Real-time Detection and Recognition of Traffic Signs

A. Martinović, G. Glavaš, M. Juribašić, D. Sutić and Z. Kalafatić
Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb
E-mail: {andelo.martinovic, goran.glavas, matko.juribasic, davor.sutic, zoran.kalafatic}@fer.hr

Abstract - Automated recognition of traffic signs is becoming a very interesting area in computer vision with clear possibilities of its application in automotive industry. For example, it would be possible to design a system which could recognize the current speed limit on the road and notify the driver in an appropriate manner. In this paper we deal with methods for automated localization of certain traffic signs, and classification of those signs according to the official designations. We propose two different approaches of determining the current speed limit after the sign was localized. A demo software system was developed to demonstrate the presented methods. Finally, we compare results obtained from the developed software, and discuss the influence of different parameters on recognition performance and quality.

I. INTRODUCTION

Traffic sign detection and recognition has found its application in many driver assistance systems, which aim to display helpful information to the driver using knowledge about the current conditions on the road. A complete system should have three distinct functions:

1. detection of a traffic sign in an image;
2. classification of the detected sign;
3. sign tracking through time.

A complete traffic sign detection and recognition system should be able to recognize all of the traffic signs used in Croatia. Croatian regulations define five sign classes: warning signs, explicit order signs, information signs, direction signs and supplemental panels [10]. Due to the limited annotated image database of traffic signs, we focused our efforts on detecting and classifying only a subset of explicit order signs. After a detailed analysis of image and video databases at our disposal, we determined that the five most common traffic signs in this category are those shown in Fig. 1.

Fig. 1. Five common signs in category of explicit order traffic signs

Since all of the signs from Fig. 1. are similar in appearance (red circles containing black or red symbols), our detection algorithm is trained to detect only circular traffic signs, while an additional classification stage is needed to separate these signs from each other.

Additionally, there is one traffic sign of specific importance: the speed limit sign (rightmost on Fig. 1). The system should also be able to determine the exact speed limit, if the corresponding sign is classified as such.

This work is organized as follows. In Section 2 we mention different approaches by various researchers. In Section 3 we briefly describe an algorithm used to detect sign in an image. In Section 4 we present two algorithms used to recognize the detected sign. Section 5 contains details about how to combine results from individual frames to achieve detection and tracking in a video. Results are illustrated and discussed in Section 6. We conclude this article in Section 7.

II. RELATED WORK

The approaches to sign detection vary in use of color and geometric information. Various color-based approaches use RGB or other color models (i.e. HSV, L*a*b, CIECAM97, etc.). Intensity decoupling color schemes [1,5] are preferred because of diverse lightning conditions usually encountered in real life applications. Some authors use simple thresholding [1,2], while other use clustering methods [3] or recursive region splitting [4,5,6].

Geometric information can be extracted with Hough transform [6,7,15], histogram of orientation vectors [5,13] or template matching [16]. Standard classifiers like SVM can be used with these geometric features [13]. A large body of work is based on the Viola-Jones detector proposed in [8]. This approach has been used in [9,10,11,12]. Most Viola-Jones based implementations extract shape information in grayscale images, while [9] uses color based Haar features. Neural networks are used for detection in [14].

Unlike some of the related work, which considers static images [5,6,7], our system works on video sequences in real time (over 20 fps) on a mainstream CPU (~2GHz).

III. SIGN DETECTION

Detecting an object in an image is a computer vision problem for which a wide variety of algorithms exist. Because we wanted our system to work in real-time, we have decided to employ the Viola-Jones detection algorithm.

A. Viola-Jones algorithm in traffic signs detection

The Viola-Jones detector works by sliding a detection window across an image. At each position, the classifier makes the decision if there is a desired object inside the window. In the vast majority of window's positions, the object is not found. The number of classifications for an image is equal to the number of windows positions, which can be in the order of 10^5 or 10^6. This is why the classification itself has to be as fast as possible.

Viola-Jones algorithm is based on a cascade of boosted Haar's features. More on the Haar's features can be found in the original paper [8]. Boosting is done through AdaBoost, a machine learning algorithm which combines weak classifiers built on Haar's features into a strong classifier. Given enough different weak classifiers, AdaBoost will produce a strong classifier with arbitrary precision. A theoretic proof and a good introduction to boosting can be found in a tutorial from AdaBoost's creators, Freund and Schapire [17]. The decision of whether the object is detected is made through voting of weak classifiers, each according to its weights. Cascading classifiers speeds up this process significantly, because more important weak classifiers get to vote first: if their decision is negative, the image is rejected and other less important classifiers do not vote at all. Object is classified positively only if it successfully passes through the cascade, positively classified in each stage. Final classifier works in real time because:

- Haar's features take constant time to calculate from integral image;
- in a classifier produced by AdaBoost, voting is done as a summation of weighted classifiers;
- on average, only a small subset of classifiers votes every time because of the cascading.

An analysis of the variants in the training process can be found in a paper by Lienhart, Kuranov and Pisarevsky [18].

B. Viola-Jones training

To train the classifier, we used 757 images of traffic signs. Each positive image contained only a cropped traffic sign normalized to the size of 24x24. 3000 images were used as negatives. We trained the cascade with OpenCV, which is an open source library of computer vision functions. We used it with the following parameters:

- Minimum hit rate of 0.995 per stage. Only one in 200 positive images is falsely rejected in every stage, the others are positively classified.
- Maximum false positives of 0.4 per stage. This allows that up to 40% of the images positively classified are false positives.
- Number of stages was set to 20.

With these parameters the theoretic hit rate is expected to be more than $0.995^{20} \approx 0{,}9$ with outmost $0.4^{20} \approx 10^{-8}$ false positives.

Training of Viola-Jones detector took approximately 16 hours on a 4 CPU computer, with OpenMP enabled. The training procedure stopped after reaching the desired number of stages.

The trained cascade was afterwards tested with a test set of 286 images. Images from the test set were taken with a different camera and under various lightning conditions. Results are shown in Table 1. The number of false positives is expressed related to the number of signs in the test set.

TABLE I
Experimental results for trained Viola-Jones detector

Scale factor	Hits	Misses	False positives
1.3	61.53%	38.46%	11.88%
1.2	67.13%	32.86%	18.88%
1.1	75.17%	24.83%	28.67%

Viola-Jones detector works by sliding a detection window across an image, and enlarging that window by a scale factor after the end of the image is found. Therefore, modifying the scale factor affects the detection quality and speed. By reducing its size, we increase the possibility a sign will be detected, but we also increase the time needed for the algorithm to finish. In our work we used the scale factor of 1.1 because it provided the best hit ratio with an acceptable frame rate (over 20 fps). False positives are expected to be removed by later stages of sign recognition.

Further analysis of the results revealed that most unsuccessful detections are caused by signs which are smaller than the samples used to train the cascade (24x24 pixels). This behavior is not unexpected; its impact diminishes when the algorithm is used on a video sequence because traffic signs grow in virtual size as the sequence progresses. In a certain moment, it will become large enough to be detected.

IV. SIGN RECOGNITION

After a sign was successfully detected in an image, the classification process begins, as to determine the type of the sign. The classifier expects an adequate input vector, which must firstly be prepared by means of image preprocessing.

A. Sign preprocessing

The resulting sign from the detection stage can have arbitrary size. In order to correctly classify the sign, a size normalization is required. In our work, we used a standard sign size of 10x10 pixels. Image resizing procedure is implemented using bilinear interpolation algorithm. A clipping operation is also required, in such a manner that only the central part of the sign remains, which holds useful information. Fig. 2 shows results from two different interpolation algorithms.

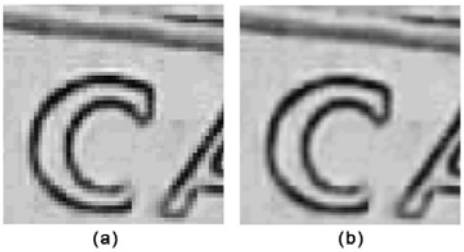

Fig. 2. (a) nearest neighbor interpolation, (b) bilinear filtering

After size normalization, color information is discarded. Conversion from 24-bit RGB space to grayscale image is conducted by ITU CCR 601 standard. The resulting grayscale image has to be transformed into a binary image (image with pixel intensities 0 and 1) using a thresholding algorithm. All pixels with intensities over a defined threshold are assigned with value 1, and the remaining pixels become 0. An iterative threshold selection method [22] is used. This method produces very good results when used on images where objects of interest are evenly illuminated (which is the case with most traffic signs). In the case of non-uniform illumination, it is advisable to use one of the adaptive thresholding methods [23].

As a result of the segmentation procedure we get a binary image with dimensions of 10*10 pixels. Input vector for the classifier is formed by taking the binary image as a one-dimensional vector with 100 elements, with one slight modification. This modification replaces values of 0 with values of -1, to improve the neural network performance by distributing values equally around zero.

B. Classification using neural networks

There are many types of neural networks (e.g. feed-forward networks, radial-basis networks, recursive networks) and possible applications for them (e.g. pattern recognition, function interpolation). It has been shown that multilayer perceptron networks with a single hidden layer and a nonlinear activation function are universal classifiers [19, 20]. Therefore, in our work we have chosen a multilayer perceptron (MLP) with back propagation (BP) training for classification.

For the purposes of this project we have developed our own software library for MLP trained according to BP algorithm. The library provides support for creating and training arbitrary MLP (arbitrary number of hidden layers with arbitrary number of units in each layer) with the sigmoidal activation function. Using the capabilities of the developed MLP library we have created and trained two different multilayer perceptrons.

- The purpose of the first network was to classify a given traffic sign (input vector) into one of five categories (as shown in Fig. 1).
- The task of the second MLP was to recognize the actual speed limit if the first network classified the input vector into speed limit category. Hence, the second network was trained to recognize decimal digits (0-9).

In the case of MLP there are always several network parameters left to be determined experimentally (the number of units in hidden layers, learning factor for weight correction, etc.) [21]. To determine the optimal parameters, we trained the MLP observing the performance on the validation set to avoid overfitting.

Table II shows optimal parameters for both multilayer perceptrons (the values for number of units in hidden layer, learning factor for weight correction, maximal number of epochs and satisfactory average epoch error were obtained empirically).

TABLE II
Neuron numbers per layer

	Input layer	**Hidden layer**	**Output layer**
General MLP	100 (10x10 pixels)	10	5
Speed limit MLP	72 (6x12 pixels)	10	10

TABLE III
Weight correction learning factors, maximum number of epochs and average errors

	Learning factor (ŋ)	**Maximum number of epochs**	**Satisfactory average epoch error**
General MLP	0.1 initially, 0.05 when error drops below 0.1	10 000	0.01
Speed limit MLP	0.01 initially, 0.005 when error drops below 0.1	50 000	0.001

Input training samples for the first network were 10x10 pixels images of traffic signs obtained by localization process on initial images. Input samples for speed limit MLP were 6x12 pixels clear images of decimal digits, and their copies with random noise added on the pattern. Noise was created by flipping 10% of the bits in the original binary image.

C. Determining the speed limit

In order to analyze the numbers in the speed limit sign, a segmentation algorithm must be employed to correctly separate the digits. A straightforward algorithm searches for maxima in the vertical projection of the input image.

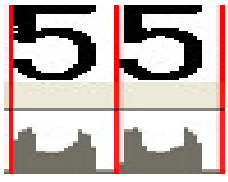

Fig. 3. Input image (top) and the corresponding vertical projection (bottom)

It is possible that some artifacts or noise will remain in the obtained digit after the segmentation is complete. To minimize such interferences, we extract the primary connected component (Fig. 4)

Fig. 4. Example of a binary image and the corresponding connected components

The extracted digit is then normalized to a size of 6x12 pixels and transformed into the input vector for the digit classifier.

Along a classification by neural network, we developed another method of digit-based classification based on structural analysis.

Structural analysis deals with more complex structures than pixels or edges. For example, it considers loops, line ends and junctions. In order to extract this high-level information from a binary image, we must obtain the skeleton of the digit by thinning the object. The procedure of skeletonization is actually a reduction of an object to a graph, and it is mathematically defined with a medial axis transform. Fig. 5. shows an example of skeletonization.

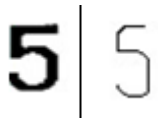

Fig. 5. Original digit (left) and its skeleton (right)

After the skeleton has been obtained, we can extract structural features from it. We consider line ends, junctions and loops, as shown in Fig. 6.

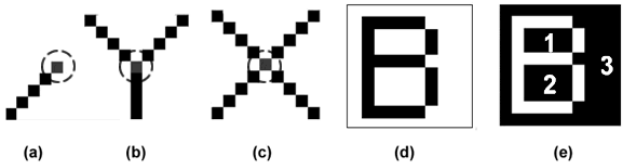

Fig. 6. Line ends (a), junctions (b),(c) and loops (d),(e)

Line end is defined simply as a black pixel with only one black neighbor pixel. Junctions can be found by counting the number of white-black transitions in the 8-pixel neighborhood of the observed pixel. For determining the number of loops, we can invert the image and search for connected components. In the end we subtract 1 from total number of components, because it represents the background.

Each digit can be described with the number of distinct features and their relative positions in the image. For example, “0” is the only digit with one loop and zero line ends. Digits “1” and “2” have the same number of features (two line ends and zero junctions and loops), but their relative and absolute positions differ. We can use this information to directly distinguish the digits.

V. SIGN TRACKING

In a video sequence, a sign will typically appear through multiple consecutive frames. Due to the imperfectness of the detection procedure, the sign might not be detected in every frame. Additionally, there is a possibility that false positives will appear. In order to efficiently track only the sign that is actually in the video, the system would have to remember information from previous frames and use it to correct the detection in the current frame.

We propose a system which uses an auto-degrading reinforcement principle. It is based on two premises:

1. *Auto-degradation*: The system should have a short-term memory which only remembers information in the certain amount of newest frames. The system „forgets“ older information.
2. *Reinforcement*: The system should be updated if an object is detected in the current frame.

The auto-degradation ensures that the information in the nearer past will have more impact than the older information.

Such system can be implemented as a cluster of accumulators. Each accumulator represents a certain object that can be tracked. The value stored in the accumulator is proportional to the number of detections of the respective object. When an object is detected, the respective accumulator's value is increased by a certain amount. Objects not detected have their accumulators' values decreased. If a value in one or more of the accumulators becomes greater than the defined threshold, the object is considered to be tracked. If the tracked object leaves the field of view, the respective accumulator's value will be decreased through time. When it falls under the threshold, the object ceases to be tracked.

VI. SOFTWARE SYSTEM AND RESULTS

In our work, we developed a software system that implements all of the described algorithms. The system consists of two front-end applications with equivalent program core. The first application can be used to detect and recognize traffic signs in a stationary image, with an easy-to-use graphical user interface (Fig. 7)

The second application uses a video file as its input and can be used to detect traffic signs in the video in real-time. The video application uses the same program core as the static image application, with the addition of an object tracking subsystem.

The developed system was tested using a test set of 146 static images (for the first application) and a video sequence in duration of 98 minutes incorporating 128 traffic signs. The results are shown in Table IV. The system has a 73% hit ratio. 15% of all errors are caused by

misdetections, while the other 13% are errors in classification

Fig. 7. Graphical user interface for detection of signs in static images

Fig. 8. Video application with a recognized speed limit traffic sign

TABLE IV
Static image application experimental results

	Number	Percentage
Total number of signs	146	100%
Correctly recognized	106	72.6%
Detection errors	22	15.1%
Classification errors	10	13%

Results obtained from the video application are shown in Table V. The noted performance excels the performance of the static image application. Since the only difference between the two applications is the addition of the object tracking algorithm, we conclude that the improved performance is the result of the increased amount of processed frames. For example, misdetection in a frame can be rectified by a correct detection in one of the following frames.

The application was tested on a dual core Athlon processor (X2 5600+) with an average speed of 21 frames per second and a dual core Intel processor (2,2 GHz) with an average speed of 30 frames per second. The improved performance on Intel processors is caused by the fact that the OpenCV library uses optimized instructions on Intel platforms, and even makes use of the Intel Performance Primitives (IPP) if they are present.

TABLE V
Video application experimental results

	Number	Percentage
Total number of signs	128	100%
Correctly recognized	106	82.8%
Detection errors	13	10.16%
Sign classification errors	7	4.6%
Speed limit errors	2	1.56%
False positives	20	

VII. CONCLUSION AND FUTURE WORK

We developed a system that recognizes traffic signs with real-time application as the main goal. With this goal in mind we used the Viola-Jones detection algorithm and neural networks for classification. The results obtained from the developed software show that our system is applicable for real-time video processing. Furthermore, we conclude that the system has better results when used on a video sequence, compared to the standard approach of traffic sign detection in static images. Relatively low error rates in classification stage indicate that the multilayer perceptron can be successfully used as a classifier of traffic signs and digits.

Further improvements of the system should include a larger number of supported traffic sign classes. Currently, the main holdback is relatively low number of training samples for less frequent traffic signs. In addition, some problems could arise when training the neural network with a larger number of classes due to the increased dimensionality of the search space. There is also a problem of similarity between some traffic sign classes.

Other improvements would include recognition of signs of different shape (triangular, for example), using more advanced methods of feature extraction such as principal component analysis, and increasing the code portability by implementing the functions from external program libraries, which could prove useful when implementing the system on different platforms or embedded systems.

REFERENCES

[1] W. D. Shaposhnikov , D. G. Shaposhnikov , Lubov. N , Er V. Golovan , A. Shevtsova, Road Sign Recognition by Single Positioning of Space-Variant Sensor, Proc. 15th International Conference on Vision Interface, 2002, pp. 213-217.

[2] Andrzej Ruta, Yongmin Lia, Xiaohui Liu, Real-time traffic sign recognition from video by class-specific discriminative features, Pattern Recognition, Vol. 43, 2010, pp. 416-430.

[3] S. Tominaga, Color image segmentation using three perceptual attributes, Proc. CVPR-86, 1986, 628–630.

[4] R. Ohlander, K. Price, D. Reddy, Picture segmentation using a recursive region splitting method, Computer Graphics Image Processing Conference 13, 1978, pp. 224–241.

[5] X.W. Gao, L. Podladchikova, D. Shaposhnikov, K. Hong, N. Shevtsova, Recognition of traffic signs based on their colour and shape features extracted using human vision models, Journal of Visual Comunication and Image Representation, 2006, pp. 675-685.

[6] John Hatzidimos, Automatic Traffic Sign Recognition in Digital Images, Proceedings of the International Conference on Theory and Applications of Mathematics and Informatics - ICTAMI 2004, Thessaloniki, Greece, pp. 174-184.

[7] V. Barrile, M. Cacciola, G. M. Meduri, F. C. Morabito, Automatic Recognition of Road Signs by Hough Transform, "International archives of the photogrammetry, remote sensing and spatial information sciences", n. XXXVI Part 5, 2008, pp. 62-67.

[8] Paul Viola and Michael Jones. Robust real-time object detection. International Journal of Computer Vision, 2004, pp. 137-154.

[9] C. Bahlmann, Y. Zhu, Visvanathan Ramesh, M. Pellkofer, and T. Koehler. A system for traffic sign detection, tracking, and recognition using color, shape, and motion information. Intelligent Vehicles Symposium, 2005, pp. 255–260.

[10] Karla Brkic, Axel Pinz and Sinisa Segvic ``Traffic sign detection as a component of an automated traffic infrastructure inventory system", in Proc. of the annual Workshop of the Austrian Association for Pattern Recognition (OAGM/AAPR), Austria, 2009, pp. 1-12.

[11] Keller, C.G. Sprunk, C. Bahlmann, C. Giebel, J. Baratoff, G., Real-time recognition of U.S. speed signs, Intelligent Vehicles Symposium, 2008 , pp. 518-523.

[12] Ach, R.; Luth, N.; Techmer, A., Real-time detection of traffic signs on a multi-core processor, Intelligent Vehicles Symposium, 2008, pp. 307–312.

[13] Sho Shimamura, Satoshi Yonemoto, Road Sign Recognition with Color and Edge based features, IEICE Tech. Rep., vol. 108, no. 471, 2009, pp. 23-28.

[14] Ghica, D., Si Wei Lu, Xiaobu Yuan, Recognition of traffic signs by artificial neural network, IEEE International Conference on Neural Networks, 1995, pp. 1444-1449.

[15] Hua Huang, Chao Chen, Yulan Jia, Shuming Tang, Automatic Detection and Recognition of Circular Road Sign, Mechatronic and Embedded Systems and Applications, MESA 2008, pp. 626-630.

[16] Bogusław Cyganek, Road Signs Recognition by the Scale-Space Template Matching in the Log-Polar Domain, Proc. of the 3rd Iberian conference on Pattern Recognition and Image Analysis, Part I, 2007, pp. 330-337.

[17] Freund, Y., Schapire R. A Short Introduction to Boosting. Journal of Japanese Society for Artificial Intelligence, 1999, pp. 771-780.

[18] R. Lienhart, A. Kuranov, V. Pisarevsky, Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection, DAGM'03, 25th Pattern Recognition Symposium, Germany, 2003, pp. 297-304.

[19] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control Signals and Systems 2,1989, pp. 303–314.

[20] K. Funahashi, On the approximate realization of continuous mappings by neural networks, Neural Networks 2, 1989, pp. 183–192.

[21] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1995.

[22] T.W. Ridler, S. Calvard, Picture Thresholding Using an Iterative Selection Method, SMC(8), 1978, pp. 629-632.

[23] L.G. Shapiro, G.C. Stockman, Computer Vision, Prentice Hall, 2002.