Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 697658, 8 pages
http://dx.doi.org/10.1155/2014/697658

Research Article

Modeling and Implementing Two-Stage AdaBoost for Real-Time Vehicle License Plate Detection

Moon Kyou Song and Md. Mostafa Kamal Sarker

Department of Electronics Convergence Engineering, Wonkwang University, 344-2 Shinyong Dong, Iksan, Jeonbuk 570-749, Republic of Korea

Correspondence should be addressed to Md. Mostafa Kamal Sarker; mksarker@wku.ac.kr

Received 6 March 2014; Accepted 31 July 2014; Published 18 August 2014

Academic Editor: Young-Sik Jeong

License plate (LP) detection is the most imperative part of the automatic LP recognition system. In previous years, different methods, techniques, and algorithms have been developed for LP detection (LPD) systems. This paper proposes to automatical detection of car LPs via image processing techniques based on classifier or machine learning algorithms. In this paper, we propose a real-time and robust method for LPD systems using the two-stage adaptive boosting (AdaBoost) algorithm combined with different image preprocessing techniques. Haar-like features are used to compute and select features from LP images. The AdaBoost algorithm is used to classify parts of an image within a search window by a trained strong classifier as either LP or non-LP. Adaptive thresholding is used for the image preprocessing method applied to those images that are of insufficient quality for LPD. This method is of a faster speed and higher accuracy than most of the existing methods used in LPD. Experimental results demonstrate that the average LPD rate is 98.38% and the computational time is approximately 49 ms.

1. Introduction

Within the last few decades, LP recognition (LPR) has become an extremely popular and active research topic in the image processing domain. With the constant increase of traffic on the roads, there is a need for intelligent traffic management systems that can detect and track a vehicle as well as identify it. Most of the previous LP detection (LPD) algorithms are restricted in certain working conditions, such as fixed backgrounds [1], known color [2], or fixed size of the LPs [3]. Therefore, detecting LPs under various complex environments remains a challenging problem.

In this paper, we evaluate how well object detection methods used in text extraction [4] and face detection [5] apply to the problem of LP detection. We present a novel method for locating the LP rapidly using the two-stage cascade AdaBoost combined with different image preprocessing procedures. The cascade AdaBoost has two phases in two stages, offline training and online detection.

In the first stage of the cascade AdaBoost, the size of positive samples is extremely important for offline training; consequently, all positive images should be the same size. Using boundary padding or boundary pixel extension [6], the training positive sample images are created of the same size. Image preprocessing is organized with a Sobel vertical operator applied to the edge of the image; the image edge is then smoothed using the Gaussian filter. Once preprocessing is finished, the AdaBoost training phase starts. After the training process is complete, the detection phase becomes ready to detect the LP. During the online detection phase, the original images are resized for faster detection, and the same image preprocessing methods as in the offline training phase are applied. If the LP is detected with the trained cascade, the detected LP is verified through the connected component analysis (CCA). If this stage detects the LP correctly, the LP is saved; otherwise, the LP image is sent to the next stage.

In the second stage, all procedures are similar to the first stage of the cascade AdaBoost, except for the image preprocessing techniques. In this second stage, we find that most of the poor quality images that are rejected from the first stage have variant illumination, blurring, ambient lighting conditions, and so forth. Therefore, we use adaptive

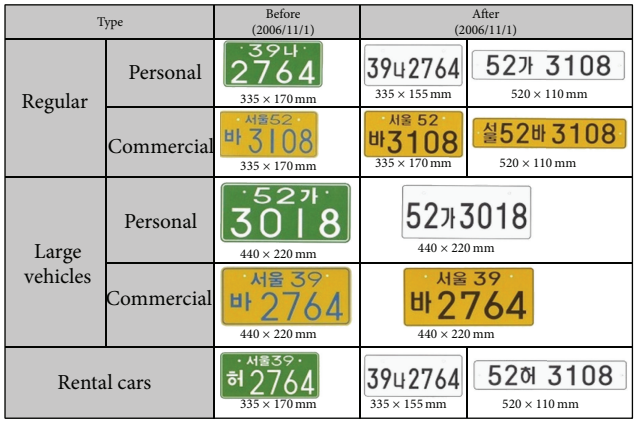

Figure 1: Different types and sizes of Korean LPs.

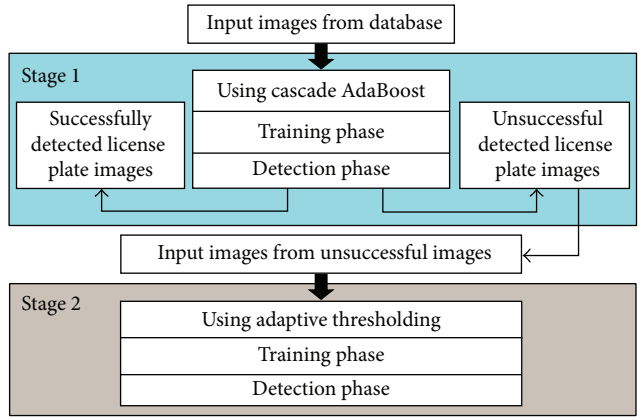

Figure 2: System architecture of the proposed LPD system using cascade AdaBoost.

thresholding to obtain maximum edge information from those images. The detection results obtained in this stage are remarkable for poor quality images. After finishing the two-stage cascade AdaBoost, we find that the average LPD rate is 98.38% with a computational time of 49 ms.

This paper is organized as follows. Background and challenges are illustrated in Section 2, and our proposed LPD method is described in Section 3. The experimental results in Section 4 show that the proposed method is able to ensure fast LPD as well as achieve sufficient accuracy. Finally, the conclusion is summarized in Section 5.

2. Background and Challenges

To properly work with LPR systems, we must manage a large variety of LPs, especially in South Korea. Each province in Korea has its own LP color, pattern, and formats of numbers and other characters. Different colors represent different types of vehicles. Moreover, there are three different sizes of LPs available in Korea, such as large (520 mm × 110 mm), medium (440 mm × 200 mm), and small (335 mm × 170 mm or 155 mm). Figure 1 shows the different types and sizes of LPs available before and after November 01, 2006, in Korea.

3. Proposed System

Our proposed LPD system consists of two parts; the first part uses cascade AdaBoost and the second part uses adaptive thresholding. (See Figure 2 for the system architecture of our proposed system.)

3.1. Using Cascade AdaBoost. Using cascade AdaBoost for LPD systems consists of two phases, offline training and online detection, as shown in Figure 3.

3.1.1. Offline Training Phase. At the core of the offline training phase are the training and combination of strong classifiers. First, a series of weak classifiers (critical features) with their weights are extracted after being trained by a large number of positive and negative examples. Then, strong classifiers are selected from the weak classifiers according to their weights. Figure 4 shows how the algorithm is structured. The strong

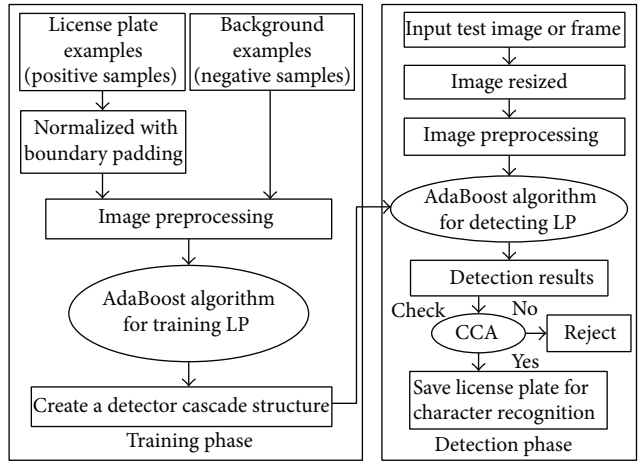

Figure 3: Framework of the proposed LPD system using cascade AdaBoost.

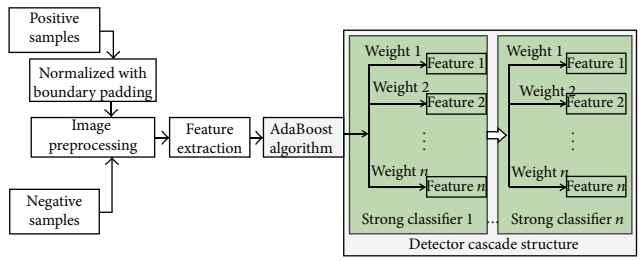

Figure 4: Structure for the offline training phase.

classifiers are then constructed in a detector cascade structure for the online recognizing module.

(1) The Training Databases. For offline training, positive sample images and negative sample images are required. The positive sample images are LP images only; the negative sample images are background images without an LP image.

(a) The Positive Samples. The Korean LP is made of three different sizes, large (520 mm × 110 mm), medium

Figure 5: Positive sample images of Korean LPs.

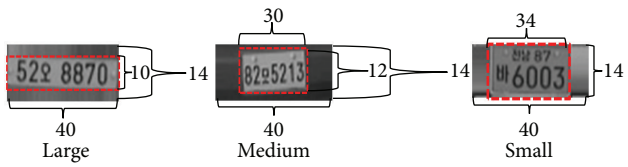

Figure 6: Normalized LP images of positive samples with boundary padding.

Figure 7: Negative sample images.

Figure 8: Examples of filtered images after using Gaussian filter: (a) positive samples and (b) negative samples.

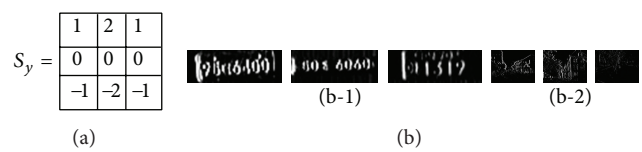

Figure 9: (a) Convolution mask of Sobel vertical edge operator, (b) examples of edge images after using Sobel vertical edge operator (b-1) positive samples and (b-2) negative samples.

(440 mm × 200 mm), and small (335 mm × 170 mm or 155 mm). A total of 15,000 images are used as the positive sample images (6,000 large; 3,000 medium; and 6,000 small) for our training experiment. Figure 5 shows some positive sample images of Korean LPs.

The AdaBoost training algorithm requires that the positive sample images be of the same size. For this reason, we must normalize all three types of Korean LP images into one equal size. Boundary padding or boundary pixel extension [6] causes all the positive sample images to be of the same size. For our training case, an image size resolution of 40×14 is applied because all the LP regions of our database images are under this resolution. Figure 6 shows the normalized LP images with boundary padding or boundary pixel extension.

(b) The Negative Samples. The negative sample images should appear without the LP; for example, they can be images of a part of the car, the road, trees, and so forth. In our training procedure, a total of 25,000 images are used for the negative samples. Figure 7 shows some negative sample images.

(2) Image Preprocessing. In this section, we describe the image converting, filtering, and edge detection methods that are applied to preprocess the training images.

(a) Image Converting. Given that RGB images have more depth, they are difficult to process; therefore, we convert the original images into gray scale images. In addition, image enhancement and edge detection are performed on the gray scale images in order to adjust the structural property of the images in preparation for LP region detection. The input images are converted from 24-bit color images to 8-bit gray scale images using

$$\text{Gray value} = 0.3 * \text{Red} + 0.59 * \text{Green} + 0.11 * \text{Blue}. \quad (1)$$

(b) Image Filtering. Gaussian filter is applied for image filtering. Gaussian filter is the weighted averaging of neighboring pixels; the weights are chosen according to the shape of a Gaussian function, which is defined as

$$g[i, j] = e^{-(i^2+j^2)/2\sigma^2}, \quad (2)$$

where i is the distance from the origin in the horizontal axis, j is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution. Figure 8 shows the filtered images using Gaussian filter.

(c) Edge Detection (Edge Image). For edge detection or edge image, the Sobel vertical edge operator [7] is applied. Figure 9 defines the convolution mask of the Sobel vertical edge operator and the edge image after using the Sobel vertical edge operator.

(3) Feature Extraction. LP location procedures classify images based on the value of simple features by using the intensity values of a pixel. These features are using the change in contrast values between adjacent rectangular groups of pixels.

The contrast variances between the pixel groups are used to determine relative light and dark areas. Two or three adjacent groups with a relative contrast variance form a Haar-like feature. These features are used for the LPD shown in Figure 10.

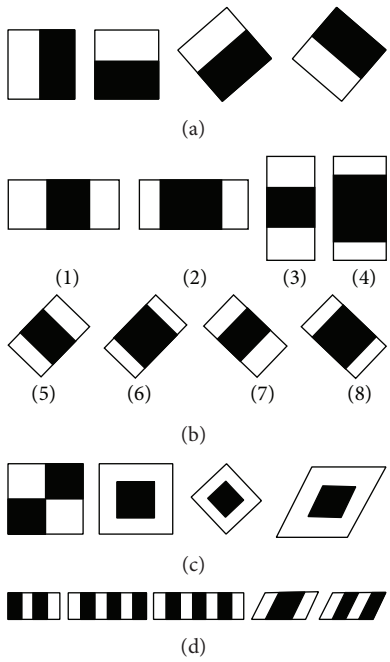

Figure 10: Haar-like prototypes used in our algorithm: (a) edge features, (b) line features, (c) center-surrounding features, and (d) plate character features.

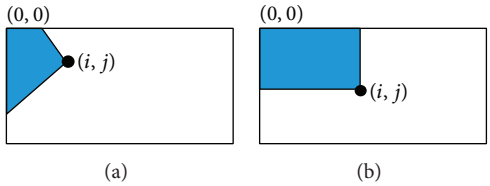

Figure 11: (a) Summed area of integral image and (b) summed area of rotated integral image.

By using a transitional depiction of an image, the simple rectangular features of an image are calculated. This is called the integral image [8]. The integral image is an array that contains the sums of the pixel intensity values located directly to the left of a pixel and directly above the pixel at location (i, j), inclusively. Therefore, if $O[i, j]$ is the original image and $\mathrm{OI}[i, j]$ is the integral image, the integral image is calculated as shown in (3) and demonstrated in Figure 11:

$$\mathrm{OI}[i, j] = \sum_{i' \le i, j' \le j} O\left(i', j'\right). \tag{3}$$

The features are rotated 45 degrees, similar to the line feature shown in Figure 10(b)(5), as presented by Lienhart and Maydt [9]. Such features require another transitional depiction called the rotated integral image or rotated sum auxiliary image. The rotated integral image is computed by finding the sum of the pixel intensity values that are located at a 45-degree angle to the left and above of the i value and below the j value. Therefore, if $O[i, j]$ is the original image and $\mathrm{OR}[i, j]$ is the rotated integral image, the integral image is calculated as shown in (4) and illustrated in Figure 11:

$$\mathrm{OR}[i, j] = \sum_{i' \le i, i' \le i - |j - j'|} O\left(i', j'\right). \tag{4}$$

(4) AdaBoost Algorithm for Training LP. The cascade boosted classifier that is created in the Haar-like features training process for several LP samples locates the LP extremely fast and correctly. For each feature, the weak learner determines the optimal threshold classification function, such that the minimum number of examples is misclassified. Thus, a weak classifier $h(x, f, p, \theta)$ consists of a feature (f), a threshold (θ), and a polarity (p) that indicates the direction of the inequality sign:

$$h(x, f, p, \theta) = \begin{cases} 1 & \text{if } pf(x) < p\theta \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

The boosting process and the AdaBoost algorithm for classifier learning are as follows.

(1) Give sample images $(x_1, y_1) \cdots (x_n, y_n)$, where $y_i = 0$ and 1 for negative and positive samples, respectively.

(2) Initialize weights ω_1, $i = 1/2\ m$ and $1/2\ l$ for $y_i = 0$ and 1, respectively, where m and l are the number of negatives and positives, respectively.

(3) For $t = 1 \cdots T$,

(a) normalize the weights

$$\omega_{t,i} = \frac{\omega_{t,j}}{\sum_{j=1}^{n} \omega_{t,j}} \tag{6}$$

such that ω_t is a probability distribution.

(b) Select the best weak classifier with respect to the weighted error

$$\varepsilon_t = \min_{f,p,\theta} \sum \omega_i \left| h(x_i, f, p, \theta) - y_i \right|. \tag{7}$$

(c) Define $h_t(x) = h(x, f_t, p_t, \theta_t)$, where f_t, p_t, and θ_t are the minimizers of ε_t.

(d) Update the weights

$$\omega_{t+1,i} = \omega_{t,i} \beta_t^{1-e_i}, \tag{8}$$

where $e_i = 0$ if example x_i is classified properly; otherwise, $e_i = 1$ and $\beta_t = e_t / 1 - e_t$.

(4) The final strong classifier is

$$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \ge \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

where $\alpha_t = \log 1/\beta_t$.

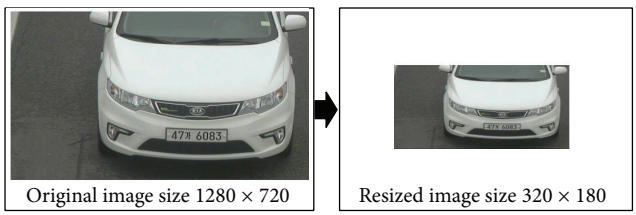

Figure 12: Resized image from the original image.

The methods used involve training a strong classifier using the AdaBoost algorithm. Over numerous sequences, AdaBoost chooses the best performing weak classifier from a group of weak classifiers acting on a single feature; once trained, AdaBoost combines the respective votes of the classifiers in a weighted manner, thus forming a strong classifier. This strong classifier is then applied to the subregions of the image that is being scanned for possible LP locations. The weak learning algorithm is designed to select the single rectangle feature that best separates the positive and negative samples. An optimization introduced by Viola and Jones [8] involves a cascade of strong classifiers, each with precisely designed false-positives and false-negatives rates, that greatly speeds up the scanning process because not all classifiers must be evaluated to exclude most non-LP subregions. A background threshold of 80, a number of training stages of 14, and a total number of features of 61,789 are used in our AdaBoost training phase. After finishing the training procedure of the AdaBoost algorithm, a detector cascade structure is created as an XML file. This XML file contains the strong classifier with features.

3.1.2. Online Detection Phase. For the online LPD, the number of test images is 1,800 with a resolution of 1280 × 720. (See Section 4.1 for an explanation of the databases.) The details of the online LPD procedure are described next.

(1) Resizing the Input Images. The size of the images in our databases is extremely large. A large image resolution requires more computational time; therefore, we resized the original test images (1280 × 720) into a resolution of 320 × 180 to accelerate the detection procedure. Figure 12 shows an image resized from the original.

(2) Image Preprocessing. The same image preprocessing techniques explained in Section 3.1.1(2) are utilized in the offline training phase: first, convert the images from RGB to gray scale; then, filter the images with Gaussian filter; finally, apply edge detection (edge image) with the Sobel vertical edge operator. Figure 13 shows the results of image preprocessing.

(3) AdaBoost Algorithm for Detecting LPs. The strong classifiers combine with each other to form a classifier cascade. The strong classifier from the first layer allows a vast majority of the image regions to be recognized and passed to the next layer; at the same time, the classifier rejects as many negative samples as possible. Thus, the classifier cascade has stronger classification abilities, and the final result is more likely to be an LP. Figure 14 shows the cascade structure of the LPD.

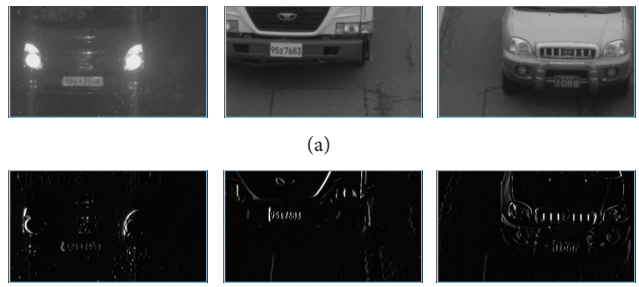

Figure 13: Results of image preprocessing: (a) examples of filtered image after using Gaussian filter and (b) examples of edge image after using Sobel vertical operator.

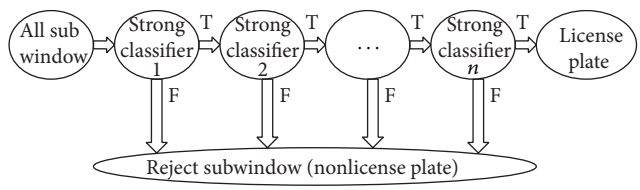

Figure 14: Cascade AdaBoost structure for LPD.

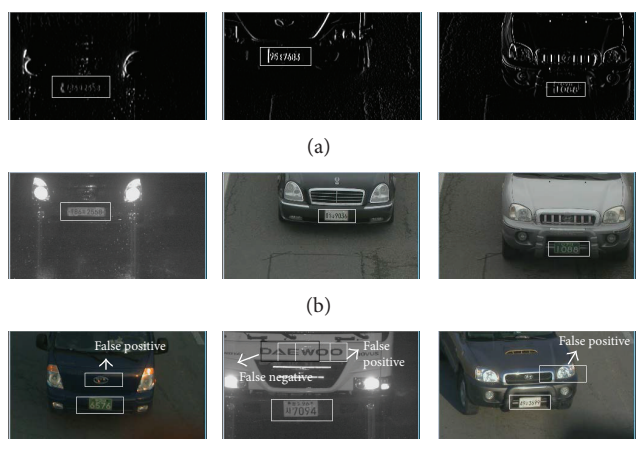

Figure 15: Results of LPD using cascade AdaBoost: (a) examples of LPD with edge images, (b) examples of LPD with resized images, and (c) examples of LPD with false positives/negatives.

During the combination process, the strong classifier that consists of more important features and an easier structure is placed at the top of the entire classifier cascade in order for the system to exclude as many negative samples as possible, thus accelerating the detection of LPs.

(4) LPD Results. Figure 15 shows the results of the LPD using our proposed cascade AdaBoost algorithm.

(5) Verify Detected LP Images with CCA. There are many false positives/negatives areas detected as LP regions using the cascade AdaBoost algorithm. To ignore such false positives/negatives, we use CCA. Figure 16 shows the procedure

Table 1: LPD using cascade AdaBoost.

Number of test images	False positives/negatives		Test accuracy	Time
	Before applying CCA	After applying CCA		
1800	17%	0%	87.94%	45 ms

(1) **If** number of Blob ≥ 6 and ≤10
(2) area = LP
(3) **Else**
(4) area = Non-LP
(5) **End**

Procedure 1: Blob (number, area).

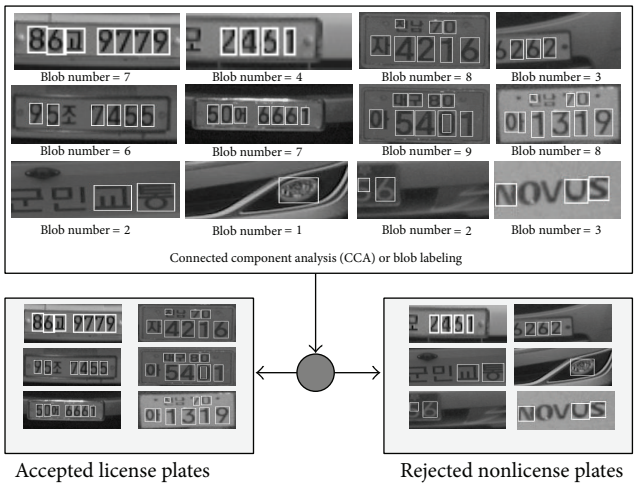

Figure 16: Verifying images with CCA and saving LP images.

for verifying detected LP images with CCA. The verification LP and non-LP procedure is as in Procedure 1.

3.2. Using Adaptive Thresholding. Using adaptive thresholding for LPD systems is similar as using the cascade AdaBoost algorithm, (see Section 3.1), with the exception of the image preprocessing method. The majority of the images rejected from the first stage have variant illumination, blurring, ambient lighting conditions, and so forth. Therefore, for the image preprocessing technique in this stage, we use adaptive thresholding to compare each pixel of an image to an average of the surrounding pixels. The procedure for adaptive thresholding is as in Procedure 2.

Figure 17 shows the LPD results after applying adaptive thresholding.

4. Experimental Results

To test the performance of our proposed method, we use our own database with 1,800 images. The details of the experimental results are presented next.

Figure 17: LPD results using adaptive thresholding.

4.1. Databases. For our test experiments, we used one database to calculate the detection rate and the computational time. The numbers of total images are 1,800. All the images in our database are rotated and illuminated; furthermore, the images were captured using a CMOS camera under different weather conditions.

4.2. Experimental LPD Results Using Cascade AdaBoost. To test the LPD using cascade AdaBoost method proposed in this paper, we applied the method to a database of 1,800 images that were captured at different times and weather conditions. The experiment is based on the conditions of a system with CPU 3.10-GHz Intel Core i3-2100 and 4.00 GB of RAM and implemented using Microsoft Visual Studio 2010 with OpenCV library. Table 1 lists the LPD rate, the percentage of false positives/negatives, and the computational time with the database.

From Table 1, we can see that the total number of detected images is 1,583 and the number of detected false positives/negatives is 306. After applying CCA, no false positives/negatives remained.

4.3. Experimental Results for LPD Using Adaptive Thresholding. The remaining 217 images that were not detected correctly (from step one of the phase that uses the cascade AdaBoost algorithm) are used as input images in this phase, adaptive thresholding is applied to them, and then the images are used for training and testing with the same procedures employed for the cascade AdaBoost phase.

From Table 2, we can see that the total number of detected images is 188, and the number of detected false positives/negatives is 35. After applying CCA, no false positives/negatives remain. The images that were not detected properly are 29.

4.4. Summary of Experimental Results. The total number of test images is 1,800. The total number of detected images is 1,771. The number of images that could not be detected

```
(1) for p = 0 to w do
(2)     sum ← 0
(3)   for q = 0 to h do
(4)     sum ← sum + in[p, q]
(5)    if p = 0 then
(6)     intImg[p, q] ← sum
(7)    else
(8)     intImg[p, q] ← intImg[p − 1, q] + sum
(9)    end if
(10)  end for
(11) end for
(12) for p = 0 to w do
(13)  for q = 0 to h do
(14)     x1 ← p − s/2 {border checking is not shown}
(15)     x2 ← p + s/2
(16)     y1 ← q − s/2
(17)     y2 ← q + s/2
(18)     count ← (x2 − x1) × (y2 − y1)
(19)     sum ← intImg[x2, y2] − intImg[x2, y1 − 1] − intImg[x1 − 1, y2] + intImg[x1 − 1, y1 − 1]
(20)     if (in[p, q]× count)_(sum × (100 − t)/100)
(21)             then out[p, q] ← 0
(22)      else
(23)         out[p, q] ← 255
(24)     end if
(25)  end for
(26) end for
```

Procedure 2: Adaptive threshold (in, out, w, h).

Table 2: LPD using adaptive thresholding.

Number of test images	False positives/negatives		Test accuracy	Time
	Before applying CCA	After applying CCA		
217	16%	0%	86.64%	80 ms

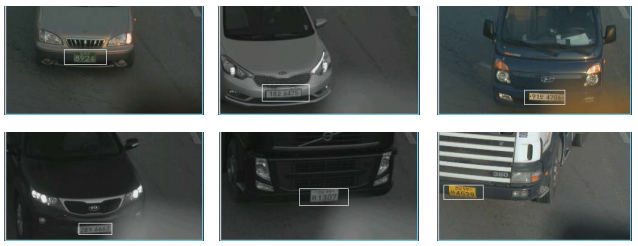

Figure 18: Successful images for LPD using our proposed method under different rotation and illumination.

properly is 29. Therefore, the average detection rate is 98.38% and the computational time is approximately 49 ms. Figure 18 shows some of the test results under different rotation and illumination.

4.5. Performance Comparison of Some Typical LPR Systems with Our Methods for LPD. See Table 3.

5. Conclusion

We demonstrated a procedure for LPD algorithms. We used two methods for our LPD system, cascade AdaBoost and adaptive thresholding. Our proposed system is separated into two stages, offline training and online detection, which make our proposed system extremely simple and effective for LPD. In this paper, we demonstrated that such simplicity and effectiveness allow our method to provide better performance than other existing methods. Most of the existing techniques are tremendously complex and are not suitable for real-time applications; however, our proposed algorithm is not complex, thus rendering it suitable for real-time applications.

Table 3: Performance comparison of some typical ALPR systems for LPD.

Methods	Main procedures for license plate detection	Database size	Image conditions	LPD Rate	Processing time	Real time	Plate format
[10]	Sliding concentric windows, histogram	40 images	640 × 480 pixels (Different distances and weather, road)	82.5%	—	—	Korean plates
[11]	Vertical edge, edge filtering, and morphological operation	350 images	Different distances and weather and road	95.2%	—	—	Iranian plates
[12]	Vertical edge detection, unwanted line elimination	664 images	640 × 480 pixels (various weather conditions, road)	91.65%	47.7 ms	Yes	Malaysian plates
[13]	Scan line, texture properties, color, and Hough transform	332 images	867 × 623 pixels (various illumination and different distances and road)	97.1%	0.53 s	No	Taiwanese plates
Our proposed method	Cascade AdaBoost algorithm and adaptive thresholding	1800 images	1280 × 720 pixels, various weather conditions and different illumination	98.38%	49 ms	Yes	Korean Plates

Using our proposed method, experimental results show that the test accuracy is 98.38% with a computational time of 49 ms, which is significantly faster than other existing methods. In regard to our proposed method, practicing and improving its accuracy and practicality are considerations for future work. Moreover, LP character recognition is our principal future work.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This paper was supported by Wonkwang University in 2013.

References

[1] H. Bai and C. Liu, "A hybrid license plate extraction method based on edge statistics and morphology," in *Proceedings of the 17th International Conference on Pattern Recognition (ICPR '04)*, vol. 2, pp. 831–834, August 2004.

[2] S. K. Kim, D. W. Kim, and H. J. Kim, "A recognition of vehicle license plate using a genetic algorithm based segmentation," in *IEEE International Conference of Image Processing (ICIP '96)*, vol. 2, pp. 661–664, 1996.

[3] S. Kim, D. Kim, Y. Ryu, and G. Kim, "A robust license-plate extraction method under complex image conditions," *International Conference on Pattern Recognition*, vol. 3, pp. 216–219, 2002.

[4] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04)*, vol. 2, pp. II366–II373, June-July 2004.

[5] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[6] R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2011.

[7] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*, Thomson Learning, 2008.

[8] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, vol. 1, pp. I-511–I-518, 2001.

[9] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in *IEEE International Conference on Image Processing (ICIP '02)*, vol. 1, pp. 900–903, 2002.

[10] K. Deb, H. Chae, and K. Jo, "Vehicle license plate detection method based on sliding concentric windows and histogram," *Journal of Computers*, vol. 4, no. 8, pp. 771–777, 2009.

[11] M. Ashoori-Lalimi and S. Ghofrani, "An efficient method for vehicle license plate detection in complex scenes," *Circuits and Systems*, vol. 2, no. 4, pp. 320–325, 2011.

[12] A. M. Al-Ghaili, S. Mashohor, A. R. Ramli, and A. Ismail, "Vertical edges-based car license plate detection method," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 1, pp. 26–38, 2012.

[13] J. M. Guo and Y. F. Liu, "License plate localization and character segmentation with feedback self-learning and hybrid binarization techniques," *IEEE Transactions on Vehicular Technology*, vol. 57, no. 3, pp. 1417–1424, 2008.