

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
XIN LI**

Dr. Zhihai He, Thesis Supervisor

DECEMBER 2010

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

VEHICLE LICENSE PLATE DETECTION AND RECOGNITION

presented by Xin Li,

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. Zhihai 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 do I learn a lot from him how to become a successful researcher but also how to become a nice person.

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 helped me a lot with my master thesis. Dr. Tony Xu Han provides me lots great ideas on how to solve problems on my research project.

As a member of Video Processing and Networking Lab, I also like to thank Xiwen Zhao, Wenqing Dai, Li Liu, York Chung, Jay Eggert and Zhongna Zhou. They are my colleagues in the lab and we spent a really nice year of encouraging and helping each other.

At last, I would like to thank my husband and my parents who always support me.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	II
LIST OF FIGURES	IV
LIST OF TABLES	VI
ABSTRACT	VII
Chapter	
1 INTRODUCTION AND BACKGROUND	1
1.1 Research Topic and Objectives	1
1.2 Challenges	1
1.3 Background and Related Work	2
1.3.1 License Plate Detection	3
1.3.2 Character Segmentation	7
1.3.3 License Plate Recognition	9
1.4 Our Previous Work	10
1.5 Overview of Our Methods	13
2 LICENSE PLATE DETECTION USING HOG FEATURES	14
2.1 Scanning window	14
2.2 HOG Features	15
2.2.1 Feature Extraction Procedure (for R-HOG)	16
2.2.2 Implementation	19
2.3 Support Vector Machine	19
2.4 Non-maximum Suppression	19
2.5 Algorithm Speedup	22
2.6 Results and Discussion	23
2.6.1 Descriptor Blocks	25
2.6.2 Result of Testing Images	26
2.6.3 Robustness with Motion Blur	36
2.6.4 Tolerance of Noise	38
2.7 Summary	40
3 LICENSE PLATE RECOGNITION	42
3.1 Digit Characters and Capital Letters Segmentation using HOG	42
3.2 Digit Characters and Capital Letters Recognition using HOG	45
3.2.1 OneVsAll	45
3.2.2 DigitOrLetter	46
3.3.1 Digit Characters and Capital Letter Segmentation using HOG	48
3.3.2 Digit Characters and Capital Letter Recognition using HOG	50
3.4 Summary	53
4 CONCLUSIONS AND FUTURE DIRECTION	54
REFERENCES	56

LIST OF FIGURES

Figure	Page
Fig. 1. 1 License plate samples in 50 states of USA.....	2
Fig. 1. 2 An illustration of the original LBP calculation.	11
Fig. 1. 3 Examples of Multi-Scale LBP.....	12
Fig. 2. 1 An overview of license plate detection procedure.....	14
Fig. 2. 2 Scan-window and image pyramid.	15
Fig. 2. 3 Variants of HOG descriptors. (a) A rectangular HOG (R-HOG) descriptor with 3×3 blocks of cells. (b) Circular HOG (C-HOG) descriptor with the central cell divided into angular sectors as in shape contexts. (c) A C-HOG descriptor with a single central cell[49].	16
Fig. 2. 4 An overview of HOG feature extraction.	16
Fig. 2. 5 A sketch of local normalization.....	18
Fig. 2. 6 Fast license plate detection image. The car rear detected is shown in yellow box. License plates are searched within the car rear region. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.	23
Fig. 2. 7 The performance of the detector with cell size = 4×4 , block size = 12×12 , block stride= 6 before (black dot) and after (red triangle) 4 round of boot strapping process.....	24
Fig. 2. 8 The miss rate at 10-4 FPPW as the cell and block sizes change. The block stride (block overlap) is fixed at half of the block size.....	26
Fig. 2. 9 License plates detected with different lighting conditions using HOG descriptor. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.	28
Fig. 2. 10 license plates with pose variations detected using HOG descriptor. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one	

	location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.	29
Fig. 2. 11	License plates detected with complex background using HOG descriptor. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.	31
Fig. 2. 12	False alarm images by HOG descriptor. Original images were shown on top of each detection images. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.	33
Fig. 2. 13	False alarm detections by HOG descriptor of Caltech datasets (1999).	35
Fig. 2. 14	One image in Caltech dataset applied with motion filters with len pixels of (a) 0 pixels, (b) 10 pixels and (c) 20 pixels in horizontal direction, and (d) 0 pixels, (b) 10 pixels and (c) 20 pixels in vertical direction.	36
Fig. 2. 15	Detection accuracy of license plate detection using HOG feature with horizontal (black curve) and vertical (red curve) motion blur with motion len size from 0 to 20 pixels.	38
Fig. 2. 16	One image in Caltech dataset applied with Gaussian white noise of mean 0 and variance 0 (a), 0.01 (b) and 0.02 (c).	39
Fig. 2. 17	Detection accuracy of license plate detection using HOG feature with Gaussian white noise of mean 0 and variance from 0 to 0.02.	40
Fig. 3. 1	Character segments of license plates. The original license plates are shown on top of each character segments images. The character segments are labeled by the green bounding boxes.	49

LIST OF TABLES

Table	Page
Table 3. 1 Characters segmentation training set details.	43
Table 3. 2 Characters segmentation training set details.	46
Table 3. 3 Training set to separate digits and capital letters.	47
Table 3. 4 Confusion matrix of the OneVsAll approach	51
Table 3. 5 Confusion matrix of the DigitOrLetter approach for digits	52
Table 3. 6 Confusion matrix of the DigitOrLetter approach for letters	52

VEHICLE LICENSE PLATE DETECTION AND RECOGNITION

ABSTRACT

In this work, we develop a license plate detection and recognition 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. A car head and rear detection method was also proposed to accelerate the time consuming scanning process. A comparison of the performance for different cell and block sizes of HOG feature is provided, and four rounds of bootstrapping was performed to achieve better detection performance. 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 the Caltech data set (1999), and achieved a detection rate of 96.0%. We also studied how its performance is impacted by different levels of noise and motion blur.

After license plate detection, we proceed to perform character segmentation and recognition using SVM classifiers with HOG features. In character segmentation, we need to deal with low contrast and tilted plates. The system performs window searching in different scales and analyzes the HOG feature using a SVM and locates their bounding boxes using Mean Shift.

CHAPTER 1

INTRODUCTION AND BACKGROUND

1.1 Research Topic and Objectives

License Plate Recognition (LPR) is a computer vision method used to identify vehicles by their license plates. During recent years, LPR have been widely used as a core technology for security or traffic applications such as in traffic surveillance, parking lot access control, and information management [1, 2].

This thesis targets at the problem of detecting and recognizing United States license plates from images and videos in real time. This will help us identify and register vehicles and provide the reference for further vehicle tracking and activity analysis. Our license plate detection approach has two major steps. First, we need to extract certain features which encode the images or frames from videos. Second, we need develop a detector, which is a classifier in our case, to determine whether a certain region in the images or frames is license plate.

1.2 Challenges

In LPR, we need to deal with a large variety of license plates, especially in United States. Each state in US has its own license plate color, pattern and formats of numbers and characters. Moreover, every few years, each state will issue new license plate patterns. Fig. 1. 1 shows the license plates from 50 states in US. And in some states, the

government allows vehicle owners to have personalized license plates which will further increase the difficulty of license plate detection and recognition.



Fig. 1. 1 License plate samples in 50 states of USA.

Another challenge in LPR is that the image quality taking by camera in real time may be affected by severe weather conditions, poor lighting conditions, and low camera resolutions. The aperture time of the camera will cause the blurring effect of the moving vehicle.

The third challenging issue we need to address in LPR is the large variations in camera perspectives when the license plate image is captured.

1.3 Background and Related Work

In most cases, License Plate Detection is a necessary procedure before LPR. Methods to locate the license plate region in images or videos from previous literature

can be grouped into the following categories: Binary Image Processing, Gray-Level Processing, Color Processing and Classifiers [2]. Character segmentation is also a very important step before character recognition. The methods for character segmentation can be grouped into Binary Image Processing, Gray-Level Processing and Classifiers. To recognize the segmented characters, a number of algorithms using the pattern/template matching or learning based classification have been developed [3-5].

1.3.1 License Plate Detection

A. Binary Image Processing

To extract license plate regions from background images, techniques based on combinations of edge statistics and morphology can achieve good results. In [6], they applied edge operators on a gray image after smoothing and normalization to extract horizontal and vertical edge maps. Statistical analysis of edges was then performed to detect the rectangle of license plate. The procedure was performed in a hierarchical manner at different scales. Several license plate regions were left after the rule-based fusion. The final decision was made based on the connected component analysis (CCA). They claimed that their algorithm can achieve 99.6% detection rate from 9825 images. Many other license plate detection algorithms [7, 8] also follow similar procedures. However, such methods are typically based on a hypothesis that the edges of the license plate frames are clear and horizontal. If the license plate frames were not clear or they had some affine transformation, these algorithms may not produce reliable results.

B. Gray-Level Processing

The large contrast between the characters and the background is exploited in [9] to detect license plates with black characters over white backgrounds. While some other algorithms assumed that the density of edges in the license plate region is larger than other regions if the contrast of the character and the license plate is sufficiently large. For example, in [10, 11] they scanned the vehicle images with N-row distance to count the existent edges. Regions with high edge density will likely have the license plate inside. Similarly, in [12], a block-based method was proposed, and blocks with high edge magnitude and variance are considered as the license plate region.

Image transformation methods based on Hough transform, Gabor filters and wavelet transform have been applied in license plate detection. Hough transform is a classic algorithm to detect straight lines. Since the shape of license plate can be defined by lines, [13] used the Hough transform to detect the boundary of a license plate. This method is practical only when the background of the image is simple. Another disadvantage of this method is that the computational complexity of Hough transform is very high. Gabor filters are often used to analyze textures as they are sensitive to textures with different directions and scales. In 2003, F. Kahraman [14] *et al.* applied Gabor filters to detect license plate and tested the algorithm with images acquired in a fixed angle and achieved a very good performance.

Vector quantization (VQ) has also been used as a feature to encode images for license plate detection [15]. In this method, vehicle images was divided into blocks and coded into strips. If a certain block contains high contrast region or details, it will split into four sub-blocks. In this case, the area of license plates with high contrast and

complex texture will be represented by small blocks. By scanning the structure and the mean value of the blocks, the license plate region can be easily located.

C. Color Processing

In many countries or regions, the format of license plates is strictly enforced. The color of the text and background is fixed, so that many algorithms use color information to detect license plates [16, 17]. However, if the lighting conditions change, the color of the license plates will vary. So the license plate detection algorithms that only rely on the color information may not achieve high detection rates. And for the countries like United States, since there are a large variety of license plates, such methods cannot be applied.

D. Classifiers

Recently, learning based license plate detection methods using different classifiers become very popular. The basic idea is to use a classifier to group the features extracted from the vehicle images into positive class (license plate region) or negative class (non-license plate region). A number of computational intelligence architectures, such as artificial neural networks (ANNs), genetic programming (GP), and genetic algorithms (GAs), were implemented for license plates detection. However, such algorithms usually need many predefined parameters. And if the parameters were not tuned properly, they may not produce satisfied results. Recently, adaptive boosting (AdaBoost) and support vector machine (SVM) have been widely used for license plate detection as they do not need a large number of parameters to obtain a decent classification performance.

The discrete-time cellular neural networks (DTCNNs) was applied for license plate detection in [18]. They extract two features of "grayness" and "texture" from the vehicle image, and used the DTCNNs to identify the pixels in the images with gray value in certain range and certain type of histogram after applying the Sobel operator. Their system identified more than 85% license plates. The pulse-coupled neural network (PCNN) was a novel neural network algorithm and was widely used in signal and image processing fields. PCNN was applied to segment license plate candidates from vehicle images, before the Fourier transform and a statistic process to locate license plate area [19]. This method achieved a detection rate of 85%.

The time-delay neural network (TDNN) was implemented by Kim *et. al* [20] for license plate detection and achieved remarkable result. A TDNN is a multilayer feed-forward network whose hidden neurons and output neurons are replicated across time. They used two TDNNs as horizontal and vertical filters to analysis the color and texture information of the vehicle images. Their system achieved 97% accuracy in detection as well as a high speed.

Another impressive license plate detection method was implemented based on the convolutional neural network (CNN) [21]. CNN have been widely used for optical character recognition (OCR) purpose. In this article, they used the convolutional sub-sampling to extract feature map, and a hierarchical approach to search the text candidate in license plate. This method achieved a detection rate of 98%.

AdaBoost was successfully used with Haar-like features in a "cascade" for face detection [22]. Using the cascade framework, the background region can be excluded to a great extent from further training. It was capable of processing images very fast with high

detection rates. The idea of AdaBoost algorithm is to combine a collection of weak classifiers to form a stronger classifier. It was demonstrated that the training error of the strong classifier approaches zero exponentially with the number of iterations [23]. In the above method was applied for license plate detection, and a detection rate of 93.6% was achieved. As they used the Haar-like features, their result was invariant to color, light, size and position of the license plates. So this algorithm can be applied with complex background. Xiangrong *et al.* extracted three set of informative features for text and used AdaBoost to detect text from natural scenes [25]. Their algorithm can detect 97.2% of the visible text from their test set, many of which are blurred.

SVM has also been widely applied for object detection recently [26, 27]. SVM is a pattern classification algorithm which minimizes an upper bound on the generalization error, while other classifiers are trying to minimize the training error. And it was tested that SVM can work well even in high dimensional space. Kim *et al.* adopted SVM to classify the color texture features followed by a continuously adaptive mean shift (CAMShift) algorithm in order to detect license plate region [28]. The detection rate of their system is 92.7% with a miss rate of 3.7%.

1.3.2 Character Segmentation

As some LPR algorithms require the single character input, after license plate detection, the preprocessing to segment the whole license plate into patches containing single characters is often needed. Any error made during this process will also affect the final LPR result.

A. Binary Image Processing

The most common and simplest way for character segmentation is to perform horizontal and vertical projections of the binary license plate image [11, 20, 29]. The idea is to calculate the sum of the binary license plate region along horizontal and vertical direction, and generate two vectors. The positions where the minimum values locate in the vectors are the place to segment the characters. The CCA can also be used for character segmentation. In [24, 30], the connected component was marked as a potential character based on the rule concerning the minimal area, width and height. And in some cases, CCA was used in conjunction with mathematical morphology or other methods for character segmentation [29].

The mathematical morphology method was applied for character segmentation [29] and can deal with seriously degraded images adaptively. The morphological thinning, thickening and pruning algorithms were applied to the binary license plate image. To search the natural segmentation points, the horizontal and vertical histogram projection was done followed by a merging operation.

Contour tracking and modeling was also used to do character segmentation. A contour tracking method known as "bug following" was implemented to segment characters [31]. In [32], a shape-driven active contour model was proposed to solve the plate character segmentation problem. The variational marching algorithm was used combined with a gradient-and-curvature-dependent speed function to segment the exact boundaries of the characters. Shape similarity statistics were used as criteria to stop the iteration when the front resembles the training characters.

B. Classifiers

Markov random field (MRF) and Hidden Markov Chains were also employed in character segmentation in images or videos. Franc and Hlavac used the Hidden Markov Chains to model a stochastic relation between an input image and corresponding character segmentation [33]. Their method needed some prior knowledge, such as the number of characters. And the characters should be segmented with equal width.

1.3.3 License Plate Recognition

A. Classifiers

Various multilayered neural networks have been used for license plate recognition. In [30], they authors used a discrete-time cellular neural networks (DTCNN's) to extract four different features (horizontal projection, vertical projection, horizontal connected component count and vertical connected component count) and an ordinary multi-layer perception network (MLP) to do the classification. A 98.5% recognition rate was reported using this method.

Chang *et al.* proposed a LPR method using self-organizing neural networks which was able to handle noisy, deformed, broken or incomplete characters in license plates [34]. The topological features of the input characters were first calculated and compared with the pre-stored character templates, which will be performed by the self-organizing character recognition procedure. An impressive 95.6% recognition rate was achieved over a large data set.

Probabilistic neural networks (PNN) were widely applied in LPR. As these types of networks can be designed and trained fast. A remarkable recognition rate of 99.5% was published in 2005 [35].

SVM-based LPR algorithms have been very popular recent years. Kim *et al.* used a SVM classifier to do LPR for Korean license plates and reported an average character recognition rate of 97.2% [20]. In [36], Arth *et al.* extracted Haar-like features from license plates in video frames, and processed LPR with a SVM classifier. They also compared the classification results using with OneVsAll and tree-like structures of different testing sets.

B. Pattern/Template Matching

Template matching technique was successfully implemented for LPR in [37]. They calculated the distance of the patch from plate image and the template, and used classifier to find the minimum distance to make a decision. The template matching method was often combined with other methods to do LPR [11].

1.4 Our Previous Work

Before we select HOG feature for license plate detection and recognition, we tried Local Binary Patterns (LBP) first to do letter detection to see if LBP feature can produce good results. LBP feature is a non-parametric kernel which summarizes the local special structure of an image and has become popular for texture classification [38-42]. The LBP feature was first introduced by Ojala *et al.* [43] and they showed the high discriminative power of this operator for texture classification. Besides, it is invariant to monotonic

gray-scale transformations, and less sensitive to changes in illumination. These advantages are the reasons why we want to test if it is suitable to do classification for letters or license plates.

Here we introduce the original LBP operator as well as multi-scale LBP and the uniform LBP. The original LBP operator [43] is a non-parametric 3×3 kernel as shown in Fig. 1. 2. At a given pixel position (x_c, y_c) in the image, LBP is defined as an ordered set of binary comparisons of pixel intensities between the center pixel and its eight surrounding pixels. For example, if the center pixel's value is greater than the neighbor, write "1", otherwise, write "0". This will generate an 8-digit binary number along a circle, i. e. clockwise or counter-clockwise. Normally, we will convert this binary number into decimal number for computational convenience.

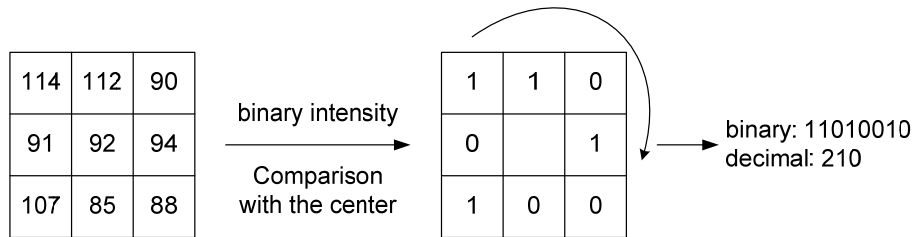


Fig. 1. 2 An illustration of the original LBP calculation.

Later, the Multi-Scale LBP, an extension of the original LBP was introduced by Ojala *et al.* [44]. This Multi-Scale LBP operator will calculate the binary comparisons of pixel intensities to a circular neighborhood of different radius size. They used $LBP_{P,R}$ to denote the LBP with P equally spaced pixel on a circle of radius R . The $LBP_{8,1}$ and $LBP_{8,2}$ are illustrated in Fig. 1. 3.

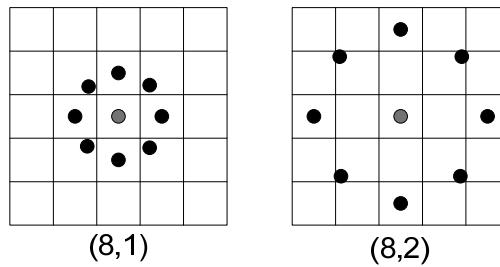


Fig. 1.3 Examples of Multi-Scale LBP.

They also noticed that the primitive micro-features such as lines, edges, corners were contained in a small subset of LBP patterns. These patterns are called uniform patterns which contain at most two bitwise 0 to 1 or 1 to 0 transitions (circular binary code). For instance, 00000000, 11110011 or 01111111 are for instance uniform patterns. $LBP_{P,R}^{u2}$ denotes the extended LBP operator (u2 for only uniform patterns, labeling all remaining patterns with a single label).

A normalization of the LBP feature will be processed before implementation. In our test, we used $LBP_{8,1}^{u2}$ on 40×80 patches with a feature dimension of 59. 303 patches of digits or letters extracted from license plate images as positive class and 30300 patches of non-digit-or-letter region as negative class were used for training. And 113 patches of digits or letters extracted from license plate images as positive class and 11300 patches of non-digit-or-letter region as negative class were used for testing. A SVM light classifier was used for classification. A cost-factor of 100 was used to deal with the imbalance of the positive and negative class. The confusion matrix for the testing data is

$$\begin{bmatrix} 10629 & 671 \\ 54 & 59 \end{bmatrix}.$$

From the confusion matrix, we can see that the classification result is far from satisfactory. We have also tried other LBP features with different P, R value, and did not

get very good result either. We think only use LBP feature will not solve our problem perfectly. We need to find another feature which is suitable for our license plate detection and recognition tasks.

1.5 Overview of Our Method

In this project, we develop a hierarchical framework for LPR. Based on a coarse-to-fine sliding window being scanned over the video frames, HOG features will be extracted and sent to the SVM classifier. License plates from all 50 states in US will be used in our training stage as positive classes in SVM. The SVM model will be bootstrapped for better performance [45]. License plate regions in video frames will be detected with a SVM classifier. A MeanShift algorithm will be applied to merge the detection regions into one final detection box. A comparison of the performance of HOG descriptor with different cell and block sizes will be provided. Our license plate detection results will demonstrate that our method is robust with noise and motion blur.

CHAPTER 2

LICENSE PLATE DETECTION USING HOG FEATURES

As discussed in section 1.4, our license plate detection will follow the procedure shown in Fig. 2. 1. In this chapter, the details of each procedure will be discussed.

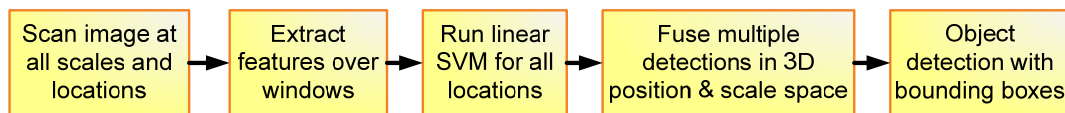


Fig. 2. 1 An overview of license plate detection procedure.

2.1 Scanning window

To search the location of license plates in an image, a scanning window needs to be defined and slide over the image at different scales and locations as shown in Fig. 2. 1. In each location, the HOG feature will be extracted from the window and sent to the SVM classifier which will decide if the region is a license plate or not. The regions with large positive response from SVM classifier will be collected as potential detection, and fused with non-maximum suppression algorithms into one final detection result. The step size of the sliding window is related to the algorithm efficiency and detection precision. Different step sizes of the sliding window will be evaluated during our experiments.

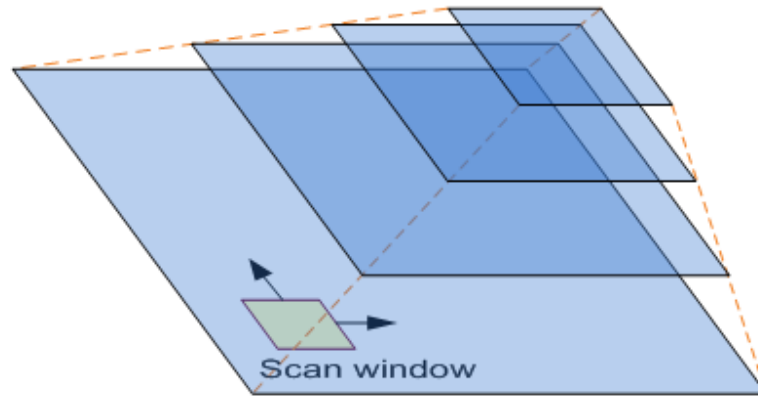


Fig. 2. 2 Scan-window and image pyramid.

2.2 HOG Features

License plate detection is a challenging task as described in Section 1.2. A robust feature set should allow the license plates to be discriminated cleanly even in cluttered backgrounds with poor lighting conditions. It has been shown that the locally normalized Histogram of Oriented Gradient (HOG) descriptors provides excellent performance in object detection over other existing feature sets including Haar wavelets [46, 47]. As mentioned in [48], HOG descriptors are reminiscent of edge orientation histograms, SIFT descriptors and shape contexts, but they are computed on a dense grid of uniformly spaced cells and they use overlapping local contrast normalizations for improved performance. The HOG descriptors have many advantages in license plate detection. It extracts the edge or gradient information which is very characteristic of local shapes, and performs with a controllable degree of invariance to local geometric and photometric transformations. Variants of HOG descriptors have been proposed such as Rectangular HOG (R-HOG), Circular HOG (C-HOG), Bar HOG, Centre-Surround HOG. The sketch

maps of R-HOG, C-HOG and Centre-Surround HOG are shown in Fig. 2. 3. As R-HOG is most conveniently to be implemented among all these descriptors, only R-HOG will be adopted and discussed in this thesis.

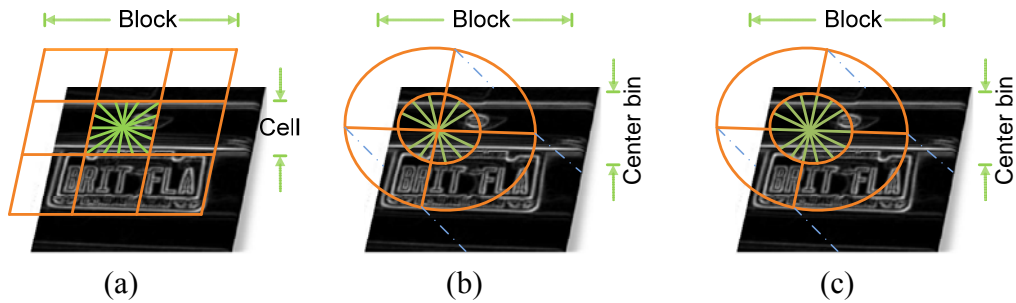


Fig. 2. 3 Variants of HOG descriptors. (a) A rectangular HOG (R-HOG) descriptor with 3×3 blocks of cells. (b) Circular HOG (C-HOG) descriptor with the central cell divided into angular sectors as in shape contexts. (c) A C-HOG descriptor with a single central cell[49].

2.2.1 Feature Extraction Procedure (for R-HOG)

Fig. 2. 4 shows the major steps to extract R-HOG features. First, the window will be divided into small regions, called cells. And for each cell, we compute a local 1D histogram of gradient directions over the pixels of the cell. The combined histogram will form the descriptor. The contrast-normalization will also be performed in order to make the descriptor invariant to illumination and shadows. This can be done by accumulating the local histogram over larger spatial regions ("blocks").

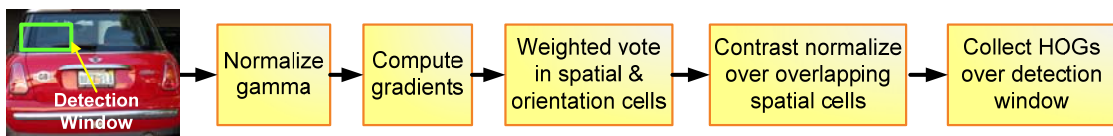


Fig. 2. 4 An overview of HOG feature extraction.

The image preprocessing of color normalization or gamma correction is often required for many other feature descriptors. However, Dalal and Triggs pointed out that this step can be omitted in HOG descriptors, as the following descriptor normalization

will achieve the same result. The gradient map can be computed by simply applying the 1D centered point discrete derivative mask in both horizontal and vertical directions. The filter masks are $[-1, 0, 1]$ and $[-1, 0, 1]^T$ for horizontal and vertical respectively, and according to [48], the simple mask works best. As the input image has three color channels (R, G, B), the gradients of each channel will be calculated separately and the one with largest norm will be used as the pixel's gradient vector.

The next step involves creating cell histograms. The structure of R-HOG is shown in Fig. 2. 3 (a). For R-HOG, the cells are square or rectangular with a certain number of orientation bins evenly distributed over $0^\circ\sim 180^\circ$ ("unsigned" gradient) or $0^\circ\sim 360^\circ$ ("signed" gradient). Each pixel in the cell is calculated a weighted vote for the orientation histogram according to its gradient, and the votes are accumulated into the orientation bins. During the voting, the bilinear interpolation is used between the neighboring bin centers in both orientation and position. The vote can be the gradient magnitude itself or a function of the gradient magnitude.

To achieve good performance, effective local contrast normalization over overlapped cells is needed to make the descriptor invariant of illumination change and foreground-background contrast. The basic idea is to group cells into larger spatial blocks and normalize the contrast of each block separately. The blocks are usually overlapped as is shown in Fig. 2. 5, so that the entries of each cell will response to several component of blocks in the final feature. Although the overlap of blocks seems a lot of redundant, experiments in [48] showed that the overlap of the blocks significantly improved performance.

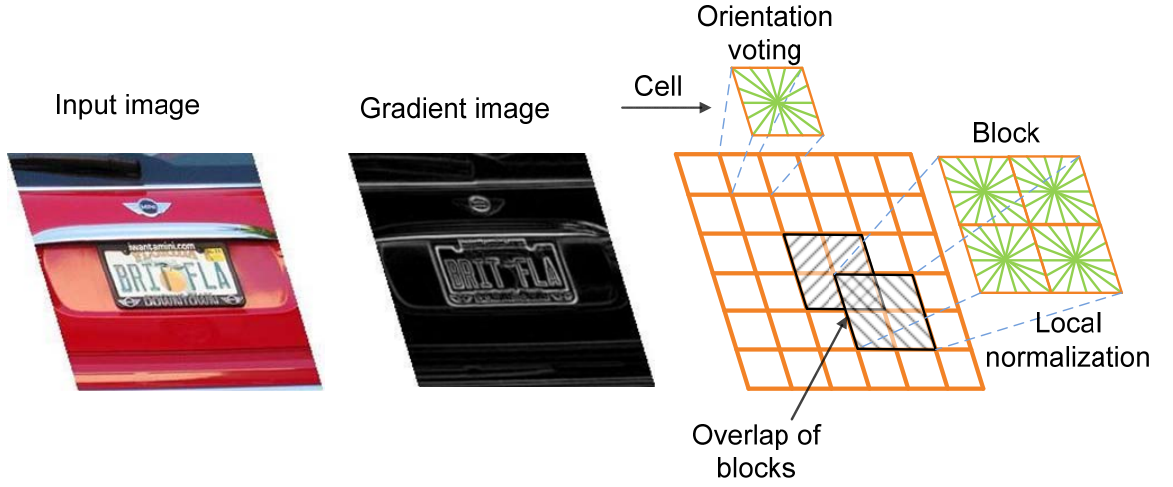


Fig. 2. 5 A sketch of local normalization.

Different block normalization schemes can be applied. Four normalization schemes were evaluated in [48].

- 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) and renormalizing;
- L1-norm: $\mathbf{v} \leftarrow \mathbf{v} / (\|\mathbf{v}\|_1 + \varepsilon)$;
- L1-sqrt: $\mathbf{v} \leftarrow \sqrt{\mathbf{v} / (\|\mathbf{v}\|_1 + \varepsilon)}$;

Where \mathbf{v} is the normalized descriptor vector, $\|\mathbf{v}\|_k$ is its k-norm for $k = 1, 2$, and ε is a small normalization constant to avoid division by zero. It was shown in [48] that L2-Hys, L2-norm and L1-sqrt normalization achieved equally good performance while L1-norm degrades the performance by 5% when compared with the other three.

2.2.2 Implementation

During our experiments, we extract the R-HOG features from the input image for its R, G, B color channels with no gamma color correction. $[-1, 0, 1]$ 1D gradient filter was applied to obtain the gradient map. In US, the license plates are made with a fixed size of 6 inches in height and 12 inches in width. So the height to width ratio of the feature extraction region is $1/2$. The detection windows have the size of 44×80 including 4 pixels of margin on all four sides. The gradient of each pixel is voted into 9 bins evenly placed over $0^\circ \sim 180^\circ$. The block size is 12×12 , and the cell size is 4×4 . L2-norm is applied as the block normalization scheme with a block spacing stride of 6 pixels. The R-HOG feature length with above parameters is 4455.

2.3 Support Vector Machine

Support Vector Machines (SVMs) are kernel-based learning systems that use N-dimensional hyper-plane that optimally separate data into two categories where the maximal margin can be obtained. It was developed by Vapnik and co-workers at AT&T Bell Laboratories. In our application, we used a dense version of SVMLight [50] which reduced memory usage for problems with large dense descriptors [49]. Linear kernels are used in SVM.

2.4 Non-maximum Suppression

After applying the SVM classifier to the HOG features extracted from the sliding windows over all scales, a number of bounding boxes may be derived by a threshold to

the prediction score. A non-maximum suppression algorithm is needed to fuse these potential detections into one final result. We used the uniform kernel Mean shift algorithms [51] to fuse the overlapped detection over all scales. All the detections can be presented in 3-D space as $\mathbf{y}_i = [x_i, y_i, s_i]$, where x_i and y_i present the position for the i -th detection, and $s_i = \log(\text{scale}_i)$. Here $K(\mathbf{y})$ is the kernel function and \mathbf{H} is a symmetric positive definite 3×3 bandwidth matrix and $\text{diag}[\mathbf{H}] = [(\exp(s_i)\sigma_x)^2, (\exp(s_i)\sigma_y)^2, (\sigma_s)^2]$ where σ_x , σ_y and σ_s are the window bandwidth. The Kernel Density Estimate (KDE) of the data is then taken to be

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

where

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

The profile of the uniform kernel is

$$k_U(y) = \begin{cases} 1 & 0 \leq y \leq \sigma \\ 0 & y > \sigma \end{cases} \quad (2.3)$$

and the kernel function is

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

where c_d is the volume of the unit 3D sphere.

The detection confidence score generated by SVM classifier is denoted as ω_i . As the weights for KDE should be greater than zero, we applied a hard clipping function $t(\omega)$ to avoid negative scores.

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

Thus, (2.1) can be written into following forms:

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

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

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

Dividing (2.8) by (2.6), we have

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

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 satisfies $\sum_{i=1}^n \bar{\omega}_i = 1$. So, 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 defines as

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

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

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

Starting from some \mathbf{y}_i until \mathbf{y}_m does not change anymore.

Those detections that finally shifted into a region with radius R will be fused into the same mode. By carefully selecting $\sigma_x, \sigma_y, \sigma_s$ and R , this mean shift algorithm can achieve desirable results for both single license plate and multiple license plates. The detections with the largest score in each mode will be selected as the final detection result.

2.5 Algorithm Speedup

Since searching the entire image for license plates is time consuming, it may be difficult to apply it in real-time. We developed a scheme to accelerate the scanning process by detecting the car head and rear parts using HOG features, as we found that almost all license plates are mounted in the front or rear of the cars. We need to train a car head and rear detector using the same method as we trained the license plate detector using SVM with HOG. We collected 6632 car head and rear patches as positive class, and 27682 negative patches. All the patches are normalized to 80×44 . The block size was 12×12 , and the cell size was 4×4 . L2-norm was applied as the block normalization scheme with a block spacing stride of 6 pixels. We can tune the detection threshold high enough to make sure the false alarm rate close to 0. For fast license plate detection, we run the car head and rear detection first. If the car head or rear is detected, we just need to search for license plates using the HOG descriptor around the car head or rear region as is shown in Fig. 2.6. This method will accelerate the detection speed several times depending on the area ratio of car head/rear to the whole image. In our experiment, it can help to complete the license plate detection for one image within 1 second.



Fig. 2. 6 Fast license plate detection image. The car rear detected is shown in yellow box. License plates are searched within the car rear region. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.

2.6 Results and Discussion

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

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

We selected 3110 positive license plate images from 50 states of US as positive training examples, together with their rotations of -12° , -6° , 6° , 12° (15550 images in all). A fixed set of 15550 patches sampled randomly from 600 training photos (containing license plates from 50 states of US mounted on various kinds of vehicles downloaded from flickr[®]) provided the initial negative set. Those photos contain vehicles, but the license plate regions were avoided when sampling. A preliminary detector was trained using SVM. 250 license plate patches and 100000 non-license plate patches from 250 testing photos (containing license plates from 50 states of US mounted on various kinds of vehicles downloaded from flickr[®]) were used to plot the DET curve. To gain better

detecting result, boot strapping is needed. 600 training photos and another 500 photos from Medialab LPR Database [2] were searched in different scales with the preliminary detector for false positives and false negatives ('hard examples'). All the training and testing photos were normalized to 500 pixels in width. The hard examples will be added to the training set for the next round of boot strapping. 4 round of boot strapping process was performed. The retraining process significantly improved the performance of the detector. Fig. 2. 7 shows the performance of the HOG detector with 4×4 cell size, 12×12 block size and 6 pixels block stride before (black dot) and after (red triangle) 4 round of boot strapping process. We can see that the retraining process significantly improves the performance of the detector by 30% at 10^{-4} False Positives Per Window (FPPW). However, additional rounds of retraining result in little improvement, so we did not use them.

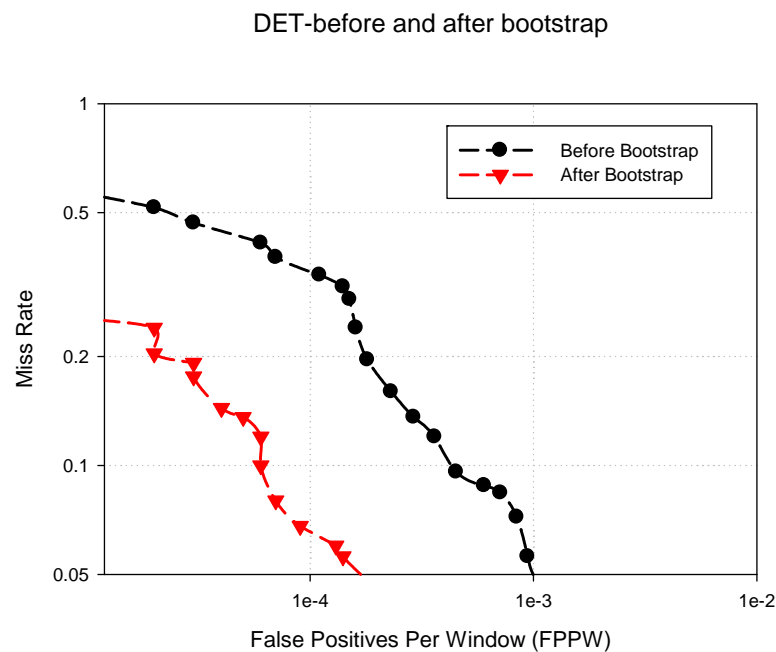


Fig. 2. 7 The performance of the detector with cell size = 4×4 , block size = 12×12 , block stride=6 before (black dot) and after (red triangle) 4 round of boot strapping process.

2.6.1 Descriptor Blocks

The local contrast normalization plays an important role for the good performance. The L2 normalization was applied due to the comparison of four different normalization methods in [48]. To evaluate how different normalization schemes would affect the final performance of the HOG descriptor in license plate detection, we basically tested the performance of the descriptor with different size of cells grouped in different size of blocks on the boot strapped dataset mentioned above. The overlap of the blocks was fixed to 50% which means the stride of the block was half the block height or width. As shown in Fig. 2. 8, we tested the descriptor with blocks that containing 1×1 , 2×2 or 3×3 cells with the cell size of 4×4 , 6×6 , 8×8 or 12×12 . We can see that the normalization scheme with cell size of 4×4 , and 3×3 cells in a block performed best among all the schemes we tested with a miss rate of 6.4% at 10^{-4} FPPW. From the figure, we can see that choosing an appropriate normalization scheme is critical for the good performance of the HOG descriptor.

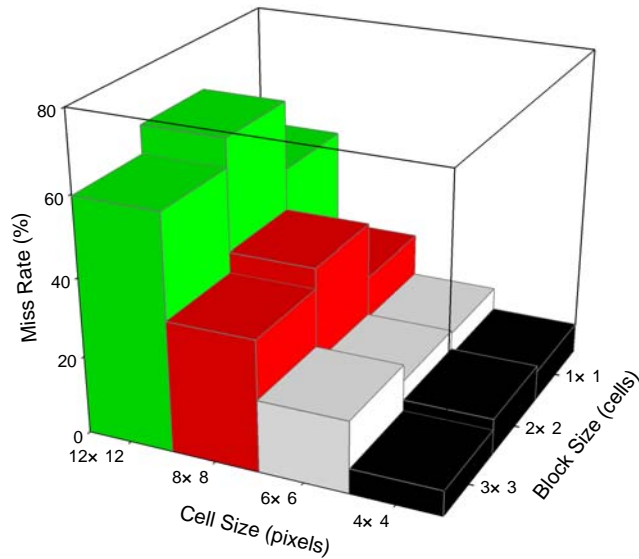


Fig. 2. 8 The miss rate at 10-4 FPPW as the cell and block sizes change. The block stride (block overlap) is fixed at half of the block size.

2.6.2 Result of Testing Images

We tested our algorithm by applying it to our 250 testing images described above. Each testing images were scanned in nine different scales which are 0.6400, 0.8000, 1.0000, 1.2500, 1.5625, 1.9531, 2.4414, 3.0518 and 3.8147 of its original size. The car head and rear detection for accelerating was not involved for this test. The step size of the scanning window sliding horizontally and vertically was fixed to 6 pixels. The HOG feature with cell size of 4×4 and block size of 12×12 was extracted from each 44×80 scanning window including 4 pixels of margin on all four sides. The well trained and bootstrapped SVM model with for corresponding cell and block size in 2.6.1 was The threshold for SVM prediction score were carefully selected when the FTTP was 0.0125. All the potential detections were merged to the red boxes in the figures after the

3D Mean shift algorithm. The green boxes were the final detections. A set of ground truth boxes were predefined. The detection would be claimed as true positive if the region of ground truth rectangular R and final detection rectangular r meet the following condition:

$$\frac{R \cap r}{R \cup r} \geq 0.45;$$
 otherwise, it would be claimed as false alarm. The window size for the 3D

Mean shift algorithm was tuned to converge for each image. For the reason that the images in our dataset contains license plates in different scales, patterns, lighting conditions, angles, and complex background, the license plates in 217 images were correctly detected among all 250 testing images. Such detection rate is acceptable for our project, because in practice we may not need such wide range of searching sales, and most of times, the backgrounds can be easily removed before detection which will greatly enhance the detection accuracy. Here we demonstrate some license plate detection results in different lighting conditions, angles, and complex background. Also we will show some false alarm images and discuss how we can enhance the preference of the detector in the future.

Fig. 2. 9 shows license plates in images with different lighting conditions correctly detected using HOG descriptor. Fig. 2. 9 (d) and (e) are acquired with adequate light, (a) and (f) are acquired in the night, and other images in Fig. 2. 9 are acquired with limited light. We can see that the license plates in these images were precisely detected even they are in a variety of lighting conditions. This means that the HOG license plate detector is tolerant of different lighting conditions and can be applied with images captured during daytime and nights.

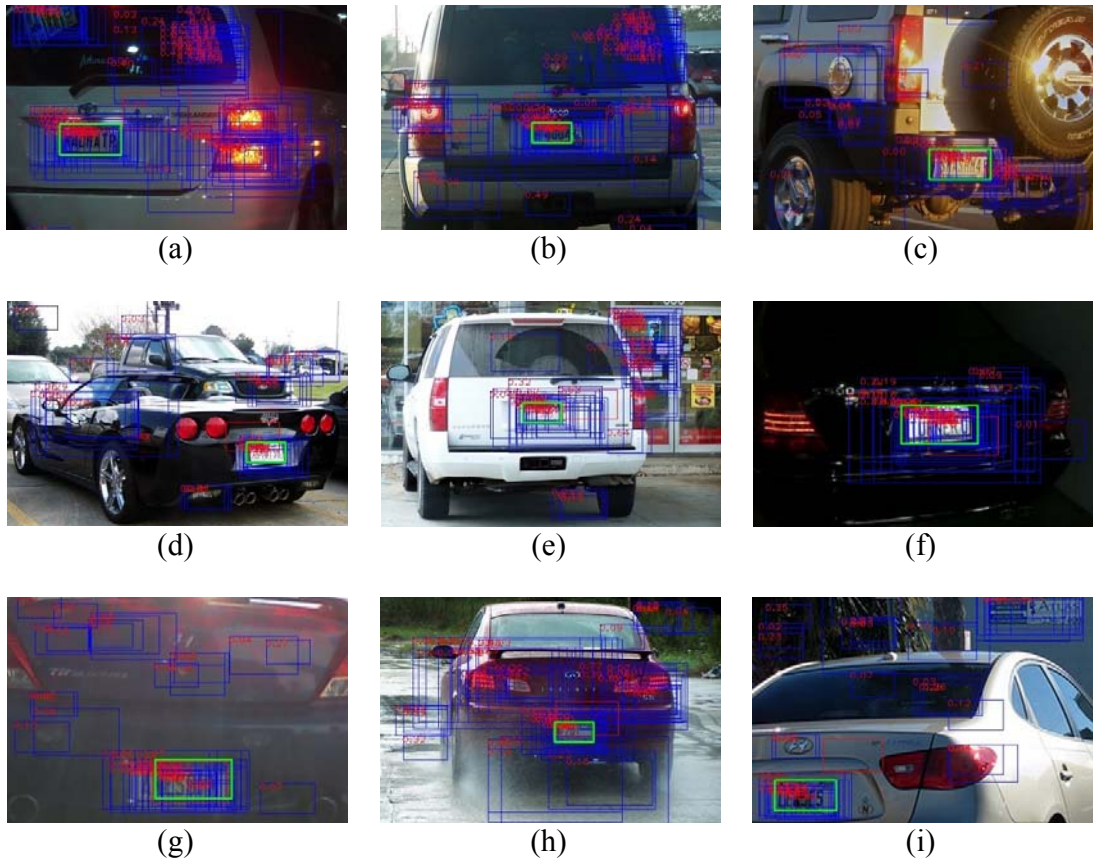


Fig. 2. 9 License plates detected with different lighting conditions using HOG descriptor. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.

Fig. 2. 10 shows license plates in images acquired with pose variations. We can take these license plates as the normal rectangular license plates with affine transformations (mainly rotation and shear). Our algorithm can detect the license plate location correctly even the license plates are rotated up to 20° (Fig. 2. 10 (b) (e) (g) (h) and (i)) or shear up to 25% of the plate height (Fig. 2. 10 (a) (b) (c) (d) (f) (g) and (i)). It shows that our HOG license plate detector exhibited a certain degree of tolerance with pose variations.

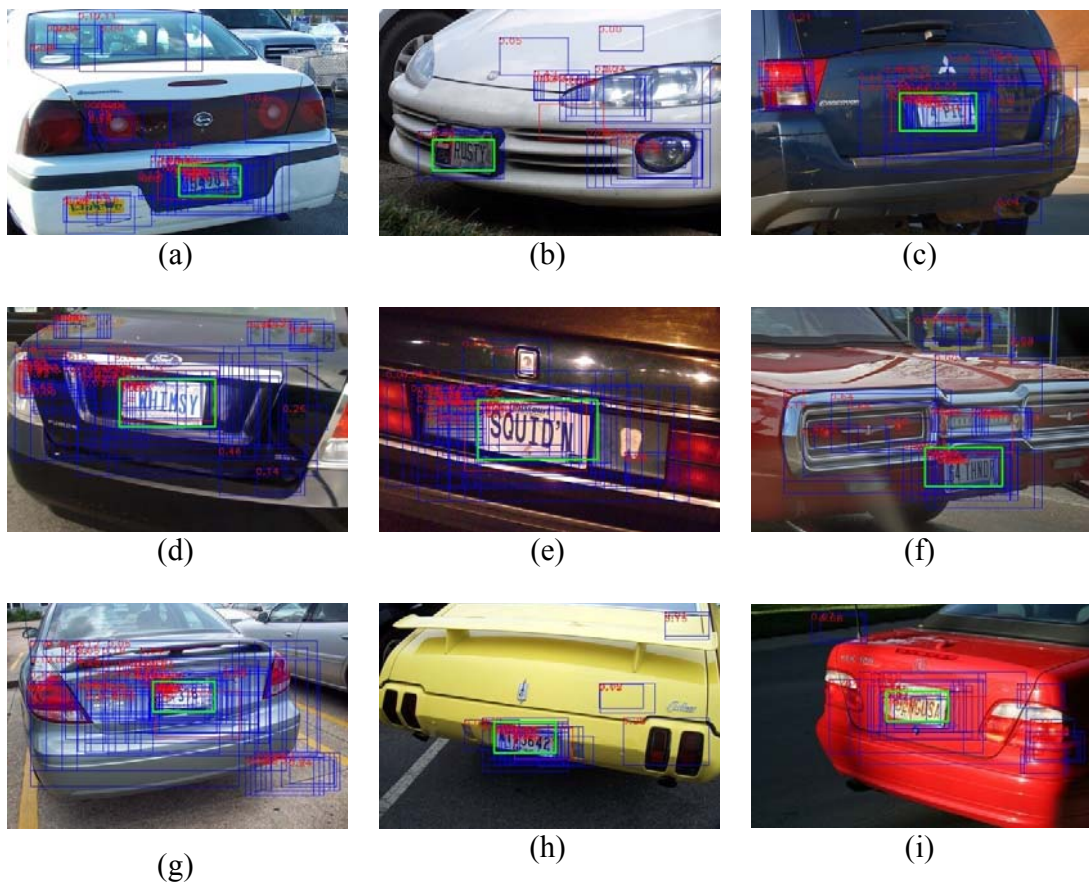


Fig. 2. 10 license plates with pose variations detected using HOG descriptor. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.

Fig. 2. 11 shows license plates in images with complex background. For example, some stickers on the car or in the background lie in Fig. 2. 11(a) -(i). Those stickers often have digit or letters and ambiguity patterns with license plates in them. There are digits and numbers in similar fonts with license plates in Fig. 2. 11(d) and (j). Fig. 2. 11(f) and (l) contain backgrounds with high complexity. And Fig. 2. 11(k) even has an advertisement plate with exactly the same size with regular license plate in it. All those elements will create confusion to detect license plates even for human eyes. However, our license plate detection algorithm can correctly detect these plates in very challenging backgrounds. That implies that the HOG license plate detector is robust to high background complexity and ambiguity.

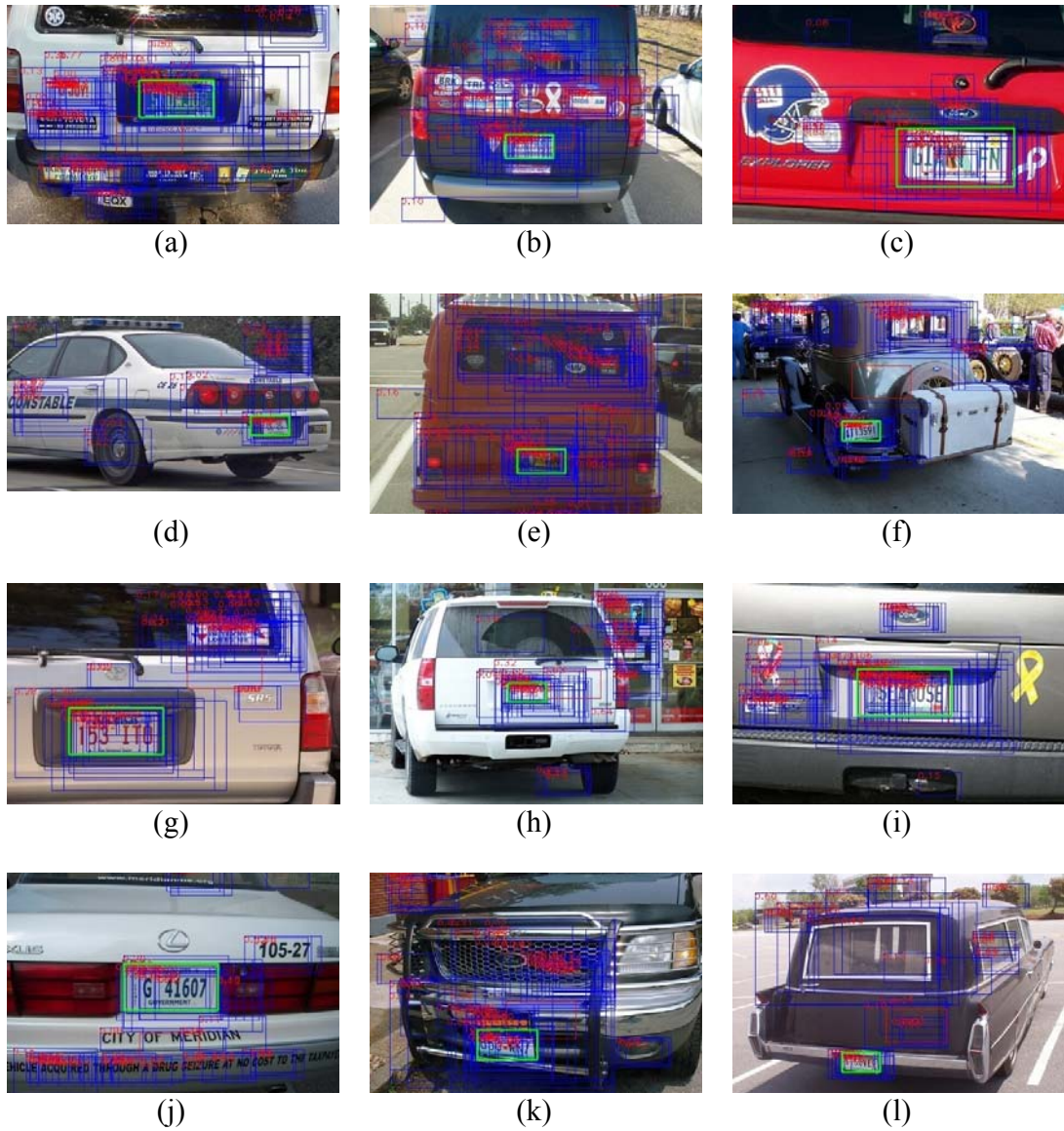


Fig. 2. 11 License plates detected with complex background using HOG descriptor. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.

Fig. 2. 12 shows some typical false alarm detections by our HOG license plate detector. Among all 250 testing images, only 33 of them were not detected correctly. In these false alarm images, 11 of them lie in license plate region but not the precise

location as is shown in Fig. 2. 12 (c), 5 of them lie in car light region as is shown in Fig. 2. 12 (a), 6 of them lie in the black car windows as is shown in Fig. 2. 12 (b), and the rest 11 false alarms lie in components of the car as is shown in Fig. 2. 12 (e) or other ambiguity patterns as are shown in Fig. 2. 12 (d) and (f). To further enhance the performance, we can examine the training data to eliminate the shifted positive patches. This will enhance the precision of the license plate detection. To avoid false alarms lying in car light or black window, we can introduce the color feature, and combine it to our HOG descriptor. This may help to reduce these false alarms as we found the color for car light and window is very distinctive from that of license plates.

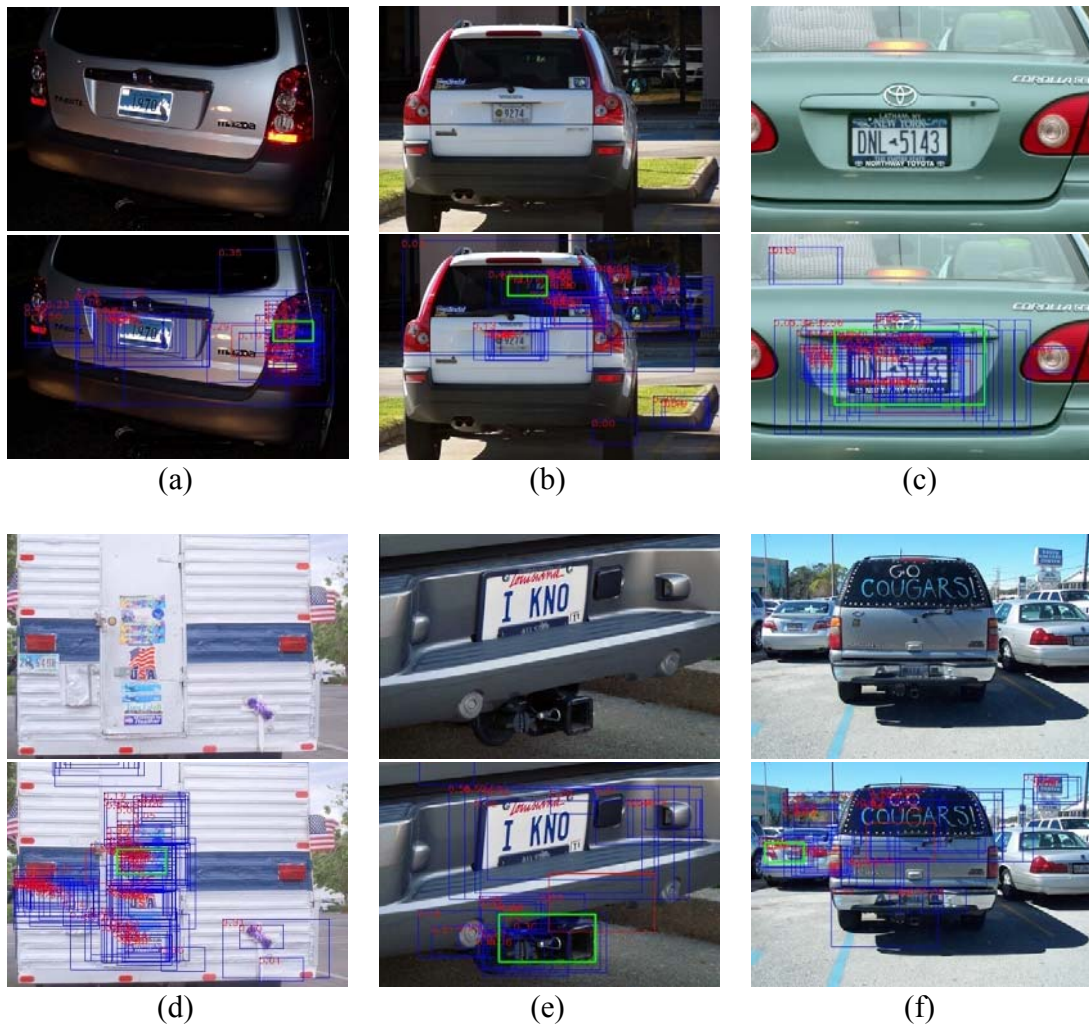


Fig. 2.12 False alarm images by HOG descriptor. Original images were shown on top of each detection images. The final license plate detection (shown in green box) is picked out among the potential detections (shown in blue boxes) that merged to the one location (shown in red) by Mean Shift algorithm. The prediction score for each potential prediction is printed in red.

To further analyze the HOG license plate algorithm, we used the car rear subset of Caltech datasets (1999). This dataset contained 126 images of cars from the rear. The scales of the images were approximately normalized, and the images were normalized to 330×500 . Each testing image was scanned in four different scales which are 1.1700, 1.3689, 1.6016 and 1.8739 of its original size. The car head and rear detection for accelerating was not involved for this test. The step size of the scanning window sliding horizontally and vertically was fixed to 6 pixels. The HOG feature with cell size of 4×4 and block size of 12×12 was extracted from each 44×80 scanning window including 4 pixels of margin on all four sides. The well trained and bootstrapped SVM model with for corresponding cell and block size in 2.6.1 was adopted. The threshold for SVM prediction score were carefully selected when the FTTP was 0.0125. The window size for the 3D Mean shift algorithm was tuned to converge for each image. From all 126 test images, only 5 of them were not detected correctly, which were shown in Fig. 2. 13. The blue boxes were the potential detections with the prediction scores above the threshold. All the potential detections were merged to the red boxes in the figures after the 3D Mean shift algorithm. The green boxes were the final detections. From all the false alarms shown in Fig. 2. 13, we can see that most of the false alarm regions lie in the area with complex texture, like trees, or bushes. The car lights also have a chance to be detected as false alarms. We are considering that combining color information with the HOG feature could probably help with this issue.

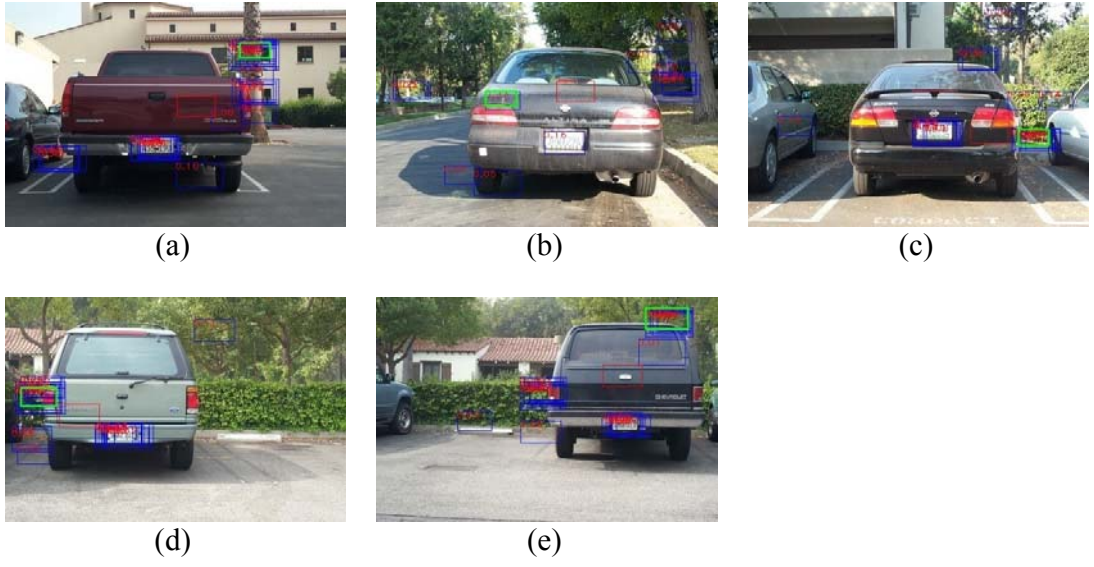


Fig. 2. 13 False alarm detections by HOG descriptor of Caltech datasets (1999).

2.6.3 Robustness with Motion Blur

Motion blur is very common during license plate detection from videos with moving vehicles. To analyze the robustness of our method to motion blur, we simulated the motion blur by applying a set of 'motion' filters to the Caltech datasets. We applied the 'motion' filters with len pixels from 0 to 20 with interval of 2 in horizontal and vertical directions respectively. As shown in Fig. 2. 14, we can see that with a 10-pixel len size motion blur (Fig. 2. 14(b) and Fig. 2. 14(e)), the characters and numbers on the license plate are difficult to recognize by human eyes; and with a 20-pixel len size motion blur (Fig. 2. 14(c) and Fig. 2. 14(f)), we can hardly see characters and numbers on the license plate.

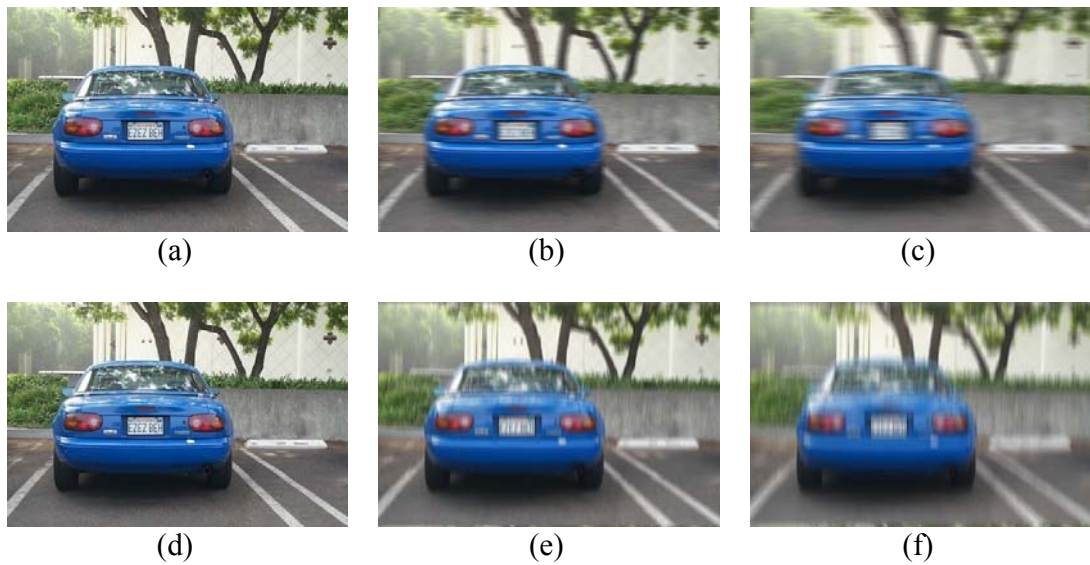
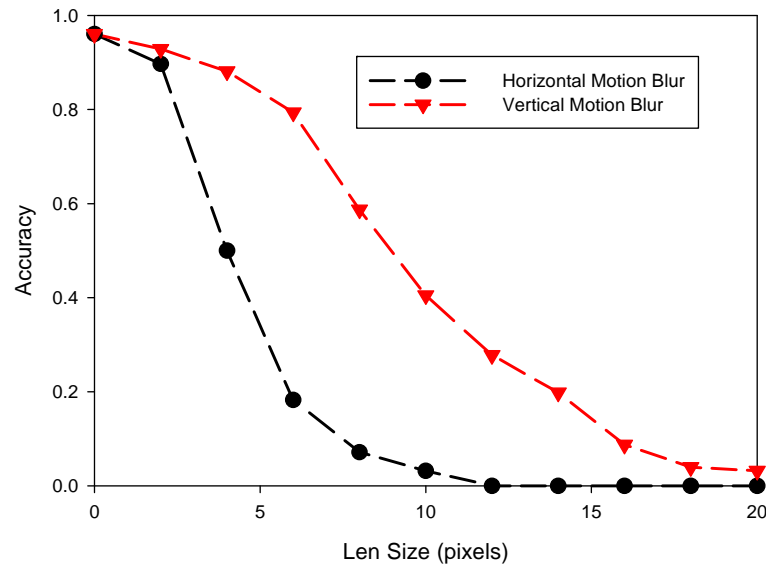


Fig. 2. 14 One image in Caltech dataset applied with motion filters with len pixels of (a) 0 pixels, (b) 10 pixels and (c) 20 pixels in horizontal direction, and (d) 0 pixels, (b) 10 pixels and (c) 20 pixels in vertical direction.

Detection accuracy vs motion blur



The curves in Fig. 2. 15 show the detection accuracy with different len pixels of 'motion' filter applied along horizontal (black curve) and vertical (red curve) directions. We can see from the chart that the detection accuracy drops significantly from 96.0% to 0 while the len size increasing from 0 to 12 pixels. And the detection accuracy remains at 0 if the len size is larger than 12 with the horizontal motion blur. However, for vertical motion blur, the detection accuracy drops slowly from 96.0% to 3.2% while the len size increasing from 0 to 20. And with the same len size, the detection accuracy is always larger with vertical motion blur than horizontal motion blur. These results indicate that for license plate detection, our method performs better with vertical motion blur than horizontal motion blur. And severe motion blur will greatly affect the detection accuracy. So motion blur should be controlled in a certain range or some motion compensation algorithms are needed when applying this method to detect license plate from video in the future.

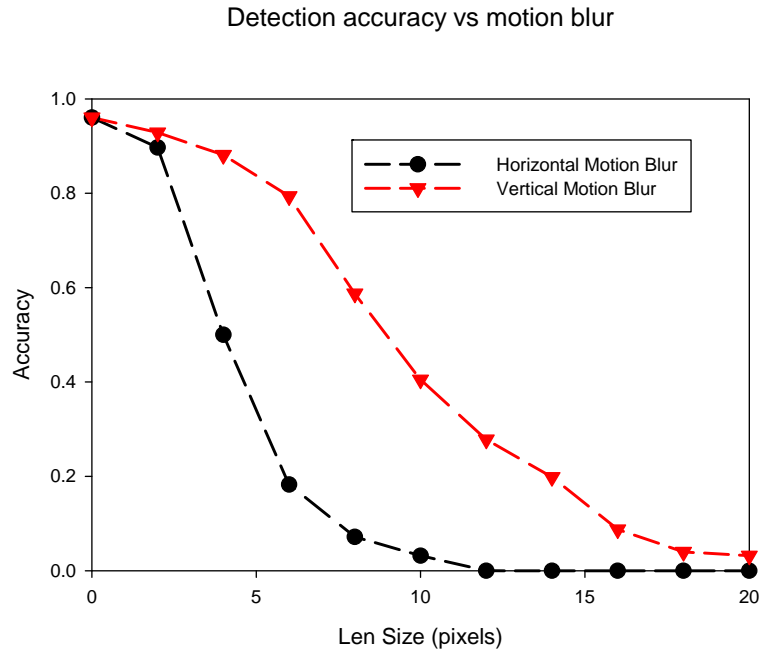


Fig. 2. 15 Detection accuracy of license plate detection using HOG feature with horizontal (black curve) and vertical (red curve) motion blur with motion len size from 0 to 20 pixels.

2.6.4 Tolerance of Noise

Images and videos in real world usually contain noises, especially for the application of license plate detection. Sometimes, the lighting conditions are restrained, for example, in dark night; and sometimes the image and video acquired may not be high quality, for the reason of device budget. These factors will introduce some noises to the images and videos. Here we tested how our license plate detection algorithm performs with different level of noises. We added Gaussian white noise with mean 0 and variance from 0 to 0.02 to the Y channel in YCbCr space of the Caltech data base to simulate the image noise in natural scene. The image with Gaussian white noise of mean 0 and variance 0 as is shown in Fig. 2. 16 (a) is the same with the original image. While we can see the one with Gaussian white noise of mean 0 and variance 0.01 shown in Fig. 2. 16

is very noisy, and the one shown in Fig. 2. 16 (c) with Gaussian white noise of variance 0.02 is severely noisy with large speckles all over the image.

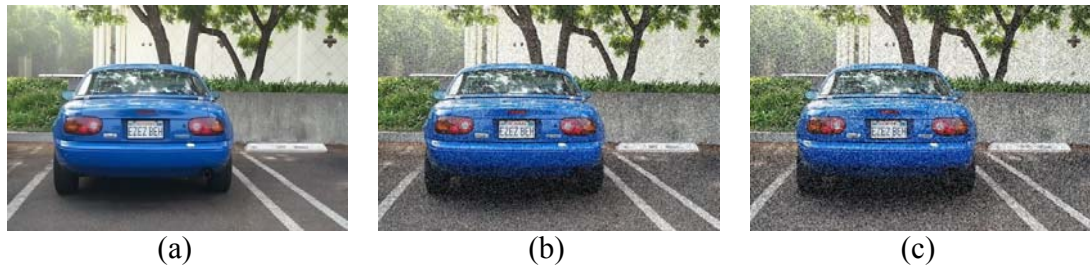


Fig. 2. 16 One image in Caltech dataset applied with Gaussian white noise of mean 0 and variance 0 (a), 0.01 (b) and 0.02 (c).

In Fig. 2. 17, we can see that with no added noise, the license plate detection accuracy can achieve 96.0%. With the variance of the Gaussian white noise increasing to 0.02, the detection accuracy gradually decreases to 73.0%. So we can see that the algorithm is tolerant to noises in the image in some extent. Even with severe noises of variance 0.02, over 73% license plates in images can still be correctly detected. This is one of the advantages that this algorithm can be applied for license plate detection in real world.

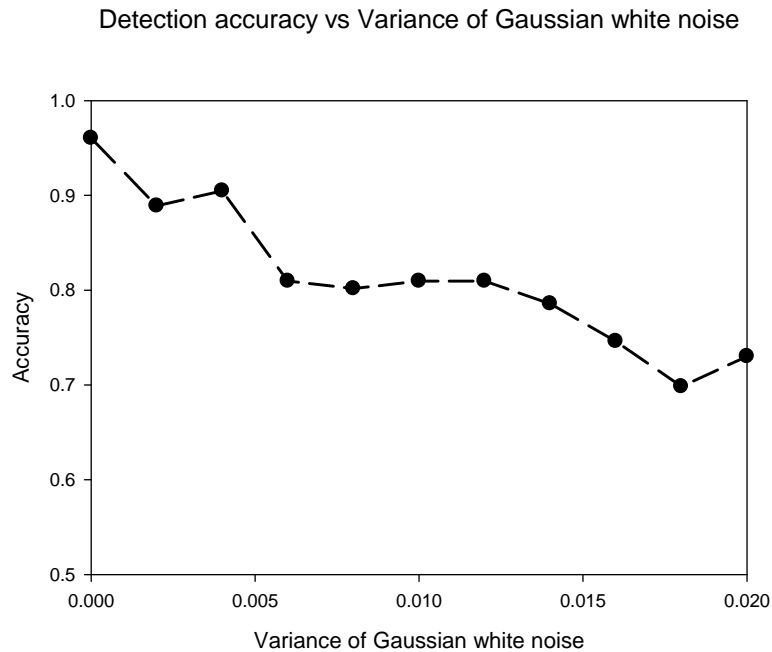


Fig. 2. 17 Detection accuracy of license plate detection using HOG feature with Gaussian white noise of mean 0 and variance from 0 to 0.02.

2.7 Summary

A license plate detection method using SVM classifier with HOG feature was developed in this work. The system performs window searching at different scales and analyzes the HOG feature using a SVM and locates their bounding boxes using Mean Shift. A car head and rear detection method was also proposed to accelerate the time consuming scanning process. Performance comparison and analysis with different cell and block sizes of the HOG feature was provided, and four rounds of bootstrapping was performed to ensure better detection results. License plate detection results show that this method is relatively insensitive to variations in illumination, license plate patterns, perspective of license plates and background complexity. Quantitative analysis was also performed on the Caltech data set (1999), and an acceptable accuracy rate of 96.0% was

reached. A test of license plate detection with different noise level and motion blur shows that the algorithm is very insensitive to the noise, and is tolerant to the motion blur in a certain range. False alarm detections were also shown with a discussion for future improvement.

CHAPTER 3

LICENSE PLATE RECOGNITION

We can see that the HOG descriptors perform fairly well in license plate detection with SVM classifiers from the previous chapter. Although the Optical Character Recognition (OCR) technology can already achieve very high accuracy in license plate recognition tasks[52], we still want to investigate if the using HOG feature with SVM can achieve acceptable result in character segmentation and recognition in vehicle license plates. That is because it will save a lot of resources if one module can do all the processing from license plate detection to recognition in implementation. As letters in lower case do not appear in US license plates, we only need to consider numeral characters and capital letters in segmentation and recognition.





































3.1 Digit Characters and Capital Letters Segmentation using HOG

Traditional character segmentation method for license plate recognition is to do the horizontal and vertical histogram projection on the edge map of license plate images. The gaps between characters can be predicted from the projection [52]. However, the accuracy of this method is not good enough, and as it relies on the edge map, the error rate is very high when the image contrast is not good enough. We are trying to use similar technique as we did license plate detection described in Chapter 2 to perform the

character segmentation in license plates. As we found that the HOG feature is not sensitive to lighting conditions and contrast, and probably, it can assist with the histogram projection method to do the character segmentation.

We collected a set of character segments from license plates positive training examples, together with their rotations of -12° , -6° , 6° and 12° . Negative patches contain those with imperfect characters, gaps between characters or none characters. All the patches were normalized to a resolution of 20×36 . The details of the training set are listed in Table 3. 1. It should be noticed that digit 0 and letter O are merged into one class.

Table 3. 1 Characters segmentation training set details.

	character	No. of patches	character	No. of patches	character	No. of patches	character	No. of patches
Positive patches		2020		2130		930		430
		2655		1175		1235		445
		2525		1450		660		470
		2395		1075		1675		470
		2260		1440		1255		595
		1785		675		1205		
		2030		725		1205		
		3080		1105		1140		
		1895		715		1460		
		1685		460		1145		
Negative patches		39597						

The SVM model was trained with cell size of 4×4 and block size of 8×8 HOG extracted from the patches with 2 pixels of margin on all four sides. The feature length is 756. A threshold for prediction score should be carefully chose to ensure good performance. We just to demonstrate this algorithm can work and did not do the bootstrapping procedure. The testing result should be even better if we bootstrapped the model.

The test images are license plate images normalized to 200×100 . During the segmentation process, each testing images were scanned in four different scales which are 0.4561, 0.5337, 0.6247 and 0.7305 of its original size. The 20×36 window with 2 pixels of margin on all four sides was scanned all over the image pyramid with a step size of 2 pixels horizontally and vertically. The corresponding HOG feature was extracted from the window with cell size of 4×4 and block size of 8×8 . A detection score was produced after the feature was analyzed by the SVM. The potential segments are produced after thresholding the detection score. The 3D Mean shift algorithm should be applied to fuse the potential segments in the 3D space. The window size for the 3D Mean shift algorithm was tuned to make sure it will converge to each character region. And the distance of the gap for clustering algorithm should also be adjusted so that it can pick several detections in a single image. Some character segmentation results were shown in Fig. 3. 1 Character segments of license plates. The original license plates are shown on top of each character segments images. The character segments are labeled by the green bounding boxes..

3.2 Digit Characters and Capital Letters Recognition using HOG

After the characters are correctly segmented, the digit characters and capital letters recognition in license plates will become a pattern recognition problem. Among 35 different classes, we need to identify which class one character patch belongs to. Normally there are two approaches to do the multi-class classification: the OneVsAll approach and Tree-like structure approach. The performance of Tree-like structure was not as good as OneVsAll approach with Haar-like feature using SVM in license plate recognition [36]. Here we use OneVsAll approach to do the license plate recognition, and meanwhile, we used a DigitOrLetter approach as a comparison. The training set includes all the positive patches for segmentation training data as is shown in Table 3. 2. HOG feature with cell size of 4×4 and block size of 8×8 was extracted from the patch with 2 pixels of margin on all four sides.

3.2.1 OneVsAll

For our current character recognition problem, the OneVsAll approach, 35 SVM models were trained with one class having a positive label and all other 34 classes having a negative label. The details of all the 35 classes are shown in Table 3.2. The thresholds of the prediction score for all 35 models will be carefully selected to minimize the sum of FPPW and Miss rate for each model. To recognize a character, all 35 SVM models will be applied. After the thresholds being applied, the class with highest prediction score is where the patch belongs to.

3.2.2 DigitOrLetter

We noticed for many states in US, the format of license plate is predefined. For example, for Missouri license plates, the 3rd and 5th characters are numerical characters, and other characters are capital letters. In this case, we just need to do the classification among all digits or all capital letters. If it is not certain that one character is a digit or a capital letter, we can use a SVM model to separate digits and capital letters first. The training set is listed in Table 3.3. The threshold for this SVM model was also selected where the sum of FPPM and Miss Rate achieved minimum. Once the digits and capital letters are separated, we apply OneVsAll approach on digits (10 classes) or capital letters (25 classes) respectively.

Table 3.2 Characters segmentation training set details.







































































character	No. of patches	character	No. of patches	character	No. of patches	character	No. of patches
	2020		2130		930		430
	2655		1175		1235		445
	2525		1450		660		470
	2395		1075		1675		470
	2260		1440		1255		595
	1785		675		1205		
	2030		725		1205		
	3080		1105		1140		
	1895		715		1460		
	1685		460		1145		

Table 3.3 Training set to separate digits and capital letters.

	character	No. of patches	character	No. of patches	character	No. of patches	character	No. of patches
Positive patches		2020		2395		2030		1685
		2655		2260		3080		
		2525		1785		1895		
		2130		1105		1255		445
		1175		715		1205		470
		1450		460		1205		470
Negative patches		1075		930		1140		595
		1440		1235		1460		
		675		660		1145		
		725		1675		430		

3.3 Results and Discussion

3.3.1 Digit Characters and Capital Letter Segmentation using HOG

Character segmentation results are shown in Fig. 3.1. We can see the license plates with very bad contrast as are shown in Figs. 3.1 (b) and (j), those with very low resolution as are shown in (a) and (f) and those tilted with a certain angle as are shown in (g) and (h) can all be segmented correctly. Those license plate images can be difficult to segment with the traditional histogram projection method. Although some segmentations are not very precise, and many false alarms appears with our current model, if we improve our SVM model for segmentation, it can at least assist with the traditional histogram projection character segmentation method to achieve better result.



Fig. 3. 1 Character segments of license plates. The original license plates are shown on top of each character segments images. The character segments are labeled by the green bounding boxes.

3.3.2 Digit Characters and Capital Letter Recognition using HOG

The confusion matrix of the OneVsAll approach is shown in Table 3.4. Here we used the text data set with 100 patches for each class with rotation of -12° , -6° , 0° , 6° and 12° . There are 3500 patches in the data set. The test patches are all normalized to 20×36 . The thresholds of the prediction score for all 35 models will be carefully selected to minimize the sum of FPPW and Miss rate for each model. To recognize a character, all 35 SVM models will be applied. After the thresholds being applied, the class with highest prediction score is where the patch belongs to. We can see that some very distinctive characters are easy to recognize, such as '3' and '4', however, the recognition rate of '0' and 'Z' is far from acceptable. Further bootstrapping will certainly help to improve the recognition rate, but to achieve very high performance we may need to find an alternative feature to do the recognition.

Table 3.4 Confusion matrix of the OneVsAll approach

		Actual																																					
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	G	H	I	J	K	L	M	N	P	Q	R	S	T	U	V	W	X	Y	Z			
Predicted	0	45	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	2	0	5	0	2	0	0	0	0	0			
	1	0	19	0	0	0	5	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	4	0	1	0	0	0	0	0	0	0			
	2	6	0	76	0	1	0	0	11	0	0	0	0	0	9	22	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	4	0	0	0	0	16		
	3	0	0	0	93	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	4	0	3	0	0	90	0	0	0	0	0	36	0	1	0	0	0	0	0	0	2	23	0	1	0	0	0	6	0	0	0	3	0	0	0	0	0		
	5	0	0	0	0	0	80	26	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0		
	6	0	0	0	0	0	7	74	0	0	0	0	5	0	0	13	15	18	5	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	7	26	53	1	0	0	0	64	0	0	0	0	0	4	0	0	0	0	20	10	0	0	0	4	0	25	0	0	21	4	0	3	0	0	0	3			
	8	0	0	0	0	0	0	0	76	0	0	19	0	0	5	0	0	0	0	0	0	0	0	0	0	0	3	1	0	0	0	0	0	0	0	0	0		
	9	0	0	1	0	8	2	0	0	4	76	4	0	0	0	2	2	0	21	0	0	0	0	2	0	15	0	18	0	0	0	4	0	0	0	0	0		
	A	0	0	0	0	0	0	0	0	0	0	60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	B	0	0	0	0	0	0	0	0	9	0	0	75	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	
	C	0	0	0	0	0	0	0	0	0	0	0	0	82	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	D	2	0	0	0	0	0	0	0	0	0	0	0	0	60	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
	E	0	0	0	0	0	0	0	0	0	0	0	0	0	0	80	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	F	0	0	0	0	0	0	0	0	0	0	0	0	0	0	64	0	3	0	0	0	0	0	11	0	4	0	0	0	0	0	2	0	0	0	0	0	0	
	G	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	82	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	H	0	0	10	0	0	0	0	0	5	0	0	0	0	0	0	0	0	67	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	I	0	15	10	0	0	0	0	9	0	0	0	0	0	2	0	0	0	0	67	0	13	3	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	J	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	K	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	L	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	86	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	M	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	3	0	73	1	0	0	1	0	0	0	0	0	0	0	0	0	0	
	N	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	68	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	P	0	0	0	3	0	0	0	3	0	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Q	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
R	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
S	0	0	0	0	0	6	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
T	0	3	0	2	0	0	0	13	0	0	0	0	4	0	0	8	0	0	6	0	47	0	0	2	0	0	10	77	0	16	0	0	60	0	0	0	0		
U	7	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	12	0	0	5	7	0	1	0	0	0	75	14	2	0	2	0	0	0		
V	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	6	0	0	0	4	0	0	0	0	0	52	0	0	2	0	0	0	0		
W	0	0	0	1	0	0	0	0	3	0	0	0	0	0	0	0	0	2	0	2	0	0	0	0	0	0	0	0	0	0	84	0	0	0	0	0	0		
X	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	2	3	13	1	1	0	0	1	0	0	3	38	0	0	0	0		
Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	3	0	36	0	0		
Z	0	0	1	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	3	0	0	0	0	1	0	0	0	0	0	0	0	5	0	0	0	0	0		

The confusion matrices of the DigitOrLetter approach are shown in Table 3.5 and Table 3.6. We used the same test dataset as in OneVsAll approach. We can see for both digits and capital letters recognized among fewer classes, the recognition rate is far better than those recognized in all 35 classes. However, a FPPW of 26.04% and a Miss Rate of 13.2% should be considered first to separate the digits from the capital letters.

Table 3. 5 Confusion matrix of the DigitOrLetter approach for digits
Actual

		Actual									
		0	1	2	3	4	5	6	7	8	9
Predicted	0	72	0	0	2	0	1	0	0	0	0
	1	0	26	0	0	0	1	0	0	0	0
	2	10	0	81	0	0	0	0	10	0	0
	3	0	0	0	95	0	0	0	0	0	6
	4	0	3	0	0	96	0	0	0	0	0
	5	0	0	2	2	0	89	22	0	0	0
	6	0	0	1	0	1	6	78	0	0	0
	7	18	71	5	1	0	0	0	89	1	0
	8	0	0	0	0	0	0	0	0	99	0
	9	0	0	11	0	3	3	0	1	0	94

Table 3. 6 Confusion matrix of the DigitOrLetter approach for letters
Actual

		Actual																										
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	P	Q	R	S	T	U	V	W	X	Y	Z		
Predicted	A	79	1	1	0	0	0	0	0	0	5	1	0	0	0	0	0	2	0	0	0	0	2	0	0	0	0	
	B	0	85	0	0	0	0	2	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
	C	0	1	74	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	3	0	0	0	0
	D	0	0	0	59	0	0	0	0	14	0	0	0	0	2	0	3	0	9	8	2	0	0	0	0	0	0	0
	E	0	9	0	0	98	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3
	F	0	0	0	0	0	58	0	4	0	0	1	0	19	0	0	0	0	0	0	0	0	3	1	4	24	0	0
	G	0	0	1	0	1	0	94	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
	H	0	4	0	0	0	15	0	81	0	0	28	0	5	6	48	0	26	0	0	0	0	5	6	0	0	0	0
	I	0	0	0	0	0	0	0	0	47	0	1	0	0	0	0	0	0	0	0	0	1	0	0	16	0	6	0
	J	20	0	0	0	0	0	0	0	0	37	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	2
	K	0	0	0	0	0	0	0	12	0	0	53	5	0	0	0	0	0	0	0	0	26	1	1	0	0	0	0
	L	0	0	0	0	0	0	0	0	0	6	0	87	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
	M	0	0	0	0	0	0	0	0	0	0	6	0	72	1	0	0	2	0	0	0	1	1	20	0	0	0	0
	N	0	0	0	0	0	0	0	0	0	0	0	0	0	65	0	0	0	0	0	0	0	0	0	0	0	0	0
	P	0	0	0	0	0	1	0	0	0	0	0	0	0	0	49	0	2	0	0	0	2	0	0	0	21	0	0
	Q	1	0	0	0	1	0	0	0	0	4	0	0	0	0	0	63	0	7	0	1	0	0	0	0	0	0	
	R	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	61	0	0	1	0	0	0	0	0	0	
	S	0	0	0	0	0	8	0	2	0	0	0	0	0	1	0	2	84	0	0	0	0	0	0	0	0	0	
	T	0	0	0	0	0	6	0	0	0	6	0	0	1	0	0	0	72	0	0	0	0	0	6	0	0	0	
	U	0	0	0	0	0	0	0	0	2	28	0	0	0	8	0	0	0	0	0	61	3	0	0	0	0	0	
	V	0	0	0	0	0	0	0	0	5	0	0	0	0	4	0	0	0	0	0	0	54	0	0	2	0	0	
	W	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	4	0	0	0	0	0	82	8	0	0	0	
	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	45	0	0	0	
	Y	0	0	0	0	0	0	0	0	2	22	0	0	0	0	0	0	0	0	1	0	8	1	0	67	0	0	
	Z	0	0	23	41	0	0	4	1	30	3	0	5	1	13	0	32	0	0	19	32	2	2	0	1	68	0	

3.4 Summary

Character segmentation and recognition in license plates using SVM classifier with HOG feature was presented. For character segmentation, the system performs window searching in different scales and analyzes the HOG feature using a SVM and locates their bounding boxes using Mean Shift. Character segmentation results in license plates show that this method can assist with the traditional histogram projection method to deal with some critical conditions such as low contrast or tilted pose. Character recognition using SVM with HOG feature was also processed in two approaches. The confusion matrices for OneVsAll and DigitOrLetter approaches were shown, and the recognition rate of them was far from pleasing. Alternative approaches need to be developed in order to achieve acceptable result.

CHAPTER 4

CONCLUSIONS AND FUTURE DIRECTION

A license plate detection method using SVM classifier with HOG feature was developed in this work. The system performs window-based searching at different scales and analyzes the HOG feature using a SVM and locates their bounding boxes using Mean Shift. A car head and rear detection method was also proposed to accelerate the time consuming scanning process. Performance comparison and analysis of the algorithm with different cell and block sizes of HOG feature was provided, and four rounds of bootstrapping was performed to ensure better detection results. License plate detection results show that this method is relatively insensitive to variations in illumination, license plate patterns, perspective of license plates and background complexity. Quantitative analysis was also performed on the Caltech data set (1999), and an acceptable accuracy rate of 96.0% was achieved. Our test of license plate detection with different noise level and motion blur shows that the algorithm is very insensitive to the noise, and is tolerant to the motion blur in a certain range. Many false alarm detections lie in the car light region, which indicate combining color information into the HOG feature may help to improve the detection results.

Character segmentation and recognition in license plates using SVM classifier with HOG feature was presented. For character segmentation, the system performs window searching in different scales and analyzes the HOG feature using a SVM and

locates their bounding boxes using Mean Shift. Character segmentation results in license plates show that this method can assist with the traditional histogram projection method to deal with some critical conditions such as low contrast or tilted pose. Character recognition using SVM with HOG feature was also processed in two approaches. The confusion matrices for OneVsAll and DigitOrLetter approaches were shown, and the recognition rate of them is promising. However, further algorithm refinement is needed to achieve better performance.

REFERENCES

- [1] C. N. E. Anagnostopoulos, I. E. Anagnostopoulos, V. Loumos, and E. Kayafs, *A license plate-recognition algorithm for intelligent transportation system applications* vol. 7. Piscataway, NJ, ETATS-UNIS: Institute of Electrical and Electronics Engineers, 2006.
- [2] 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.
- [3] 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 *Fuzzy Information, 2004. Processing NAFIPS '04. IEEE Annual Meeting of the*, 2004, pp. 357-362 Vol.1.
- [4] Q. Jin, S. Quan, Y. Shi, and Z. Xue, "A fast license plate segmentation and recognition method based on the modified template matching," in *Image and Signal Processing, 2009. CISP '09. 2nd International Congress on*, 2009, pp. 1-6.
- [5] L. Xiaoping, L. Xiaoxing, W. Shuaizong, Z. Lin, L. Yinxiang, and D. Hongjian, "Research on the recognition algorithm of the license plate character based on the multi-resolution template matching," in *New Trends in Information Science and Service Science (NISS), 2010 4th International Conference on*, 2010, pp. 341-344.
- [6] B. Hongliang and L. Changping, "A hybrid license plate extraction method based on edge statistics and morphology," in *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 2 - Volume 02*, 2004, pp. 831-834.
- [7] D. Zheng, Y. Zhao, and J. Wang, "An efficient method of license plate location," *Pattern Recogn. Lett.*, vol. 26, pp. 2431-2438, 2005.

- [8] P. V. Suryanarayana, S. K. Mitra, A. Banerjee, and A. K. Roy, "A morphology based approach for car license plate extraction," in *INDICON, 2005 Annual IEEE*, 2005, pp. 24-27.
- [9] P. Comelli, P. Ferragina, M. N. Granieri, and F. Stabile, "Optical recognition of motor vehicle license plates," *Vehicular Technology, IEEE Transactions on*, vol. 44, pp. 790-799, 1995.
- [10] J. Cano and J.-C. Pérez-Cortés, *Vehicle License plate segmentation in natural images* vol. 2652. Berlin, ALLEMAGNE: Springer, 2003.
- [11] J. Barroso, E. L. Dagless, A. Rafael, and J. Bulas-Cruz, "Number plate reading using computer vision," in *Industrial Electronics, 1997. ISIE '97., Proceedings of the IEEE International Symposium on*, 1997, pp. 761-766 vol.3.
- [12] L. Hsi-Jian, C. Si-Yuan, and W. Shen-Zheng, "Extraction and recognition of license plates of motorcycles and vehicles on highways," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, 2004, pp. 356-359 Vol.4.
- [13] V. Kamat and S. Ganesan, "An efficient implementation of the Hough transform for detecting vehicle license plates using DSP'S," in *Proceedings of the Real-Time Technology and Applications Symposium*, 1995, p. 58.
- [14] 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.
- [15] R. Zunino and S. Rovetta, "Vector quantization for license-plate location and image coding," *Industrial Electronics, IEEE Transactions on*, vol. 47, pp. 159-167, 2000.
- [16] 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.
- [17] K. Deb and J. Kang-Hyun, "HSI color based vehicle license plate detection," in *Control, Automation and Systems, 2008. ICCAS 2008. International Conference on*, 2008, pp. 687-691.

- [18] M. H. ter Brugge, J. H. Stevens, J. A. G. Nijhuis, and L. Spaanenburg, "License plate recognition using DTCNNs," in *Cellular Neural Networks and Their Applications Proceedings, 1998 Fifth IEEE International Workshop on*, 1998, pp. 212-217.
- [19] M. I. Chacon M and A. Zimmerman S, "License plate location based on a dynamic PCNN scheme," in *Neural Networks, 2003. Proceedings of the International Joint Conference on*, 2003, pp. 1195-1200 vol.2.
- [20] K. K. Kim, K. I. Kim, J. B. Kim, and H. J. Kim, "Learning-based approach for license plate recognition," in *Neural Networks for Signal Processing X, 2000. Proceedings of the 2000 IEEE Signal Processing Society Workshop*, 2000, pp. 614-623 vol.2.
- [21] Y.-N. Chen, C.-C. Han, C.-T. Wang, B.-S. Jeng, and K.-C. Fan, "The application of a convolution neural network on face and license plate detection," in *Proceedings of the 18th International Conference on Pattern Recognition - Volume 03*, 2006, pp. 552-555.
- [22] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vision*, vol. 57, pp. 137-154, 2004.
- [23] E. S. Robert, F. Yoav, B. Peter, and L. Wee Sun, "Boosting the margin: a new explanation for the effectiveness of voting methods " in *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997.
- [24] Z. Huaifeng, J. Wenjing, H. Xiangjian, and W. Qiang, "Learning-based license plate detection using global and local features," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, 2006, pp. 1102-1105.
- [25] C. Xiangrong and A. L. Yuille, "Detecting and reading text in natural scenes," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2004, pp. II-366-II-373 Vol.2.
- [26] E. Osuna, R. Freund, and F. Girosit, "Training support vector machines: an application to face detection," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, 1997, pp. 130-136.

- [27] Y. Zhong and A. K. Jain, "Object localization using color, texture and shape," presented at the Proceedings of the First International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition, 1997.
- [28] K. I. Kim, K. Jung, and J. H. Kim, "Color texture-based object detection: an application to license plate localization," in *Proceedings of the First International Workshop on Pattern Recognition with Support Vector Machines*, 2002, pp. 293-309.
- [29] S. Nomura, K. Yamanaka, O. Katai, H. Kawakami, and T. Shiose, "A novel adaptive morphological approach for degraded character image segmentation," *Pattern Recognition*, vol. 38, pp. 1961-1975, 2005.
- [30] J. A. G. Nijhuis, M. H. Ter Brugge, K. A. Helmholt, J. P. W. Pluim, L. Spaanenburg, R. S. Venema, and M. A. Westenberg, "Car license plate recognition with neural networks and fuzzy logic," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, 1995, pp. 2232-2236 vol.5.
- [31] K.-B. Kim, S.-W. Jang, and C.-K. Kim, *Recognition of car license plate by using dynamical thresholding method and enhanced neural networks* vol. 2756. Berlin, ALLEMAGNE: Springer, 2003.
- [32] A. Capar and M. Gokmen, "Concurrent segmentation and recognition with shape-driven fast marching methods," in *Proceedings of the 18th International Conference on Pattern Recognition - Volume 01*, 2006, pp. 155-158.
- [33] V. Franc and V. Hlaváč, *License plate character segmentation using hidden markov chains* vol. 3663. Berlin, ALLEMAGNE: Springer, 2005.
- [34] C. Shyang-Lih, C. Li-Shien, C. Yun-Chung, and C. Sei-Wan, "Automatic license plate recognition," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 5, pp. 42-53, 2004.
- [35] Y. Hu, F. Zhu, and X. Zhang, *A novel approach for license plate recognition using subspace projection and probabilistic neural network* vol. 3498. Berlin, ALLEMAGNE: Springer, 2005.

- [36] C. Arth, F. Limberger, and H. Bischof, "Real-time license plate recognition on an embedded DSP-platform," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, 2007, pp. 1-8.
- [37] H. Yo-Ping, L. Shi-Yong, and C. Wei-Po, "A template-based model for license plate recognition," in *Networking, Sensing and Control, 2004 IEEE International Conference on*, 2004, pp. 737-742 Vol.2.
- [38] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face recognition with local binary patterns," in *Computer Vision - ECCV 2004*. vol. 3021, T. Pajdla and J. Matas, Eds., ed: Springer Berlin / Heidelberg, 2004, pp. 469-481.
- [39] G. Heusch, Y. Rodriguez, and S. Marcel, "Local binary patterns as an image preprocessing for face authentication," in *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, 2006, pp. 6 pp.-14.
- [40] H. Xiangsheng, S. Z. Li, and W. Yangsheng, "Jensen-Shannon boosting learning for object recognition," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005, pp. 144-149 vol. 2.
- [41] Y. Rodriguez and S. Marcel, "Face authentication using adapted local binary pattern histograms," in *Computer Vision – ECCV 2006*. vol. 3954, A. Leonardis, *et al.*, Eds., ed: Springer Berlin / Heidelberg, 2006, pp. 321-332.
- [42] Z. Jiali, W. Haitao, R. Haibing, and K. Seok-Cheol, "LBP discriminant analysis for face verification," in *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, 2005, pp. 167-167.
- [43] J. S. Weszka, C. R. Dyer, and A. Rosenfeld, "A comparative study of texture measures for terrain classification," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-6, pp. 269-285, 1976.
- [44] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, pp. 971-987, 2002.

- [45] C. Huajie and W. Wei, "Pseudo-example based iterative SVM learning approach for gender classification," in *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, 2006, pp. 9528-9532.
- [46] A. Mohan, C. Papageorgiou, and T. Poggio, "Example-based object detection in images by components," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 23, pp. 349-361, 2001.
- [47] P. Viola, M. J. Jones, and D. Snow, "Detecting pedestrians using patterns of motion and appearance," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, 2003, pp. 734-741 vol.2.
- [48] 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.
- [49] N. Dalal, "Finding people in images and videos," PhD, Institut National Polytechnique De Grenoble, 2006.
- [50] T. Joachims, "Making large scale SVM learning practical ", ed: Universität Dortmund, 1998.
- [51] 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.
- [52] 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.