HOG (C-HOG) descriptor with central cell divided into angular sections as in shape contexts.

2.2.1 Feature Extraction Procedure (for Dense-HOG)

The region covered by the sliding window will firstly be divided into smaller regions (called cells). A local one dimensional histogram of gradient directions is computed for each cell. While concatenating the histograms together, we basically get the HOG descriptor for the region of the sliding window. Accumulating local histograms over larger spatial regions (called blocks) will improve the performance because it can normalize the contrast, and therefore render the descriptor more resistance to shadows and illumination. The feature extraction procedure follows the steps described below:

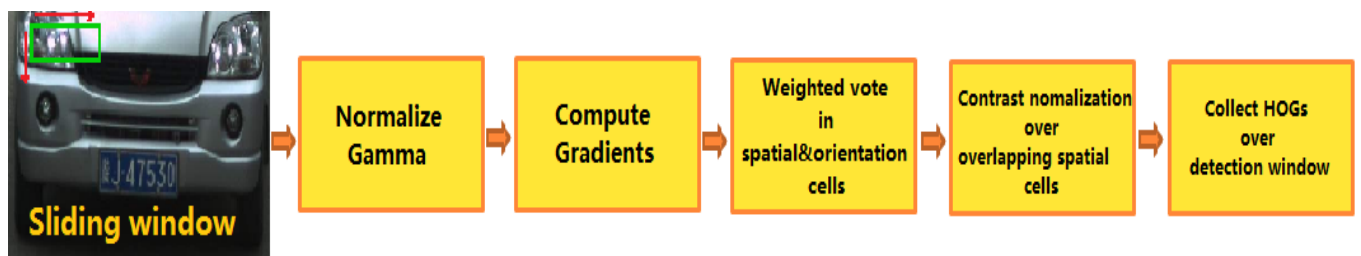


Fig. 2. 5. An overview of HOG feature extraction

It is often required for many features that the color normalization or gamma correlation is performed as a pre-processing step, however, Dalal and Triggs pointed out that HOG descriptors is one exception as the following descriptor normalization has the same effect and achieve the same result. The gradient map is computed simply by applying 1D centered point discrete derivative mask in horizontal and vertical directions, whose filter masks are $[-1, 0, 1]$ and $[-1, 0, 1]^T$ respectively. Since the input image consists of three color channels (R, G, B), of which the gradients will be calculated individually and the one with the largest norm will be the pixel's gradient vector.

The step afterwards is the creation of cell histograms. The structure of R-HOG has been shown in Fig. 2. 4 (a). The cells of R-HOG are square or rectangular with a certain number of orientation bins distributed over $0^\circ \sim 180^\circ$ or $0^\circ \sim 360^\circ$. Each of the pixels in the cell is computed for a weighted vote, which contributes to the orientation histogram. The votes from all the pixels will accumulate into the orientation bins. Bilinear interpolation is applied between the neighboring bin centers in both orientation and position during the voting. The vote is a gradient magnitude or a function of it.

In order to achieve better performance, effective local contrast normalization over overlapped cells is a necessity to make the descriptor invariant of foreground-background contrast and illumination change. What happens is the cells are grouped into larger spatial blocks, of whom the contrast is normalized separately. These blocks are often overlapped, as shown in Fig. 2. 6., in order that entries of each cell will reflect several component of blocks in the final

feature. The overlapping is not abundant, for it contributes to the significant improvement of performance.

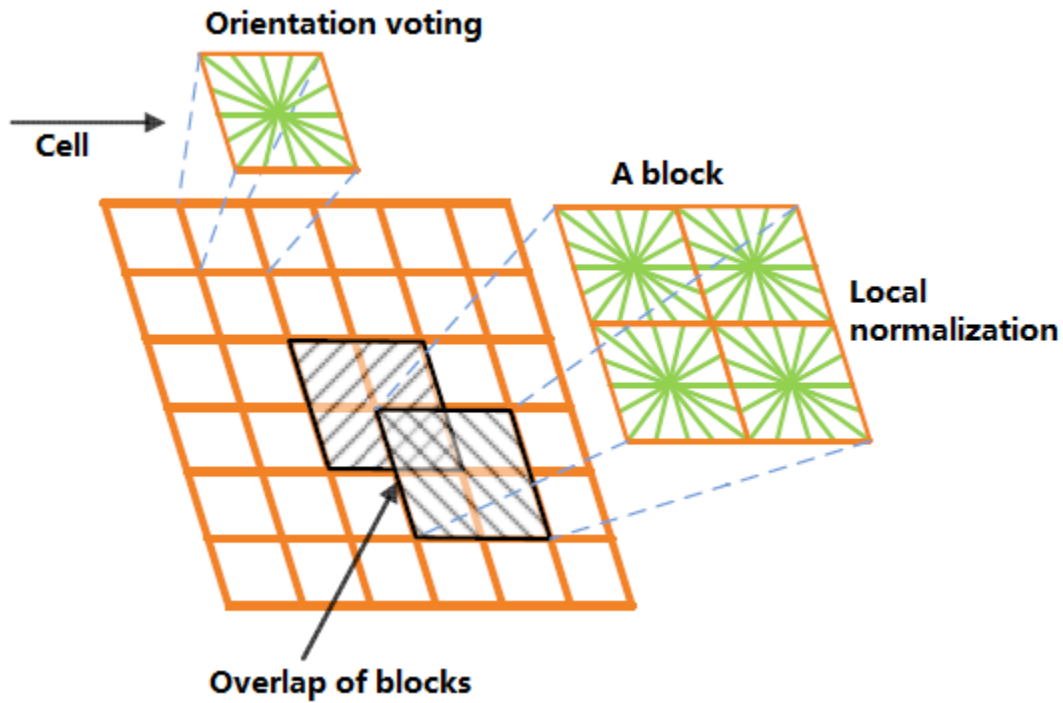


Fig. 2. 6. A sketch of local normalization.

Several different normalization schemes are among the choices. Four of them are evaluated in [87]:

- L2-norm: $\mathbf{v} \leftarrow \mathbf{v} / \sqrt{\|\mathbf{v}\|_2^2 + \varepsilon^2}$;
- L2- Hys: L2-norm followed by clipping (limiting the maximum values of \mathbf{v} to 0.2) ;
- L1-norm: $\mathbf{v} \leftarrow \mathbf{v} / (\|\mathbf{v}\|_1 + \varepsilon)$;
- L1-sqrt: $\mathbf{v} \leftarrow \mathbf{v} / \sqrt{\|\mathbf{v}\|_1 + \varepsilon}$;

Where \mathbf{v} is the normalized descriptor vector and $\|\mathbf{v}\|_k$ is its k-norm for $k= 1,2$, and ϵ is a small normalization constant to prevent a division by zero. In [87], L2-Hys, H2-norm and L1-sqrt achieved the same performance, all of which is 5% greater than that of L1-norm.

2.2.2 Implementation

Our implementation adopts R-HOG features to extract from the input images for R, G, B color channels with no gamma color correction. The gradient map is derived by applying the simple 1D gradient filter $[-1, 0, 1]$ to the original image.

Detection window size is 24 by 72. The bin number is 9 and the gradient of each pixel is placed over $0^\circ \sim 180^\circ$. The blocks are normalized by L1-sqrt function: $\mathbf{v} \leftarrow \mathbf{v} / \sqrt{\mathbf{v} / (\|\mathbf{v}\|_1 + \epsilon)}$. The block size is 8 by 8 grids, meaning 64 cells in a block. And each cell is 4 by 4 in pixels. Block spacing stride is 4 pixels. The overall HOG feature length with above features is 3060.

2.3 Support Vector Machine

Support Vector Machine (SVM) is a kernel-based supervised learning model that optimally classifies data into two categories in N-dimensional hyper-space by maximizing the margin in between. The SVM algorithm was originally invented by Vladimir N.Vapnik and then developed into current standard incarnation by Vapnik and Corina Cortes. In our application, a dense version of SVMlight [80] is used as it reduces the memory usage when large dense descriptors come along [79] and linear kernels are chosen.



Fig. 2. 7. The training images of the SVM classifier. Positive.

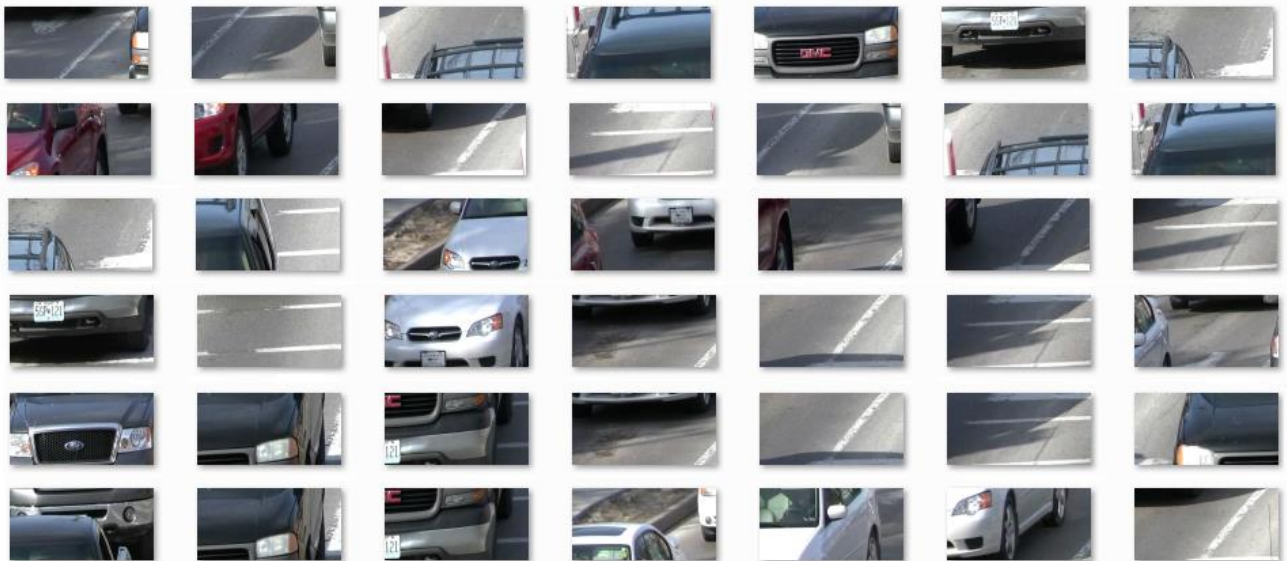


Fig. 2. 8. The training images of the SVM classifier. Negative.

2.4 Non-Maximum Suppression

As multiple overlapping detections are often obtained for each instance of a license plate, a greedy procedure for eliminating repeated detections are adopted via non-maximum suppression [81]. A number of bounding boxes will be derived from the multiple-scale scan of the sliding window, each having an SVM score. With a threshold eliminating some weak detections, multiple windows may still be left, where non-maximum suppression fuse the remaining windows into a final decision. A uniform Kernel Mean Shift algorithm [83] is used for the fusion of multi-scale overlapped detections.

All the detections are represented by a 3-D vector: $\mathbf{y}_i = [x_i, y_i, s_i]$, where i symbolizes the i -th detection window. x, y indicate the x, y coordinates and s stands for scale. $K(\mathbf{y})$ is the kernel function and \mathbf{H} is a symmetric positive definite 3 by 3 bandwidth matrix. $\text{diag}[\mathbf{H}] = [(\exp(s_i) \sigma_x)^2, (\exp(s_i) \sigma_y)^2, (\sigma_s)^2]$ where σ_x, σ_y , and σ_s are the window bandwidth. The kernel Density Estimation (KDE) is

$$\hat{f}(\mathbf{y}) = \frac{1}{n} \sum_{i=1}^n K_H(\mathbf{y} - \mathbf{y}_i) \quad (2.1)$$

where

$$K_H(\mathbf{y}) = |\mathbf{H}|^{-\frac{1}{2}} K(\mathbf{H}^{-\frac{1}{2}} \mathbf{y}) \quad (2.2)$$

The profile of the uniform kernel is

$$k_U(\mathbf{y}) = \begin{cases} 1, & 0 \leq x < \sigma \\ 0, & y \geq \sigma \end{cases} \quad (2.3)$$

and the kernel function is

$$K_U(\mathbf{y}) = \begin{cases} c_d & \|\mathbf{y}\| \leq \sigma \\ 0 & \|\mathbf{y}\| > \sigma \end{cases} \quad (2.4)$$

where c_d is the volume of the unit 3D sphere. The SVM score of each detection window is denoted as w_i . Since the weights for KDE should be no less than zero, a hard clipping function is applied to ensure that the scores are positive.

$$t(w) = \begin{cases} 0 & \text{if } w < c \\ w - c & \text{if } w \geq c \end{cases} \quad (2.5)$$

Therefore, equation (2.1) can be written as the following form:

$$f(\mathbf{y}) = \frac{c_d}{n} \sum_{i=1}^n |\mathbf{H}_i|^{-1/2} t(\omega_i) D^2[\mathbf{y}, \mathbf{y}_i, \mathbf{H}_i] \quad (2.6)$$

where

$$D^2[\mathbf{y}, \mathbf{y}_i, \mathbf{H}_i] \equiv (\mathbf{y} - \mathbf{y}_i)^T \mathbf{H}_i^{-1} (\mathbf{y} - \mathbf{y}_i) \quad (2.7)$$

is the Mahalanobis distance between \mathbf{y}_i and \mathbf{y} . The gradient of (2.6) is given by

$$\nabla \hat{f}(\mathbf{y}) = \frac{2c_d}{n} \sum_{i=1}^n |\mathbf{H}_i|^{-1/2} |\mathbf{H}_i|^{-1} (\mathbf{y} - \mathbf{y}_i) t(\omega_i) D^2[\mathbf{y}, \mathbf{y}_i, \mathbf{H}_i] \quad (2.8)$$

Dividing (2.8) by (2.6), then we have

$$\frac{\nabla \hat{f}^2(\mathbf{y})}{f^2(\mathbf{y})} = \sum_{i=1}^n \bar{\omega}_i(\mathbf{y}) |\mathbf{H}_i|^{-1} \mathbf{y}_i - \left(\sum_{i=1}^n \bar{\omega}_i(\mathbf{y}) |\mathbf{H}_i|^{-1} \right) \mathbf{y} \quad (2.9)$$

where

$$\bar{\omega}_i(\mathbf{y}) = \frac{|\mathbf{H}_i|^{1/2} t(\omega_i) D^2[\mathbf{y}, \mathbf{y}_i, \mathbf{H}_i]}{\sum_{i=1}^n |\mathbf{H}_i|^{-1/2} t(\omega_i) D^2[\mathbf{y}, \mathbf{y}_i, \mathbf{H}_i]} \quad (2.10)$$

And since $\sum_{i=1}^n \bar{\omega}_i = 1$, we have

$$\mathbf{H}_h^{-1}(\mathbf{y}) = \sum_{i=1}^n \bar{\omega}_i(\mathbf{y}) \mathbf{H}_i^{-1} \quad (2.11)$$

Be the data weighted harmonic mean of the covariance matrices \mathbf{H}_i computed at \mathbf{y} .

From (2.9) and (2.11), the variable bandwidth mean shift vector is defined as

$$\mathbf{m}(\mathbf{y}) = \mathbf{H}_h \frac{\nabla f^2(\mathbf{y})}{f^2(\mathbf{y})} \equiv \mathbf{H}_h(\mathbf{y}) \left[\sum_{i=1}^n \bar{\omega}_i(\mathbf{y}) \mathbf{H}_i^{-1} \mathbf{y}_i \right] - \mathbf{y} \quad (2.12)$$

At the mode location, the gradient $\nabla \hat{f}(\mathbf{y}) = 0$, implying $\mathbf{m}(\mathbf{y}) = 0$. Equation (2.12) suggests that the mode can be estimated iteratively by computing this:

$$\mathbf{y}_m = \mathbf{H}_h(\mathbf{y}_m) \left[\sum_{i=1}^n \bar{\omega}_i(\mathbf{y}_m) \mathbf{H}_i^{-1} \mathbf{y}_i \right] \quad (2.13)$$

And this starts from some \mathbf{y}_i until \mathbf{y}_m does not change no more.

We check those detections who finally shift into a region with radius R , and fuse them into the same mode. This mean shift algorithm may achieve reliable results for single license plate scenario or even multiple license plate scenario, if we choose variables wisely, such as carefully selecting σ_x , σ_y , and σ_s . The detections with the greatest score in each mode are selected as the final detection result. If there are multiple modes, it indicates that there are multiple license plates detected.

2.5 Refinement of the Algorithm

2.5.1 Edge Information

The sliding window method is time-consuming to scan the whole image since most of the candidates are not possible to be a license plate. Edge information can be analyzed to accelerate the scanning process as it is necessary for license plate candidate to contain edges.

Canny edge detection is applied as a preprocessing stage prior to sliding window so that the scanning region and therefore the computation time is minimized. Only detection windows containing enough edge pixels will be classified by SVM.

Usually, the images captured on the highway are mostly smooth since most part is the road and if there is a vehicle, most parts are smooth as well. So the overall computation time can be saved around ten times, from being 3s to around 200ms. And the plate will never be missed by using the edge information.



Fig. 2. 9. Using Edge information. (a) The canny edge image; (b) The scanning regions that will be classified by SVM.

2.5.2 Scale Adaption

The intuition behind this is that for certain cameras, with their angle and height fixed, they capture images of vehicles whose license plates are almost of the same size. The images need to be scanned in multi-scales, however, once the scale is fixed, redundant searching is not necessary.

For a set of images that are captured by the same camera, the sliding window will be performed at multiple scales within the first several frames. If the scales of these frames at which the license plates are found are the same, it is considered that this camera is among such cameras and the scale will be fixed in order to save time.

After the detection with its corresponding scale is found, the scale will be fixed unless no license plate can be found. If no plate can be found, the multi-scale search will be restarted to ensure that no license plates are missed. This will deal with cameras that are not so ideally located, which is most of the cases. But at least some time can be saved for such scenarios. The time can be shortened to 100ms for frames whose scale is fixed.

2.6 Results and Discussion

We use Detection Error Tradeoff (DET) curves on a log-log scale, i.e. miss rate versus FPPW, to quantify the performance of the detector. The miss rate is calculated as 1-Recall, or

$$\frac{FalseNeg}{TruePos + FalseNeg}$$

A license plate database from a Chinese traffic surveillance system is tested. The database contains vehicle (e.g., truck, car) images collected both at daylight or night. The images

are captured at two provinces of China, Shan'xi and Guangzhou. There are totally 105 images in this dataset. Additionally, 3378 extra American License Plate images are used for training the SVM classifier to detect the license plates.

The total computation time for each 1920 by 1104 image, which later is normalized to 903 by 523 pixels, is only approximately 200ms on Intel (R) Core (TM)2 Quad CPU Q8200 @2.33GHz. The images are multi-scale searched. The scales are 0.61, 0.72, 0.85 and 1.00. For single fixed scale, the time for detection can be around 100ms.

The detection would be claimed as true positive if the detection window can be aligned to fit the license plate by our method, which will be explained further in the next chapter. Usually, if the detection can be aligned, the following condition should be necessary: $\frac{R \cap r}{R \cup r} \geq 0.5$, where R is the area of the ground truth rectangle while r is the area of the detection rectangle. The test on the 105 images of the dataset is successful that false positives can be eliminated with no false negatives. Therefore, with no license plates missed, the miss rate is 0.

2.7 Summary

We developed a license plate detection method using SVM classifier with HOG features in this work. The license plate detection system searches at different scales for license plate candidates whose HOG features are extracted and analyzed by using an SVM as the classifier. The highly overlapped candidates are finalized into one decision by Mean Shift. Edge information is used to accelerate the time-consuming scanning process, together with scale adaption mechanism. A detection rate of 100% is achieved on the dataset, both in the daytime and at night. This method is comparatively insensitive to lighting conditions.

CHAPTER 3

LICENSE PLATE RECOGNITION

Overview of the License Plate Recognition:

Based on accurate alignment, we can use k-means to derive two clusters from the license plate. One is the characters, and one is the background of the plate. Set the major cluster with more points white and the minor cluster black so the characters will be black. Segmentation is based on the binary license plate.



Fig. 3. 1. Aligned binary license plate



Fig. 3. 2. Segmented license plate

The segmentation is done with a model. The model is shown in Fig. 3.3. It is from prior knowledge that we know the number of characters from Chinese license plates is seven, and that there is a dot between the second and the third character.

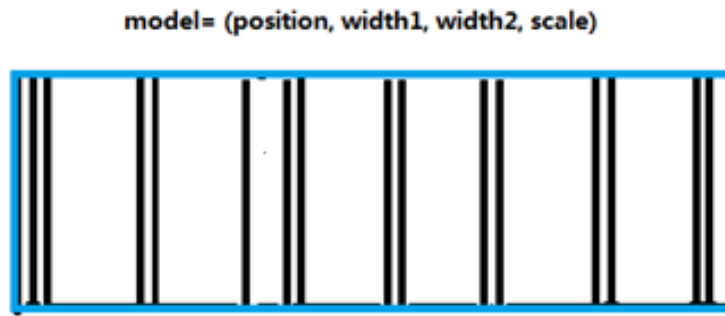


Fig. 3. 3. Segmentation model.

The model is deformable with its position and width and scale. Scan vertically at each x axis of the plate, the 255 to 0 inversion contributes to the score. In Fig.3.4, the y axis shows the value of histogram $h(x)$ at the corresponding x axis of the plate.

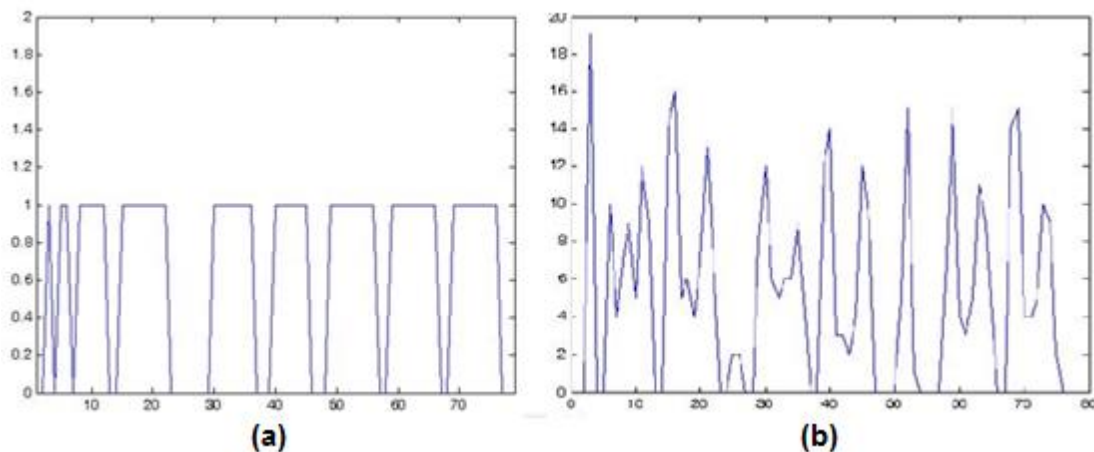


Fig. 3. 4. The analysis of the best fit model. Left figure is with threshold; right figure is without threshold.

The character patches segmented by this method will be recognized respectively by a SVM classifier using similar features, including 255-to-0 inversion and raw features. The overview of License Plate Recognition process is shown in Fig. 3.5.



Fig. 3. 5. The overview of the license plate recognition process.

3.1 License Plate Alignment Using Color Information

3.1.1 License Plate Alignment Without Angles

The license plate detection procedure described in the last chapter provides the plate region to the recognition part of the LPR system. This region is the fusion of many detection windows via NMS. And consequently, the region may not be well aligned with the license plate in the image. Therefore, a method to well align the detected license plate is proposed. Fig. 3.6 shows the detected license plate before and after the alignment.

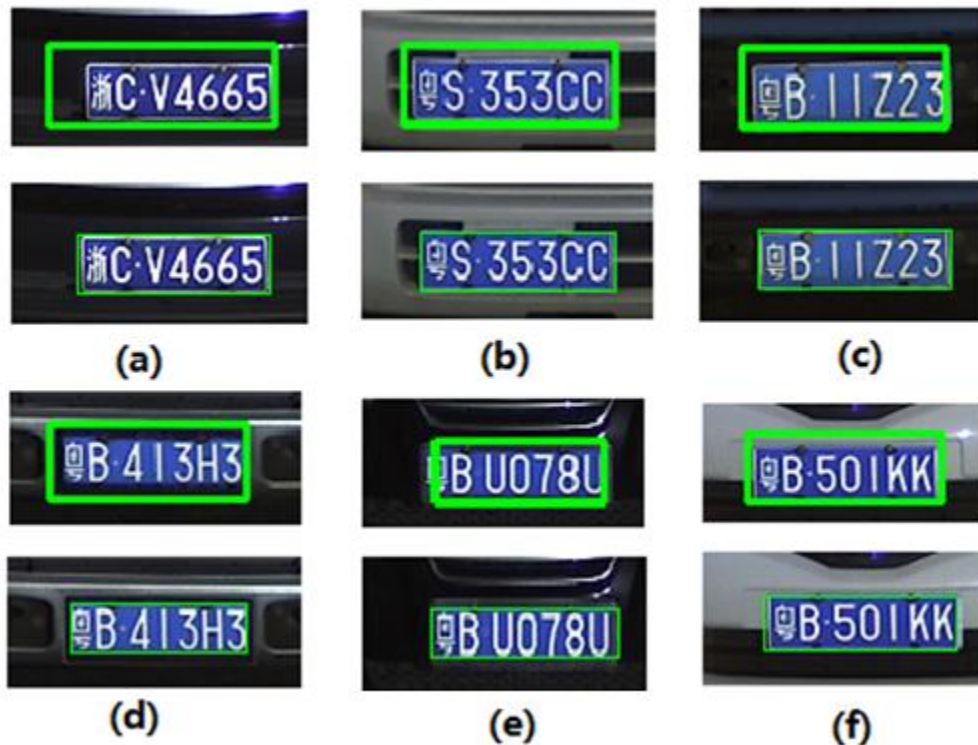


Fig. 3. 6. The detected license plate before and after alignment

The method is basically optimizing the boundaries of the detection window in order to ensure the colors on both sides of the boundaries are most different, as shown in Fig. 3.7. In this figure, A, B, C symbolizes regions that are used to compute histograms. L1, L2, L3, and L4 are the four lines that will eventually form the aligned license plate. As the lines slide within its range, histograms are computed and the color difference between regions is measured. The positions of lines where a maximum difference is achieved will then form the aligned result rectangle, which is the accurate license plate region.

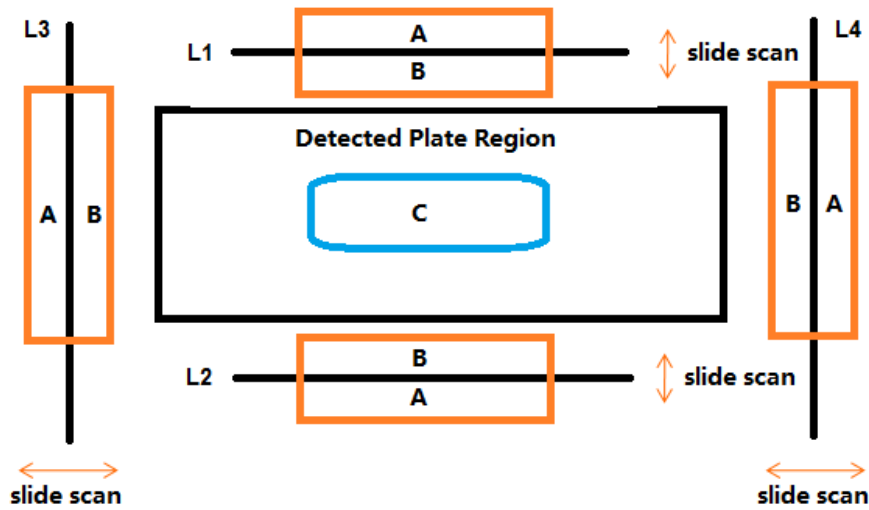


Fig. 3. 7. The alignment model. L1-L4 are lines that will slide within a certain range. Region A and Region B are two regions that are symmetrical to each line. Region C is a region in the inner-middle of the detected license plate rectangle.

The optimization process is performed by maximizing a fitness function.

$$F = F_{left} + F_{right} + F_{top} + F_{bottom}, \quad (3.1)$$

Where

$$F_{left} = d_{AC,left} - w_{left} \cdot d_{BC,left} \quad (3.2)$$

$$F_{right} = d_{AC,right} - w_{left} \cdot d_{BC,right} \quad (3.3)$$

$$F_{top} = d_{AC,top} - w_{left} \cdot d_{BC,top} \quad (3.4)$$

$$F_{bottom} = d_{AC,bottom} - w_{left} \cdot d_{BC,bottom} + d_{AB} \quad (3.5)$$

F_{bottom} is slightly different from other fitness functions because the bottom part of license plates on the vehicles are often adhere to vehicles parts that have similar structures as the license plate itself. Therefore, not only plate region C should be different from region A and similar with region B, but Region A should be also different from region B.

The way to measure color difference is to compute the histogram distance d . The histogram in RGB color space has 3 channels. Matrix wise, each histogram describing a region can be regarded as a 3 by 32 matrix, whose column represents the intervals while row represents the channels. Fig. 3. 8 shows the matrix and the histogram.

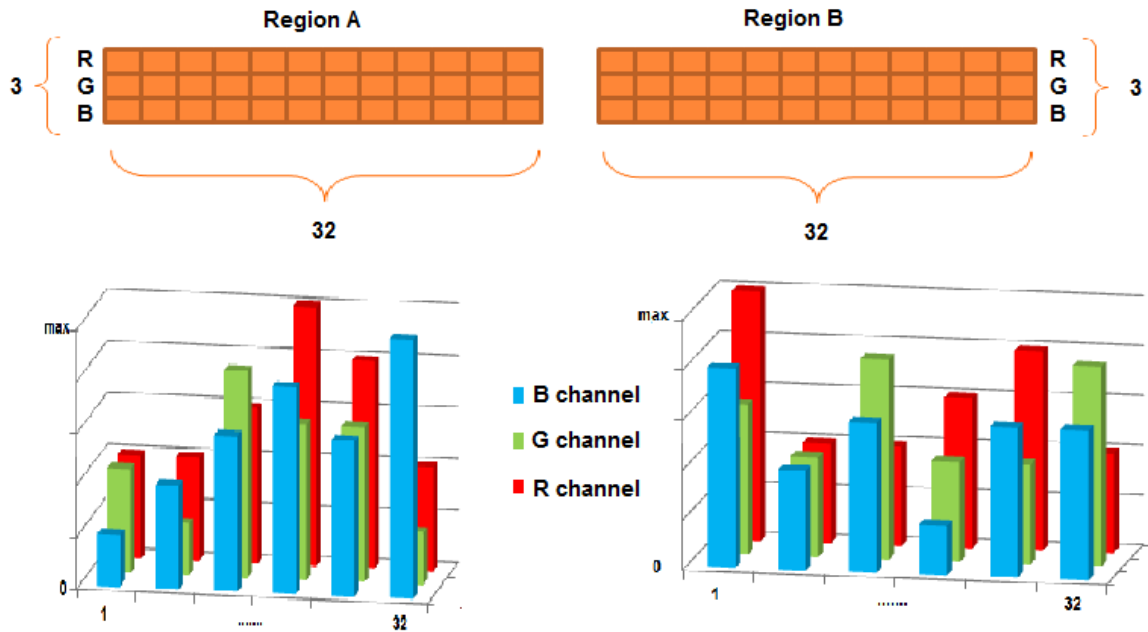


Fig. 3. 8. The histograms and their corresponding matrix for two regions: region A and region B.

Since each pixel's value ranges from 0 to 255, by separating the values into 32 intervals, every 8 values fall into the same interval, as shown in Fig. 3. 9.

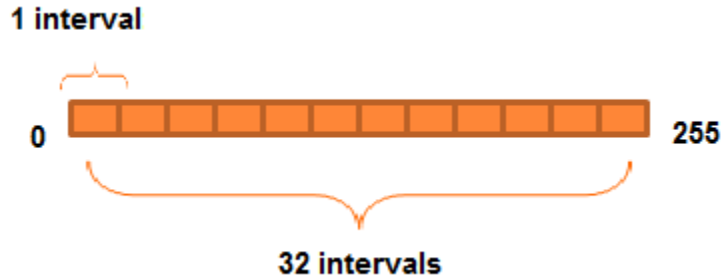


Fig. 3. 9. Quantize each pixel value into 32 intervals. Pixels whose value of a channel is within the range of the corresponding interval will vote that bin, which forms the 3 by 32 histogram matrix.

The distance between these histograms will be computed based on this function:

$$d_{AB} = \sum_{i=1}^3 \sum_{j=1}^{32} \frac{[h_A(i, j) - h_B(i, j)]^2}{h_A(i, j) + h_B(i, j)} \quad (3.6)$$

This is the histogram distance between region A and region B.

The reason for dividing the Euclidean distance $[h_A(i, j) - h_B(i, j)]$ by $h_A(i, j) + h_B(i, j)$ is that for histograms whose max value is different, this will normalize the difference.

For example, comparing line L1 with line L3 in Fig. 3.7, since the width of a license plate is usually bigger than the height, the overall pixels in region A/B of line L1 will be more than the number of pixels in region A/B of Line L3. In this case, L1's regions will generally have higher values in its histograms than that of L3 without the normalization. Therefore, with the same color difference level, L1 will contribute more color distance than L3 because the distance is of second power. As discussed in chapter 3. 1. 2, if the license plate is tilted, then angles are also searched.

3.1.2 License Plate Alignment with Angles

Originally the alignment described in this thesis was more complicated as it searches 12 angles for the best one that will optimally fit the license plate. So if the license plate is tilted, the four lines that make up the license plate boundary will be a parallelogram, and it still can be found by the alignment method. The segmentation process, consequently, will be a little different too. Rather than always being vertical, the segment lines will have the same angle as the lines, which is determined by alignment, its previous processing unit.

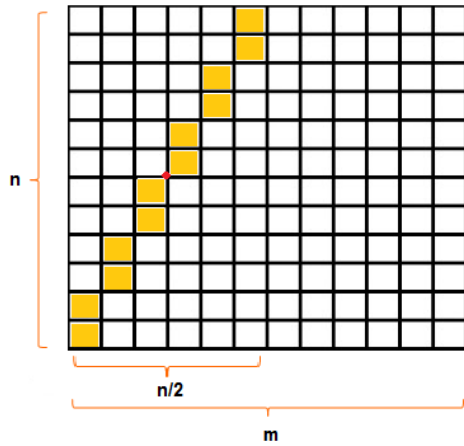
The angle is searched after the rectangle is found. It is described in detail in Chapter 3.1.1 how to derive the alignment rectangle. The basic idea is to find the rectangle by searching the neighbor area of the detected license plate. Since the lines of the rectangle partition the space into two parts, in a restricted area near the license plate, the color difference on two sides should be most salient.

The angle corresponds to a mode. There are many modes in total, but the search times for the angles depend on the width *wid* and height *ht* of the obtained rectangle. Here is how the modes are defined and related to the angle:

$$\theta = \arctan(mode) \quad (3.7)$$

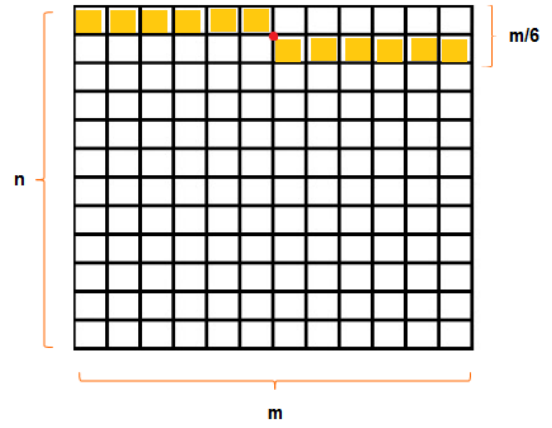
$$\theta' = \arctan(1/mode) \quad (3.8)$$

where in my implementation, $mode \in [-n, -12] \cup [12, n]$. And θ is the angle for the vertical line of the rectangle, while θ' is the angle for the horizontal line of the rectangle.



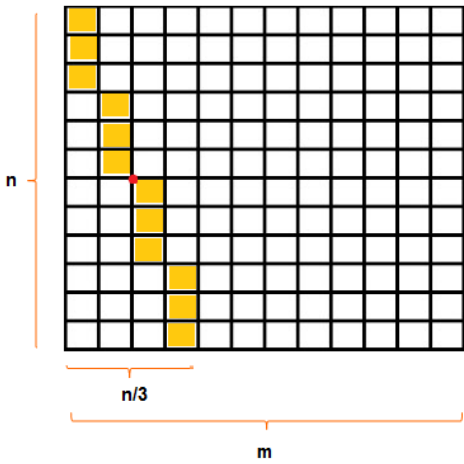
$$\theta = \arctan 2$$

(a)



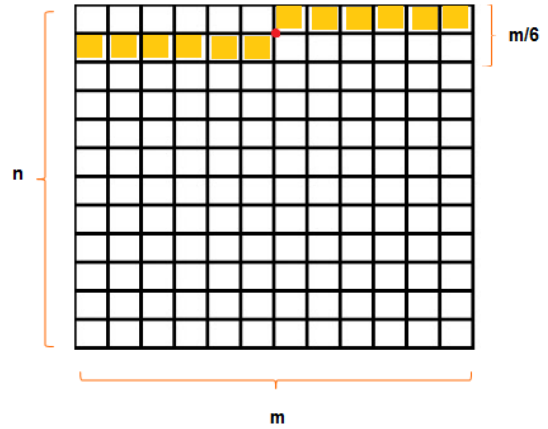
$$\theta' = \arctan (1/6)$$

(b)



$$\theta = \arctan (-3)$$

(c)



$$\theta' = \arctan (-1/6)$$

(d)

Fig. 3. 10. The modes of angles. Some examples: (a) For the vertical line of the rectangle, the mode is 2; (b) For the horizontal line of the rectangle, the mode is 1/6; (c) For the vertical line of the rectangle, the mode is -3; (d) For the horizontal line of the rectangle, the mode is -1/6.

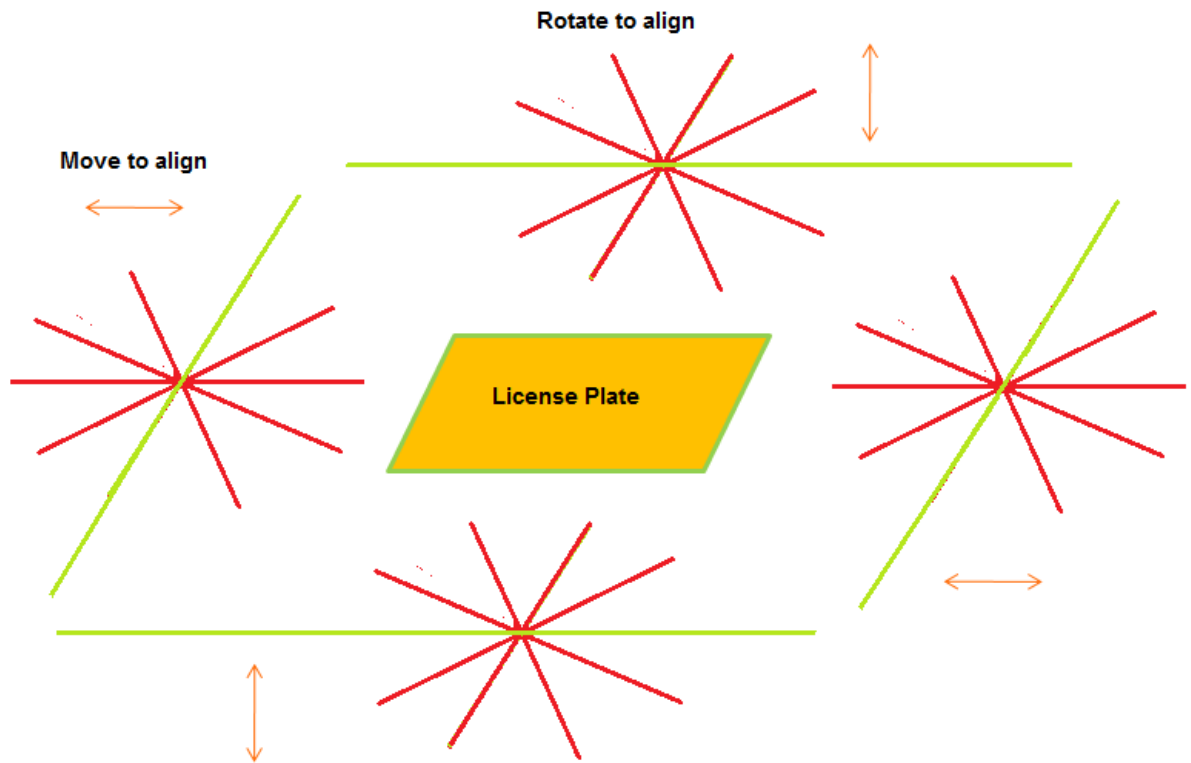


Fig. 3. 11. The alignment process. The lines are searching for position as well as the angle.

This is implemented but not used because the license plates in this dataset for testing are seldom very tilted, and the license plate with the slight tilt can still be successfully segmented. The angle searching is turned off to save some computation time.

3.2 Plate Binarization Using K-means Clustering

With the license plate successfully aligned, the region of interest (ROI) is a rectangle that contains only the text on the plate and the background of the plate. The Chinese license plates only have two colors. One is the color of the digits, which is usually black; one is the background

color of the plate, whose color is uniform. With the two conditions met, it is eligible to perform k-means clustering to binarize the license plate pixels. Basically, k-means classify the license plate pixels into two clusters. One cluster is the pixels of the foreground, which is the text, the other cluster being the background.

Since the number of pixels that belong to the text is always smaller than that of the background, it is simple for us to distinguish the two clusters and therefore assign corresponding binary values. Fig. 3. 12 shows the results of k-means clustering on the detected license plates.



Fig. 3. 12. Results of k-means clustering on aligned license plates

3.2.1 K-means Clustering

Clustering is the process that partitions or groups a given set of patterns into disjoint clusters. It is done such that patterns in the same cluster resemble each other and patterns belonging to two different clusters are distinct. K-means is a commonly used method for clustering in unsupervised learning. It is an NP hard problem but with refinement algorithms it

may converge quickly to local minimum. K-means method has been shown to be an effective clustering algorithm for many practical applications [90].

The number of clusters k is assumed to be fixed in k-means clustering. Let the k prototypes (w_1, \dots, w_k) be initialized to one of the n input patterns (w_j, \dots, i_l) . We have,

$$w_j = i_l, j \in \{1, \dots, k\}, l \in \{1, \dots, n\} \quad (3.9)$$

C_j is the j^{th} cluster and its value is a disjoint subset of input patterns. An error function is used to determine the quality of the clustering:

$$E = \sum_{j=1}^k \sum_{i_l \in C_j} |i_l - w_j|^2 \quad (3.10)$$

The appropriate choice of k is a problem, which is domain dependent. Usually a user tries several values of k . In the context of binarizing aligned license plate, however, the k is certainly set to 2, assuming there are only two colors of the aligned license plate, the foreground color which is the color of the text, and the background color which is the plate color. The general k-means clustering algorithm is described in the following figure:

```

function Direct-k-means()
  Initialize  $k$  prototypes  $(w_1, \dots, w_k)$  such that
   $w_j = i_l, j \in \{1, \dots, k\}, l \in \{1, \dots, n\}$ 
  Each cluster  $C_j$  is associated with prototype  $w_j$ 
  Repeat
    for each input vector  $i_l$ , where  $l \in \{1, \dots, n\}$ ,
      do
        Assign  $i_l$  to the cluster  $C_{j^*}$  with nearest
        prototype  $w_{j^*}$ 
        (i.e.,  $|i_l - w_{j^*}| \leq |i_l - w_j|, j \in$ 
         $\{1, \dots, k\}$ )
      for each cluster  $C_j$ , where  $j \in \{1, \dots, k\}$ , do
        Update the prototype  $w_j$  to be the
        centroid of all samples currently
        in  $C_j$ , so that  $w_j = \sum_{i_l \in C_j} i_l / |C_j|$ 

  Until  $E$  does not change significantly or cluster membership
  no longer changes

```

Fig. 3. 13. The general k-means clustering algorithm

3.3 License Plate Segmentation

In order to recognize the characters with the binary information within the aligned plate, the plate is first segmented into patches where each patch should contain one and only one character, which will be classified by an SVM with simple features.

As shown in Fig. 3. 14, the segmentation is based on a model, specifically designed for Chinese license plates. The prior knowledge we can infer from the Chinese license plates is the layout of the characters: There are in total 7 characters within a plate, and the first two characters are separated from the latter five by a dot. The width of each character should be the same while the dot interrupts the consistency with its own width.

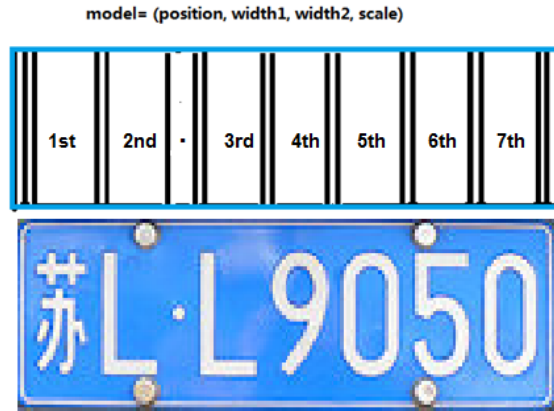


Fig. 3. 14. The layout of a Chinese license plate

Here is how this model works. The aligned plate will first be extended on the left and right side to guarantee the range of scan, as shown in Fig. 3. 15.

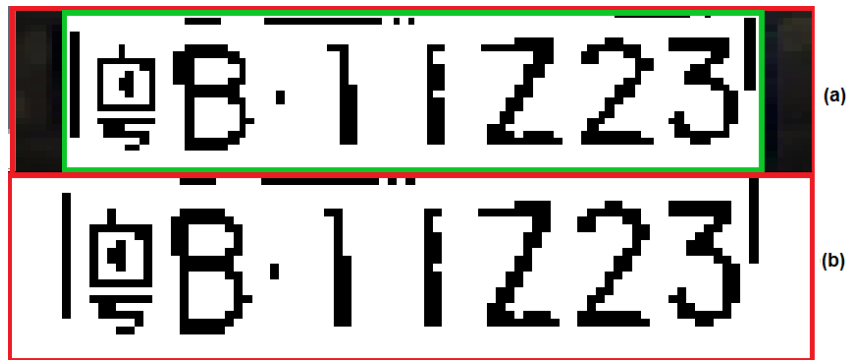


Fig. 3. 15. The pre-processing of segmentation: (a) The green rectangle shows the alignment result of the license plate. The rectangle will be extended to the red rectangle to ensure some scan range. (b) The pixels of extended region will be assigned 255 as they are considered to be background pixels.

The binary plate is scanned vertically at each horizontal location in order to compute the histogram $h(x)$, which will be used to calculate the fitness score for each model. The model is deformable with its position and width and scale, therefore the fitness score will be a variable dependent on the model. The target problem of segmentation is to find the optimum score and its

corresponding model which is considered to be the model that best describes the segmentation. It is described below how the histogram $h(x)$ is computed.

Scan vertically at each x axis of the plate, the 255 to 0 inversion contributes to the score, which indicates that there is a character stroke at this y axis. The reason for counting the occurrence of inversion rather than counting the occurrence an absolute value is that the stroke could be of some width, and the same stroke should not be counted for multiple times. Otherwise, the histogram is not normalized for characters whose size is different. In Fig. 3.16, the y axis shows the score at the corresponding x axis of the plate.

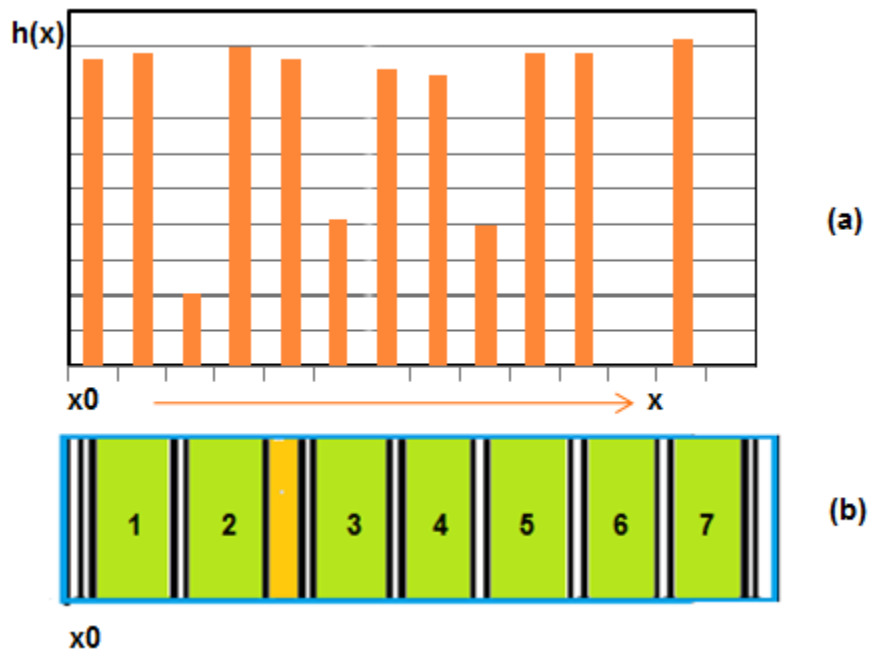


Fig. 3. 16. The histogram of the license plate derived from the vertical 0-1 inversion scan. (a) The histogram $h(x)$; (b) The corresponding model.

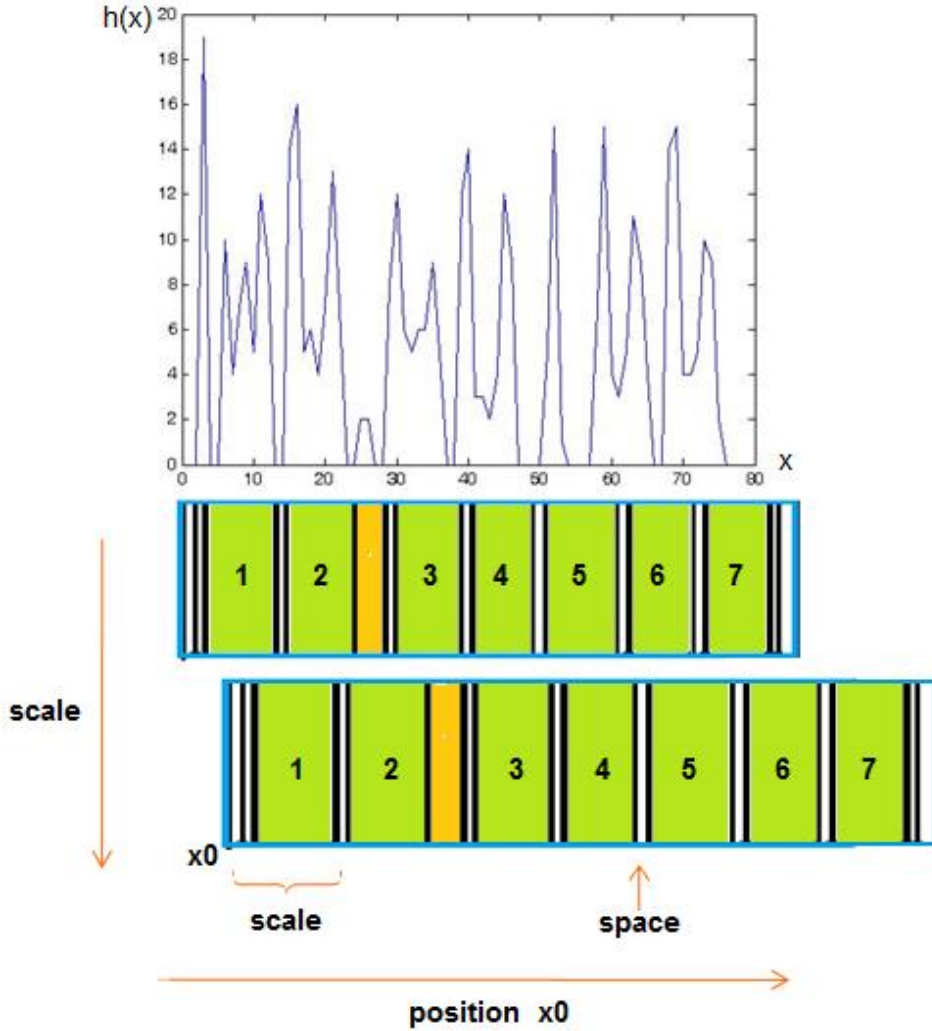


Fig. 3. 17. The variables that will be explored: (scale, x, space). The green segment is character segment, while the yellow segment symbolizes dot segment. The white segments are spaces.

The optimization function is:

$$f(x) = \sum_{x=0}^{max} s(x) \quad (3.11)$$

where $s(x)$ is the score of each x column.

The score is computed based on the histogram derived by vertical scanning of 0-1 pattern.

$$s(x) = \begin{cases} h(x), & x \text{ belongs to character segment} \\ -h(x), & x \text{ belongs to space segment or dot segment} \end{cases} \quad (3.12)$$

where $h(x)$ is the histogram.

Experiments have been conducted to explore the necessity of giving the raw histogram a threshold, as shown in Fig. 3.18. It may look better than the raw histogram, however, the histogram after thresholding need proper thresholds that are dependent on the character in the segment. Besides, some information is lost. The small column in Fig . 3. 18 (b) pointed by an orange arrow, for example, corresponds to the small dot between the 2nd and the 3rd characters of the license plate, which is not reflected in the histogram after thresholding as shown in Fig. 3. 18 (a).

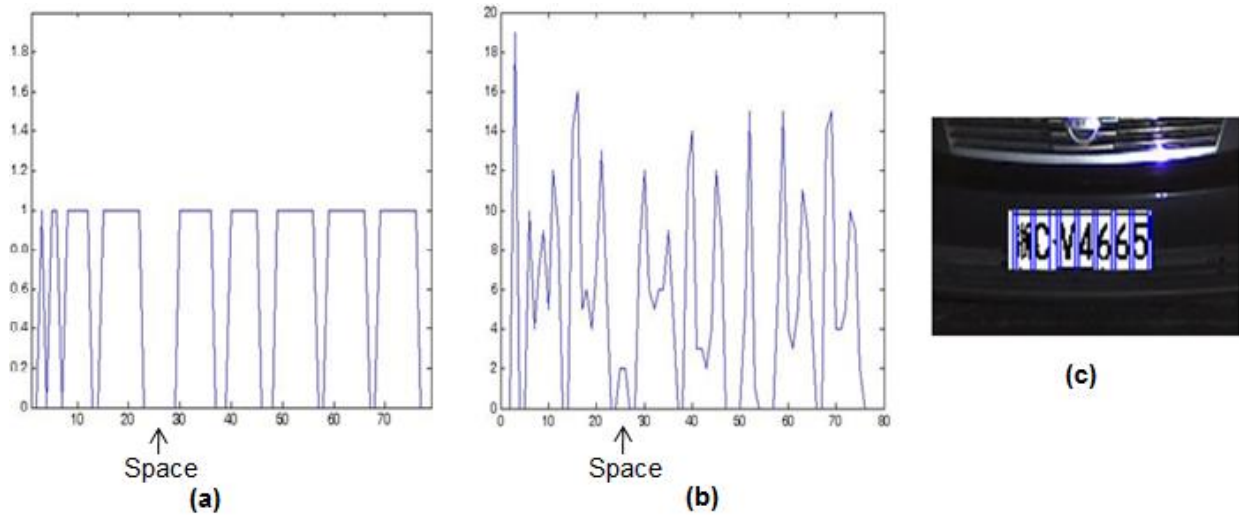


Fig. 3. 18. The process of license plate segmentation: (a) the histogram after thresholding; (b) the raw histogram; (c) the segmented license plate

Here are some results of segmenting the binary license plates obtained, as shown in Fig. 3. 19.

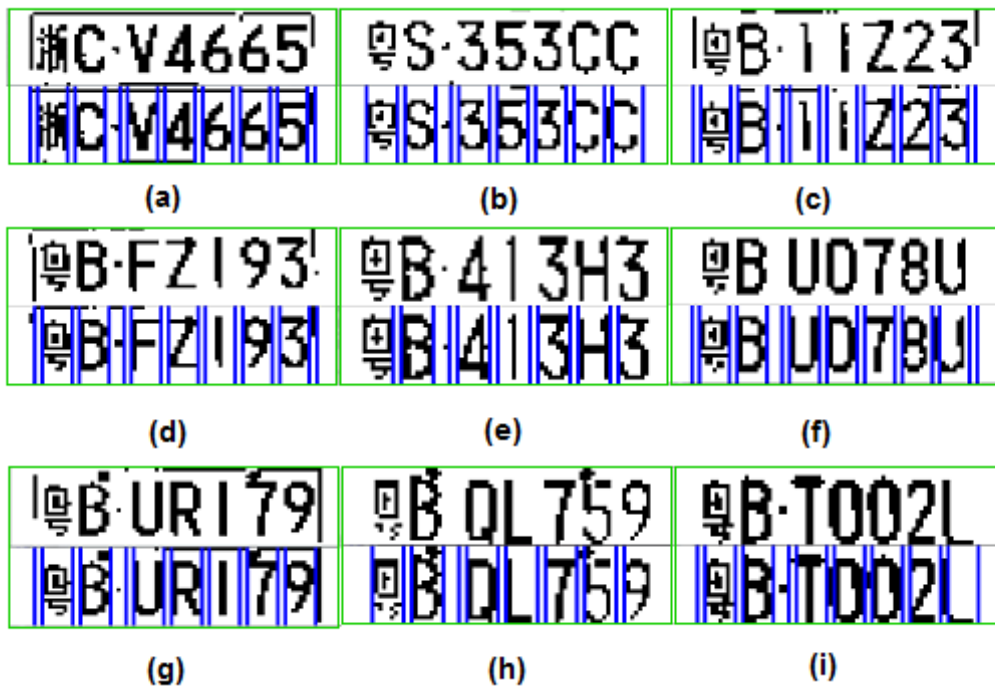


Fig. 3. 19. The results of segmentation

3.4 Digit Characters and Capital Letters Recognition Using Simple Robust Features

Character recognition is a 1 V.S N classification problem. We use SVM to classify multiple times until classification result is positive. Before SVM, we also tried a Bag-of-words model, and experimented with four voting schemes.

There are 5 set of features in all. They are concatenated and the overall feature length is: 576.

The segmented patches are all resized to 16 by 32. Such features are extracted from patch:

- (1). The histogram of vertical 0-1 inversion pattern.
- (2). The histogram of horizontal 0-1 inversion pattern.

- (3). The histogram of 0/1 ratio vertically.
- (4). The histogram of 0/1 ratio horizontally.
- (5). The raw feature.

3.4.1 Using Bag-of-words Model

The characters were initially recognized by a bag-of-words model. We used 100 templates for the voting. The character will get the corresponding label if it receives most votes from a certain class.

The Euclidean distance is computed between the features of the to-be-recognized character patch and the features of each template. The distances fill form a matrix where the column index symbolizes different individuals of templates from the same class, while the row index symbolizes different classes.

The recognition rate of the letters were 93%. Then different voting schemes were tried and experiments were conducted on which voting scheme is the best. The result is shown in the following table:

Result comparison of four voting schemes

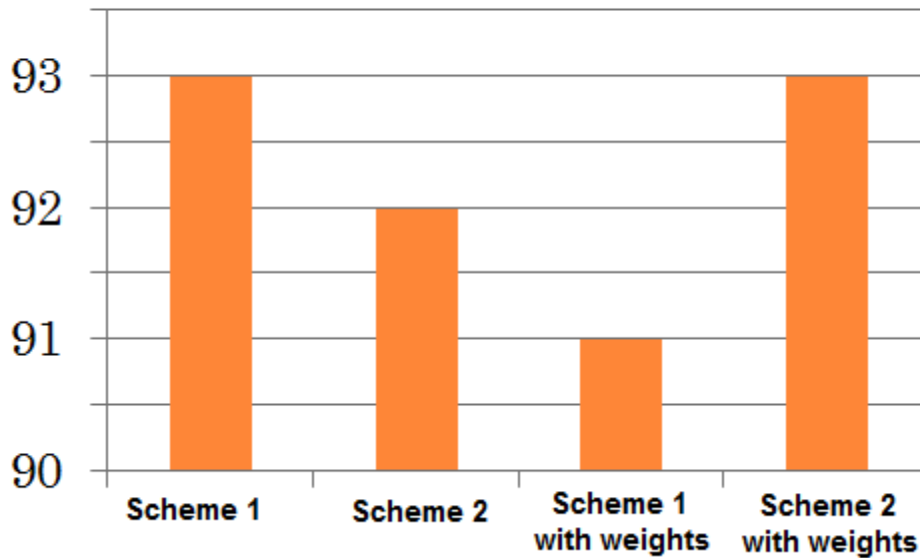


Fig. 3. 20. The results of four different voting schemes.

Voting scheme 1 is the scheme that we find the best 10 votes whose distance scores are the minimum top 10. And we count the 10 top votes for each class. The class that get the most votes will be considered the class this character belongs to. If there are any classes that have equal votes, the overall scores of votes will be computed for the classes, and that which has the minimum score is considered where this character belongs.

Voting scheme 2 is such a scheme that we find the two (or more) templates for each class that have the minimum feature distance from the query character patch, after which we compute the average score of these two (or more) distances as the final score. The class that has the minimum final score will be regarded the one this character should belong to.

Another two different schemes are the same as the two schemes described above, except that automatic feature-weighting has been added respectively [88]. The weights for the feature are trained based on the inner and inter scores computed between the templates. The inner scores are computed between templates that are of the same class, while the inter scores are computed between templates that belong to different classes. This method explores the template feature space and renders the weights that will help provide the features that best represent the templates, which means that the template features within a same class are optimally close in feature space and that the template features among different classes are most remote. It therefore utilize the voting scheme.

The weights are computed as

$$w(n) = 10 \times \frac{(\bar{m}_{inner} - \bar{m}_{inter})^2}{\sigma^2_{inner} + \sigma^2_{inter}} \quad (3.13)$$

where \bar{m} and σ^2 represent mean and variance separately. The mean and variance are computed for both inner and inter pair groups, where the inner group consist all the possible pairs of templates from the same class and similarly, the inter group is consisted of all the possible pairs that come from two different classes. Here n represents the n th feature of the feature vector. For each feature, a weight is trained and computed as described above and thus we have a weight vector corresponding to the feature vector used for character recognition. Each element in the feature vector scaled by the corresponding weight is the element of the new feature, which is used for the weighted voting scheme.

The result shows that the weighting does help with vote scheme 2, since the scores computed are optimized. However, for vote scheme 1, the weighting reduces the recognition rate, which is due to the fact that for some classes, there exist templates that are not quite

representative of this class. These templates sometimes provide medium distance scores, which are not so good as scores the templates of the true class could provide, but are good enough to be top 10 and therefore able to vote. A wrong class is determined by such low-quality votes if the number of them is greater. Usually this happens when the character patch has some dirt on it. The recognition rate drops because the features with the weight produce better scores for such unrepresentative templates too.

The result was then raised to 95% by a 2-level voting scheme, which is a combination of two different voting schemes. The voting scheme is based on an observation of the results from the above four different voting schemes. We noticed that we can trust scheme 2 for its decision if the ground truth of the characters are from 7-9. If the result of it is from 0-8, then the result can be right or wrong, and the confidence is not very high. Scheme 1 has an equally confidence for each class, so it is set at the second cascade. It recognizes whatever character that scheme 2 cannot be sure of. And its result is steady. This 2-level voting scheme is shown in the following flowchart.

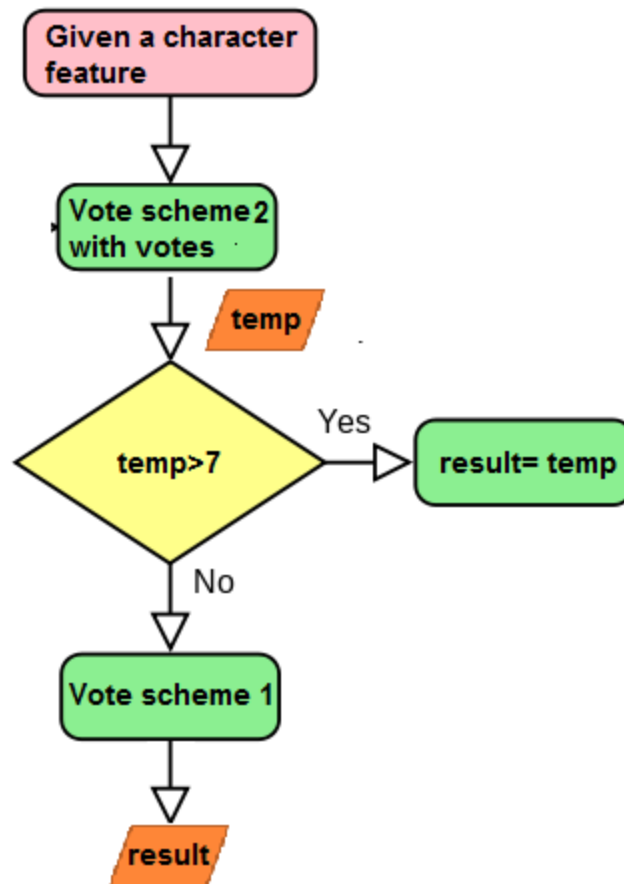


Fig. 3. 21. Flowchart of the 2-level voting scheme

The recognition rate is 95% for this combined voting scheme, higher than any of the above four voting schemes. But the recognition rate is still not very satisfactory. Therefore, we finally decided to adopt a support vector machine (SVM) to perform classification on the character patch images in this thesis, which has been described in detail in chapter 2.

3.4.2 Using SVM Classifier

The features (1)-(4) are 16 dimensional each, while feature (5) is 512 dimensional. These four sets of features are concatenated into a single vector as the feature to serve for the SVM classifier, which is used to recognize the character patches by classifying them into

corresponding categories. 1 vs All scheme is used. There are 10 digits and 26 capital letters in all, and 36 classes is the minimum number.

Experiments have been conducted on digits-only classifier and digit-and-letter classifier. The recognition rate of the digits-only classifier is 99%. The recognition rate of the digit-and-letter classifier is 95%. With appropriate features, SVM can achieve fairly good results.

3.5 Dataset and Results

The plate detection is tested on two datasets, one is daylight and contains 43 images, the other one is night and contains 62 images. The detection rate is both 100% and there is no false positives. All of them are well aligned.

For night dataset, the speed for detection is around 200ms on average with 3 scanning scales and the step width being 8. For the daylight dataset, the plates are fairly small, and in order to ensure 100% detection rate, the scanning is performed at 3 scales while the step width to scan is set down to 2. And the average detection time is around 1000ms. If we tolerate some miss, the time could be around 200ms too.

If less than 10% of the character is cut off by segmentation, it is considered that the segmentation is correct. In this sense, the segmentation is 100% correct for the dataset.

For digits-only recognition, we trained 400 characters and tested 100 characters. The training and testing characters are derived from the segmentation output patches. The digits-only recognition rate for segmented single characters is 99%. For digit-and-letter recognition, since the letter patches are not abundant, we used synthetic letter patches for training and the segmentation output patches for testing. The digit-patches trained and tested for digits-only

recognition are added respectively to training and testing groups. There are 993 patches trained and 228 patches tested in total. And the digit-and-letter recognition rate is 95%.





Fig. 3. 22. Sampled results of global alignment, k-means binarization, segmentation and character recognition. The recognition part ignores Chinese characters and output a "*" instead.

3.6 Summary

In this chapter, we introduced a model to align the license plates detected by SVM, which is described in chapter 2. The alignment ensures that the boundary of the ROI window is accurate enough to binarize the license plate based on color information. K-means clustering is applied to group the pixels within the license plate into two categories, the foreground and the background. Based on the binary license plate, a deformable model is proposed to segment the license plate into character patches, which explores in its parameter space for an optimum model that maximizes the fitness score. The character patches are then used to extract robust features, with which the bag-of-words model and SVM are experimented to recognize the patches. The SVM achieves better performance, with a recognition rate of 99% on digits and 95% on letters with digits.

CHAPTER 4

REAL TIME EMBEDDED SYSTEM

This chapter is about implanting the License Plate Detection and Character Recognition program to Ts-7250 or other hardware with necessary peripherals to simulate the process of vehicle retrieval. After segmentation of the plate into several characters, multiple threads will be created to recognize each character. If one of those threads has met a problem, at least some of the characters will be recognized. This is very important in that if one character is missing, the program would still provide important information for the traffic police.

4.1 Hardware Part

Multiple TS-7250 boards and one auxiliary board are needed. One TS-7259 board will play the role of "main board", others serve as slaves, called "child boards". The "main board" performs segmentation on license plate images. Its auxiliary board provides time to capture/load license plate images. "Child boards" need to receive segmented characters from the "main board". The characters are transmitted through ports. The "child boards" will recognize the character received and then send the recognized number back to the "main board" through ports. The "main board" subsequently exhibits the recognized numbers.

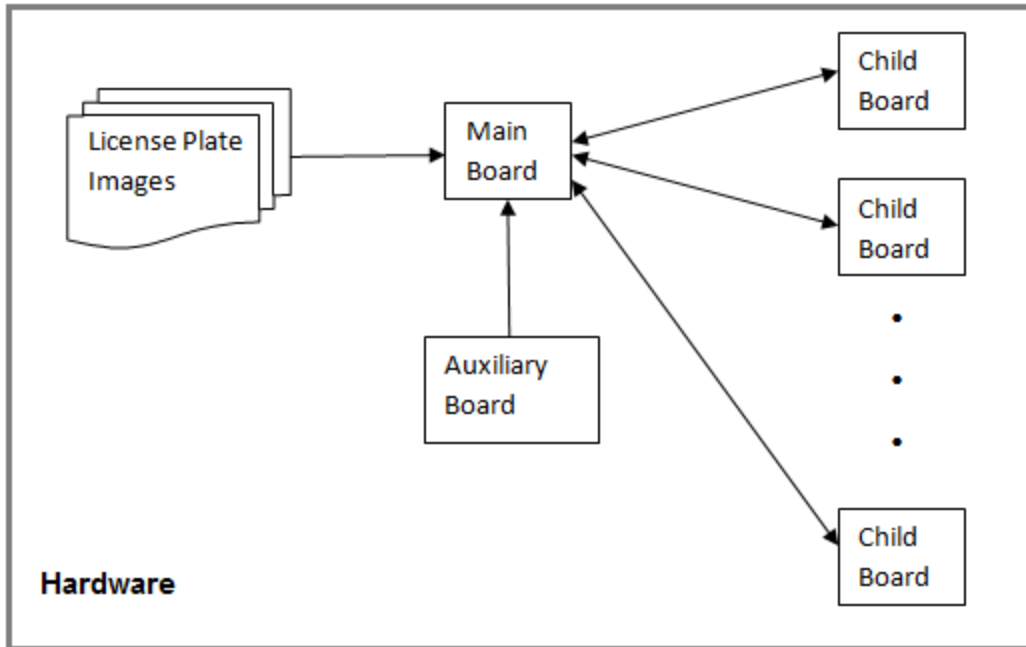


Fig. 4. 1. Diagram of the hardware part.

4.2 Software Part

4.2.1 The "Main Board" Side

The module for the "main board" will check the B0 button of its auxiliary board. Once B0 button on the auxiliary board of the TS-7250 "main board" is pressed, which simulates the process of this Vehicle retrieval system capturing the image of vehicles, the module will send the message through a fifo to the main thread, who will then read license plates from a pre-defined directory. The main thread will thereafter perform segmentation of the plate.

The user space program for the "main board" has three set of threads. Aside from the main thread mentioned above, one set of threads will ask from the main thread for the segmented characters through a shared buffer, then send the segmented characters to "child" boards via the ports. Another set of threads will do the job of receiving recognized numbers from "child boards".

Once the numbers are received, the threads send them to the main thread through several named pipes.

Once the main thread receives the recognized numbers, since it knows which number comes from which thread, it then prints out the numbers in the same order as they are shown on the plate. Semaphores will be used as a form of synchronization. The main thread only prints out the results after the threads that send the numbers are synchronized. The whole process is shown in Fig. 4.2.

4.2.2 The "Child Boards" Side

There are two threads for the user space program(for each of the child board). For each child board, one thread receives the corresponding character from the "main board", put them in a shared buffer. The other thread reads from the shared buffer the segmented character, perform character recognition, and then send the recognized number back to its corresponding "main board"-thread.

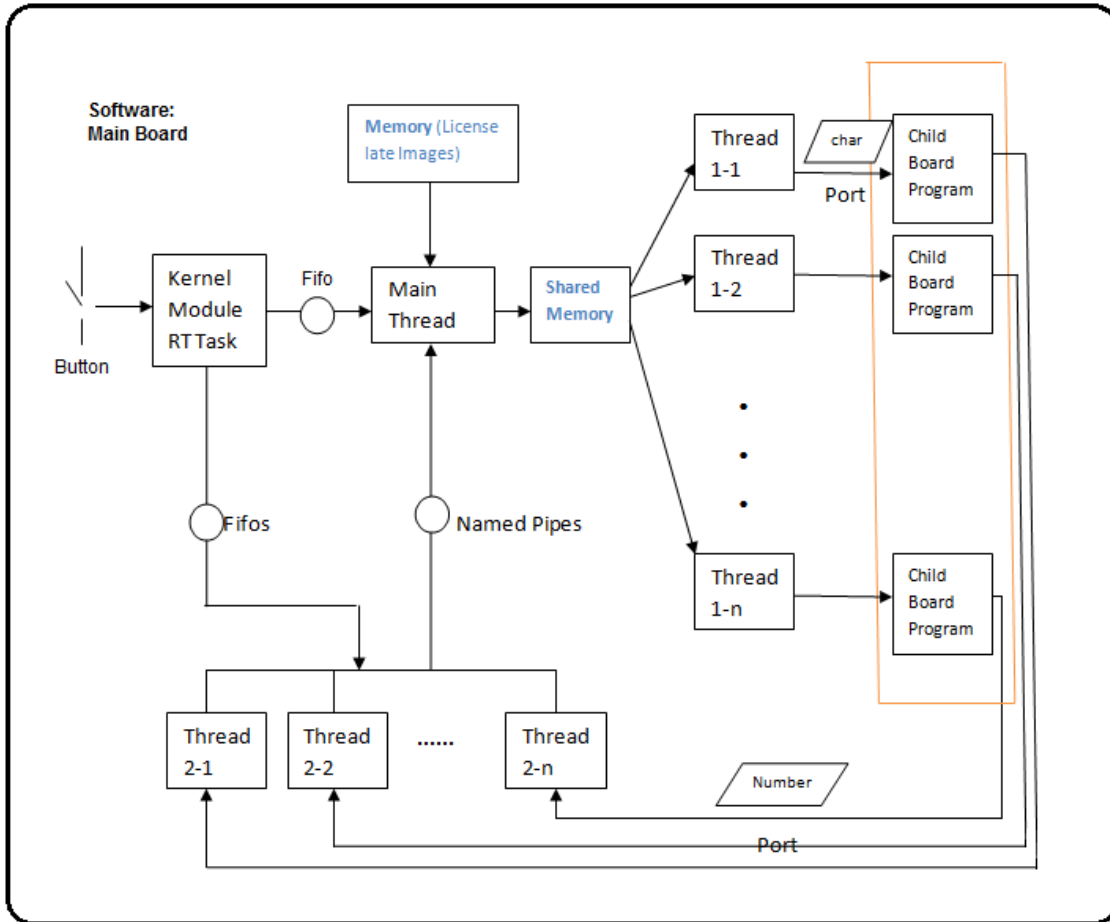


Fig. 4. 2. Software part- "main board" side. The part within orange rectangle belongs to "child board program"

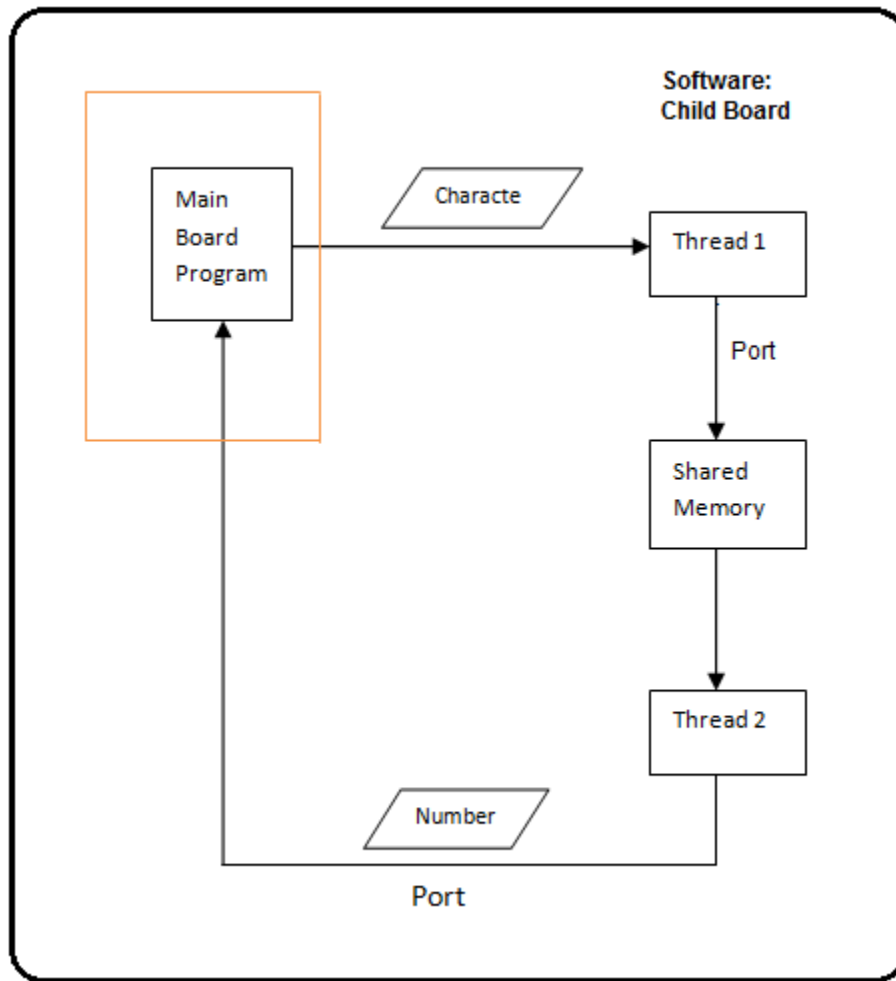


Fig. 4. 3. Software part- "child board" side. The part within orange rectangle belongs to the "main board"

4.3. Implementation

4.3.1 Kernel Module

The kernel module is implemented only for the trigger of reading in an image. Once the button on the auxiliary board is pressed, the kernel module sends a flag into the user space program of the main board through FIFO to acknowledge that the camera is triggered, so an image should be read. In real life experience, this trigger should come from a pair of sensors on

the high-speed road that supervises vehicles. Here it is done by hand to decide when to read an image. The images are license plates images, which are already extracted from frames of vehicles on the road.

The kernel module is implemented by constantly checking the port B of the auxiliary board. If any of the buttons is pressed, the value of the corresponding register will change and this will trigger the kernel module sending the flag.

4.3.2 Main Board

The Main board reads the image files in "bmp" format and store them in structure CImage. The implementation of this is from open source, contributed by Chen, Lee[89]. The image is stored in shared memory.

Multiple threads are created to perform segmentation of the image into several segments. The segmentation is explained in Chapter 3. Another set of multiple threads are created to listen for the transmitting of recognized numbers from child boards. Flags are used for the synchronization. Each digit corresponds to a flag, which indicates that the digit is received. When all the flags are indicating positive, the main thread of the board will display the recognized plate number. And it clears all these flags afterwards. Unknown digits will be displayed as "*".

4.3.3 Socket and TCP

Instead of using UDP, TCP is chosen as a more reliable way to transmit images. Besides, the ports are pre-allocated to boards by hand. Multiple sockets will be created and they use different ports. The main board creates multiple sockets and is the server of them to send

segments to child boards. At the same time, the main board is also the client of each child board server, which sends recognized digits to the main board. Likewise, each child board plays both roles of server and client, but in different sockets.

In the implementation, the main board program creates sockets as a server first. When child board programs are executed, the child board programs will connect to the main board, and then creates their own sockets as a server. When the button of the auxiliary board is pressed for the first time, threads will be created for the main board to join and connect to child board programs as a client. Again, the synchronization is done with flags.

4.3.4 Child Board

The child board program has two threads, one creating a socket and acting as the server, the other one connecting to the main board and being the client. The server thread sends the recognized number, while the client thread receives segments and perform the character recognition, whose theories are explained in previous chapter.

4.4. Results and Discussion:

The framework of this implementation is steady, with the communication being reliable and stable. It is also flexible to re-configure into other real-life applications. And the recognition rate is 99% based on a 400 training characters/100 testing images trial. The segmentation, however, needs future work to reinforce in the embedded system. Without OpenCV, some elegance in the code is gone. Segmentation is to be re-written to deal with the raw data more properly to avoid loss of information.

Future work will be based on the framework. Since the license plate detection, practically speaking, does not necessarily need multi-processor to run and send results back through

network, it is the framework that counts. Other cases, for example, tasks like data training of deep learning that takes a long time to finish, would require multiple processors to work at the same time. Another scenario will be, when a task is on computational intelligence, for example, running EDA algorithm or just genetic algorithm to decide, for instance, a deformable vehicle model. Then the results for each run could be different, and the final result will be based on a statistical evaluation. The multi-processor frame will be useful in that it can be multiple processors running the same program on the same data, and they may render different results. They can send their own results back to the core program to decide the final result based on statistical evaluation. Other examples will be without number, which on the other hand proves this implementation to be worthy and useful.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

In this work, a sliding window based plate detection method is developed using SVM classifier with dense HOG features. The searching is performed at multiple different scales on proportional image. Edge information is used to determine the possible proportion and accelerate the time consuming process to scan. Candidates are finalized with mean shift, an algorithm of non-maximum suppression. An effective method to align the license plate detection result is proposed. And after applying k-means clustering, the aligned license plate can be segmented into character patches using a novel layout-based model.

SVM classifier with some simple features is used to recognize the character patches. It achieved better results than voting by the bag-of-words model, with or without the second level voting scheme and automatic weighting method.

The LPR system was then developed into an embedded system to simulate the vehicle retrieval system by using multiple TS-7250 boards and an auxiliary board.

Future work need to be done to detect and recognize the license plates from video clips in real time. Tracking based on motion compensation can be applied to restrict scanning window area.

REFERENCES

- [1] Shan Du; Ibrahim, M.; Shehata, M.; Badawy, W., "Automatic License Plate Recognition (ALPR): A State-of-the-Art Review," *Circuits and Systems for Video Technology, IEEE Transactions on* , vol.23, no.2, pp.311,325, Feb. 2013
- [2] K. Fatih, K. Binnur, and G. Muhittin, *License plate character segmentation based on the Gabor transform and vector quantization* vol. 2869. New York: Springer- Verlag, 2003.
- [3] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, I. D. Psoroulas, V. Loumos, and E. Kayafas, "License plate recognition from still images and video sequences: a survey," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 9, pp. 377-391, 2008.
- [4] L. Zheng, X. He, B. Samali, and L. T. Yang, "Accuracy enhancement for license plate recognition," in *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, 2010, pp. 511-516.
- [5] L. Miao, F. Wang, and H. Wang, "Automatic license plate detection based on edge density and color model," in *Control and Decision Conference, 2009. CCDC '09. Chinese*, 2009, pp. 3718-3721.
- [6] K. Kanayama, Y. Fujikawa, K. Fujimoto, and M. Horino, "Development of vehicle-license number recognition system using real-time image processing and its application to travel-time measurement," in *Proc. IEEE Veh. Tech. Conf.*, May 1991, pp. 798-804.
- [7] V. Kamat and S. Ganesan, "An efficient implementation of the Hough transform for detecting vehicle license plates using DSPs," in *Proc. Real-Time Tech. Applicat. Symp.*, 1995, pp. 58-59.
- [8] C. Busch, R. Domer, C. Freytag, and H. Ziegler, "Feature based recognition of traffic video streams for online route tracing," in *Proc. IEEE Veh. Tech. Conf.*, vol. 3. May 1998, pp. 1790-1794.
- [9] S. Zhang, M. Zhang, and X. Ye, "Car plate character extraction under complicated environment," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, vol. 5. Oct. 2004, pp. 4722-4726.
- [10] S. Wang and H. Lee, "Detection and recognition of license plate characters with different appearances," in *Proc. Int. Conf. Intell. Transp. Syst.*, vol. 2. 2003, pp. 979-984.
- [11] F. Faradji, A. H. Rezaie, and M. Ziaratban, "A morphological-based license plate location," in *Proc. IEEE Int. Conf. Image Process.*, vol. 1. Sep.-Oct. 2007, pp. 57-60.
- [12] H. Bai and C. Liu, "A hybrid license plate extraction method based on edge statistics and morphology," in *Proc. Int. Conf. Pattern Recognit.*, vol. 2. 2004, pp. 831-834.

- [13] D. Zheng, Y. Zhao, and J. Wang, "An efficient method of license plate location," *Pattern Recognit. Lett.*, vol. 26, no. 15, pp. 2431–2438, 2005.
- [14] A. M. Al-Ghaili, S. Mashohor, A. Ismail, and A. R. Ramli, "A new vertical edge detection algorithm and its application," in *Proc. Int. Conf. Comput. Eng. Syst.*, 2008, pp. 204–209.
- [15] C. Nelson Kennedy Babu and K. Nallaperumal, "An efficient geometric feature based license plate localization and recognition," *Int. J. Imaging Sci. Eng.*, vol. 2, no. 2, pp. 189–194, 2008.
- [16] M. Sarfraz, M. J. Ahmed, and S. A. Ghazi, "Saudi Arabian license plate recognition system," in *Proc. Int. Conf. Geom. Model. Graph.*, 2003, pp. 36–41.
- [17] H.-J. Lee, S.-Y. Chen, and S.-Z. Wang, "Extraction and recognition of license plates of motorcycles and vehicles on highways," in *Proc. Int. Conf. Pattern Recognit.*, 2004, pp. 356–359.
- [18] V. Kamat and S. Ganesan, "An efficient implementation of the Hough transform for detecting vehicle license plates using DSPs," in *Proc. Real-Time Tech. Applicat. Symp.*, 1995, pp. 58–59.
- [19] T. D. Duan, D. A. Duc, and T. L. H. Du, "Combining Hough transform and contour algorithm for detecting vehicles' license-plates," in *Proc. Int. Symp. Intell. Multimedia Video Speech Process.*, 2004, pp. 747–750.
- [20] T. D. Duan, T. L. H. Du, T. V. Phuoc, and N. V. Hoang, "Building an automatic vehicle license-plate recognition system," in *Proc. Int. Conf. Comput. Sci. RIVF*, 2005, pp. 59–63.
- [21] D.-S. Kim and S.-I. Chien, "Automatic car license plate extraction using modified generalized symmetry transform and image warping," in *Proc. IEEE Int. Symp. Ind. Electron.*, vol. 3, Jun. 2001, pp. 2022–2027.
- [22] J. Xu, S. Li, and Z. Chen, "Color analysis for Chinese car plate recognition," in *Proc. IEEE Int. Conf. Robot. Intell. Syst. Signal Process.*, vol. 2, Oct. 2003, pp. 1312–1316.
- [23] B.-F. Wu, S.-P. Lin, and C.-C. Chiu, "Extracting characters from real vehicle license plates out-of-doors," *IET Comput. Vis.*, vol. 1, no. 1, pp. 2–10, 2007.
- [24] Z. Qin, S. Shi, J. Xu, and H. Fu, "Method of license plate location based on corner feature," in *Proc. World Congr. Intell. Control Automat.*, vol. 2, 2006, pp. 8645–8649.
- [25] J. Matas and K. Zimmermann, "Unconstrained license plate and text localization and recognition," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Sep. 2005, pp. 225–230.
- [26] P. Wu, H.-H. Chen, R.-J. Wu, and D.-F. Shen, "License plate extraction in low resolution video," in *Proc. Int. Conf. Pattern Recognit.*, vol. 1, 2006, pp. 824–827.
- [27] N. Bellas, S. M. Chai, M. Dwyer, and D. Linzmeier, "FPGA implementation of a license plate recognition SoC using automatically generated streaming accelerators," in *Proc. IEEE Int. Parallel Distributed Process. Symp.*, Apr. 2006, pp. 8–15.

- [28] M. M. I. Chacon and S. A. Zimmerman, "License plate location based on a dynamic PCNN scheme," in *Proc. Int. Joint Conf. Neural Netw.*, vol. 2. 2003, pp. 1195–1200.
- [29] K. Miyamoto, K. Nagano, M. Tamagawa, I. Fujita, and M. Yamamoto, "Vehicle license-plate recognition by image analysis," in *Proc. Int. Conf. Ind. Electron. Control Instrum.*, vol. 3. 1991, pp. 1734–1738.
- [30] H.-K. Xu, F.-H. Yu, J.-H. Jiao, and H.-S. Song, "A new approach of the vehicle license plate location," in *Proc. Int. Conf. Parallel Distr. Comput. Applicat. Tech.*, Dec. 2005, pp. 1055–1057.
- [31] Y. S. Soh, B. T. Chun, and H. S. Yoon, "Design of real time vehicle identification system," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, vol. 3. Oct. 1994, pp. 2147–2152.
- [32] R. Parisi, E. D. D. Claudio, G. Lucarelli, and G. Orlandi, "Car plate recognition by neural networks and image processing," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 3. Jun. 1998, pp. 195–198.
- [33] V. S. L. Nathan, J. Ramkumar, and S. K. Priya, "New approaches for license plate recognition system," in *Proc. Int. Conf. Intell. Sens. Inform. Process.*, 2004, pp. 149–152.
- [34] V. Seetharaman, A. Sathyakhala, N. L. S. Vidhya, and P. Sunder, "License plate recognition system using hybrid neural networks," in *Proc. IEEE Annu. Meeting Fuzzy Inform.*, vol. 1. Jun. 2004, pp. 363–366.
- [35] C. Anagnostopoulos, T. Alexandropoulos, S. Boutas, V. Loumos, and E. Kayafas, "A template-guided approach to vehicle surveillance and access control," in *Proc. IEEE Conf. Adv. Video Signal Based Survei.*, Sep. 2005, pp. 534–539.
- [36] C.-T. Hsieh, Y.-S. Juan, and K.-M. Hung, "Multiple license plate detection for complex background," in *Proc. Int. Conf. Adv. Inform. Netw. Applicat.*, vol. 2. 2005, pp. 389–392.
- [37] F. Yang and Z. Ma, "Vehicle license plate location based on histogramming and mathematical morphology," in *Proc. IEEE Workshop Automa. Identification Adv. Tech.*, Oct. 2005, pp. 89–94.
- [38] R. Bremananth, A. Chitra, V. Seetharaman, and V. S. L. Nathan, "A robust video based license plate recognition system," in *Proc. Int. Conf. Intell. Sensing Inform. Process.*, 2005, pp. 175–180.
- [39] R. Zunino and S. Rovetta, "Vector quantization for license-plate location and image coding," *IEEE Trans. Ind. Electron.*, vol. 47, no. 1, pp. 159–167, Feb. 2000.
- [40] C.-N. E. Anagnostopoulos, I. E. Anagnostopoulos, V. Loumos, and E. Kayafas, "A license plate-recognition algorithm for intelligent transportation system applications," *IEEE Trans. Intell. Trans. Syst.*, vol. 7, no. 3, pp. 377–392, Sep. 2006.
- [41] K. Deb, H.-U. Chae, and K.-H. Jo, "Vehicle license plate detection method based on sliding concentric windows and histogram," *J. Comput.*, vol. 4, no. 8, pp. 771–777, 2009.
- [42] H. Caner, H. S. Gecim, and A. Z. Alkar, "Efficient embedded neural-network-based license plate recognition system," *IEEE Trans. Veh. Tech.*, vol. 57, no. 5, pp. 2675–2683, Sep. 2008.

- [43] F. Kahraman, B. Kurt, and M. Gokmen, *License Plate Character Segmentation Based on the Gabor Transform and Vector Quantization*, vol. 2869. New York: Springer-Verlag, 2003, pp. 381–388.
- [44] Y.-R. Wang, W.-H. Lin, and S.-J. Horng, “A sliding window technique for efficient license plate localization based on discrete wavelet transform,” *Expert Syst. Applicat.*, vol. 38, pp. 3142–3146, Oct. 2010.
- [45] H. Zhang, W. Jia, X. He, and Q. Wu, “Learning-based license plate detection using global and local features,” in *Proc. Int. Conf. Pattern Recognit.*, vol. 2. 2006, pp. 1102–1105.
- [46] L. Dlagnekov, *License Plate Detection Using AdaBoost*. San Diego, CA: Computer Science and Engineering Dept., 2004
- [47] W. Le and S. Li, “A hybrid license plate extraction method for complex scenes,” in *Proc. Int. Conf. Pattern Recognit.*, vol. 2. 2006, pp. 324–327.
- [48] S. Z. Wang and H. J. Lee, “A cascade framework for a real-time statistical plate recognition system,” *IEEE Trans. Inform. Forensics Security*, vol. 2, no. 2, pp. 267–282, Jun. 2007.
- [49] X. Shi, W. Zhao, and Y. Shen, “Automatic license plate recognition system based on color image processing,” *Lecture Notes Comput. Sci.*, vol. 3483, pp. 1159–1168, 2005.
- [50] E. R. Lee, P. K. Kim, and H. J. Kim, “Automatic recognition of a car license plate using color image processing,” in *Proc. IEEE Int. Conf. Image Process.*, vol. 2. Nov. 1994, pp. 301–305.
- [51] S.-L. Chang, L.-S. Chen, Y.-C. Chung, and S.-W. Chen, “Automatic license plate recognition,” *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 1, pp. 42–53, Mar. 2004.
- [52] S. K. Kim, D. W. Kim, and H. J. Kim, “A recognition of vehicle license plate using a genetic algorithm based segmentation,” in *Proc. Int. Conf. Image Process.*, vol. 2. 1996, pp. 661–664.
- [53] S. Yohimori, Y. Mitsukura, M. Fukumi, N. Akamatsu, and N. Pedrycz, “License plate detection system by using threshold function and improved template matching method,” in *Proc. IEEE Annu. Meeting Fuzzy Inform.*, vol. 1. Jun. 2004, pp. 357–362.
- [54] W. Jia, H. Zhang, X. He, and Q. Wu, “Gaussian weighted histogram intersection for license plate classification,” in *Proc. Int. Conf. Pattern Recognit.*, vol. 3. 2006, pp. 574–577.
- [55] W. Jia, H. Zhang, X. He, and M. Piccardi, “Mean shift for accurate license plate localization,” in *Proc. IEEE Conf. Intell. Transp. Syst.*, Sep. 2005, pp. 566–571.
- [56] W. Jia, H. Zhang, and X. He, “Region-based license plate detection,” *J. Netw. Comput. Applicat.*, vol. 30, no. 4, pp. 1324–1333, 2007.
- [57] L. Pan and S. Li, “A new license plate extraction framework based on fast mean shift,” vol. 7820, pp. 782007-1-782007-9, Aug. 2010.

- [58] F. Wang, L. Man, B. Wang, Y. Xiao, W. Pan, and X. Lu, "Fuzzy-based algorithm for color recognition of license plates," *Pattern Recognit. Lett.*, vol. 29, no. 7, pp. 1007–1020, 2008.
- [59] Boris Epshtein, Eyal Ofek, and Yonatan Wexler, "Detecting Text in Natural Scenes with Stroke Width Transform," *CVPR2010*
- [60] S. Draghici, "A neural network based artificial vision system for license plate recognition," *Int. J. Neural Syst.*, vol. 8, no. 1, pp. 113–126, 1997.
- [61] F. Alegria and P. S. Girao, "Vehicle plate recognition for wireless traffic control and law enforcement system," in *Proc. IEEE Int. Conf. Ind. Tech.*, Dec. 2006, pp. 1800–1804.
- [62] W. T. Ho, H. W. Lim, Y. H. Tay, and Q. Binh, "Two-stage license plate detection using gentle Adaboost and SIFT-SVM," in *Proc. 1st Asian Conf. Intell. Inform. Database Syst.*, 2009, pp. 109–114.
- [63] H. W. Lim and Y. H. Tay, "Detection of license plate characters in natural scene with MSER and SIFT unigram classifier," in *Proc. IEEE Conf. Sustainable Utilization Development Eng. Tech.*, Nov. 2010, pp. 95–98.
- [64] Z. Chen, C. Liu, F. Chang, and G. Wang, "Automatic license plate location and recognition based on feature salience," *IEEE Trans. Veh. Tech.*, vol. 58, no. 7, pp. 3781–3785, 2009.
- [65] S. Mao, X. Huang, and M. Wang, "An adaptive method for Chinese license plate location," in *Proc. World Congr. Intell. Control Automat.*, 2010, pp. 6173–6177.
- [66] T. Nukano, M. Fukumi, and M. Khalid, "Vehicle license plate character recognition by neural networks," in *Proc. Int. Symp. Intell. Signal Process. Commun. Syst.*, 2004, pp. 771–775.
- [67] V. Shapiro and G. Gluhchev, "Multinational license plate recognition system: Segmentation and classification," in *Proc. Int. Conf. Pattern Recognit.*, vol. 4. 2004, pp. 352–355.
- [68] B.-F. Wu, S.-P. Lin, and C.-C. Chiu, "Extracting characters from real vehicle license plates out-of-doors," *IET Comput. Vision*, vol. 1, no. 1, pp. 2–10, 2007.
- [69] Y. Cheng, J. Lu, and T. Yahagi, "Car license plate recognition based on the combination of principal component analysis and radial basis function networks," in *Proc. Int. Conf. Signal Process.*, 2004, pp. 1455–1458.
- [70] C. A. Rahman, W. Badawy, and A. Radmanesh, "A real time vehicle's license plate recognition system," in *Proc. IEEE Conf. Adv. Video Signal Based Surveillance*, Jul. 2003, pp. 163–166.
- [71] H. A. Hegt, R. J. Haye, and N.A. Khan, "A high performance license plate recognition system," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, vol. 5. Oct. 1998, pp. 4357–4362.
- [72] B. Shan, "Vehicle license plate recognition based on text-line construction and multilevel RBF neural network," *J. Comput.*, vol. 6, no. 2, pp. 246–253, 2011.

- [73] J. Barroso, E. Dagless, A. Rafael, and J. Bulas-Cruz, "Number plate reading using computer vision," in *Proc. IEEE Int. Symp. Ind. Electron.*, Jul. 1997, pp. 761–766.
- [74] Q. Gao, X. Wang, and G. Xie, "License plate recognition based on prior knowledge," in *Proc. IEEE Int. Conf. Automat. Logistics*, Aug. 2007, pp. 2964–2968.
- [75] A. Capar and M. Gokmen, "Concurrent segmentation and recognition with shape-driven fast marching methods," in *Proc. Int. Conf. Pattern Recognit.*, vol. 1. 2006, pp. 155–158.
- [76] D.-J. Kang, "Dynamic programming-based method for extraction of license plate numbers of speeding vehicle on the highway," *Int. J. Automotive Tech.*, vol. 10, no. 2, pp. 205–210, 2009.
- [77] Y. S. Huang and C. Y. Suen, "A method of combining multiple experts for the recognition of unconstrained handwritten numerals," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 1, pp. 90–93, Jan. 1995.
- [78] C. J. Ahmad and M. Shridhar, "Recognition of handwritten numerals with multiple feature and multistage classifier," *Pattern Recognit.*, vol. 2, no. 28, pp. 153–160, 1995.
- [79] N. Dalal, "Finding people in images and videos," PhD, Institut National Polytechnique De Grenoble, 2006.
- [80] T. Joachims, "Making large scale SVM learning practical ", ed: Universit ät Dortmund, 1998.
- [81] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32(9), 2010.
- [82] Shuangxuan Tian, Shijian Lu, Bolan Su, and Chew Lim Tan, " Scene Text Recognition using Co-occurrence of Histogram of Oriented Gradient," *ICDAR, 2013*
- [83] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, pp. 603-619, 2002.
- [84] ChuCai Yi, Xiaodong Yang, and Yingli Tian, "feature representations for scene text character recognition a comparative study," *ICDAR, 2013*
- [85] "ICDAR 2003 robust reading competitions", Proceedings of Seventh International Conference on Document Analysis and Recognition, 2003, pp. 682-68
- [86] K. Wang and S. Belongie. Word spotting in the wild. In ECCV, 2010. 1, 2, 3, 4, 5
- [87] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection, "in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005, pp. 886-893 vol. 1.
- [88] Nima Razavi, Juergen Gall, Pushmeet Kohli, and Luc van Gool, "*Latent Hough Transform for Object Detection*," *European Conference on Computer Vision, 2012. ECCV 2012*.

- [89] <http://blog.csdn.net/carson2005/article/details/7614125>
- [90] Alsabti, Khaled; Ranka, Sanjay; and Singh, Vineet, "An efficient k-means clustering algorithm" (1997). *Electrical Engineering and Computer Science*. Paper 43.