University of Aveiro& VSB – Technical University of Ostrava
Departments of Mechanical Engineering

Detection of Traffic Lightsand Signs Using Image Processing

Writer: Ing. Jaromír Zavadil
Supervisor: Prof. Vítor Manuel Ferreira dos Santos

Aveiro: 28th June 2011

Content

1 Introduction

This work deals with the task of perceiving traffic signs for autonomous driving. It is concerned with the competition of fully autonomous robots that takes place in a track with the shape of a traffic road. The problem to be solved, concerns recognition of the symbols on traffic lights by a robot during the competition. There are a lot of techniques how to detect the traffic lights, but it is not so easy to solve this problem. Many conditions influence theimage recognition, especially the light condition.

The task of this work is to choose a several techniques and try to create solution useful for the competition.At first some fundamentals needed for this work are introduced. Solution and results of testing will follow.The program will be created using MATLAB. To solve this task two techniqueswill be used namely blob detection and the pattern recognition.

2 Fundamentals

Digital image representation in MATLAB and the basics of work with images using this tool will be outlined.

The power of MATLAB in the field of digital image processing is an extensive set of functions for processing multidimensional arrays. Moreover, when combined with Image Processing Toolbox it can even more extend the capability of the MATLAB environment. It makes many image processing operations easy to write, thus providing an ideal software prototyping environment.

In this work, the following tools are used: Image Acquisition Toolbox for image data acquisition from a camera, video preview and storing image data to the MATLAB Workspace; Image Processing Toolbox – set of functions for image processing

2.1 Digital Image Representation

In general, an image can be defined as two dimensional function,*f(x,y)*, where x and y are spatial coordinates and the amplitude of *f* at any pair of coordinates *(x,y)* is called the intensity of the image at that point.

Digital imagesarestored as a two dimensional arrays (matrices of size M x N), where each element of the array corresponds to a single pixel in the image. The convention used in Image Processing Toolbox uses the notation *(r,c)* to indicate rows and columns. The origin of the coordinate system is at *(r,c) = (1,1)*withthe range of *r* from *1 to M* and *c* from *1 to N* in integer increments.

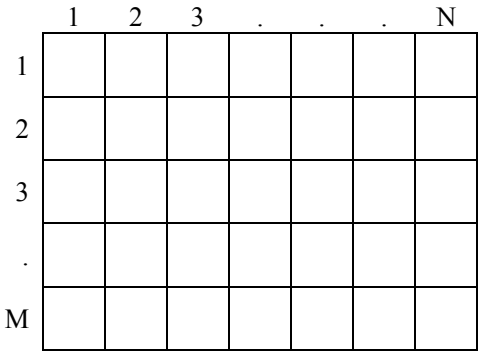


Figure 2.1 – Coordinate convention used in IPT

Image representation as a matrix:

$$f = \begin{bmatrix} f(1,1) & f(1,2) & \dots & f(1,N) \\ f(2,1) & f(2,2) & \dots & f(2,N) \\ \vdots & \vdots & & \vdots \\ f(M,1) & f(M,2) & \dots & f(M,N) \end{bmatrix}$$

Monochrome images are stored as two dimensional arrays where each element of this array represents intensity of the image at that point. The term gray level is used. Color images are stored in MATLAB as a three dimensional array. They are formed by combination of three two dimensional arrays, where each array represents an individual color component (e.g. RGB). So, each pixel in a color image is formed from the corresponding pixels of this three component images.

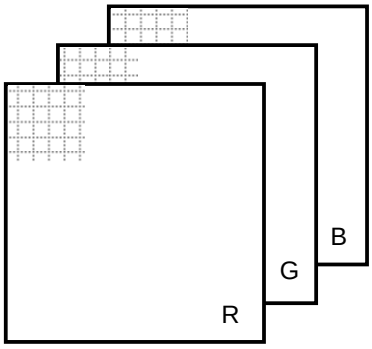


Figure 2.2 – Schema of components of RGB picture

Selecting Array Dimensions

`>> k = size(A, 1)` - size of A along its first dimension

`>> d = ndims(A)` - number of dimensions of array

2.2 Image types

There are four basic types of images definedin the Image Processing Toolbox:

- Binary images (black and white)
- Indexed images (pseudo color image)
- Grayscale images (intensity image)
- Truecolor images (RGB images)

Binary images contain only two discrete values 0 and 1, which represent black and white. These images are stored as a logical array. Indexed images consist of an array and color map matrix. The values in the array are direct indices into a colormap. A grayscale image is an array whose values represent intensities within some range. The range depends on used data class (see following chapter).Truecolor images are stored in MATLAB as an m-by-n-by-3 array that defines red, green and blue color components for each individual pixel.

2.3 Data Classes

MATLAB and Image Processing Toolbox support various data classes for representing pixel values. The most frequently used classes in image processing applications are double unit8 or unit16 and class logical. List of all supported classes can be seen inTable 2.1.

Table 2.1 Supported data classes

Name	Description
double	Double-precision, floating-point numbers in the approximate range -10^{308} to 10^{308} (8 bytes per element).
unit8	Unsigned 8-bit integers in the range [0,255] (1 byte per element).
unit16	Unsigned 16-bit integers in the range [0,65535] (2 bytes per element).
unit32	Unsigned 32-bit integers in the range [0,4294967295] (4 bytes per element).
int8	Signed 8-bit integers in the range [-128,127] (1 byte per element).
int16	Signed 16-bit integers in the range [-32768,32767] (2 bytes per element).
int32	Signed 32-bit integers in the range [-2147483648,2147483647] (4 bytes per element).
single	Single-precision floating-point numbers with values in the approximate range -10^{38} to 10^{38} (4 bytes per element).
char	Characters (2 bytes per element).
logical	Values are only 0 or 1 (1 byte per element).

2.4 Reading Images

To import an image from image file into the MATLAB environment the `imread()` functionis used. For example:

```
>>image = imread('image.bmp');
```

or

```
>>image = imread('D:\images\image.bmp');
```

When no path information is included, `imread()` reads the file from current directory.However any part of the directory path can be specified.

Function `size()` can be used for information about dimensions of an image:

```
>>[M, N] = size(image);
```

2.5 Displaying Images

Images can be displayedusing function `imshow()`.

```
>>imshow(image)
```

When anotherimage is displayed, the image on the screen is replaced with the new one. For keeping the first image and showing more images, function `figure` can be used to create a new window:

```
>>figure, imshow(image2)
```

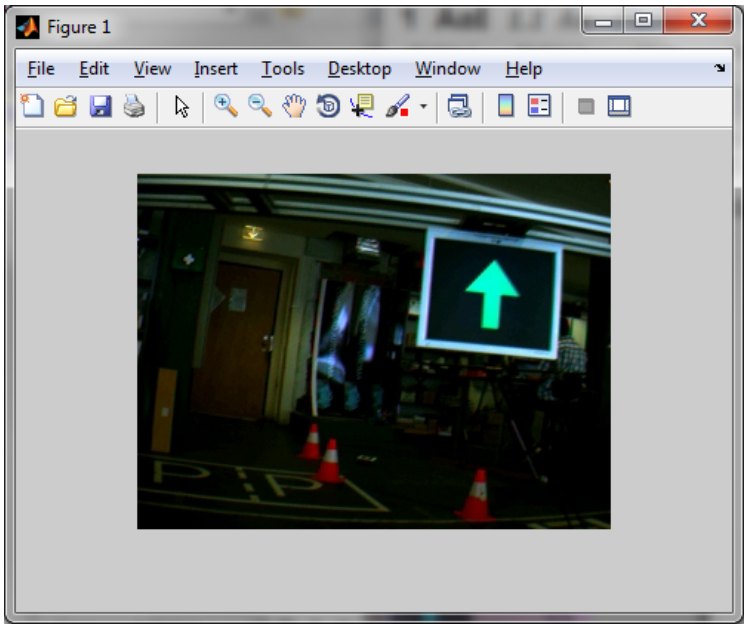


Figure 2.3 – Dysplaying images using imshow()

2.6 Writing Images

`Imwrite()` function is used to save data from the MATLAB workspace to a graphic file. See the following example:

```
>>imwrite(image, 'image.bmp');
```

or

```
>>imwrite (image, 'D:\images\image.bmp');
```

When no path information is included, `imwrite()` saves the file to the current directory. Supported data formats can be seen inTable 2.2.

Table 2.2 –Supported data formats

Format		**Extension**
BMP	Windows Bitmap	.bmp
GIF	Graphic Interchange Format	.gif
HDF4	Hierarchical Data Format	.hdf
JPEG	Joint Photographic Experts Group	.jpg, .jpeg
JPEG 2000	Joint Photographic Experts Group 2000	.jp2, .jpx
PBM	Portable Bitmap	.pbm
PCX	Windows Paintbrush	.pcx
PGM	Portable Graymap	.pgm
PNG	Portable Network Graphic	.png
PNM	Portable Anymap	.pnm
PPM	Portable Pixmap	.ppm
RAS	Sun Raster	.ras
TIFF	Tagged Image File Format	.tif, .tiff
XWD	X Window Dump	.xwd

3 Image processing using MATLAB

In this chapter some MATLAB functions for image processing as well as some techniques used in this work will be shown. First part is about image acquisition, the second contains a brief introduction to color spaces and last part is about blob analysis using `regionprops()` function.

3.1 Functionsfor image acquisition

Image Acquisition Toolbox is used for image acquisition using MATLAB. This toolbox provides a set of functions to support a wide range of image acquisition operations like acquiring images through many types of image acquisition devices, viewing a preview of the live video stream, bringing the image data into the MATLAB workspace etc.

imaqhwinfo()

`Imaqhwinfo()` provides information about available image acquisition hardware. For example following command returns information about all accessible devices through a particular adaptor (for more information look in MATLAB Product Help):

```
>> info=imaqhwinfo('winvideo');

info =

       AdaptorDllName: [1x81 char]
    AdaptorDllVersion: '4.0 (R2010b)'
          AdaptorName: 'winvideo'
DeviceIDs: {[1]  [2]}
DeviceInfo: [1x2 struct]
```

For information about supported formats can be used:

```
>>info.SupportedFormats
```

videoinput()

This command create video input object. Syntax is following:

```
>>vid = videoinput('winvideo',1,'IYUV_640x480');
```

where ‘winvideo’ specifies the name of used adaptor, 1 is device ID and last argument specifies requested video format.

preview()

Preview of live video stream. Syntax is simple:

```
>>preview(vid)
```

For close preview can be used`closepreview()`command.

```
>>closepreview (vid)
```

getsnapshot()

Return single image frame from the video input object. Frame is returned to the MATLAB workspace.

```
>>image = getsnapshot(vid);
```

3.2 Color Spaces

MATLAB represents colors as RGB values in an RGB image or in an indexed image, where the colormap is stored in RGB format. But there are more other color spaces supportedbythe toolbox. These are NTSC, ZCbCr, HSV, CMY, CMYK and HSI color spaces. I will introduce here RGM and HSV color spaces.

RGB Color Space

RGB color space usually is shown graphically as an RGB color cube. Where each color pixel corresponding to the red, green and blue components of an RGB image. If all components are identical, the result is a gray-scale image.

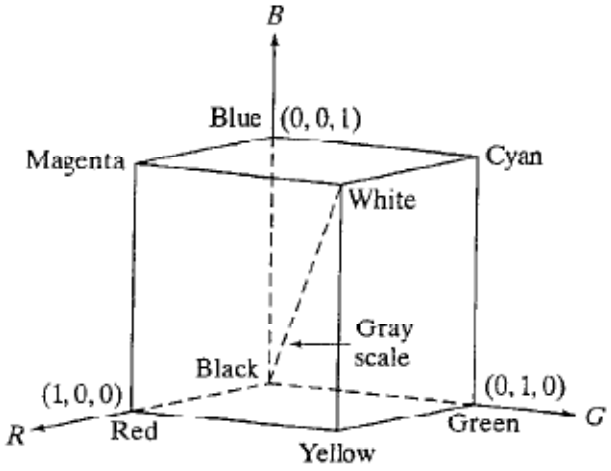


Figure 3.1 – RGB cube [1]

Extraction of RGBcomponent images:

```
>>fR = rgb_image(:, :, 1);
>>fR = rgb_image(:, :, 2);
>>fR = rgb_image(:, :, 3);
```

HSV Color Space

HSV color space represents color image as a combination of three components – hue, saturation and value. This color space is closer than the RGB system to the way in which people experience and describe color sensations.

Hue is expressed as an angle around a color hexagon, typically with red as the 0°. Value is presentedby axis of the cone. The value V = 0 on one end of the axis is black and V = 1 on the opposite end is white. Saturation is measured as a distance from the V axis. See on image below.

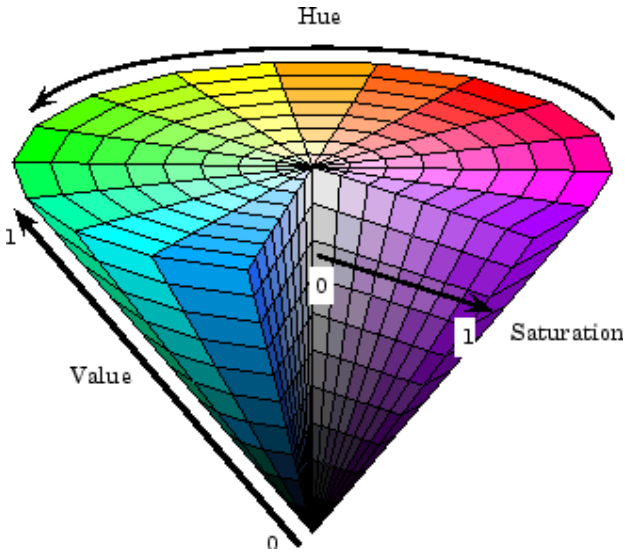


Figure 3.2 HSV color space [MathWorks]

Converting from RGB to HSV color space

```
>>hsv_image=rgb2hsv(image_rgb);
```

3.3 Blob analysis

Thistechniquedetects regions in the image that have different surrounding. In the Image processing Toolbox there are several functions which can be used for this task.Combination of `bwlabel()` and `regionprops()` functions can be used to get information about blobs.

bwlabel()

Label connected components in 2D binary image. For example

```
>>L_image = bwlabel(image, 4);
```

returns a matrix 'L_image', of the same size as 'image', containing labels for the connected objects in 'image'. The elements of L_image are integer values, where the pixels labeled 0 are the background;the pixels labeled 1 make up one object; the pixels labeled 2 make up a second object; and so on.The value4 specifies 4-connected object (if the argument is omitted, it defaults to 8).

Full description of `bwlabel()`can be found in MATLAB Product Help. In older versions of MATLAB, `bwlabel()` command was necessary for using `regionprops()`. But in MATLAB 2010 do not have to be used`bwlabel()` before calling `gegionprops()`. Also in new MATLAB the function `bwlabel()` was replaced by `bwconncomp()` which uses less memory. But in this work `bwlabel()`was used because of compatibility with older versions of MATLAB.

regionprops()

Is function for measuring properties of image regions. Syntax is following:

```
>>  properties = regionprops(L_image, 'all')
```

This command measures all properties for each labeled region in the label matrix L_image. Or required properties can be specified using comma separated list of strings. Full description of `regionprops()` and its properties can be found in Help. Only the list of properties used in this work will be shown here.

Table 3.1 – Used properties of regionprops()

Area	Eccentricity	FilledArea
BoundingBox	EulerNumber	Orientation
Centroid	Extent	Solidity

Other used functions

ismember() – returns array elements that are members of set

```
>>array = ismember(A, B);
```

Array is the same size as A and contains logical 1 where the elements of A are in the set B, and logical 0 elsewhere.

numel() – number of elements in array

```
>> n = numel (A)
```

find() – returns indices and values of nonzero elements. For example you can specify elements of array. Following command returns indices of the array which have value bigger than 200.

```
>>idx = find([array]> 200);
```

4 Application 1 (Blob Analysis)

In this chapter specific problem to be solved as well as a solution and results of this work will be introduced. This work belongs to the task of autonomous driving. In this case, it is concerned with the competition of fully autonomous robots. At first an introduction to this task will be made and then the solution and the results will be presented.

4.1 Competition

The problem to be solved, concerns recognition of the light signaling for a robot during Autonomous Driving Competition Robotica 2011. Full text of the rules and technical specifications can be found on the website of the competition (http://robotica2011.ist.utl.pt/). Here will be introduced only parameters concerning the light signals.

The task was to recognize five different symbols, which are presented on the signaling panel (17' TFT panel with resolution 1024x768 pixels). The colors used for signals are red, green and yellow on a black background. Dimensions of symbol image area are 600x600 pixels.A scheme of signaling panels as well as images of the symbols presented in these panels can be seen below (Figure 4.1 and Figure 4.2).

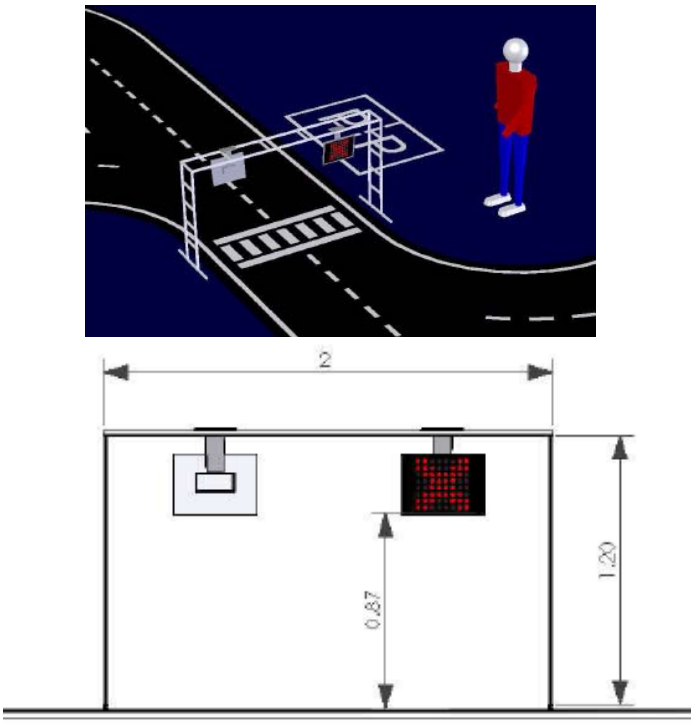


Figure 4.1 – Signaling panels [2]

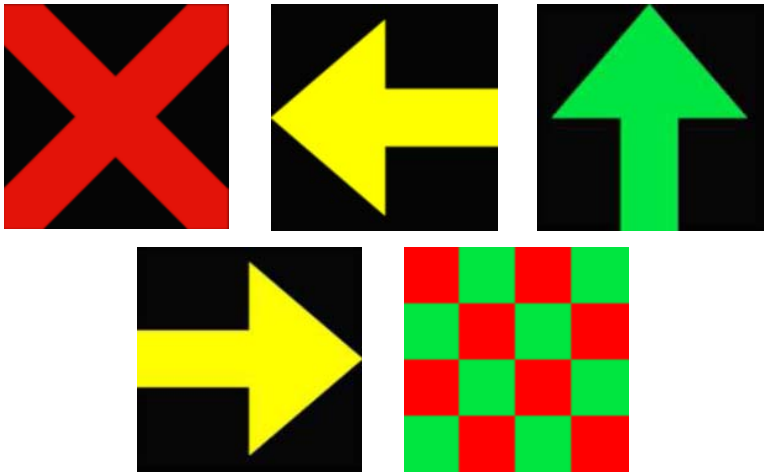


Figure 4.2 – Symbols presented in the panels

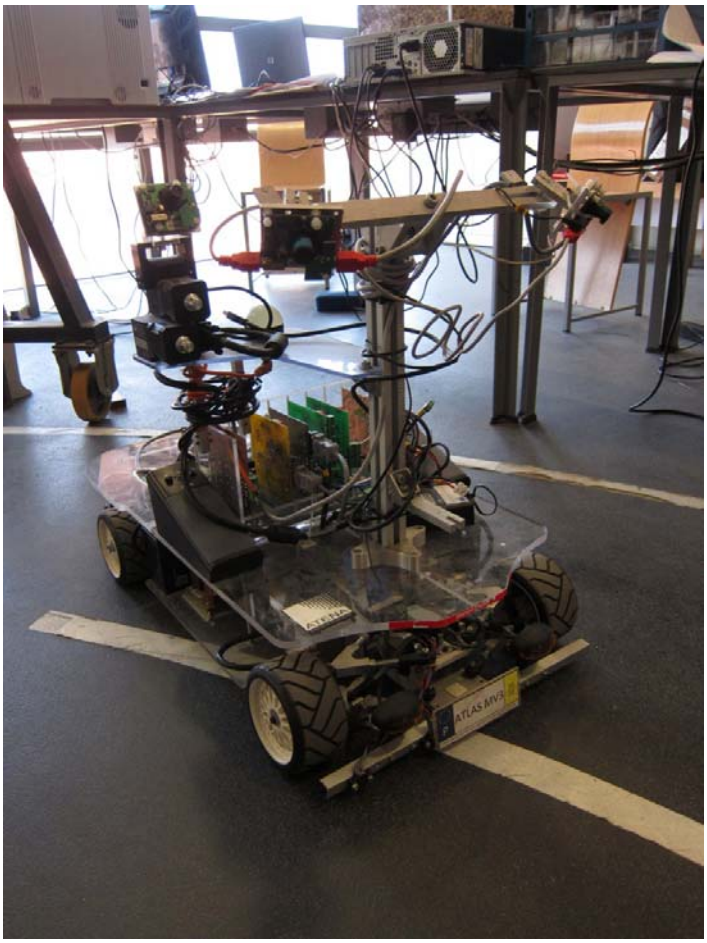


Figure 4.3 – Robot used for competition

4.2 Solution

To solve this task a technique that consists of color segmentation and blob detection was chosen.To recognize above-mentioned symbols the `regionprops()` functions will be used. Here it will be described how the program works as well as some important parts of the used algorithm. Full code can be found in the appendix.

Running the program

The aim of the created program is to capture and process images from a connected camera. Captured images and results of recognition are stored. Simultaneously with the running program the preview of video stream can be seen as well as the processing of the image. The control of the program is done using menu (Start/Stop). After pressing the Start button the program will start running in an endless loop until Stop button is pressed. Saved images as well as text file with results of processing are stored in the same directory as the program. Captured images are stored in JPEG format under the generic name img*.jpg.Numbering starts from 1 and continues until the program is stopped. The results of image recognition are stored in a text file named Statistic.txt. For each run of the program one text file is created with results of processing all images acquired during the test (until you press Stop button).

TXT file includes the name of each acquired picture and statistic for these pictures. So, later any statistic with these data can be made (e.g. in Excel). Example of this text file is shown in Table 4.1. The first column contains the name of the corresponding image file. Next three columns (Red, Green, Yellow) contain values (number of pixels) obtained only using color segmentation. Next three columns (bRed, bGreen, bYellow) contain number of pixels obtained using blob analysis. The last column (Symbol) specifies the type of recognized symbol. Results of some tests can be seen in graphs. Tables of data can be found in appendix.

Table 4.1 – Example of a text file with statistics

Name	Red	Green	Yellow	bRed	bGreen	bYellow	Symbol
img1.jpg	708	203	393	427	0	0	1
img2.jpg	628	503	293	449	0	0	1
img3.jpg	227	862	294	0	354	0	2
img4.jpg	301	817	341	0	343	0	2
img5.jpg	689	683	425	355	369	0	4
img6.jpg	705	743	374	365	381	0	4

Description of the algorithm

The first step after acquisition of an image is the conversion to HSV color space and color segmentation. According to the used symbols I am looking for red, green and yellow components. Used ranges for each color component are inTable 4.2. The values were reached by testing of several sets of images (2000 images approximately).After color segmentation follows the blob analysis using `regionprops()`.The last step is the determination of the presence of any symbol in the picture. Determination is done by "if" sequence.The next pages will describe important parts of MATLAB code. Flowchart will follow.

Table 4.2– Used color ranges values (obtained by testing approximately 2000 images)

	H	S	V
red	0 - 0.07 & 0.96 - 1	> 0.5	> 0.5
green	0.26 - 0.54	> 0.4	> 0.4
yellow	0.15 - 0.2	> 0.5	> 0.4

Color segmentation for red component

```
value = ((image_H>=0&image_H<0.07)|(image_H>=0.96 &image_H<=1)) ...
&image_S>0.5 &image_V>0.5;
```

Blob analysis for red component

At first searching for blobs bigger than 40 pixels with eccentricity smaller than 0.97. This step is searching for smaller blobs of symbol End of the Trial.

```
idx = find([BlobStats.Area] > 40 & [BlobStats.Eccentricity] < 0.97);
```

If the number of found blobs is smaller than 7 searching for the Red Cross (bigger blob).

```
idx = find([BlobStats.Area] > 200 & [BlobStats.Eccentricity] < 0.7 ...
& [BlobStats.Solidity] < 0.8 & [BlobStats.Solidity] > 0.4 ...
& [BlobStats.Extent] > 0.3 & [BlobStats.Extent] < 0.8);
```

In the end, there is a sum of pixels that satisfies the blob analysis.

Color segmentation for green component

```
value = (image_H>0.26 &image_H<0.54) &image_S>0.4 &image_V>0.4;
```

Blob analysis for red component

Searching for blobs bigger than 40 pixels with eccentricity smaller than 0.97 (symbolEnd of the Trial).

```
idx = find([BlobStats.Area] > 40 & [BlobStats.Eccentricity] < 0.97);
```

If number of found blobs is smaller than 7 searching for the Green Arrow (bigger blob)

```
idx = find([BlobStats.Area] > 200 & [BlobStats.Eccentricity] < 0.9 ...
& [BlobStats.Extent] > 0.4 & [BlobStats.EulerNumber] > -20 ...
& [BlobStats.Solidity] < 0.83 ...
& ([BlobStats.Orientation] < -60 | [BlobStats.Orientation] > 60));
```

In the end is sum of pixels that satisfy the blob analysis.

Color segmentation for yellow component

```
value =(image_H>0.15 &image_H<0.2) &image_S>0.5 &image_V>0.4;
```

Blob analysis for yellow component

```
idx = find([BlobStats.Area] > 200 & [BlobStats.Eccentricity] < 0.9 ...
& [BlobStats.Solidity] < 0.83 & [BlobStats.Extent] > 0.35 ...
& [BlobStats.Orientation] > -25 & [BlobStats.Orientation] < 25 ...
& [BlobStats.EulerNumber] > -8);
```

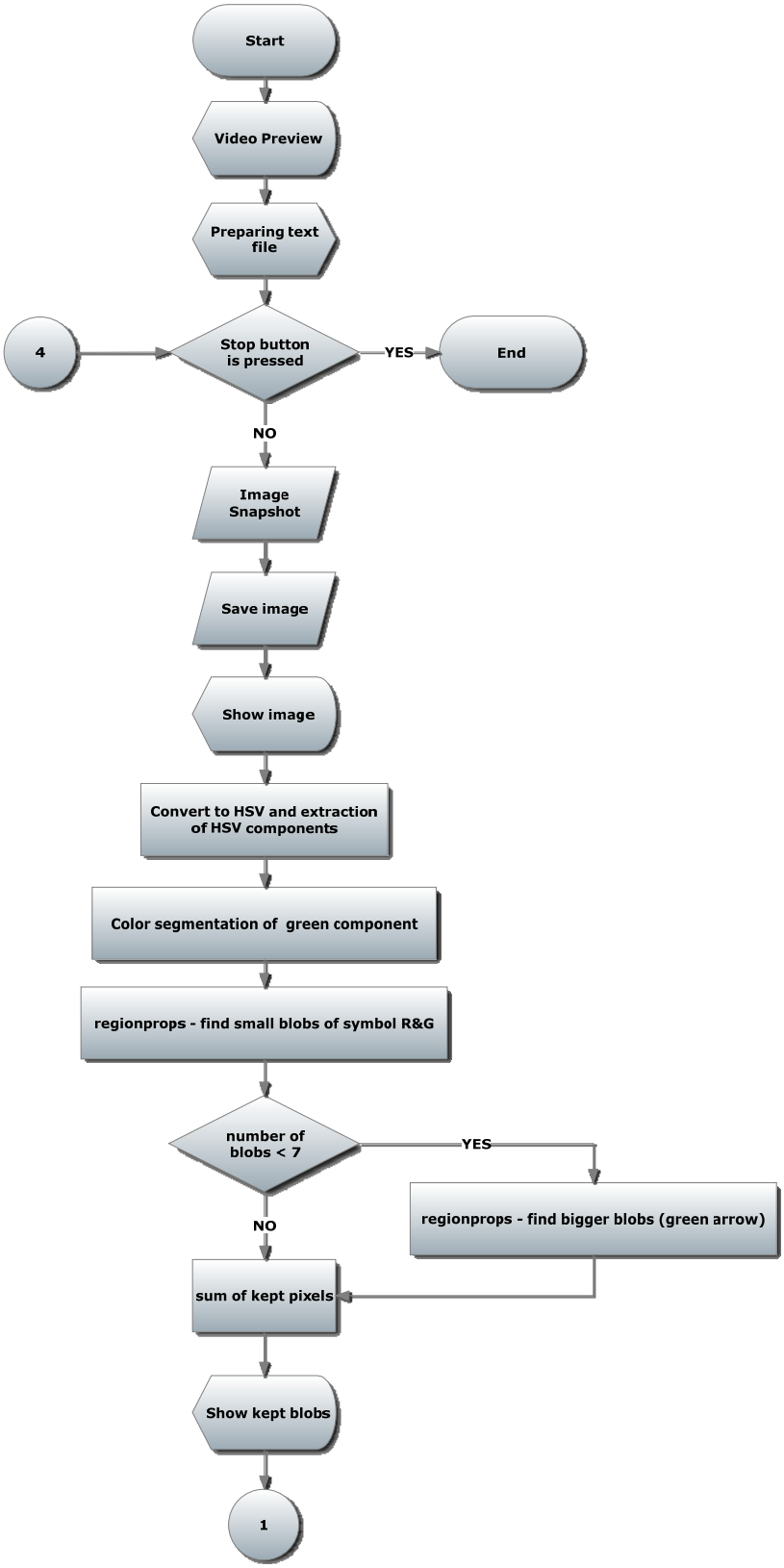
Determination of an arrow direction

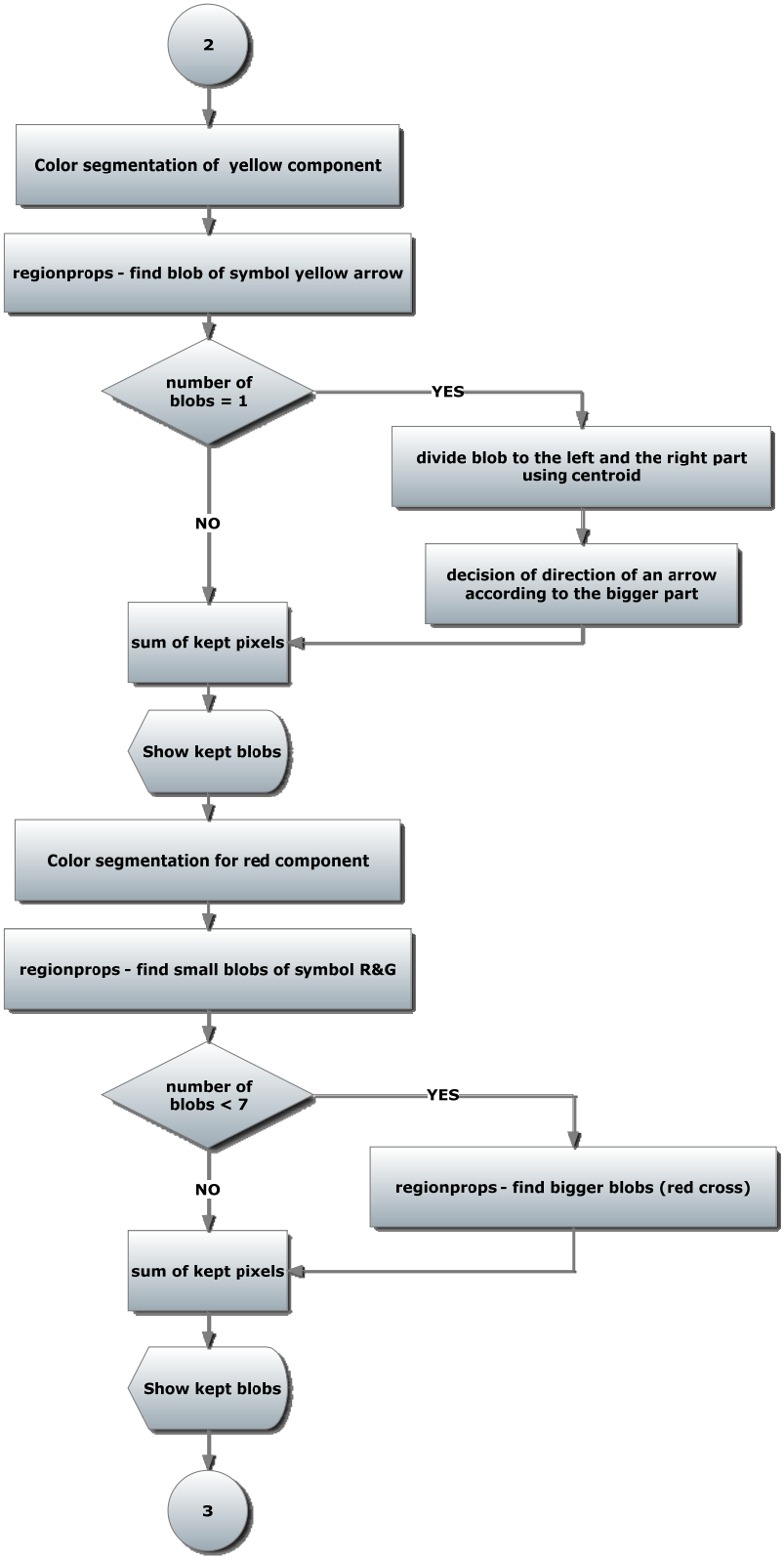
This task is solved by dividing the arrow to left and right part using centroid and comparing the number of pixels in left and right parts. Part with higher number of pixels specifies the direction.

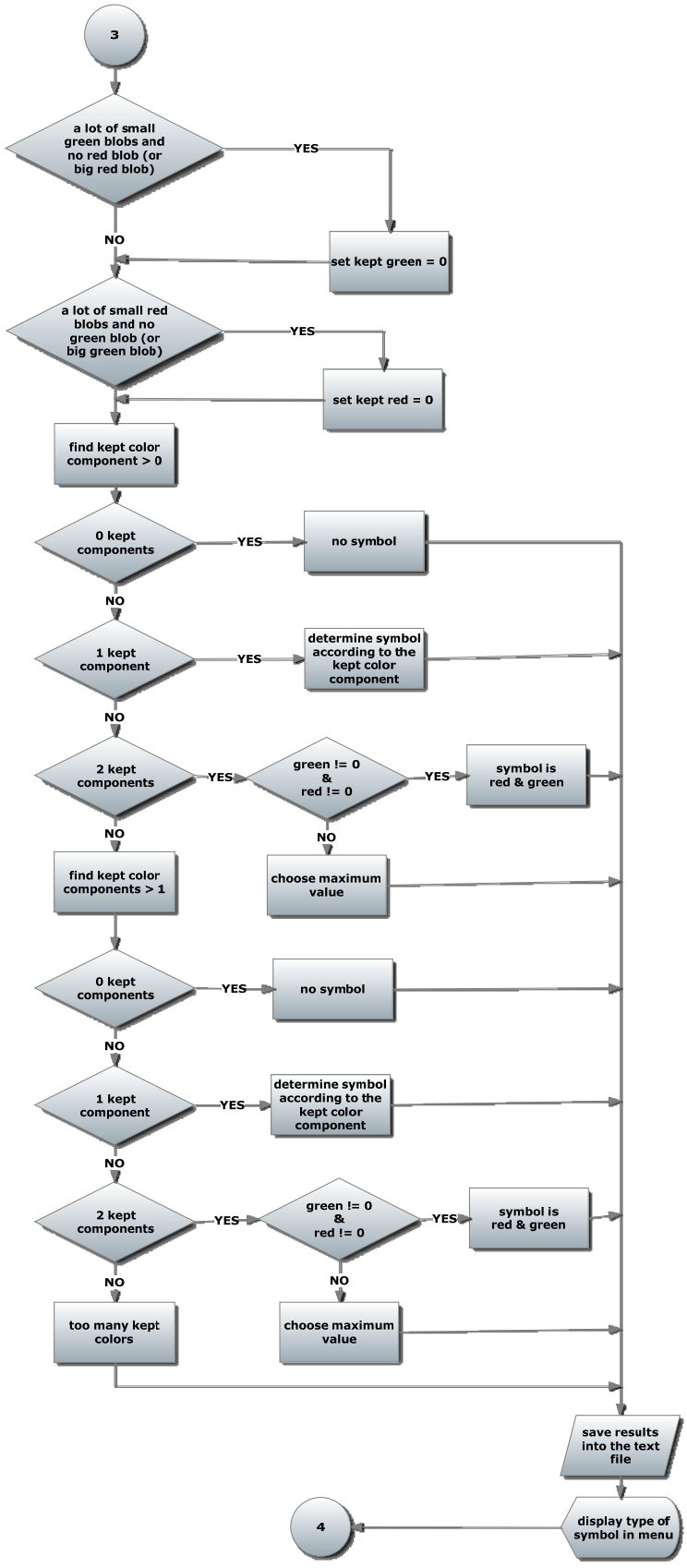
Determination of dominant color

The last step is the comparison of kept pixels of each color. If there is only one color, the symbol is determined by this color. If there are two kept colors and these are red and green the symbol is Red and Green. If there are two other colors program chooses the bigger value. If there are components of each color component then the process is done again but for one percent bigger values.

Flowchart


Start
Video Preview
Preparing text file
4
Stop button is pressed
YES
End
NO
Image Snapshot
Save image
Show image
Convert to HSV and extraction of HSV components
Color segmentation of green component
regionprops - find small blobs of symbol R&G
number of blobs < 7
YES
regionprops - find bigger blobs (green arrow)
NO
sum of kept pixels
Show kept blobs
1


2
Color segmentation of yellow component
regionprops - find blob of symbol yellow arrow
number of blobs = 1
YES
divide blob to the left and the right part using centroid
decision of direction of an arrow according to the bigger part
NO
sum of kept pixels
Show kept blobs
Color segmentation for red component
regionprops - find small blobs of symbol R&G
number of blobs < 7
YES
regionprops - find bigger blobs (red cross)
NO
sum of kept pixels
Show kept blobs
3


3
a lot of small green blobs and no red blob (or big red blob)
YES
NO
set kept green = 0
a lot of small red blobs and no green blob (or big green blob)
YES
set kept red = 0
find kept color component > 0
0 kept components
YES
no symbol
NO
1 kept component
YES
determine symbol according to the kept color component
NO
2 kept components
YES
green != 0 & red != 0
YES
symbol is red & green
NO
NO
find kept color components > 1
choose maximum value
0 kept components
YES
no symbol
NO
1 kept component
YES
determine symbol according to the kept color component
NO
2 kept components
YES
green != 0 & red != 0
YES
symbol is red & green
NO
NO
too many kept colors
choose maximum value
save results into the text file
4
display type of symbol in menu

4.3 Results

Some results of testing of the created program will be presented here. Because of a lot of measured values, the tables can be found in the appendix. The results in graphs can be seen here. In the beginning it should be remarked that the results depend very much on settings parameters of the camera. The good operation of the program also depends on light conditions. Sometimes, depending on the light conditions, the Black Level settings of the camera have to be changed. Usually when the pictures are darker, the results are better.

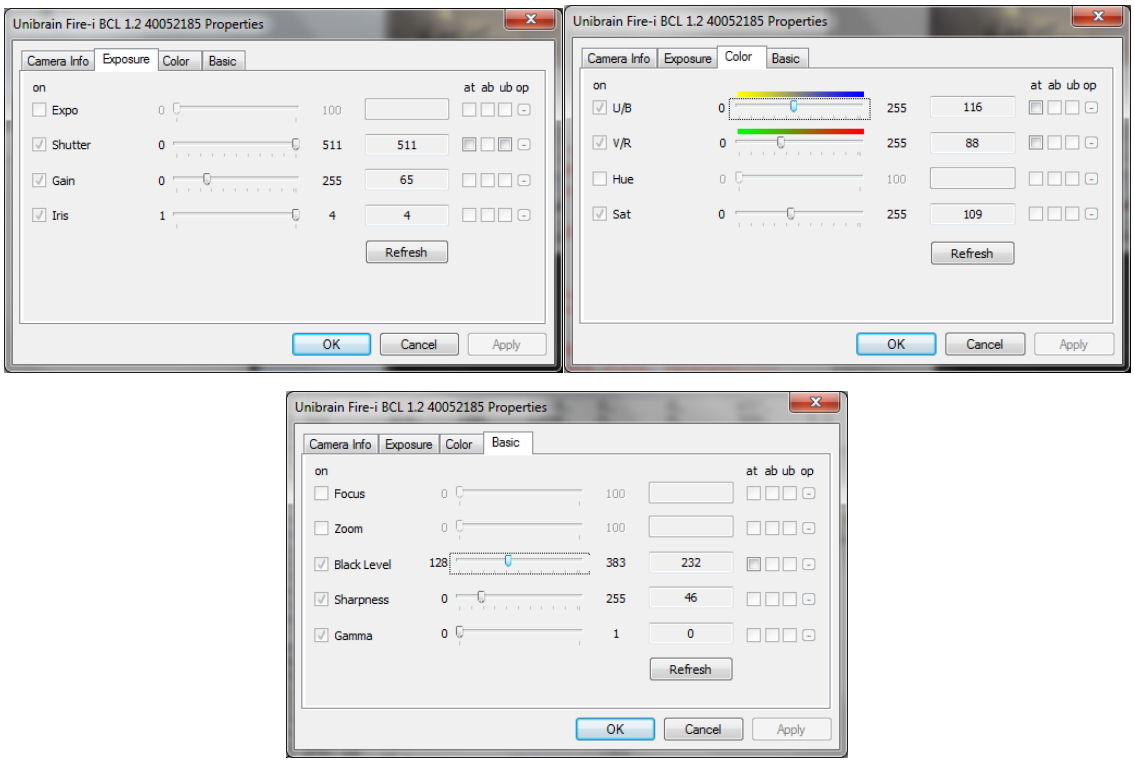


Figure 4.4 – Used settings of the camera

It should be noted that standardFireWire color digital camera was used in this test,specifically unibrain Fire-i™ Digital Camera.Set in following video mode: 320x240 YUV 4:2:2 (16bits).Used settings of the camera can be seen in Figure 4.4. The program was running on a standard 2.26GHz dual-core laptop with 4GB RAM.Windows 7 64bit operating system and MATLAB R2010b were used.

First tests

First results of two tests with brighter pictures will be shown. The first one was done closer to the lights while the second one was done in a variety of distances and at various angles. . It should be remarked that the background of the lights was very complex according to a number of objects (red cones etc.) with a similar color as symbols indicated on the Signaling panel. Also the first two tests were done with wider color range for yellow color component.

Table 4.3 – Used color ranges values (obtained by testing approximately 2000 images)

	H	S	V
red	0 - 0.07 & 0.96 - 1	> 0.6	> 0.4
green	0.26 - 0.54	> 0.4	> 0.4
yellow	0.11 - 0.2	> 0.5	> 0.4

Overall 110 images were capturedin the first test. One image from four without any symbol was recognized as a red (see Figure 4.6). Four images from forty four Red & Green symbols were recognized as a red (see Figure 4.7). The rest of the tested images were recognized well. Complete results can be found in the appendix. On the next page the Table 4.4 shows summary of results.

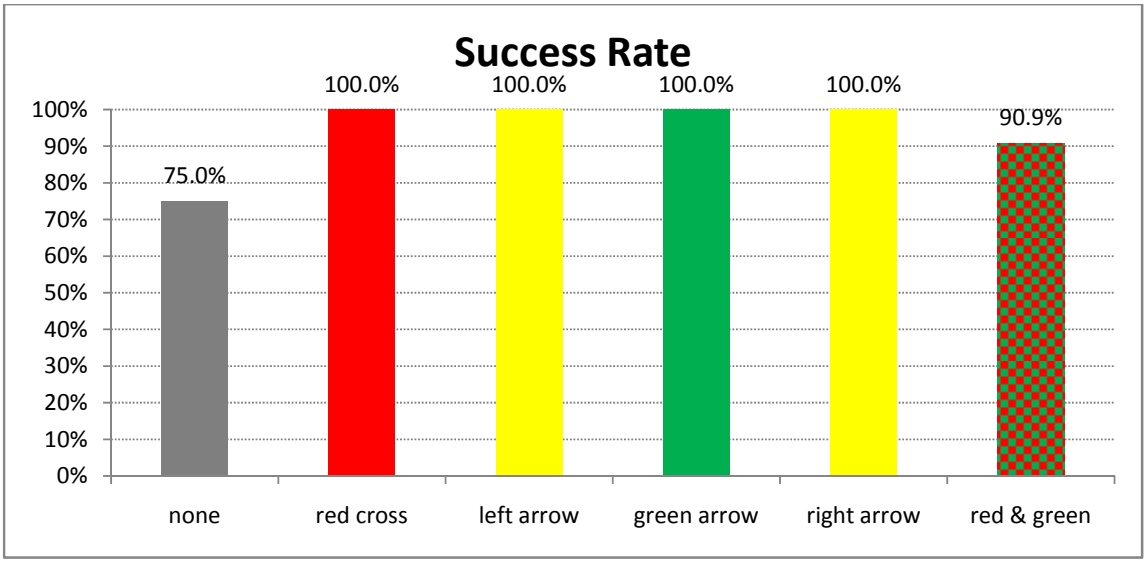


Figure 4.5 – Results of the first test(0526) using 110 images

Table 4.4 Results of the first(0526) test

Tested Images							Results	
red cross	left arrow	green arrow	right arrow	red & green	without light	total	missed	wrong
14	9	17	22	44	4	110	0	5

In the case of a picture without a symbol, problem was caused by the red cones in the background. The problem with red cones was partially solved by modifying the hue range for color segmentation of red.

Figure 4.6 – Example of false positive

Another problem was with Red & Green symbols. Sometimes they were recognized only as red. It was also caused by red cones. This problem was also solved by adding a condition.

Figure 4.7 – Problem with red cones in background

Figure 4.8–Examples of well recognized symbols

Second test

Second test was more complex, with bigger variety of distances and angles from which the pictures were taken. Again, there was a lot of demanding objects in background of symbols or in images without symbol. This fact influenced much the result of the test. Especially in case when there were not any symbols in the image.In this test 161 pictures were captured. 6 false positive influenced the results of symbol recognition.But there were more false positive which did not influence the result of the test.

Table 4.5 Results of the second test (0526/2)

Tested Images							Results	
red cross	left arrow	green arrow	right arrow	red & green	without light	total	missed	wrong
53	21	20	33	28	6	161	5	5

All problems with false positive occurred only in the yellow color range (see Figure 4.10).Subsequently, problems with undesirable yellow objects were solved by changing the

hue range for color segmentation of yellow. Also analysis of pictures with red cones was improved. New conditionsinto the code were added.

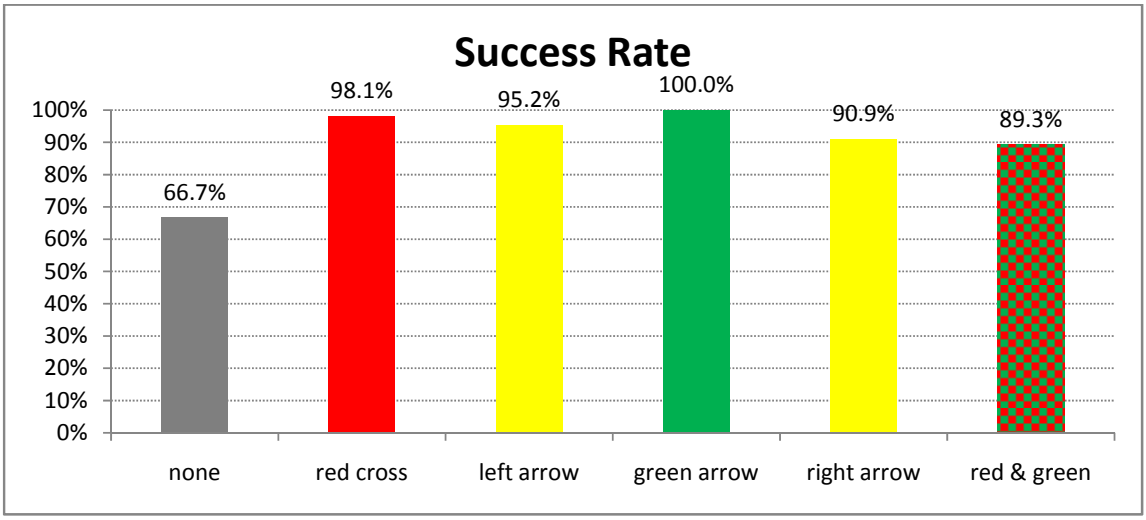


Figure 4.9 – Results of the second test (0526/2) using 161 images

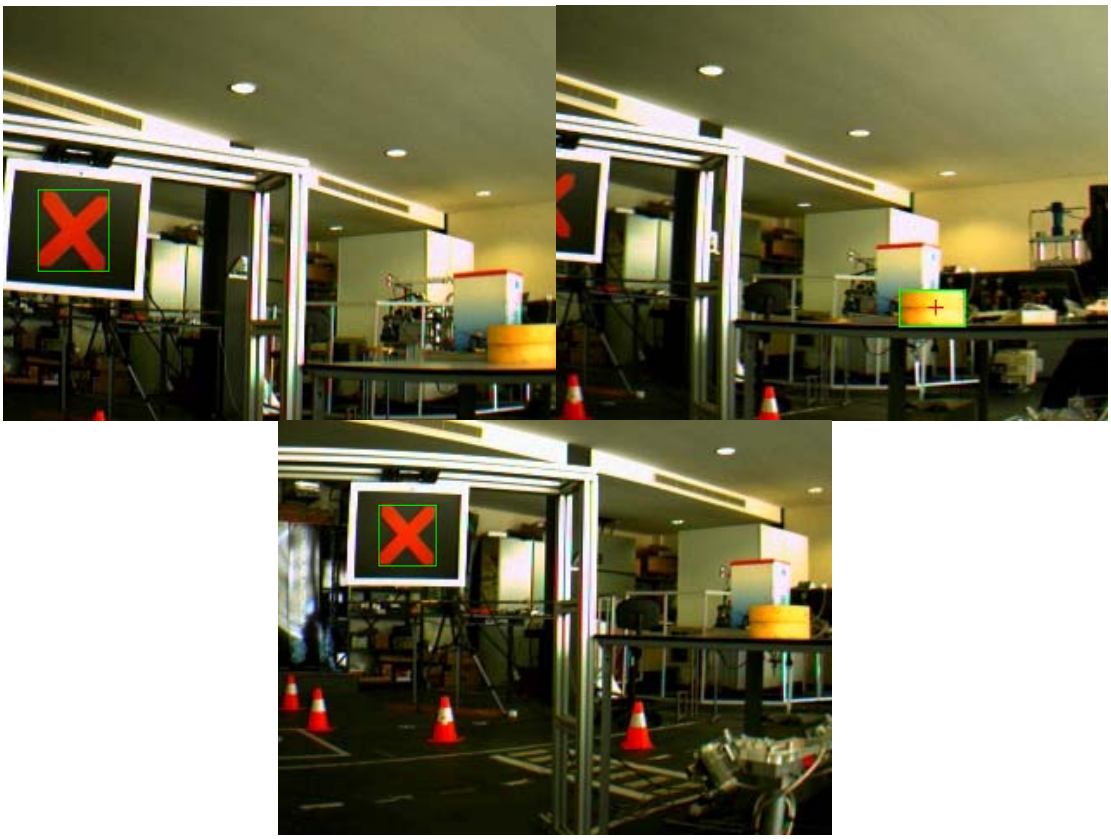


Figure 4.10 –First well recognized image with the Red Cross, then bad recognized imagewithout any symbol and last image with Red Crossagain

Final test

As mentioned, the final test was done with darker images and also with changed hue range for yellowcomponent. Problem with false positives in this color rangewas reduced significantly.

Table 4.6 – Used color ranges values (obtained by testing approximately 2000 images)

	H	S	V
red	0- 0.07 & 0.96 - 1	> 0.5	> 0.5
green	0.26 - 0.54	> 0.4	> 0.4
yellow	0.15 - 0.2	> 0.5	> 0.4

As can be seen in the graph, the results are better,especially the results of recognition on images without any symbols are much better.In this case success rate was 95%. Worse result of recognition of the green arrow was caused by complex angles during capturing images. However for a real situation during competition these angles are not so relevant.The robot should not get to such a position. In other cases there was problem with a big distance between the camera and the signaling panel. Recognition is more successful in distance under two meters from the signaling panel. In distances over two and half meters from signaling panel the success rate declines, especially with symbol Red & Green Chessboard. In this test 2images were recognized wrong and34symbols weremissed. In total 450 images were processed.

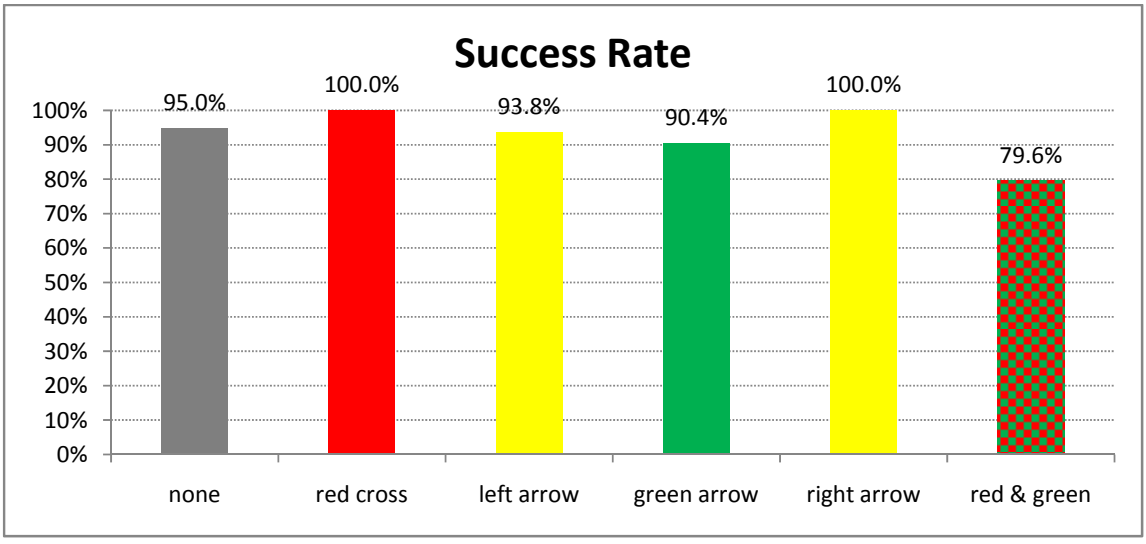


Figure 4.11 – Results of the final test (0603)using 450 images

Table 4.7– Results of the final test (0603)

Tested Images							Results	
red cross	left arrow	green arrow	right arrow	red & green	without light	total	missed	wrong
92	113	73	49	103	20	443	34	2

Some examples of missed symbols can be seen on following pictures (Figure 4.12). In most cases it was caused by big distance of camera from the signaling panel or by blurred image.Maximum distance for symbols of Green Arrow and Red Cross is approximately 3 meters. However maximum distance for reliablerecognition of symbol Red & Green is only 2 meters. Maximum distance for symbol Yellow Arrow is approximately 2.5 meters. On Figure 4.13 false positives that occurred during this testcan be seen. First the R&G Chessboard was wrongly recognized like as a Green Arrow.Second, there was yellow blob detected on the wall.

Figure 4.12 – Examples of symbols which were not recognized by reason of big distance or angle.

Figure 4.13 – False positives

4.4 Proposals of improvement

Finally several proposals to future improvement of the created program will be mentioned. First, it might be better to exclude the bottom quarter (or third) of the image from processing. In this area there should not be any traffic light. Hence, it is highly probable that the objects found in this part of the image will be false positives.Second, it might be good to compare actual image with previous or with more previous images.Tracking could improve accuracy of detection.Finally, it could be better to use pattern recognition or any more complex method as a hard features or neural networks.

5 Application 2 (Pattern Recognition)

To improve the success rate of traffic lights detection the Pattern Recognition will be used in this chapter.There are a lot of approaches how to compute distance of a given object from the reference set. The minimum distance classifier will be used in this work. The basic isEuclidean distance

$$d_E = \sqrt{(X-Y)^T(X-Y)}$$

where X is the reference sample and Y is a object to be classified.

Next well known distance is the Mahalanobis distance. It differs from the Euclidean distance in that it takes into account the correlations of the data set and is scale invariant.

$$d_M(X,Y) = \sqrt{(X-Y)^T \sum^{-1} (X-Y)}$$

whereΣ is the covariance matrix.

$$\sum = \begin{bmatrix} \sigma_{11}^2 & & & \\ & \sigma_{22}^2 & & \\ & & \ddots & \\ & & & \sigma_{n\,n}^2 \end{bmatrix}$$

Along the diagonal of tis matrix are the variances of the data set.

In MATLAB there is the mahal() function for computing the Mahalanobis distance. Syntax of this function is following

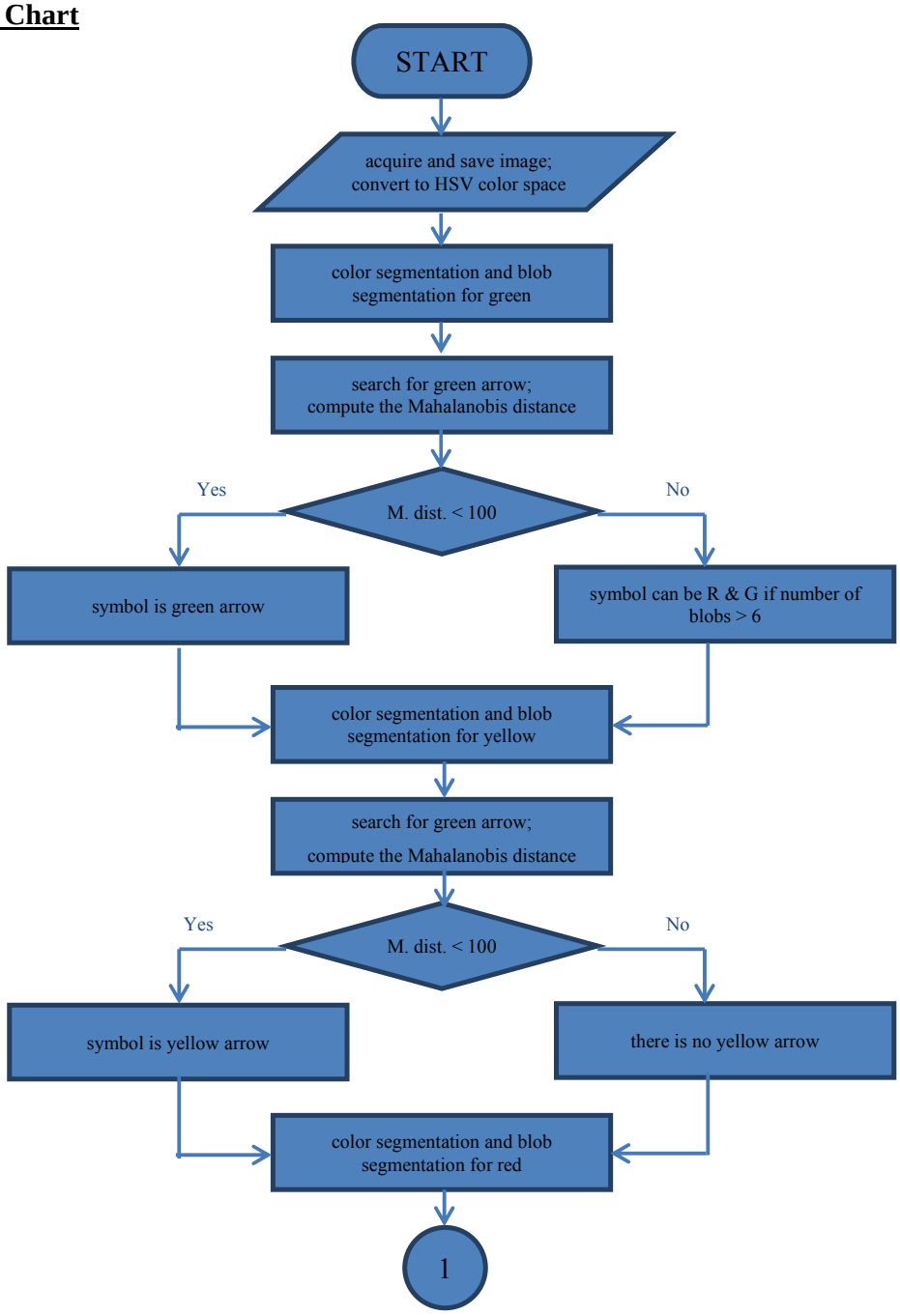
```
>> d = mahal(X, Y);
```

X and Y must have the same number of columns, but can have different numbers of rows. X must have more rows than columns. For more information see MATLAB Help.

5.1 Proposed solution

The Mahalanobis distance will be used to compute the distance of candidates in this application. Due to bigger accuracy of Pattern Recognition the color ranges of the green and the yellow can be changed to wider ones (Table 5.1). After color segmentation a Blob Analysis will be used to find the candidates. After that the Mahalanobis distance will be computed for these candidates.

Flow Chart



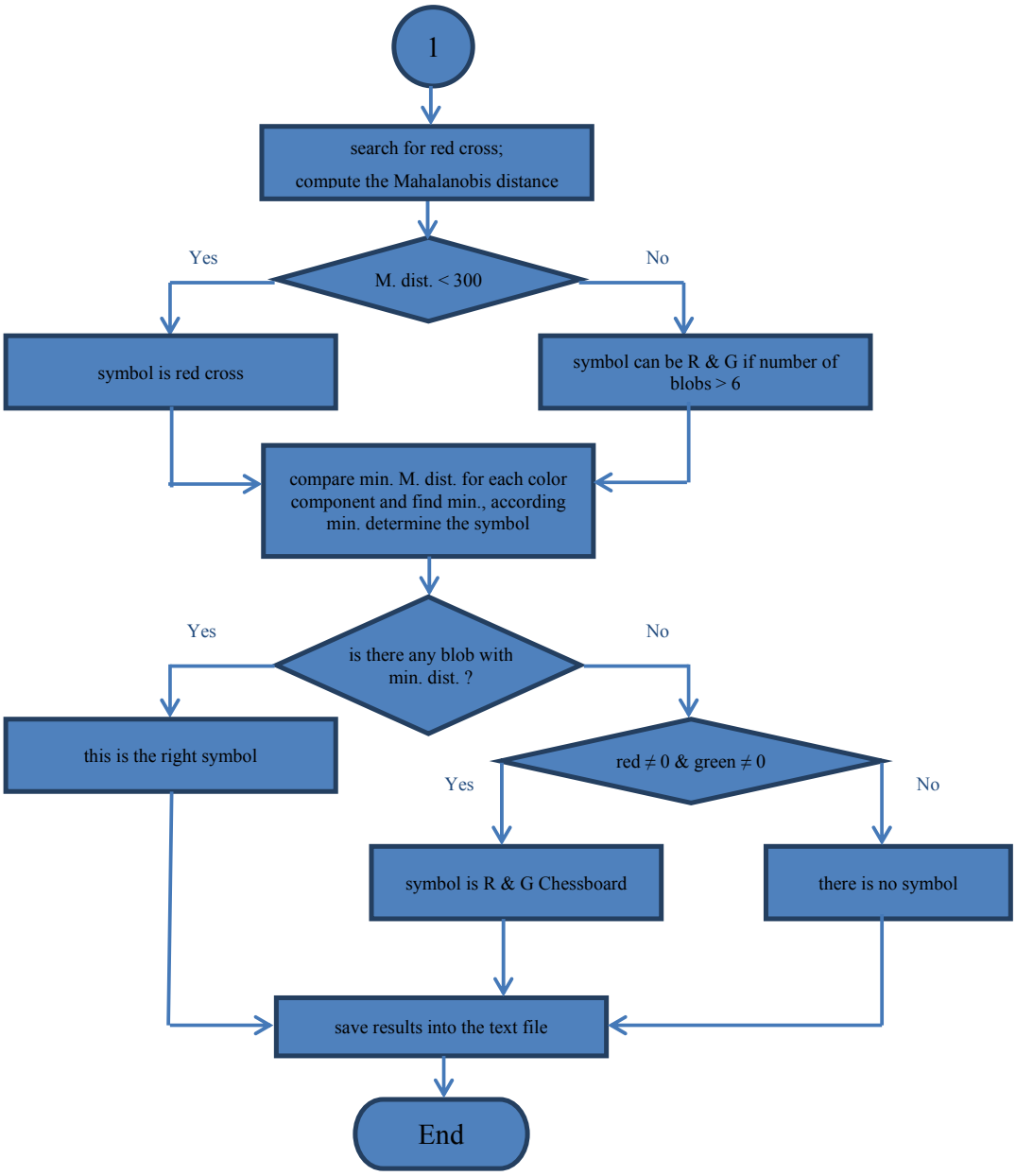


Table 5.1 – Used color ranges values (obtained by testing approximately 2000 images)

	H	S	V
red	0 - 0.07 & 0.96 - 1	> 0.5	> 0.5
green	0.2 - 0.54	> 0.4	> 0.4
yellow	0.11 - 0.2	> 0.5	> 0.4

5.2 Results

The accuracy of symbol detection is considerably better with Pattern Recognition. This test was made using the same set of images as in final test in previous chapter. So, the results can be compared. No false positiveoccurred in this case. Also the number of missed images was reduced. From whole set of 450 images only 23 images were missed. In most cases it was caused by large distance of the camera from the Signaling Panel. In other cases it was caused by high angle or by blurring of an image. Results can be seen in graph (Figure5.1). Time taken by processing one image is approximately 1 s.

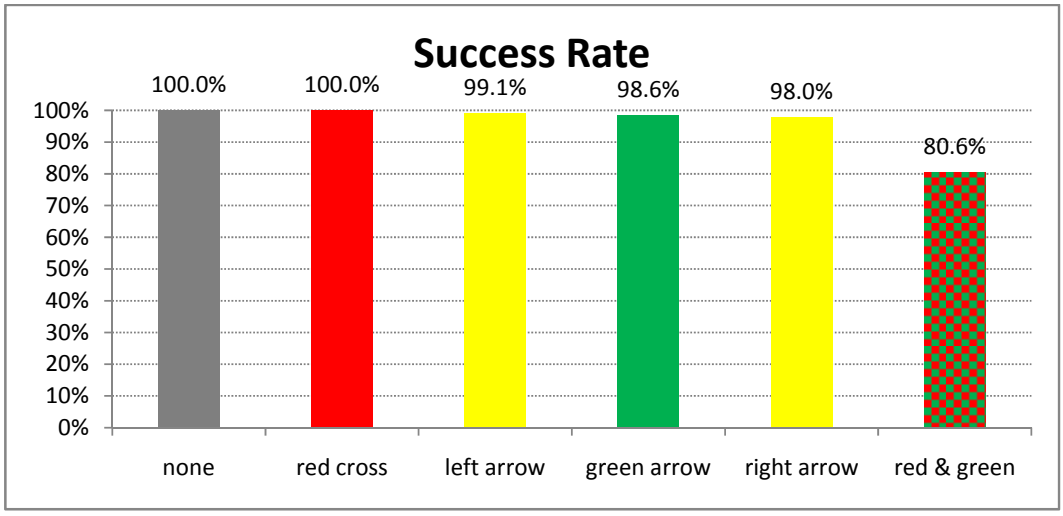


Figure5.1 – Results of the Pattern Recognition test (0603) using 450 images

Worse result in case of Red & Green Chessboard symbol was caused by a long distance. In case of Arrows the symbols were missed because of high angle or blurring the image. Examples of these images can be seen inFigure 5.4.

Table 5.2– Results of the Pattern Recognition (0603)

Tested Images							**Results**	
red cross	left arrow	green arrow	right arrow	red & green	without light	total	missed	wrong
92	113	73	49	103	20	450	23	0

The yellow arrow is recognized well even in case when is rotated till± 25° (Figure 5.2).Still, the direction of such an arrow can be determined using centroid.Arrows rotated more than ± 25° are removed by blob selection.

Saved text file includes following information (see Table 5.3):

- Name of corresponding image file
- Red, Green and Yellow are the quantities of each color component after color segmentation
- bRed, bGreen and bYellow are quantities of corresponding color components after blob selection
- dMred, dMgreen and dMyellow are the minimum Mahalanobis distances for selected blob in each color component
- Symbol represent the type of found symbol (1 = red cross, 2 = left arrow, 3 = green arrow, 4 = right arrow,5 = end)
- MahalDist is the Mahalanobis distance of found symbol

Table 5.3 – Example of a text file with rwsults

Name	Red	Green	Yellow	bRed	bGreen	bYellow	dMred	dMgreen	dMyellow	Symbol	MahalDist
img0.jpg	14	1045	542	0	334	541	Inf	1 012,8	5,3	2	5,3
img1.jpg	13	1025	537	0	188	535	Inf	2 273,5	4,2	2	4,2
img10.jpg	8	510	266	0	0	259	Inf	Inf	5,5	2	5,5
img100.jpg	4	620	402	0	0	309	Inf	Inf	11,9	2	11,9
img101.jpg	8	346	539	0	63	280	Inf	5 657,1	46,1	2	46,1
img102.jpg	11	290	680	0	57	259	Inf	5 062,2	9,4	2	9,4
img103.jpg	16	248	828	0	0	273	Inf	Inf	14,1	2	14,1
img104.jpg	6	110	935	0	0	0	Inf	Inf	Inf	0	Inf
img105.jpg	1	153	936	0	0	0	Inf	Inf	Inf	0	Inf
img106.jpg	7	93	996	0	0	0	Inf	Inf	Inf	0	Inf

Figure 5.2–Wellrecognizedyellow arrow (numbers represent pixels of the area)

Figure 5.3– Example of well recognized images

Figure 5.4 – Examples of missed images. First two were missed because of higher number ofMahalanobis distance then the threshold value. Last two were missed because of the small blobs.

Second test using 661 images

In Figure 5.5 the results of another test can be seen. In total 661 images ware processed. There was only 1 image without any symbol, so the result for this case is not reasonable. Also during this test no false positive occurred. Also here the worse result in case of Red & Green Chessboard symbol was caused by a long distance.

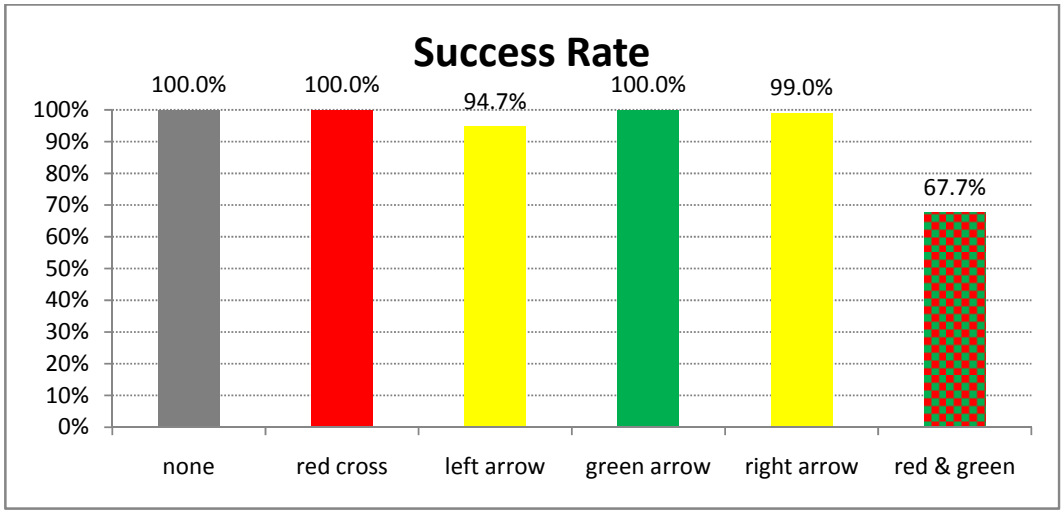


Figure 5.5 – Results of the Pattern Recognition test(0527) using 661 images

Table 5.4– Resultsof the second test (0527)

Tested Images							Results	
red cross	left arrow	green arrow	right arrow	red & green	without light	total	missed	wrong
232	76	90	98	164	1	661	58	0

Figure 5.6 – Example of well recognized images

Tests of images from 2010

Finally, the last test using images from the 2010 was done.These images were taken in different lights conditions during the competition. Parameters for color segmentation were the same as in previous test (see Table 5.5).

Most of the images in this test were without any symbol. Hence, the results for other symbols are not so relevant. Too many images were taken from long distance. That fact influenced the results very much. However, this test is good demonstration of recognition images without any symbol. Only 1 false positive occurred in this test (Figure 5.10). One object in red color component was found and classified as a Red Cross.Mahalanobis distance of this object was 81,5.

Table 5.5 – Used color ranges values

	H	S	V
red	0 - 0.07 & 0.96 - 1	> 0.5	> 0.5
green	0.2 - 0.54	> 0.4	> 0.4
yellow	0.11 - 0.2	> 0.5	> 0.4

Table 5.6 – Results of the test with images from 2010

Tested Images							**Results**	
red cross	left arrow	green arrow	right arrow	red & green	without light	total	missed	wrong
18	104	38	26	243	2885	3314	108	1

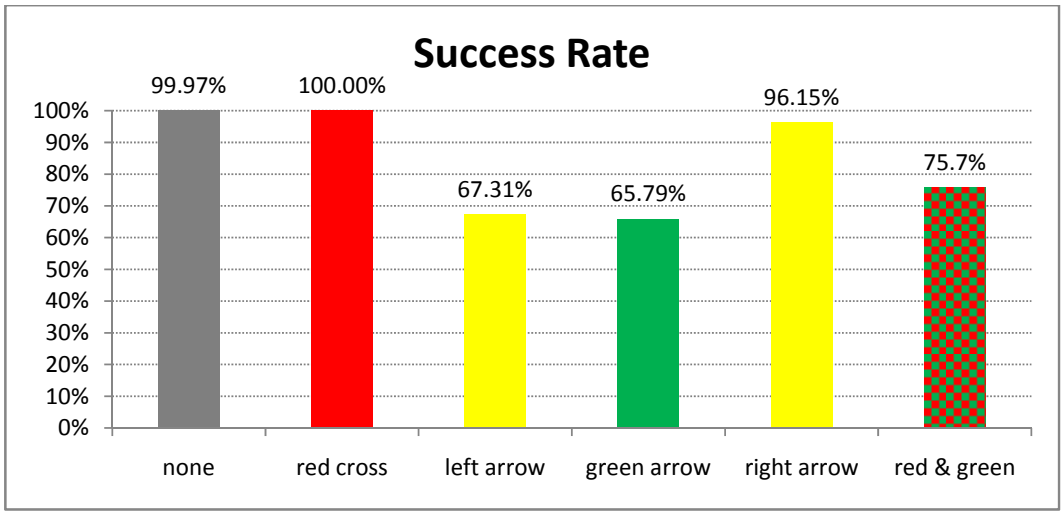


Figure5.7 – Resultsof the test with images from 2010 (in total 3314 images)

Figure 5.8 – Example of well recognized images

Figure 5.9 – Examples of images taken from very long distance

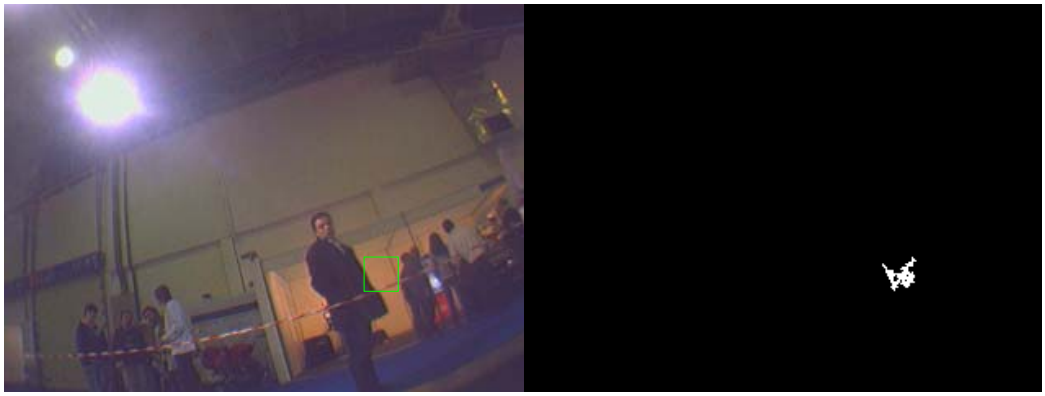


Figure 5.10 – False positive in red color component (the red component can be seen in the right image)

To solve the problem with false positive the color range for red component was changed (see Table 5.7). The problem with false positive was solved.There were no false positives during the test with new color range.

Table 5.7 – New color ranges values

	H	S	V
red	0 - 0.05& 0.96 - 1	> 0.5	> 0.5
green	0.2 - 0.54	> 0.4	> 0.4
yellow	0.11 - 0.2	> 0.5	> 0.4

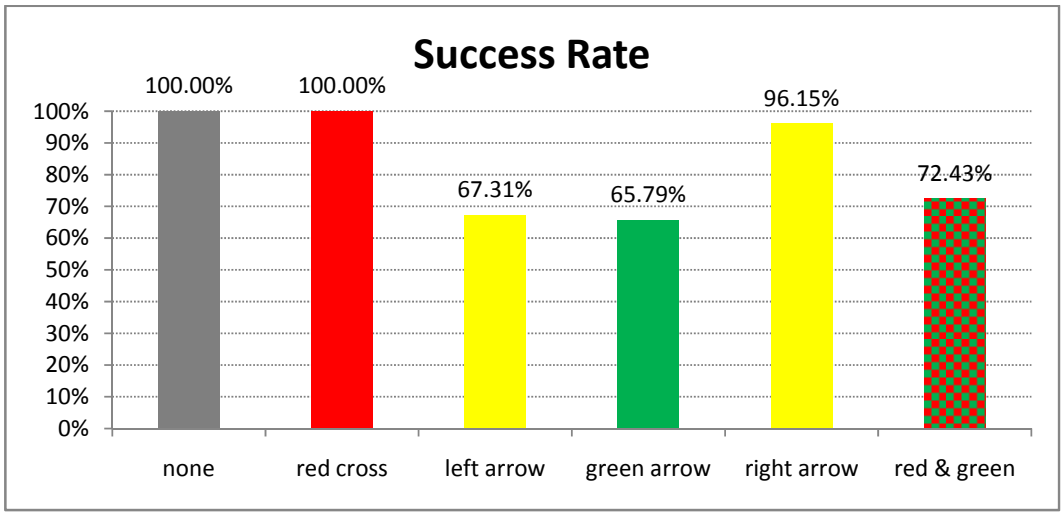


Figure5.11 – Results of the test with new color range for red component

Table 5.8 – Results of the test with images from 2010

Tested Images							Results	
red cross	left arrow	green arrow	right arrow	red & green	without light	total	missed	wrong
18	104	38	26	243	2885	3314	115	0

To better compare results of symbols recognition, the last test only with images taken approximately till distance of 2 meters from the signaling panel was done. When only images taken from shorter distance are included, the results are significantly better. Only 1 yellow arrow and 14 Red & Green symbols were missed. It was caused by bad taken images. Examples of these missed images can be seen in Figure 5.12. The success rate in other cases is 100% (see Figure 5.13).

Figure 5.12 – Example of missed images from short distance

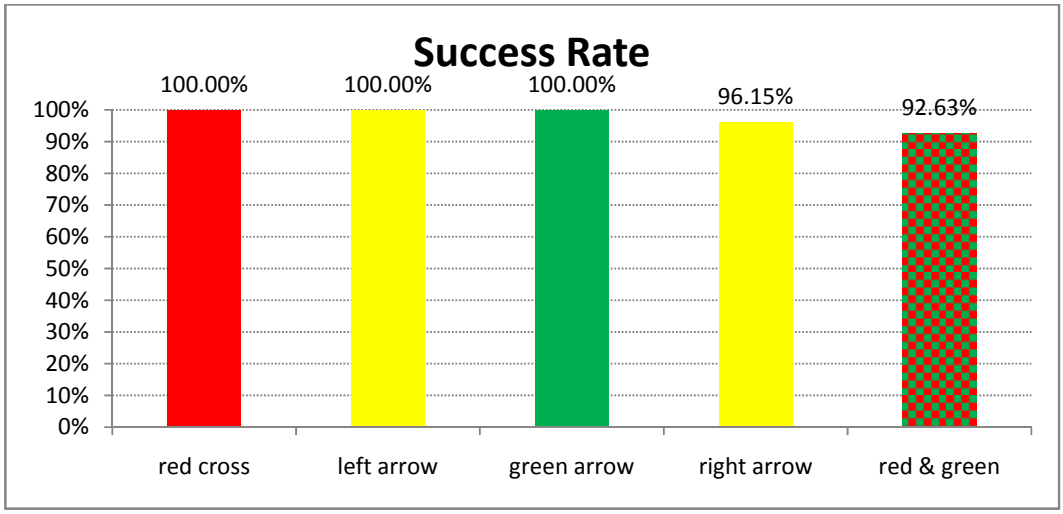


Figure 5.13 – Results of the test with images taken till 2m distance

Table 5.9 – Results of the test with images taken till 2m distance

Tested Images						Results	
red cross	left arrow	green arrow	right arrow	red & green	total	missed	wrong
18	70	17	26	190	321	15	0

5.3 Proposals to improvement

At first it should be better to build a bigger reference set of good examples. It will improve computation of Mahalanobis distance and thereby the recognition of the symbols.Then it will be useful to compute the probability of detected symbols. Next step should be also to try to use a neural network for classification.

6 Future work–Traffic Signs

Future part of this work will be focused on the task of traffic signsrecognition. Traffic signs recognition is currently very important part of tasks in the area of smart advanced driving assistance systems. These systems support a driver and provide a lot of information for safe driving.At first, background research of already published traffic sign recognition approaches is presented.

6.1 Background research

In the majority of recently published sign detection approaches following technique is used: At first, color segmentation and shape recognition are done for selection of traffic sign candidates. Subsequently, a final recognition is done using some classification algorithm, generally based on machine learning.

X.W. Gao, L. Podladchikova, D. Shaposhnikov, K. Hong, N. Shevtsova[10] have used color model CIECAM97 to extract color information and to segment traffic signs and shape features were extracted using FOSTS model.Bahlmann, C., Zhu, Y., Visvanathan Ramesh, Pellkofer, M., Koehler, T.[11] have used a set of color sensitive Haar wavelet features for detecting signs. The set was obtained from Ada-Boost training. After that classification is performed using Bayesian generative modeling.AryuantoSoetedjo and Koichi Yamada[12] have used tracking to improve traffic sign detection. Their approach is based on color segmentation in two color layers simultaneously with blob detection. Finally, they used blob matching using a ring-partitioned method. Fabien Moutarde, AlexandreBargeton, Anne Herbin, and LowikChanussot[13] have used only shape detection (rectangles or circles) for detection of traffic signs and segmenting digits inside the sign candidates to classify them. Final step is applying neural network digit recognition. Marcin L. Eichner, Toby P. Breckon[14] proposed sign detection via color and shape, then classification of sign using trained neural network. Jun Miura, Tsuyoshi Kanda, and Yoshiaki Shirai[15] have proposed a system composed of two cameras, one with a wide angle lens and second with telephoto lens. They have used a PC with image processing board. The system first detects candidates for traffic signs in the wide-angle image using color, intensity, and shape information. For each candidate, the telephoto camera is directed to its predicted position to capture the candidate in a larger size in the image. Identification of the sign is done by pattern matching. DušanZámečník, Ivo Veselý[16]have used color segmentation in HSV color space

for red signs followed by using radiometric, brightness descriptors and descriptors derived using Hough transformation of sign contour. Finally classification is done by a few simple neuron networks.T. Hwang, I. Joo, and S.I. Cho[17] suggested a method for traffic light detection that processes color thresholding, finds center of traffic light by Gaussian mask and verifes the candidate of traffic light using suggested existence-weight map.

8 Conclusion

The created programs solve recognition symbols of traffic lights which are used for robotic competition. Theywere created especially for the competition Robotica 2011 Festival National de Robótica.Two different approaches were chosen. First is based on blob detection whilst the second one is based on Pattern Recognition or a combination of both.

Even using blob detection the results of created algorithm are quite good. However, it depends a lot on surrounding light conditions.Sometimes, depending on the light conditions, the Black Level settings of the camera have to be changed. Usually when the pictures are darker, the results of the recognition are better. A result also differs by symbol in the picture. The best results are obtained with symbol Red Cross (almost always 100% success rate). The most complicated symbol is Red & Green Chessboard. This is caused by the small size of blobs.Maximum distance for recognition symbols of Green Arrow and Red Cross is approximately 3 meters. However maximum distance for reliable recognition of the symbol Red & Green is only 2 meters. Maximum distance for symbol Yellow Arrow is approximately 2.5 meters.When the robot is on the road then the angle should not have a big influence on recognition.Object recognition based on blob analysis works good for objects (blobs) of completely different proportions. However this technique is not very reliable for even slightly similar objects.

Better results were obtained using the Pattern Recognition.Light conditions do not have so big influence to results as in case of using only blob detection. But still the results depend on surrounding light conditions.The results are better with darker images. The accuracy of this method is significantly better. There were no false positive during this test. But the method is sensitive to distortion of the symbols. Due this fact some images were missed. However, the distances for good recognition of symbols are similar as in previous case.

Finally, the test with images taken during the competition in 2010 was done. There was only one false positive in whole set of 3314 images. No false positive occurred after modification of the color range for recognition of the red color component. Success rate of recognition images taken till 2 meters distance from signaling panel is almost 100%.

In the end of this work some proposals to improvement are stated. Last part is a proposal to future work and brief background research of already published traffic sign recognition approaches. Traffic sign recognition should be the next step of this work.

9 References

[1] Gonzalez, Woods, Eddins, *Digital Image Processing Using MATLAB,* Prentice Hall, 2003, 728 p., ISBN: 978-0130085191

[2] Robotica 2011, Festival Nacional de Robotica, *Competition, Rules and Technical Specifications,* 2011, 27 p., Accesible on www: <http://robotica2011.ist.utl.pt/docs/2011_Conducao_Autonoma-regras-en.pdf>

[3] C. Solomon, T. Breckon*, Fundamentals of Digital Image Processing, A Practical Approach with Examples in MATLAB*, Willey-Blackwell, 2011, 344 p., ISBN: 978 0 470 84472 4

[4] MathWorks, Inc.,*Image Processing Toolbox™ 7: User's Guide*, September 2010

[5] The MathWorks, Inc.,*Product Help*, MATLAB R2010b.

[6] Web pages of the MathWorks company, [online], 2011, Accessible on www: <http://www.mathworks.com>

[7] S. Theodoridis, K. Koutroumbas, *An Introduction to Pattern Recognition: A MATLAB Approach*, Elsevier Inc., 2010, 219p.

[8] S. Theodoridis, K. Koutroumbas, Pattern Recognition, Second Edition, Elsevier, 2003, 689 p., ISBN: 0-12-685875-6

[9] M. N. Murty, V. S. Devi, *Pattern Recognition, An Algorithmic Approach*, Springer, 263p., ISBN: 978-0-85729-494-4

[10] 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*,School of Computing Science, Middlesex University, Bounds Green, London N11 2NQ, UK,*Journal of Visual Communication and Image Representation*, vol. 17, Aug. 2006, pp. 675-685.

[11] 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*, Siemens Corporate Research, Inc.,*Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, 2005, pp. 255-260.

[12] AryuantoSoetedjo and Koichi Yamada, *Improving the performance of traffic sign detection using blob tracking*,*IEICE Electronic Express*, Vol. 4, No. 21, 2007, pp. 684-689

[13] F.Moutarde, A.Bargeton, A.Herbin, and L.Chanussot, *Robust on-vehicle real-time visual detection of American and European speed limit signs, with a modular Traffic Signs Recognition system*, *Intelligent Vehicles Symposium, 2007 IEEE*, 2007, pp. 1122-1126.

[14] M. L. Eichner, T. P. Breckon, *Integrated Speed Limit Detection and Recognition from Real-Time Video, Intelligent Vehicles Symposium, 2008 IEEE*, 2008, pp. 626

[15] J.Miura,T.Kanda, Y.Shirai, *An Active Vision System for Real-Time Traffic Sign Recognition*,Department of Computer-Controlled Mechanical Systems, Osaka University,*Intelligent Transportation Systems, 2000. Proceedings. 2000 IEEE*, 2000, pp. 52-57

[16] D.Zámečník, I.Veselý, *Rozpoznávaniečervenýchdopravnýchznačiek s použitímHoughovejtransformácie a neurónovýchsietí*,Fakultaelektrotechniky a komunikačníchtechnologií VUT v Brně, Brno,*elektrorevue*, 2010,ISSN 1213 – 1539

[17] T. Hwang, I. Joo, and S.I.Cho,*DetectionofTrafficLightsfor Vision-Based Car NavigationSystem*, In Proceedingsof PSIVT, 2006, pp.682-691.