Efficient Recognition of Speed Limit Signs

Jim Torresen, Jorgen W. Bakke and Lukas Sekanina

***Abstract*— An automatic traffic sign detection system would be important in a driver assistance system. In this paper, an approach for detecting Norwegian speed limit signs is proposed. It consists of three major steps: Color-based filtering, locating sign(s) in an image and detection of numbers on the sign. About 91% correct recognition is achieved for a selection of 198 images.**

I. INTRODUCTION

Being able to recognize traffic signs would probably be important for vehicle safety systems in the future. Such systems could assist drivers on signs they did not notice before passing them. Specifically speed limit sign recognition – studied in this paper, could inform drivers about the present speed limit as well as giving an alert if a car is driven faster than the speed limit. In the future, autonomous vehicles would probably have to be controlled by automatic road sign recognition.

Recognizing road images have been studied for a long time. The first known attempt for making a real-time system was by Akatsuka and Imai [1]. Many techniques have been proposed since then.

The standard technique for detecting and recognizing road signs consists of three steps [2]. First, color segmentation or color thresholding is applied to emphasize possible signs in an image. Thus, restricting the search area in the image. Second, template matching is applied for shape detection. Third, specific signs are detected using template matching or neural networks.

A color image is often transformed from the RGB (Red, Green, Blue) color space into the HSV (Hue, Saturation, Value) color space. Color segmentation then becomes easier (by applying it on the hue value only rather than the three RGB values). The hue value is invariant for the illumination as well. However, the hue is not suited for grey-level segmentation since it has a constant value along the "grey-level" axis. The value will be unstable near this axis too. Small perturbations in the RGB signals may cause large variations in the hue.

A traffic sign detection system consisting of four stages has been proposed by Wei et al [3]. First, a color-based filtering is applied to filter out image regions that have color characteristics similar to the color found in one of the signs known to the system. That is, the color input image is converted into a binary image using the filter. Second, the boundaries of the regions are smoothed by applying close and open morphological operations. Third, shape analysis is performed to evaluate the similarity between the region and a given shape. This includes major and minor axis as well as comparing area. Finally, the holes within the regions are analyzed. The system performs well for four different signs in different environments as well as at different orientation and scales.

Another system proposed by Hsu and Huang [4] is based on three steps for road sign detection followed by sign recognition based on Matching Pursuit (MP) filter. The detection locates regions with signs in the unknown input image by first using a priori knowledge about position, shape and color (HSV). Second, template matching is performed in the extracted regions to find road signs. Third, the road sign is extracted by further template matching and normalization.

This paper concerns detection of speed limit signs specifically. An early version of the system (with few results) was introduced in [5]. Results from our experiments with sign number classification in evolvable hardware are given in [6]. We have found very little work on speed limit sign classification. There exists a system based on Global Position System (GPS) and digital road maps with speed limits included [7]. However, such a system depends on much external infrastructure. Further, problems like lack of updated speed limits on road maps question the reliability of such a system.

The next section introduces the Norwegian speed limit signs. This is followed by an outline of the proposed algorithm in Section III and results from the implementation in Section IV, respectively. Finally, Section V concludes the paper.

II. NORWEGIAN SPEED LIMIT SIGNS

Speed limit signs have features making them feasible for automatic detection. First, there is a limited number of signs to distinguish.

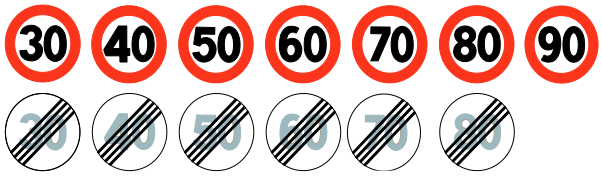

Fig. 1. Norwegian speed limit signs (standard).

In Norway, the following speed limit signs – see Figure 1, are used: 30, 40, 50, 60, 70, 80 and 90 (100 are being tested at the moment). The speed is measured in kilometers per

J. Torresen and J.W. Bakke is with Department of Informatics, University of Oslo, P.O. Box 1080 Blindern, N-0316 Oslo, Norway {jimtoer,jorgenwa}@ifi.uio.no

L. Sekanina is with Faculty of Information Technology, Brno University of Technology, Bozetechova 2, 612 66 Brno, Czech Republic sekanina@fit.vutbr.cz

Fig. 2. An example of an input image.

hour. A typical input image is shown in Figure 2. The outer circle of a sign is in *red* color. Further, there are signs to nullify 30 and 40 (set speed limit to 50), 50, 60 and 70 (set speed limit to 80) and 80 (set speed limit to 90). The nullify signs have not yet been considered by the algorithm outlined in this paper.

Second, there are rules (named a road grammar) for how signs can be placed along the road. After a “80” sign you will never find a “30” sign, but rather another “80” sign or a “60” sign. Third, the numbers on the signs are positioned vertically making the matching simpler. In curves only, the signs may be marginally tilted. These features make it promising to undertake speed limit sign detection with a very high rate of correct prediction.

Speed limit signs in several other countries use the same format (color and numbers) as in Norway. However, in some countries like in the US both the color and the numbers are different. Analyzing such signs would have to also consider the second digit since it can be either 0 or 5.

Making a speed limit sign detection system commercially applicable, the safety is very important. If the detection system do not have a distinct best match, it should rather output that it “don’t know” the speed at the moment. Thus, it is important that the system refrain from indicating a speed rather than indicating a *wrong* speed. The system presented in the next section is based on a novel heuristic developed through a large range of experiments.

III. SPEED LIMIT SIGN RECOGNITION ALGORITHM

The algorithm is divided into *three* parts: 1) Image filtering to emphasize the red circle on the sign(s), 2) Template matching to locate the sign(s) in an image and 3) Sign number recognition. These will be presented in the following sections below.

A. Image Filtering

A specialized robust color filter is applied on the image to mark the *red circle* surrounding the speed limit numbers. It consists of two main parts. The first part – called Image filtering RWB (Red White Black), extracts the three colors red, white and black. The second part - called Red Reduction, reduces the number of red pixels in the image to minimize the succeeding template matching processing. The detection algorithm has to use a broad definition of red and white colors since various intensities and saturations can occur in real images (depending on weather, day/night-light, etc.). On the other hand, only red color *on* traffic signs should be detected to minimize the number of starting points for the following template matching algorithm. Thus, detection of red color *outside* a traffic sign (like on buildings, cars, etc.) should be minimized.

The RGB color model was found useful for detection. RGB is also appropriate for hardware implementation since this will be the output from the camera. Converting into e.g. the HSV model would be computational expensive. The colors are defined (by experiments) as follows in the Image filtering RWB:

- A pixel is RED if: (R>77) AND (R-G>17) AND (R-B>17)
- A pixel is WHITE if: (R>108) AND (G>108) AND (B>108)
- A pixel is BLACK if: (R<122) AND (G<122) AND (B<122)

RED, WHITE and BLACK are symbolic names that will be used below. As one can see, the definition of the basic colors is broad to assure robustness. WHITE and BLACK are used mainly in different parts of the algorithm. Thus, there is a value overlap (108 to 122) between BLACK and WHITE. A high classification rate is achieved by combining these colors. The output from filtering the image in Figure 2 is given in Figure 3.

The brightness varies a lot between the images. To be able to find signs in either light or dark images, the pixel values have to be made darker or lighter, respectively. This is undertaken if the average pixel value in an image is less than 125 (too dark image) or above 150 (too light image). The pixel values are to be modified by the difference between the average pixel value and the dark or light threshold values given above. Adjusting the value of each pixel in an image adds much computation time. Thus, the *thresholds* for the colors (RED, WHITE and BLACK) are modified instead. This results in the same recognition performance but less time is needed.

The Red Reduction algorithm reduces the number of red pixels in the following way:

1) Find all 2x2 WHITE squares in the image (since there are white pixels *inside* the red circle on the sign).
2) Find (recursively) all RED neighbors of the located WHITE squares.
3) Apply template matching (described in the next section) to the RED pixels found.

This algorithm effectively limits the number of red pixels in the image to be further processed.

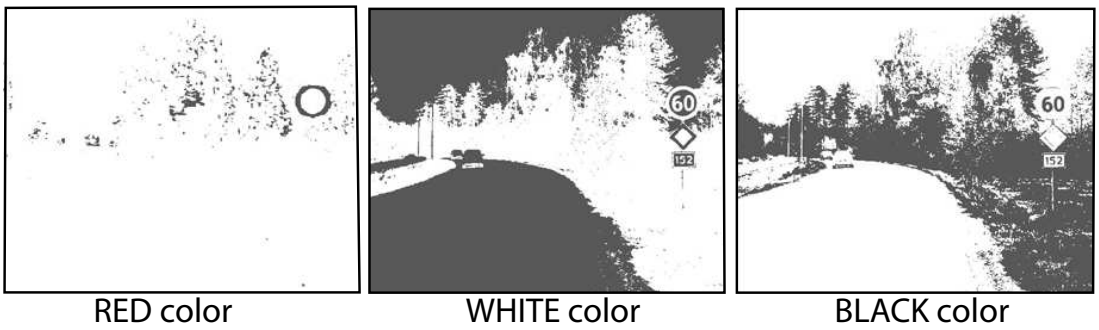

Fig. 3. The result of each of the three color filters. (Dark color mark pixels with the filtered color in each of the images.)

Besttotalmatch = 0
1) FOR (every template) DO
  BestP = 0
  2) FOR (every RED pixel at even line and even column in the Image2) DO
    a) P = Percentage of pixels that match with a template.
    b) IF (there are some RED pixels in the center of the sign) THEN P = 0 (avoid crossing lines)
    c) IF (RED pixels are not distributed symmetrically) THEN P = 0 (avoid triangles)
    d) IF (NOT BLACK pixels in the template center) THEN P = 0 (avoid signs without numbers)
    e) IF (there is more than 95% of WHITE pixels) THEN P = 0
    f) IF (there is less than 10% of WHITE pixels) THEN P = 0
    g) IF (P > BestP) THEN update BestP
  3) IF (BestP > Besttotalmatch) THEN update record of the Besttotalmatch
IF (Besttotalmatch < 50%) THEN Print: No speed limit
ELSE continue by recognition of number on the sign

Fig. 4. The template matching algorithm.

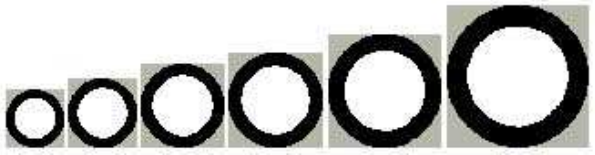

Fig. 5. Templates used for locating signs in an image.

B. Locating Sign by Template Matching

We locate sign in the image by searching for the *red circle* surrounding the numbers. The search is based on template matching. One reason for this is that template matching can be very effectively implemented in hardware. To limit the computation time, we have found that six templates are sufficient. This was based on analyzing sign sizes in images taken from a vehicle. The templates are of size 32x32, 38x38, 46x46, 52x52, 62x62 and 78x78 pixels and are used to detect the *position* of a speed limit sign. The templates are shown in Figure 5. A very high reliability of the system is needed and small templates tend to produces incorrect classification of the numbers. Thus, no smaller templates than 32x32 have been used. The template matching algorithm is shown in Figure 4. In addition to locating the sign, the algorithm tries to reject objects that are not signs and signs that are not speed limit signs. That is, improving recognition at the same time as reducing the computation time.

C. Sign Number Recognition

The last part of the algorithm is to detect the speed limit *number* on a sign. This is conducted as follows:

1) Clean the region defined by the best template (remove RED and surrounding colors), but keep the numbers.
2) Find boundaries of the numbers (width and height).
3) Work only with the first number (the second is always zero).
4) Create a 7 (rows) x 5 (columns) bit array of the given number (down-scaling): Set each bit of the array to 1 if there is more BLACK than WHITE pixels, else set the bit to 0.
5) Classify the bit array using a classifier system.

Fig. 6. Examples of extracted arrays from real images.

Some randomly picked examples of different extracted numbers (bit arrays) are included in Figure 6. To classify

the numbers, a feed-forward neural network trained by the back-propagation algorithm was used. It consisted of 35 inputs (7 x 5 bit array), 35 hidden units and 6 outputs – one for each speed limit (30 – 80) to be classified.

IV. RESULTS

This section reports about the experimental work undertaken. As a part of the experiments, a Graphical user interface (GUI) was developed and is the first to be described below.

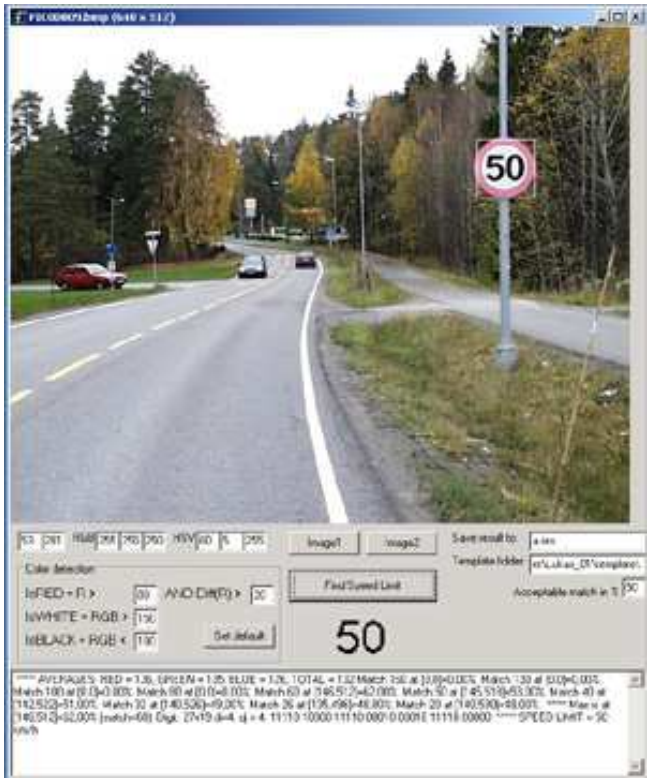

Fig. 7. The Graphical User Interface.

Graphical User Interface

Figure 7 shows a picture of the main part of the GUI. The purpose of the control elements and indicators are as follows:

- RGB – informs about RGB pixel values at the mouse position.
- HSV – informs about HSV pixel values at the mouse position.
- Save result to: denotes a file where the report will be stored.
- Template folder: denotes a folder where templates used for recognition are stored.
- Acceptable match in %: if a match is less than this % value, then there is not any speed limit sign in the image.
- Color detection – allows the user to define the thresholds for the Image filtering RWB.
- Set defaults – set default thresholds for the Image filtering RWB.
- Image1 – shows the image after application of the Red Reduction filter. Position of the best match is indicated.
- Image2 – shows the image after thresholding and cleaning inside a template (mainly for user information).
- Find Speed Limit – activates speed limit detection. Image1 and Image2 are created, a report (in the message box) is created and the final decision is presented.
- Possible final decisions (50 in the given picture):
  - NOT – no speed limit found.
  - ??? – probably no speed limit sign since a template matched well, but no number was recognized.
  - x? – unknown (not supported) number recognized.
  - The speed limit number detected.

The user interface is very useful for studying single images. However, to be able to process a batch of images, we have extended the system to accept specification of a *set of* images.

A. Performance Results

A database of 198 images from traffic situations were used in the experiments. 115 contained a speed limit sign and 83 contained other signs or no sign at all. Many of the images were in various ways “difficult” (different brightness on sign, faded color on sign etc). The results were as follows:

- Is there a speed limit sign? A speed limit sign was found (and correctly classified) in 100 of the 115 images (87%) containing a speed limit sign. In those not found, the system stated that a speed limit sign was not present in the image.
- 78 of the 83 images without a speed limit was correctly refused (94%). Thus, only five images – see an example of such an image in Figure 8, were sent to the final sign number recognition. Two out of these were refused in the final stage by having an ambiguous/low output from the neural network. Those three matching a number could most probably be avoided by matching on the “0” number (not yet implemented).
- In summary, 180 out of the 198 images (90.9%) were correctly classified.

For *all* the 100 images that the system correctly detected a speed limit sign, the number on the sign was also correctly classified. The number of images for the different speed limits is given in Table I.

In addition to achieve a high rate of correct classification, we have made much effort at the same time to reduce the processing time. This would be highly needed in a future real-time system. The average processing time for an image is 130ms, which represents a capacity of processing about 8 images per second. This is measured on a PC with a 1.4 GHz Athlon AMD processor (512Mbyte RAM). The image filtering is the most computational demanding with about

TABLE I

THE NUMBER OF SPEED LIMIT SIGNS CLASSIFIED IN THE SIGN NUMBER RECOGNITION PART.

Speed limit	Number of images
30	4
40	11
50	32
60	35
70	12
80	6

Fig. 8. An extracted part of an image where the sign was classified as a speed limit sign.

90ms average processing time, while template matching and number recognition all together use about 40ms.

With the promising results so far, future work consists of further improving the algorithms and implementing the most computational demanding parts in special hardware. Testing on more image data would also be important to ensure reliability. Further, to allow for a real prototype, the nullify signs would also have to be classified.

V. CONCLUSIONS

The paper has presented a novel approach to locate speed limit signs and detect the numbers on them. It is focused on reducing computation time at the same time as getting high recognition performance. Good results are achieved for detecting the signs.

VI. ACKNOWLEDGMENTS

The research is funded by the Research Council of Norway through Norwegian Government Scholarships 2001/2002.

REFERENCES

[1] H. Akatsuka and S. Imai. Road signpost recognition system. In *Proc. of SAE vehicle highway infrastructure: safety comptatilbility*, pages 189–196, 1987.

[2] M. Lalonde and Y. Li. Road sign recognition, survey of the state of the art. In *CRIM/IIT (Centre de recherche informatique de Montreal)*, 1995.

[3] G. Wei et al. Traffi c sign detection and recognition for safe driving. In Dagli et al., editors, *Smart Engineering System Design: Neural Networks, Fuzzy Logic, Evolutionary Programming, Data Mining, and Complex Systems, Proc. of ANNIE'99*. ASME Press, November 1999.

[4] S.H. Hsu and C.L. Huang. Road sign detection and recognition using matching pursuit method. *Image and Vision Computing*, 19:119–129, 2001.

[5] L. Sekanina and J. Torresen. Detection of Norwegian speed limit signs. In *Proc. of the 16th European Simulation Multiconference (ESM2002)*, pages 337–340. SCS Europe, June 2002.

[6] J. Torresen, J.W. Bakke, and L. Sekanina. Recognizing speed limit sign numbers by evolvable hardware. In *Proc. of Parallel Problem Solving from Nature VIII (PPSN VIII)*, Lecture Notes in Computer Science. Springer-Verlag, 2004.

[7] R. Thomas. Less is more [intelligent speed adaptation for road vehicles]. *IEE Review*, 49(5):40–43, May 2003.