Traffic Signs Recognition System

Krzysztof Dyczkowski, Paweł Gadecki, Adam Kułakowski
Faculty of Mathematics and Computer Science,
Adam Mickiewicz University,
Umultowska 87, 61-614 Poznań, Poland
(email: chris@amu.edu.pl)

***Abstract*— This paper presents a computer system that detects traffic signs in the input video stream and tries to recognize them using a knowledge base. Detection is done in the HSB colorspace using fuzzy color segmentation. Afterwards, detected segments are analyzed by a fuzzy rule-based system which checks whether a pair of segments represents the same sign. Finally, detected objects are checked against the knowledge base using a designed similarity measure.**

I. Introduction

Contemporary driving aid systems consist of many modules, responsible for providing the driver with information. One of the important parts of such a system is a module designed for recognition of the traffic signs passed by a car. The identification is based on video camera image analysis. First it has to detect areas containing traffic signs on the captured image, then identify the signs, and finally convey the information to the driver.

Efficient sign recognition software has to be a real-time system. This means that analysis of a single video frame must be completed sufficiently quickly, that the received information is remains up to date.

II. System Architecture

The system consists of three main parts:

Knowledge base – stores signs information and representatives associated with them, allowing fast finding of the representations most similar to the received encoded pattern (the signs most similar to it).

Learning module – makes it possible to create and extend the knowledge base of the system by adding connections: *fragment of image taken from video material ↔ traffic sign*.

Recognizing module – responsible for processing the input image so as to identify areas which may contain traffic signs, to make the best choice of them, to compare them with data in the knowledge base and to communicate the result to the user (Fig. 1).

Fig. 2 shows a simplified packages and classes diagram of the whole system.

A. Image Preprocessing

The goal of image preprocessing is to define all the parts of the picture that may contain traffic signs. First we have to determine in which part of the picture we are looking for the traffic signs: it is assumed that we are trying to detect only the signs appearing on the right side of the road. Therefore, we look for signs only on the right side of the picture. Then we have to perform segmentation of the input image to isolate fragments that may contain traffic signs.

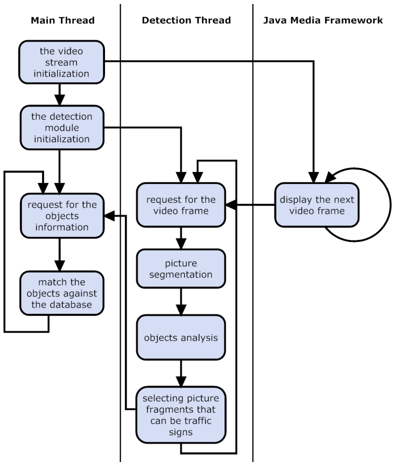

Main Thread
Detection Thread
Java Media Framework
the video stream initialization
the detection module initialization
request for the objects information
match the objects against the database
request for the video frame
picture segmentation
objects analysis
selecting picture fragments that can be traffic signs
display the next video frame

Fig. 1. Simplified outline of the action of the recognizing module

B. Fuzzy Color Segmentation (cf. [1], [2])

Color segmentation produces a binary image. In this binary image white pixels are considered to be part of traffic signs. Black ones constitute the background, and these will not be considered during further processing.

First we used the RGB color model. After preliminary work it was found to be too unnatural. Furthermore, defining the fuzzy sets based on the RGB elements was quite difficult.

In the next step we tried to employ the HSB color model. HSB stands for "Hue, Saturation and Brightness". Hue describes what color it is, taking values from $0°$ to $360°$. Saturation represents the amount of color, taking values from 0.0 to 1.0 (or from 0% to 100%). Brightness, the last value, also ranges from 0.0 to 1.0. The higher the value, the brighter the color.

In the segmentation process we examined every pixel and judged if it could belong to some traffic sign. In order to do this, we defined a model of three colors that are typical for traffic signs: red, blue and yellow. These are represented by three fuzzy sets. We calculated to what degree a pixel is a member of these color classes. We also defined the threshold that must be reached. If the maximum value of

The research project is supported by Ministry of Science and Higher Education grant NN519384936

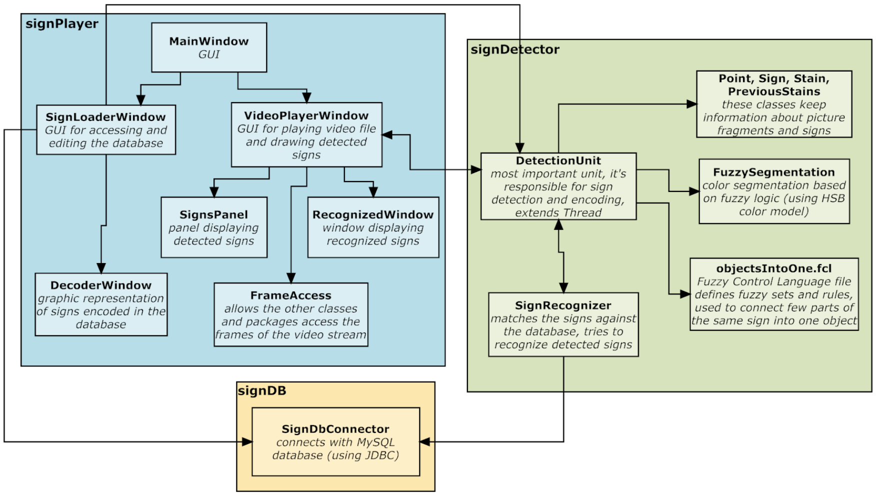

signPlayer
MainWindow
GUI
SignLoaderWindow
GUI for accessing and editing the database
VideoPlayerWindow
GUI for playing video file and drawing detected signs
SignsPanel
panel displaying detected signs
RecognizedWindow
window displaying recognized signs
DecoderWindow
graphic representation of signs encoded in the database
FrameAccess
allows the other classes and packages access the frames of the video stream
signDetector
Point, Sign, Stain, PreviousStains
these classes keep information about picture fragments and signs
DetectionUnit
most important unit, it's responsible for sign detection and encoding, extends Thread
FuzzySegmentation
color segmentation based on fuzzy logic (using HSB color model)
objectsIntoOne.fcl
Fuzzy Control Language file defines fuzzy sets and rules, used to connect few parts of the same sign into one object
SignRecognizer
matches the signs against the database, tries to recognize detected signs
signDB
SignDbConnector
connects with MySQL database (using JDBC)

Fig. 2. System architecture

the degree to which the considered pixel belongs to these three colors is greater than or equal to the threshold then the pixel is accepted as a part of a traffic sign. The membership functions of the fuzzy sets representing the color model are presented in Fig. 3.

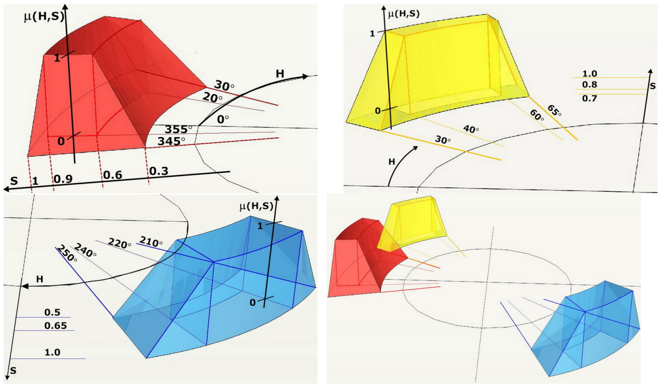


Fig. 3. Membership functions for the fuzzy sets "red", "yellow" and "blue"

Fig. 4 illustrates the results of color segmentation on a random video frame.

C. Objects Analysis. (cf. [3], [4], [5])

Now we are able to work with the binary image. First, it is necessary to find all the regions on the picture. By a region we understand a group of adjoining pixels. A region is defined by the position of a rectangle which circumscribes it. To find all the regions in a binary image we created an iterative algorithm because the recursive one had consumed too much memory. The results of the region detecting algorithm are shown in Fig. 7.

From now on by an object we understand a rectangle circumscribing a region in the binary image. As we can see in Fig. 7 a traffic sign can be divided into two or more objects. We define a fuzzy rule based system which determines if a pair of objects represents two parts of the same traffic sign.

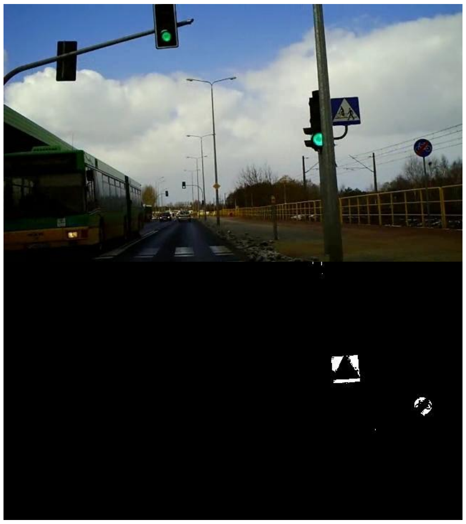


Fig. 4. Example of Fuzzy color segmentation

We check this for every pair of objects. If the answer is 'yes' we connect these objects into one larger one (represented by the rectangle circumscribing both of them).

As input the fuzzy controller takes the following parameters:

1) $d_{hor} = d_{h1}/d_{h2}$ – relative horizontal distance between two objects;
2) $d_{ver}/d_{v1}/d_{v2}$ – relative vertical distance between two objects;
3) $l_{hor} = (p_{h1} + p_{h2})/p_{h3}$ – relative horizontal location difference between two objects;
4) $l_{ver} = (p_{v1} + p_{v2})/p_{v3}$ – relative vertical location difference between two objects;
5) $area_A$ – area of object A;
6) $area_B$ – area of object B;
7) $inter = \frac{ab}{\min(area_A, area_B)}$ – intersection of two objects;
8) $ratio$ – aspect ratio of the object.

These formulas are based on the variables defined in Fig. 5.

On the basis of the above-mentioned variables we defined

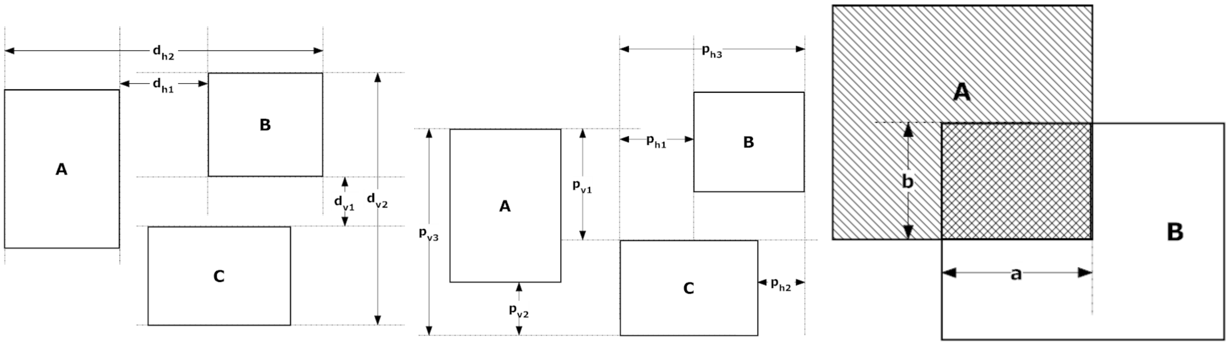


Fig. 5. Variables used to calculate distance and location difference between two objects and their intersection

fuzzy sets describing the mutual relations between the objects. Membership functions of these fuzzy sets are shown in Fig. 6.

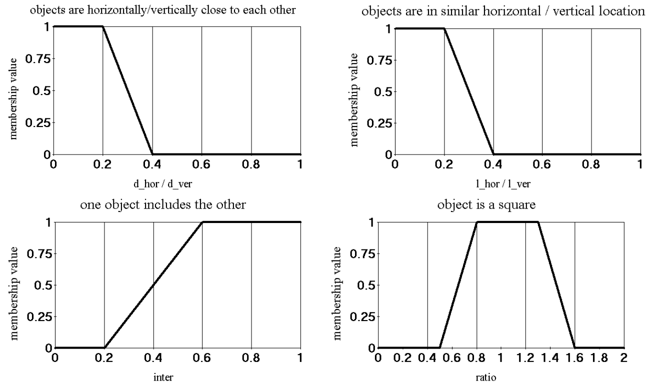


Fig. 6. Membership functions describing the mutual relations between the objects

Then we defined the fuzzy rules:

DEFAULT: *'objects are parts of the same sign'* `IS` *'No'*;

RULE 1: `IF` *inter* `IS` *'one object includes the other'* `THEN` *'objects are parts of the same sign'* `IS` *'Yes'*;

RULE 2: `IF` d_{hor} `IS` *'objects are close to each other'* `AND` l_{ver} `IS` *'objects are in similar location'* `THEN` *'objects are parts of the same sign'* `IS` *'Yes'*;

RULE 3: `IF` ($ratio_A$ `IS NOT` *'square'* `OR` $ratio_B$ `IS NOT` *'square'*) `AND` d_{ver} `IS` *'objects are close to each other'* `AND` l_{hor} `IS` *'objects are in similar location'* `THEN` *'objects are parts of the same sign'* `IS` *'Yes'*.

By default we assumed that the examined objects are not parts of the same traffic sign. The first rule says that if the intersection of these objects is big enough then they belong to the same traffic sign. The second rule says that if the objects are close to each other in the x dimension and they are placed similarly in the y dimension, they are different parts of the same traffic sign. The third rule is analogous to the second, but there is one additional condition: two objects lying one below the other are not parts of the same traffic sign if both of them are squares. Quite often we can see a group of two or three traffic signs one under the other. Thanks to this additional condition in the third rule we avoid connecting such signs like into one object.

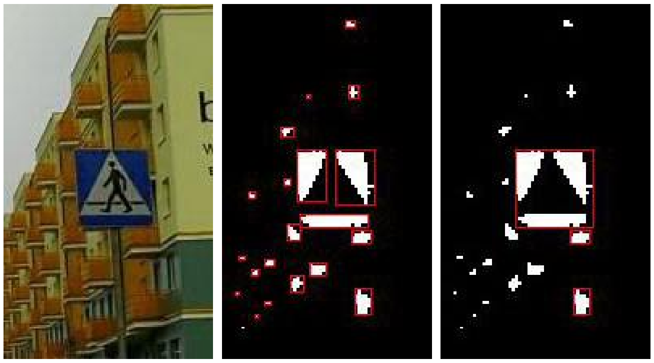


Fig. 7. Example of segmentation, region retrival, connection of objects and rejection of those that were too small

We have connected all the objects that were supposed to represent different parts of the same traffic sign. Now we go through the objects list again. We remove an object from the list if:

- **It is too small** – even if it is a traffic sign we are not able to recognize it since it is very small. Furthermore, in the process of encoding we will use a 32×32 matrix. We remove the object from the list if its width or height is less than 32 pixels.
- **Its aspect ratio is too large or too small**. We defined aspect ratio as follows: $ratio = height/width$. We remove the object from the list if $ratio \notin (0.24, 4.0)$.

The results of the analysis, connection and rejection of objects are shown on Fig. 7 (picture on the right).

D. Traffic Sign Pattern Encoding (cf. [6], [7])

The process of traffic sign pattern encoding involves describing the pattern in the form of an interchangeable fixed set of numbers, where similar sets represent our traffic signs in the database. Using appropriate similarity measure we will be able to search the database for the signs which are most similar to the one being analyzed.

Assuming that encoded picture fragment is a traffic sign, we assign to each of its pixels a certain value corresponding to its color (black, white, yellow, blue or red). Every picture

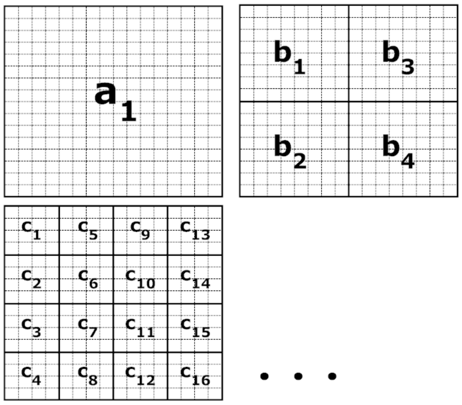


Fig. 8. Image division during encoding.

fragment representing a particular traffic sign is encoded into six vectors, each of them being a more detailed description of the picture. First vector is a single number standing for the whole picture. Then we divide the picture into four sections and compute four values for the second vector (Fig. 8).

Each component of every vector is a weighted mean of the values of the pixels belonging to the section described by that element (pixels of a non-sign color were assigned lower weights than the others, but they are considered to be a part of the encoded picture). The result of the sign pattern encoding is a set of six vectors (of lengths: 1, 4, 16, 64, 128 and 256).

$$A = [a_1],\ B = [b_1, b_2, b_3, b_4],\ \ldots,\ F = [f_1, f_2, \ldots,\ f_{256}]$$

The entire examined area is described by the first vector with one number, containing reliable information about the colors within that area. The next already says something more about the distribution of these colors. Every subsequent division supplies additional information. This approach is intended to enable fast recognition of when the compared pair of signs is too different, by comparing the first two or three vectors. Let us notice here that the first vectors are very short, which it allows us to index the database on the columns used to store them. This means that we can very quickly get to the signs from the database which are quite similar to the investigated one. Only for this selected group of signs will we compare the subsequent (much longer) vectors.

In Fig. 9 we can see an example of a graphical representations of the sign "pedestrian crossing". In the top right we can see the fragment of the image (recognized as the traffic sign). Below that we can see the effect of segmentation of this fragment. On the right-hand side we can see graphical representatives of the pattern – each of six fragments represens the image in a more and more detailed manner, circumscribing the vector.

As one can see, although the traffic signs are positioned somewhat a differently on these images and have different sizes, the first three vectors describing them are very similar to each other.

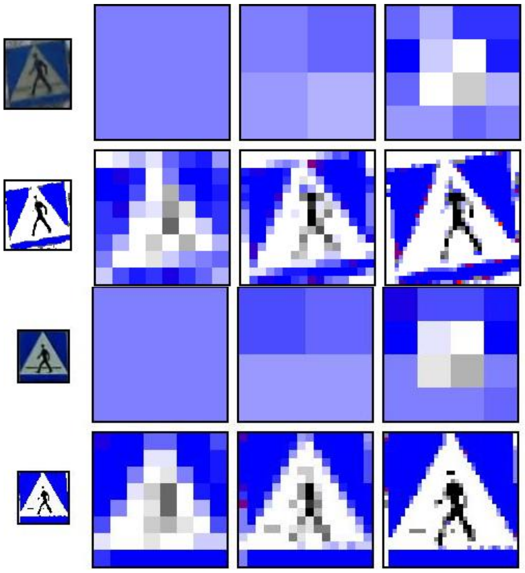


Fig. 9. Process of encoding traffic signs on the example of two traffic signs for "pedestrian crossing"

E. Building the Knowledge Base

The knowledge base is a database of associations between encoded traffic signs and signs descriptions to which they refer. It consists of three elements:

sign_images – table that contains encoded traffic signs, columns `d1...d6` store sign pattern vectors and `signId` is a foreign key referring to traffic sign description. We set up an index on the columns `d1...d4` to speed up searching through its records;

signs – table with traffic signs descriptions and their model images;

sign_pics – table containing pictures of encoded traffic signs stored in the `sign_images` table (this data is stored for the knowledge base content analysis and integrity check).

Detected but unrecognized pieces of the input image are stored for the purpose of expanding the knowledge base. The user employing the application is able to link these images (if they represent traffic signs) with their traffic sign descriptions. If such an association is made the **SignLoader** application stores it in the knowledge base (see Fig. 10).

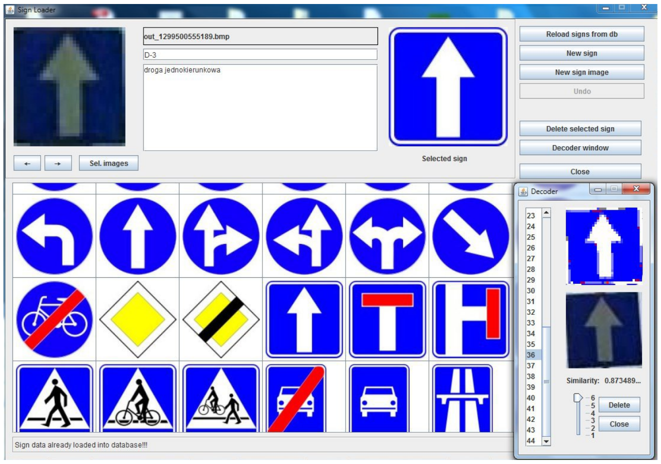
F. Similarity Measure

In order to recognize a detected sign we have to find a best matching representation in the database. Assuming that our knowledge base holds thousands of encoded patterns it would be too time-consuming to compare the detected sign with all of them. Our approach is first to extract a certain number of encoded patterns from the base and compare them with the detected sign.

1) Data extraction: The detected sign is represented by six encoded vectors:

$$A = [a_1], B = [b_1, b_2, b_3, b_4], \ldots, F = [f_1, f_2, \ldots, f_{256}]$$

This will be denoted by $R = (A, B, C, D, E, F)$. For the first four of these we create down and up-limit vectors



out_1299500555189.bmp

D-3

droga jednokierunkowa

Similarity: 0.873489...

Sign data already loaded into database!!!

Fig. 10. SignLoader application window

A_{min},..., D_{min} and A_{max}, ..., D_{max} :

$$A_{min} = (a_1 - \delta_1), \ldots, D_{min} = (d_1 - \delta_4, d_2 - \delta_4, \ldots, d_{64} - \delta_4)$$

$$A_{max} = (a_1 + \delta_1), \ldots, D_{max} = (d_1 + \delta_4, d_2 + \delta_4, \ldots, d_{64} + \delta_4)$$

where $\delta_1, \ldots, \delta_4$ have experimentally determined values. Using these artificially created limit vectors and indexed columns `d1`,..., `d4` we are able to extract very rapidly a certain number of encoded traffic sign representations which are quite similar to the detected sign.

2) Sign comparison: To find the best matching traffic sign representation we designed a similarity measure Sim which for two representations R_1 and R_2, both given by the set of vectors: $R_1 = (A_1, B_1, C_1, D_1, E_1, F_1)$ and $R_2 = (A_2, B_2, C_2, D_2, E_2, F_2)$, returns a value from the range $[0, 1]$, where the higher the value, the closer the similarity:

$$\begin{aligned} Sim(R_1, R_2) = 1 - (&0.01 * D_1(R_1, R_2) + 0.02 * D_2(R_1, R_2) \\ &+ 0.07 * D_3(R_1, R_2) + 0.1 * D_4(R_1, R_2) \\ &+ 0.2 * D_5(R_1, R_2) + 0.6 * D_6(R_1, R_2)) \end{aligned}$$

where

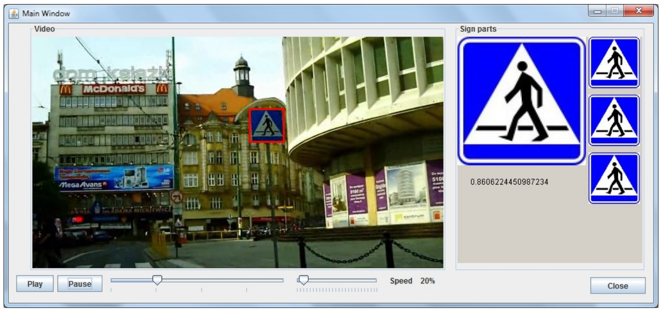
$$\begin{aligned} D_1(R_1, R_2) &= Dif(|a1_1 - a2_1|) \\ D_2(R_1, R_2) &= \frac{\sum_{j=0}^{4} Dif(|b1_j - b2_j|)}{4} \\ D_3(R_1, R_2) &= \frac{\sum_{j=0}^{16} Dif(|c1_j - c2_j|)}{16} \\ D_4(R_1, R_2) &= \frac{\sum_{j=0}^{64} Dif(|d1_j - d2_j|)}{64} \\ D_5(R_1, R_2) &= \frac{\sum_{j=0}^{128} Dif(|e_j - e2_j|)}{128} \\ D_6(R_1, R_2) &= \frac{\sum_{j=0}^{256} Dif(|f_j - f2_j|)}{256} \end{aligned}$$

and Dif is a difference degree function:

$$Dif(x) = \begin{cases} \frac{arctan(x-5)+\frac{\pi}{2}}{\pi} & if\ x > 0 \\ 0 & if\ x = 0 \end{cases}$$

G. Implementation

The whole system is written in Java (the architecture is presented in Fig. 2 and the main window of the application is presented in Fig. 11). The Fobs4JMF library is used for video stream playback and frame capture (which is an implementation of the Java Multimedia Framework). The knowledge base is stored in a MySQL database with JDBC connection. Some of the fuzzy logic operations were implemented using the JFuzzyLogic library.



0.8606224450987234

Fig. 11. Main window of the application

III. Results

To test the system we collected about five hours of video recording. The system automatically separated from this material fragments of the image which might be traffic signs. Next by hand, we chose representatives of traffic signs out of those image fragments of which in the database we recognized as significant. We tried to avoid entering two representatives of the same traffic sign into the database if they were very close to each other. We regard different pictures as being of the same traffic sign if they fulfilled at least one of the following conditions:

- they differ in size;
- they come from recordings made in different weather conditions; traffic signs differ depending on shade, brightness or color saturation;
- the traffic signs are viewed under slightly different angles.

Table I gives the number of representations collected in the database for individual traffic signs chosen for the tests.

A. Test Methodology

To achieve test scores as far as possible in accordance with reality we adopted a number of assumptions:

- the test recording will have a length of 2 hours and it will consist of recordings made under different weather conditions and lighting;
- we will examine the quantity of signs detected for each kind of sign;
- among the detected signs we will count correctly recognized signs;
- the examination will take into consideration only signs with a large number of representations in the knowledge

TABLE I

NUMBER OF REPRESENTATIONS OF THE TRAFFIC SIGNS IN THE DATABASE

Sign symbol	Traffic sign	No. of represent.
A-7	Give way 	96
B-21	No left turn 	61
B-36	No stopping 	187
C-4	Left turn only 	50
D-1	Road with priority 	150
D-6	Pedestrian crossing 	156

base, constituting representatives of all groups of signs as regards shape and color (see Table I);

- the result is a weighted average (with importance proportional to the number of detected/recognized signs).

B. Tests Results

Scores from the tests are presented in Table II.

TABLE II

TESTS RESULTS

Sign	Detecting module	Recognizing module
	Detected signs	Recognized signs
A-7	57%	100%
B-21	100%	100%
B-36	88%	68%
C-4	100%	100%
D-1	53%	75%
B-6	88%	95%
Average	**83.0%**	**84.4%**

C. Comparing with Similar Systems

In Table III the effectiveness of our system is presented in comparison with four other systems:

M. Schneier ***Road Sign Detection and Recognition*** (see [8]) – a system detecting traffic signs based on rules determining the color, shape and supposed situation of the sign in a picture. Detects signs by comparing with templates in the database and following through image sequences.

M. Taha Khan ***Real–Time Recognition System For Traffic Signs*** (see [9]) – For detecting signs a row of graphical filters was used in the space of HSV colors (segmentation, searching for homogeneous areas, labeling, filtering the size of the detected object). Recognizing the sign involves fitting the shape of the image to one of the patterns.

M. Fifik, et al. ***Experiments with a Transform-based Traffic Sign Recognition System*** (see [10]) – The system search for signs with the help of algorithms for segmentation of colors and shapes. Distortions of the image are eliminated by applying Hough and Trace transforms. Recognizing the sign involves through categorizing detected area in one of the groups on the basis of features describing it.

J. Miura, et al. ***An Active Vision System for Real-Time Traffic Sign Recognition*** (see [11]) – The system uses two cameras. The first, a with wide-angle lens, is used for detecting candidate signs using information about the color, brightness and shape. The second camera, equipped with a telephoto lens, is aimed at the predicted item to picking up more accurate image. Recognizing the sign involves fitting it to one of the templates.

Detailed descriptions of these systems can be found in the articles included in the bibliography.

TABLE III

EFFECTIVENESS COMPARISON

System	Detecting module	Recognizing module
	Detected	Recognized
M. Schneier [8]	88%	78%
M. Taha Khan [9]	–	95%
M. Fifik, et al.[10]	–	86%
J. Miura, et al.[11]	97%	42%
Our system	**83.0%**	**84.4%**

IV. USE OF BIPOLAR SIMILARITY <EASURE

At present we are working on the construction and applications of a bipolar similarity measure based on IF-sets for the sign recognition process. The major advantage of this approach is the use of additional information on a margin of uncertainty in assigning intensity of similarity. The idea consists in simultaneously calculating the values of similarity and dissimilarity measure. This enable us to estimate our level of ignorance about similarity.

The concept of IF-sets was proposed by Atanassov [12]. It can be viewed as a generalization of fuzzy sets as introduced by Zadeh in 1965 [13]. An IF-set A in the universe X is defined as:

$$A = \{\langle x; \mu(x); \nu(x)\rangle : x \in X\}$$

where $\mu_A : X \to [0,1]$ and $\nu_A : X \to [0,1]$ are a membership and non-membership function, respectively. We assume that $0 \leq \mu(x) + \nu(x) \leq 1$ for each $x \in X$. $\mu(x)$ and $\nu(x)$ are understood as degree of membership and non-membership, respectively.

A consequence of the above inequality is the concept of hesitation margin (i.e. intuitionistic fuzzy index) defined as follows:

$$\pi_A(x) = 1 - \mu_A(x) - \nu_A(x)$$

$\pi_A(x)$ expresses the size of our ignorance as to the membership degree of x to an incompletely known fuzzy set modeled by A.

More details on the theory of Atanassov intuitionistic fuzzy sets can be found in [14] and [15], [16].

The first experiments show that the use of bipolar similarities increases the effectiveness of recognition of traffic signs.

A similar approach was successfully used in a lip shapes recognition system (see [17]). This kind of information modeling has also been successfully applied in many other fields such as intelligent data analysis (see [18]), group decision making (see [19]) and image processing (see [20]).

V. Summary

The traffic sign recognition system which we have created meets all expectations made of it: it operates in real time, it allows extension of the knowledge base, and it uses fuzzy methods in decision making. Thanks to solutions on which we decided in the design phase and our validation of the system in the course of implementation, it is possible to recognize signs of all types, and as can be seen in Table III, the system performs very well compared with other systems of this type.

The methodology described in this work, thanks to the application of the fuzzy rules and of image recognition based on the "from whole to detail" method, can be applied for detecting and recognizing practically any objects in a video stream (after carrying out analysis and modification of the steering rules). A detailed description of the system can be found in [21].

References

[1] H. Lars and R. Bernd, “Fuzzy techniques in image processing,” ser. Studies in Fuzziness and Soft Computing, M. Nachtegael and E. E. Kerre, Eds. Physica-Verlag Heidelberg, 2000, ch. Fuzzy color Processing, pp. 267–285.

[2] H. R. Kang, *Color technology for electronic imaging devices*. Bellingham, Washington: SPIE - The International Society for Optical Engineering, 1996.

[3] W. Pedrycz and F. Gomide, *Fuzzy Systems Ecgineering*. Wiley, 2007.

[4] O. Basir, H. Zhu, and F. Karray, “Fuzzy filters for image processing,” ser. Studies in Fuzzines and Soft Computing, M. Nachtegael, D. Van der Weken, D. Van De Ville, and E. E. Kerre, Eds. Springer-Verlag, 2003, vol. 122, ch. Fuzzy Based Image Segmentation, pp. 101–128.

[5] J. Bezdek, *Fuzzy models and algorithms for pattern recognition and image processing*, ser. The Handbooks of Fuzzy Sets Series. Springer, 2005.

[6] R. Choras, *Komputerowa Wizja, Metody interpretacji i identyfikacji obiektow. (in Polish)*. EXIT, 2007.

[7] F. Y. Shih, *Image Processing and Pattern Recognition: Fundamentals and Techniques*. Hoboken, New Jersey: John Wiley & Sons, Inc., 2010.

[8] M. Shneier, “Road sign detection and recognition,” *IEEE Computer Society International Conference on Computer Vision and Pattern Recognition*, 2005.

[9] M. Taha Khan, “Real–time recognition system for traffic signs,” Master’s thesis, Department of Computer Engineering, Dalarna University, 2008.

[10] M. Fifik, J. Turán, and L. Ovseník, “Experiments with a transform based traffic sign recognition system,” *Proc. of 17th International Conference on Systems, Signals and Image Processing*, 2010.

[11] J. Miura, T. Kanda, and Y. Shirai, “An active vision system for real-time traffic sign recognition,” *Proc. of Intelligent Transportation Systems, 2000*, pp. 52–57, 2010.

[12] K. Atanassov, *Intuitionistic Fuzzy Sets: Theory and Applications*. Heidelberg: Springer, 1999.

[13] L. Zadeh, “Fuzzy sets,” *Information and Control*, vol. 8, pp. 338–353, 1965.

[14] A. Pankowska and M. Wygralak, “On hesitation degrees in if-set theory.” in *Artificial Intelligence and Soft Computing. LNAI*, L. Rutkowski, J. Siekmann, R. Tadeusiewicz, and L. Zadeh, Eds. Springer, Heidelberg, 2004, vol. 3070, pp. 338–343.

[15] E. Szmidt and J. Kacprzyk, “Distances between intuitionistic fuzzy sets,” *Fuzzy Sets and Systems*, vol. 114, pp. 505–518, 2000.

[16] ——, “A similarity measure for intuitionistic fuzzy sets and its application in supporting medical diagnostic reasoning,” in *Artificial Intelligence and Soft Computing - ICAISC 2004*, ser. LNCS. Springer-Verlag Berlin Heidelberg, 2004, vol. 3070, pp. 388–393.

[17] K. Dyczkowski, “Application of if-sets to modeling of lip shapes similarities,” in *Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Methods*, ser. Communications in Computer and Information Science, E. Hüllermeier, R. Kruse, and F. Hoffmann, Eds. Springer Berlin Heidelberg, 2010, vol. 80, pp. 611–617.

[18] E. Szmidt and J. Kacprzyk, “An intuitionistic fuzzy set based approach to intelligent data analysis (an application to medical diagnosis),” in *Recent Advances in Intelligent Paradigms and Applications*, A. Abraham, L. Jain, and J. Kacprzyk, Eds. Springer, Heidelberg, 2002, pp. 57–70.

[19] ——, “An application of intuitionistic fuzzy set similarity measures to a multi-criteria decision making problem,” in *ICAISC 2006*, ser. LNCS (LNAI), L. Rutkowski, R. Tadeusiewicz, L. Zadeh, and J. Zurada, Eds. Springer, Heidelberg, 2006, vol. 4029, pp. 314–323.

[20] H. Bustincea, M. Pagola, P. Melo-Pinto, E. Barrenechea, and P. Couto, “Use of AtanassovÍs intuitionistic fuzzy sets for modelling the uncertainty of the thresholds associated to an image,” in *Fuzzy Sets and Their Extensions: Representation, Aggregation and Models*, ser. Intelligent Systems from Decision Making to DataMining, Web Intelligence and Computer Vision. Springer, Berlin, 2008.

[21] P. Gadecki and A. Kułakowski, “Knowledge based image recognition - an experiment. Traffic signs recognition system (in Polish),” Master’s thesis, Faculty of Mathematics and Computer Scence, Adam Mickiewicz University, Poznań, Poland, 2010.