Traffic Sign Recognition in Disturbing Environments

Hsiu-Ming Yang, Chao-Lin Liu, Kun-Hao Liu, and Shang-Ming Huang

Department of Computer Science, National Chengchi University
Taipei 11605, Taiwan
chaolin@cs.nccu.edu.tw

Abstract. Traffic sign recognition is a difficult task if we aim at detecting and recognizing signs in images captured from unfavorable environments. Complex background, weather, shadow, and other lighting-related problems may make it difficult to detect and recognize signs in the rural as well as the urban areas. We employ discrete cosine transform and singular value decomposition for extracting features that defy external disturbances, and compare different designs of detection and classification systems for the task. Experimental results show that our pilot systems offer satisfactory performance when tested with very challenging data.

1 Introduction

Computer vision has been applied to a wide variety of applications for intelligent transportations systems. For instance, researchers have developed vision-based techniques for traffic monitoring, traffic-related parameter estimation, driver monitoring, and intelligent vehicles, etc. [1]. Traffic sign recognition (TSR) is an important basic function of intelligent vehicles [7], and TSR problems have attracted attention of many research groups since more than ten years ago [16]. In this paper, we report experiences in detection and recognition of traffic signs when images of the traffic signs are severely disturbed by external factors.

Detection and recognition are two major steps for determining types of traffic signs [6]. Detection refers to the task of locating the traffic signs in given images. It is common to call the region in a given image that potentially contains the image of a traffic sign the region of interests (ROI). Taking advantages of the special characteristics of traffic signs, TSR systems typically rely on the color and geometric information in the images to detect the ROIs. Hence, color segmentation is common to most TSR systems, so are edge detection [9,14] and corner detection techniques [4].

After identifying the ROIs, we extract features of the ROIs, and classify the ROIs using the extracted feature values. Researchers have explored several techniques for classifying the ideographs, including artificial neural networks (ANNs) [4], template matching [14], chain code [15], and matching pursuit methods [9].

Detection and recognition of traffic signs become very challenging in a noisy environment [14]. Traffic signs may be physically rotated or damaged for different reasons. View angles from the car-mounted cameras to the traffic signs may lead to artificially rotated and distorted images. External objects, such as tree leaves, may occlude the traffic signs, and background conditions may make it difficult to detect traffic signs. Bad weather conditions may have a detrimental effect on the quality of the images.

To confront these challenges, researchers have designed techniques for raising the quality of the recognition results. Sandoval et al. develop methods for generating convolution masks that are then used for position-dependent edge detection of circular signs [17]. Kehtarnavaz and Ahmad apply Fourier and log-polar-exponential grid transformations for extracting invariant feature values of the traffic signs [11]. Piccioli et al. focus more on detection of traffic signs in cluttered background. Assuming constant orientation of images of the detected signs, they apply template-matching methods to pick candidate signs, and claim 98% of correct classification [14].

We consider that occluded and poor-quality images of traffic signs are not uncommon in reality. Images in Fig. 1 and the Appendix illustrate some of these challenging scenarios. As a result, we believe that templates alone will not always work perfectly for traffic sign recognition. Even if images of the traffic signs are partially occluded by objects or interfered by shadow, human may guess what the signs are using the limited available information, so we charge ourselves with the task of detection and recognition of 45 triangular signs in challenging scenes. Besides some selected raw image data, we employ discrete cosine transform [5] and singular value decomposition methods [2] to acquire some invariant features of the traffic signs as features, and apply these features in different ways. Similar to many other researchers, we use the extracted feature values as input to ANNs for classifying the captured images. We also test the integration of ANNs and Naïve Bayes models, and examine the applicability of *k* nearest neighbor methods. As we will explain, different setups of our systems recognize around 70% of the test signs.

Fig. 1. Selected "hard" traffic signs. The left sign did not face the camera directly, and had a red background. The middle picture was taken in the dusk. The signs in the rightmost image were in the shadow.

Section 2 provides details of our approaches to traffic sign detection and recognition. Section 3 presents empirical results of our recognition systems. It turned out that our systems detected and recognized signs in the leftmost and the rightmost images in Fig. 1. We wrap up this paper with discussions in Section 4.

2 Traffic Sign Detection and Recognition

2.1 ROI Detection

Transportation engineers design traffic signs such that people can recognize them easily by using distinct colors and shapes for the signs. Similar to many other countries, Taiwan uses triangles and circles for signs that carry warning and forbidding messages, respectively. These signs have thick and red borders for visibility from apart. Hence, we may use color and shape information for detecting traffic signs.

Color Segmentation

Identifying what pixels of the images are red is a special instance of the *color segmentation* problems. This task is not easy because images captured by cameras are affected by a variety of factors, and the "red" pixels as perceived by human may not be encoded by the same pixel values all the time.

Assuming no directly blocking objects, lighting conditions affect the quality of the color information the most. Weather conditions certainly are the most influential

factor. Nearby buildings or objects, such as trees, may also affect quality of the color information because of their shadows. It is easy to obtain very dark images, e.g., the middle image in Fig. 1, when we are driving in the direction of the sun.

As a consequence, "red" pixels can be embodied in a range of values. Hence, we attempt to define the range for the red color. We convert the original image to a new image using a pre-selected formula. Let R_i, G_i, and B_i be the red, green, and blue component of a given pixel in the original image. We encode the pixels of the new image by R_o, G_o, and B_o. Based on results of a few experiments, we found that the following conversion most effective: $R_o = \max(0, (R_i - G_i) + (R_i - B_i))$, $G_o = 0$, and $B_o = 0$. After the color segmentation step, only pixels whose original red components dominate the other two components can have a nonzero red component in the new image most of the time.

Region of Interests

We then group the red pixels into separate objects, apply the *Laplacian of Gaussian* (LoG) edge detector [8] to this new image, and use the 8-connected neighborhood principle for determining what pixels constitute a connected object. We consider any red pixels that are among the 8 immediate neighbors of another red pixel *connected*.

After grouping the red pixels, we screen the object based on four features to determine what objects may contain traffic signs. These features are areas, height to width ratios, positions, and detected corners of the objects.

According to the government's decrees for traffic sign designs, all traffic signs must have standard sizes. Using our camera, which is set at a selected resolution, to take pictures of warning signs from 100 meter apart, the captured image will occupy 5x4 pixels. Due to this observation, we ignore objects that contain less than 40 red pixels. We choose to use this threshold because it provided a good balance between recall and precision, defined in (3) in Section 3.2, when we applied the *Detection* procedure to the training data. Two other reasons support our ignoring these small objects. Even if the discarded objects were traffic signs, it would be very difficult to recognize them correctly. Moreover, if they are really traffic signs that are important to our journey, they would get closer and become bigger, and will be detected shortly.

The decrees also allow us to use shapes of the bounding boxes of the objects to filter the objects. Traffic signs have specific shapes, so heights and widths of their bounding boxes must also have special ratios. The ratios may be distorted due to such reasons as damaged signs and viewing angles. Nevertheless, we can still use an interval of ratios for determining whether objects contain traffic signs.

Positions of the objects in the captured images play a similar role as the decrees. Except driving on rolling hills, we normally see traffic signs above a certain horizon. Due to this physical constraint and the fact that there are no rolling hills in Taiwan, we assume that images of traffic signs must appear in a certain area in the captured image, and use this constraint for filtering objects in images.

Since we focus on triangular signs in our experiments, we can rely on features specific to triangles for determining whether the objects are likely to contain triangular signs. We divide the bounding boxes of the objects into

Fig. 2. Using corners for identifying triangular borders

nine equal regions, and check whether we can detect corners in selected regions. The leftmost image in Figure 2 illustrates one of these patterns by the blue checks. More patterns are specified in the following *Detection* procedure. If none of the patterns is satisfied, chances are very low that the object could contain a triangular sign. Using this principle, we were able to detect the rightmost four signs in Fig. 2.

Procedure *Detection* (Input: an image of 640x480 pixels; Output: an ROI object list)
Steps:

1 Color segmentation
2 Detect edges with the LoG edge detector.
3 Remove objects with less than 40 red pixels.
4 Mark the bounding boxes of the objects.
5 Remove objects whose highest red pixel locates below row 310 of the original images, setting the origin (0,0) of the coordinate system to the upper-left corner.
6 Remove objects with height/width ratios not in the range [0.7, 1.3].
7 Check existence of the corners of each object.
  7.1 Find the red pixel with the smallest row number. When there are many such pixels, choose the pixel with the smallest column number.
  7.2 Find the red pixels with the smallest and the largest column numbers. If there are multiple choices, choose those with the largest row numbers.
  7.3 Mark locations of these three pixels in the imaginary nine equal regions, setting their corresponding containing regions by 1.
  7.4 Remove the object if these pixels do not form any of the patterns listed aside.

```
010 010 010 100 001
100 001 000 000 000
001 100 101 101 101
```

8 For each surviving bounding box, extract the corresponding rectangular area from the original image and save it into the ROI list.

Fig. 3 illustrates how we detect a triangular sign with the *Detection* procedure. Notice that the sing in (f) is not exactly upright. The tree trunks and red sign behind the sign made our algorithm unable to extract the complete red border. All objects detected by *Detection* are very likely to contain a triangular traffic sign. They will be used as input to the recognition component after the preprocessing step.

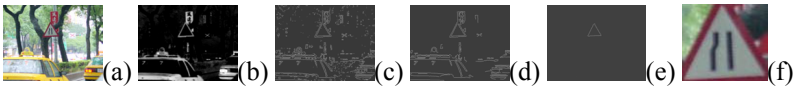


Fig. 3. An illustration of detection steps: (a) the original image; (b) result of color segmentation; (c) result of edge detection; (d) result of removing small objects; (e) results of filtering objects by step 7; (f) the enlarged image of the detected sign

2.2 Preprocessing

Procedure *Preprocessing* (Input: an ROI object list; Output: an object list)
Steps:
For each object in the ROI list, do the following:

1 Normalize the object to the standard size 80x70.
2 Extract the rectangle of 32x30

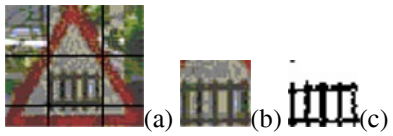


Fig. 4. An illustration of preprocessing steps

pixels from (25,30).

3 Remove remaining red pixels.

4 Convert the object to a gray-level image.

As the first step of the preprocessing, we normalize all objects to the 80x70 standard size. After a simple analysis of the 45 standard triangular signs, we found that the ideographs appear in a specific region in the normalized images. As shown in Fig. 4(a), we can extract the ideographs from a particular rectangular area in the image. We extract the ideograph from a pre-selected area of 32x30 pixels from the normalized image. The coordinates of the upper left corner of the extracted rectangle is (25, 30). Notice that, although we have attempted to choose the rectangular area such that it may accommodate distorted and rotated signs, the extracted image may not include all the original ideographs all the time. Fig. 4(b) shows that the bottom of the ideograph was truncated. Similarly, the extracted area may contain noisy information.

After extracting the rectangular area that might contain the ideograph, we remove red pixels in the extract. We use a more stringent standard for defining "red." Let *R*, *G*, and *B* be the red, green, and blue component of a pixel. A pixel is red if $R>20$, $(R-B)>20$, and $(R-G)>20$.

After removing the red pixels, we convert the result into a gray-level image. We adjust pixel values based on the average luminance to increase contrast of the image. We compute the YIQ values [18] of each pixel from its RGB values, set their gray levels to their luminance values, and compute the average gray levels of all pixels. Let the average be α. We invert the colors of the pixels by deducting the amount of $(\alpha-100)$ from the gray levels of all pixels.

Then, pixels whose remaining gray levels are smaller than 70 are set to 0, and others are set to 255. However, if using 70 as the threshold gives us less than 10 pixels with value 255 or 10 pixels with value 0, we apply another slightly more complex method. We calculate the average gray level of the pixel values, and use this average, λ, as the cutting point for assigning pixel values in the gray-level image. Pixels whose gray levels are less than λ are set to 0, and others are set to 255. Fig. 4(c) shows such a gray-level image.

2.3 Traffic Sign Recognition

After the preprocessing procedure, each object becomes a rectangle of 32x30 pixels. We can use these raw data as features for recognition. In addition, we employ the discrete cosine transform (DCT) and the singular value decomposition (SVD) procedures for extracting the invariant features of the ideographs.

DCT is one of the popular methods for decomposing a signal to a sequence of components and for coding images [5]. We concatenate rows of a given object, generated at step 5 in *Preprocessing*, into a chain, and apply the one-dimension DCT over the chain, and use the first 105 coefficients as the feature values.

We apply singular value decomposition to the matrices of the objects that are obtained at step 4 in the *Preprocessing* procedure for extracting features of the objects. Let $U\Sigma V^T$ be the singular value decomposition [2] of the matrix that encodes a given object. We employ the diagonal values in Σ as the feature values of the given object. Since the original matrix is 32x30, we obtain 30 feature values from Σ.

We investigate the effectiveness of artificial neural networks, *k*-nearest-neighbor

(*k*NN) models, and naïve Bayes models for the recognition task. All these three alternatives are commonly used for pattern recognition tasks [3]. We use the collected features as input to different combinations of these techniques, test the integrated systems with test data, and report their performance in the next section.

3 Empirical results

3.1 Data Source

We trained our classifiers with feature values of perfect signs, real-world signs, and artificially created signs. We created more than 10000 imperfect signs using perfect images of the 45 triangular signs. We gradually rotated the signs both clockwise and counterclockwise by the amount of one degree. The first row of Fig. 5 shows ten rotated signs. We added Gaussian noise of varying means and variances to the perfect signs. We created signs in the middle row by adding Gaussian noise of increasing means, and the leftmost five signs in the last row by adding Gaussian noise of increasing variances. We also created training patterns by shifting the perfect signs to their north, east, south, and west. The rightmost five signs in the bottom illustrate such operations.

Fig. 5. Sample training patterns

We used the *Preprocessing* procedure to process our training data. As described in Section 2.3, for each pattern, we obtained the values of the 32x30 pixels, the 105 DCT coefficients, and the SVD vector of 30 components. (For easy reference, we refer to the vectors formed by the diagonal of the Σ matrix by *SVD vectors* henceforth.)

3.2 Experiments and Results

We tested our systems with 210 signs in pictures that were not used in the training phase, and all of these pictures are shown in the Appendix. The pictures were taken at the 640x480 resolution with Nikon Coolpix 775 digital camera. We report results of five sets of experiments. For each experiment, we measured the performance by standard definitions of *precision* (*P*), *recall* (*R*), and *F* [12]. Precision is the portion of total number of correctly classified signs in the total number of classified objects. Recall is the portion of total number of correctly classified signs in the total number of real signs in the testing images. *F* measure is a function of *P* and *R*, and $F(\beta) = ((\beta^2 + 1)PR)/(\beta^2 P + R)$. We set β to 1 because we considered precision and recall are equally important for TSR.

Basic Methods

In the first experiment, we used the pixel values and their DCT coefficients as input to an ANN for recognition. The ANN had 1065 input units, each for the 32x30 pixel values and the 105 DCT coefficients. There were 400 hidden units and 45 output units. The values of the output units indicated the likelihood of the detected object being the sign represented by the unit.

In the second experiment, we rely on the SVD vectors of the objects for classification. We computed the Euclidean distances of SVD vector of the object to those of the training patterns, and applied the *k*NN principle for classification. The closest 100 neighbors voted for the class of the test object.

Integrated Methods

We combine the original pixel values, DCT coefficients, and SVD vectors for recognition in three ways. Here, we reused the ANN in the first experiment, but used SVD vectors in a slightly different way. We normalized each SVD vector by dividing its components by the largest component before we computed distances. This was partially due to the fact the outputs of our ANN were normally below 2, while values of components of original SVD vectors fell in a much wider range. We normalized the values in SVD vectors to balance the influences of the ANN and SVD vectors.

The first integration of ANN and SVD assigned equal weights to the scores given by ANN and SVD. We separately normalized ANN scores and the SVD distances to the range [0,1]. Let $Score_A(sign)$ be the likelihood of the object being the *sign* determined by output units of the ANN, and $Score_S(sign)$ be the distances between the SVD vector of the test object and the SVD vector of the perfect image of *sign*. The possibility of the test object being a particular sign is determined by the following score function. The subtraction was due to the fact that we computed distances with SVD vectors, and large values suggest different signs.

$$Score(sign) = Score_A(sign) - Score_S(sign)$$

We may also assign different weights to the scores given by ANN and SVD. We determined the weights based on the classification performances of setups used in the first and the second experiments over the training data.

We used the ANN for the first experiment and the SVD vectors for the second experiment to classify the training data, and collected their *F* measures for individual sign classes. These *F* measures reflected how well the ANN-based and SVD-based classifiers performed on the training data, so we used this information to weight their predictions for the test data. Let F_A and F_S be the *F* measures so collected for the ANN and SVD, respectively. In this experiment, we computed the scores of a sign class, s_j, using the following formula to choose the best candidate solutions.

$$Score(s_j) = F_A(s_j) \cdot Score_A(s_j) + F_S(s_j) \cdot (2 - Score_S(s_j))$$

The last experiment employed the components of the SVD vectors as features in a Naïve Bayes model. We assumed that the distributions of the feature values are mutually independent and normally distributed given the signs. Let v_i be the i^{th} component in the SVD vector. As shown below, we used the training data to estimate the means, u_{ij}, and variances, σ_{ij}^2, of the distributions of v_i given sign s_j.

$$f_{ij}(v_i|s_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} \exp\left(-(v_i - \mu_{ij})^2 / (2\sigma_{ij}^2)\right)$$

Because each SVD vector contained 30 components, and we had 45 triangular signs, we would have to estimate 2700 parameters for1350 distributions. Since we had only slightly more than 10000 training data, which did not appear to be sufficient for estimating 2700 parameters, we chose to use only the largest 10 components in the SVD vectors for this experiment, so we had to estimate only 900 parameters for

450 normal distributions.

To apply the naïve Bayes model, we needed to know the prior distribution over the 45 signs. Normally, this information would be trained with the frequencies of the signs in the training data. This standard method could not work for us. We could not guarantee that the frequencies of signs that were collected in our training set reflected the relative likelihood of coming across the signs. As an alternative, we employed the output of the ANN component as the surrogate for the prior distribution. As just mentioned, the ANN component gave a score for each of the different targeted signs. We mapped the ANN scores to the interval [0,1], and used the results as $\Pr(s_j)$.

Let v_k be the k^{th} largest component in the SVD vector of the object being recognized. At the recognition phase, we computed the following score, and assigned the object the sign s_j that maximized the following score.

$$Score(s_j) = \Pr(s_j)\prod_{k=1}^{10} f_{kj}(v_k \mid s_j) \quad (1)$$

In all of these experiments, we used the absolute values of the ANN scores and SVD distances as extra filters to determine what objects were very unlikely to contain a triangular sign. The current object would not be considered as a traffic sign if any of the following conditions satisfied.

1. if the SVD distance between the current object to all perfect signs is greater than 5
2. if the highest ANN score of the current object is less than 0.3
3. if all the following conditions hold: the highest ANN score is less than 1.5 times the second highest ANN score; the second highest ANN score is less than 2 times the third highest ANN score; the third highest ANN score is less than 3 times the fourth highest ANN score

Results

Using the output of the *Detection* procedure as the gauging point for computing the precision and recall of our detection component, we achieved 93% in recall, and 78% in precision. We deliberatively allowed higher recall and lower precision at the detection phase because it offered better prospect of higher recognition rate of the overall system.

Table 1. Experimental results of classifying objects into 45 triangular signs

Classifiers	Single Candidate			Single Candidate		
	Precision	**Recall**	**F**	**Precision**	**Recall**	**F**
EXP1: ANN	63%	71%	67%	68%	78%	73%
EXP2: kNN	28%	40%	34%	36%	50%	43%
EXP3: ANN+SVD	66%	76%	71%	72%	83%	78%
EXP4: ANN+SVD+F	62%	73%	68%	72%	83%	78%
EXP5: ANN+SVD+NB	45%	54%	50%	56%	65%	61%

Table 1 shows the recognition rates of our experiments over the test data that are publicized in the Appendix. The **Single Candidate** column shows the performance measures when our recognizer returned only the most possible traffic sign as the answer. The **Three Candidates** column shows the performance measures when we allowed our recognizer to return the three most possible signs.

Although SVD distances alone did not perform very well, according to results of the second experiment, they worked quite well with the ANN scores. The direct inte-

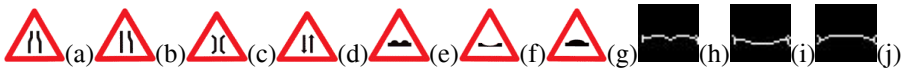
gration of ANN scores and SVD distances turned out to be the best classifier, followed by their integration using the F measures as the weights. Integrating ANN scores and SVD distances via the naïve Bayes model did not perform as well as we expected. After analyzing the errors, we found that using the product formula in (1) allowed SVD-based features to dominate the classification decisions, which is not a very good design as already suggested by results of the second experiment. In contrast, the third and the fourth experiments gave more balanced weights to ANN scores and SVD distances, and achieved better performances.

In Exp4, which provides the best performance, the average time spent on detection was 4.05 seconds for each input image. The standard deviation was 0.501 seconds. The average time spent on recognition for each ROI was 0.168 seconds, and the standard deviation was 0.0146 seconds. The timing statistics were collected when we ran our system on a 1 GHz Pentium III CPU with 256 Mbyte SDRAM, using interpreted Matlab programs.

4 Discussions

Although we have discussed designs of our experiments using the RGB color system, we did have tried the HSI system [18]. The HSI system may be more resilient to the disturbance caused by lighting problems, but did not improve performances of our systems significantly.

The problems we are tackling are more difficult than we thought. Consider the following signs. Telling signs (a) through (d) apart in a noisy environment can be a difficult job even for human eyes. Signs (e) through (g) form another confusing group. We found that, in many cases, drivers could guess types of the signs by contextual information while driving.


(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

It has been tempting to us to apply low-level information, such as morphological information [10], for recognition. One may compute the skeleton of the ideographs for sign recognition. Although we did not report results of our effort on this front, our experiences indicated that skeletons alone might not be very fruitful. For instance, skeletons of ideographs in (e) through (g), shown in (h) through (j), may look very similar when we take their images in noisy environments.

Nevertheless, the extremely similar signs in these sample images strongly suggest the necessity of low-level information for high quality of recognition. A robust template-based matching can be very useful if integrated with an active vision system [13]. If we use one camera to search for ROIs that may contain traffic signs in the viewing area, and use another camera to zoom in the ROIs for clearer images of the candidate areas, we may apply template-based matching for high performance systems. However, we are not sure if it is feasible to install two cameras on passenger cars while maintaining the affordability.

The applicability of our current system is quite limited by the facts that it employs several human selected parameters and rules. Many of these settings are not well supported by any theories, but were set to their current values based on limited experiments over the training data. Improving the recognition rate and the generalizability of our methods requires a lot of more future work.

Acknowledgments

We thank anonymous reviewers for the invaluable comments on the original manuscript. Although we cannot follow all the comments partially due to page limits, we will do so in an expanded version of this paper. We also thank Mr. Chang-Ming Hsieh of the Taichung Bureau of Transportation for his providing perfect images of the traffic signs. This project was supported in part by Grant NSC-91-2213-E-004-013 from the National Science Council of Taiwan.

References

We use ITS and IV for *Intelligent Transportation Systems* and *Intelligent Vehicles*, respectively.

1. *Proc. of the IEEE 5th Int'l Conf. on ITS*, 2002.
2. Ž. Devčić and S. Lončarić, SVD block processing for non-linear image noise filtering, *J. of Computing and Information Technology*, **7**(3), 255–259, 1999.
3. R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, John-Wiley & Sons, 2001.
4. A. de la Escalera, L. E. Moreno, M. A. Salichs, and J. M. Armingol, Road traffic sign detection and classification, *IEEE Trans. on Industrial Electronics*, **44**(6), 848–859, 1997.
5. O. Egger, P. Fleury, T. Ebrahimi, and M. Kunt, High-performance compression of visual information: A tutorial review, Part I: still pictures, *Proc. of the IEEE*, **87**(6), 976–1011, 1999.
6. D. M. Gavrila. Traffic sign recognition revisited, *Proc. of the 21st DAGM Symp. für Mustererkennung*, 86–93, 1999.
7. D. M. Gavrila, U. Franke, C. Wöhler, and S. Görzig, Real-time vision for intelligent vehicles, *IEEE Instrumentation & Measurement Magazine*, **4**(2), 22–27, 2001.
8. R. Haralick and L. Shapiro, *Computer and Robot Vision*, Vol. 1, 346–351, Addison-Wesley, 1992.
9. S.-H. Hsu and C.-L. Huang, Road sign detection and recognition using matching pursuit method, *Image and Vision Computing*, **19**(3), 119–129, 2001.
10. G. Y. Jiang, T. Y. Choi, and Y. Zheng, Morphological traffic sign recognitions, *Proc. of the 3rd Int'l Conf. on Signal Processing*, 531-534, 1996.
11. N. Kehtarnavaz and A. Ahmad, Traffic sign recognition in noisy outdoor scenes, *Proc. of the IEEE IV 1995 Symp.*, 460–465, 1995.
12. C. D. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1999.
13. J. Miura, T. Kanda, and Y. Shirai, An active vision system for real-time traffic sign recognition, *Proc. of the IEEE 3rd Int'l Conf. on ITS*, 52–57, 2000.
14. G. Piccioli, E. De Micheli, P. Parodi, and M. Campani, A robust method for road sign detection and recognition, *Image and Vision Computing*, **14**(3), 209–223, 1996.
15. L. Priese, R. Lakmann, and V. Rehrmann, Ideograph identification in a realtime traffic sign recognition system, *Proc. of the IEEE IV 1995 Symp.*, 310–314, 1995.
16. W. Ritter, Traffic sign recognition in color image sequences, *Proc. of the IEEE IV 1992 Symp.*, 12–17, 1992.
17. H. Sandoval, T. Hattori, S. Kitagawa, and Y. Chigusa, Angle-dependent edge detection for traffic signs recognition, *Proc. of the IEEE IV 2000 Symp.*, 308–313, 2000.
18. M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis, and Machine Vision*, PWS Publishing, 1999.

Appendix **http://www.cs.nccu.edu.tw/~chaolin/papers/ismis03/testdata.html**

(Please contact the author should the link became inaccessible in the future.)