

## **VHDL DESIGN FOR A SPEED LIMIT SIGN SEGMENTATION ALGORITHM**

Hanene Rouabeh, Chokri Abdelmoula and Mohamed Masmoudi

*Micro Electro Thermal Systems, National Engineers School of Sfax, University of Sfax, Tunisia*

### **ABSTRACT**

One of the most important and challenging tasks in the image processing field is color segmentation. In this paper, we propose a red color segmentation algorithm which is developed to be used as a mandatory task of a complete vision based intelligent speed limit road sign recognition system. This type of sign is characterized by a red border. For this reason the segmentation remains important in the detection task. A simple and accurate technique was proposed which presents a modified and improved thresholding based method that uses the Red, Green and Blue color space components. In order to improve the robustness and efficiency, a pretreatment was proposed in our work to be applied on images before thresholding. This pretreatment is divided into two steps; the first one is classification. The aim is to classify images according to different illumination conditions. The second one consists in color adjustment and correction. These two algorithms were introduced to solve problems caused firstly by the use of same thresholding values for different images, and secondly by the sensibility of colors to lightening conditions variation. The comparison with existent methods has shown robustness and accuracy mainly for images of different day times and for faded and blurred signs. In addition to that low computation time is achieved. The software implementation of the developed segmentation algorithm using MATLAB tool is discussed in this paper, as well as the simulation results of the VHDL Circuit using ModelSim tool. Computational time reduction, low complexity and low resource utilization have shown the effectiveness of our approach.

### **KEYWORDS**

Image processing, Color segmentation, Color spaces, VHDL implementation

## **1. INTRODUCTION**

In recent decades, the development of new technologies in computer science and data processing have improved many applications related to different fields such as autonomous navigation, intelligent transportation and driver assistance systems. The combined use of automatic perception modules and intelligent data analysis and processing algorithms, has

attracted a big interest among researchers. In this context, many vision based road sign recognition systems were proposed, but the efficiency was limited according to some problems caused essentially by the variation of scenes conditions in which images were captured and the influence of lightening and weather conditions on color intensities. Most existent issues for this task are divided into two stage; sign detection and recognition such as: (Thongchai et al., 2013), (Xu Qingsong et al., 2010) and (Usman Zakir et al., 2012). Existent approaches for the detection task can be classified into color based detection approaches and shape based approaches. In our work we are interested in color based techniques. Many issues were developed for red color segmentation using the different color spaces. Authors in (Benallal et al., 2003) have proposed an approach to segment road signs from the simple comparison of the RGB components taken two by two. They have inspired this method from the influence of illumination variation on RGB components of several road signs. HSV color space was used in (Malik et al., 2007) to segment red regions, whereas the HSI color space was used in (Khodadadzadeh et al., 2010). HSI color space was also used in (Sallah et al., 2011) to segment red, yellow, blue and white road signs. The performance of these techniques varies according to the non-uniform lightening and weather conditions.

In many issues developed for pattern and object recognition color constancy algorithms were used to improve and enhance the detection task. Taking into account color casts caused by illumination variation, the role of color constancy algorithms consists in color correcting. Issues proposed in (Chambah, 2006), (Xiaozhao, 2009) and (Sinan et al., 2014) have discussed different algorithms developed for color correction in machine vision. These algorithms were used to maintain color constancy and to retrieve standard images. In our work a color adjustment algorithm is used as a pretreatment to improve the segmentation task.

In this paper, we are interested to develop an accurate road sign segmentation approach. The main challenges consist in increasing the performance and achieving a high accuracy and high detection rate. A thresholding task is carried on the RGB color space to segment red pixels and to extract ROI suspected to be a speed limit road sign. In order to enhance results, a classification task is used as a pretreatment to classify images into five classes according to illumination variation and weather conditions. To achieve a good classification, various existent classification approaches were evaluated such as those in (Xiaojuan et al., 2009), (Manthira et al., 2011), (Madokoro et al., 2012) and (Agarwal et al., 2011). This classification helps to automatically choose the thresholding values that best fit the processed image. A color constancy algorithm is used as a secondary stage to improve the segmentation method.

The remainder of this paper is organized as follows. Section 2 details the proposed method. Section 3 discusses the VHDL implementation of the proposed segmentation approach. A summary of the key points and future works concludes the paper in section 4.

## 2. PROPOSED METHOD AND CONTRIBUTION

This section describes the red color segmentation algorithm. The segmentation serves as an obvious pre-processing task of an automatic speed limit road sign recognition system. This system uses a vision based perception module. It is developed to be implanted on an FPGA card to be used as a driver assistance application. The object of our work is to recognize speed limit road signs in road scene images. The main characteristic of this type of signs is the red border. So the segmentation task will be focused on the red color segmentation that helps to

extract red objects from other ones in the image. This contributes to select region of interest suspected to be a road sign. The RGB color space is used for this purpose due to many advantages that can be obtained such as simplicity and robustness. A method proposed in (Benallal et al., 2003) for red color segmentation was developed based on the simple R, G and B components thresholding. Some disadvantages can be faced such as the sensibility of colors to illumination variation and lightening conditions. In our work we have proposed a modified version of this technique that increases the performance and efficacy of the segmentation. The advantages of the proposed solution consists firstly in the use of the best thresholds that best fits the processed image according to illumination condition instead of using same thresholds as in (Benallal et al., 2003) Secondly it consists in the use of a color adjustment algorithm that enhances the segmentation result. As demonstrated in (Hanene et al., 2016), thresholds used to efficiently segmenting a daytime image have not lead to same result for a nighttime image. In fact a set of different thresholding values have been used to obtain good results. Various tests have been carried on different images captured in different illumination conditions. They all have lead to the conviction that same thresholds couldn't be used for different images.

The proposed solution consists in classifying road scene images into five classes according to illumination conditions and daytime. Then, a set of four thresholding values are fixed for each image class. A big number of images have been used to fix these thresholds for each class. So the first step in the segmentation task is the image classification task which is done using one of the existent learning based classification algorithms. A comparison and evaluation of two classification algorithms have been discussed in (Hanene et al., 2014) to choose the suitable classifier for this task.

To overcome imperfection in some cases a color correction algorithm was used. This algorithm was used to correct and adjust colors intensities in order to improve the thresholding task. So the overall segmentation approach is divided into three basic steps as depicted in Figure 1.



Figure 1. Segmentation Process

Image Classification is achieved using some features extracted from the Hue, Saturation and Lightness Histograms. The choice of a Decision Tree classifier was justified in (Hanene et al., 2014). This classifier has shown effectiveness and accuracy. Indeed low computational time and low complexity have been approved. Adjustment step consists in color correction and normalization in order to enhance the final thresholding task which is carried on the new R, G and B components obtained after the adjustment process. Thresholds are chosen automatically after the fixation of the image class. These three steps will be discussed in next sub sections.

## 2.1 Classification

This section discusses the developed image classification approach. The aim of the classification is to improve the accuracy and efficacy of the real-time road sign segmentation system. Working with RGB color space for image segmentation may present a real-time and

low computational requirement, but a lack of precision can be caused by the sensibility of this color space to the variation of light conditions. Indeed, it is not preferable to use same thresholds to segment images captured under different lightening conditions. For this reason, we have tried to develop a method that helps to decide the suitable thresholding values according to the image characteristics.

A Recent issue in daytime nighttime image classification has involved the HSV color space to classify images in (Ajay et al., 2012), in which Ajay Raghavan et al have proposed a method based on the Hue and Value components thresholding to classify images into two classes; daytime and nighttime images. They have demonstrated that pixels of daytime images have mostly contribution in blue-green range as well as pixels of nighttime images have mostly contribution in red-yellow range.

In our work we are interested in classifying images into five classes: class 1 represents image of a bright day, class 2 refers to image of a bad weather day, class 3 describes image of Sunrise/Sunset time, class 4 illustrates image of a bright night and class 5 denotes image of a dark night. In order to improve the red color segmentation algorithm, the classification task is arranged towards classifying images into these five classes. In fact the segmentation technique that will be discussed later is based on the simple R, G, B color space thresholding. It was demonstrated in many studies that the same set of thresholding values cannot be used to segment images of different lighting conditions. Due to the influence of illumination's variation on the red color intensity's, a set of thresholds is fixed for every image class. To achieve our proposed classification method five features have been extracted from the H, S and L components Histograms. In fact, the analysis of these histograms have shown that the Hue Histogram can be used to classify images into two big classes; daytime and nighttime images. For images of class 1 and 2, the largest number of pixels have a Hue value in the range [0.2; 0.8]. To differentiate these two classes the saturation Histogram is used. The analysis of this Histogram shows that the majority of pixels of the class 2 images have low saturation's values (less than 0.2). Indeed low saturation means low intensity and low vivacity. Image of class 3 can be distinguished using the mean value of lightness histogram. For this image class, the majority of pixels have lightness value less than 0.5 so the mean value should be less than 0.5 contrary to images of class 1 and 2. To differentiate class 4 and 5 from other classes, the majority of pixels of these images have Hue values outside of the range [0.2; 0.8]. But to differentiate class 4 from class 5, the mean of lightness for the class 5 image is less than 0.1 and in the same time the majority of pixels have low saturation values. The analysis of Histograms of a big number of images has shown that these hypotheses are verified for the majority of images and some special cases can be faced. A thresholding process is carried on the H, S and L Histograms to extract three thresholds to be used with the mean values of the Hue and the Lightness components as features for classification. These features are presented as follows:

- N1: The number of pixels having Lightness value less than or equal to 0.2.
- N2: The number of pixels having Saturation value greater than or equal to 0.2.
- N3: The number of pixels having Hue value in the range [0.2 , 0.8]
- H\_Mean: Presents the mean value of the Hue component calculated using (1):

$$H\_Mean = \frac{\sum_{i=1}^N H(i)}{N} \quad (1)$$

- L\_Mean: Presents the mean value of Lightness component calculated using (2):

$$L\_Mean = \frac{\sum_{i=1}^N L(i)}{N} \quad (2)$$

With  $N$  is total number of pixels in an image,  $H$  and  $L$  are respectively Hue and Lightness values.

A Decision Tree based classifier was designed in (Hanene et al., 2014) to realize this classification task. The input variables are the five extracted features and the output is the image class. Figure 2 shows the classification process.



Figure 2. Classification model

The RGB to HSL color space conversion is done using equations mentioned in (Hanene et al., 2016). The different features are then determined after the calculation of the  $H$ ,  $S$  and  $L$  components' Histograms. Subsequently, the five calculated values are used as inputs for the designed Decision Tree classifier. The image class is then determined following the decisions from the beginning node down to the leaf node containing the suitable output class.

## 2.2 Color Adjustment

The outputted image class obtained by the classification task is used to automatically choose the suitable thresholding values that are fixed for each class. The fixation of these values was done using various testing images for each class. In order to improve segmentation results, the adjustment task is used. The goal is to render specific colors correctly. In fact in some cases color intensities must be transformed to new values that are more significant and more corrected. The adjustment is done directly on the  $R$ ,  $G$  and  $B$  pixel's values.

Figure 3 summarizes the results obtained by the evaluation of the segmentation algorithm on different image classes before and after the application of the color adjustment task. Results show that this task enhances the segmentation and more performant results are obtained after adjustment step.

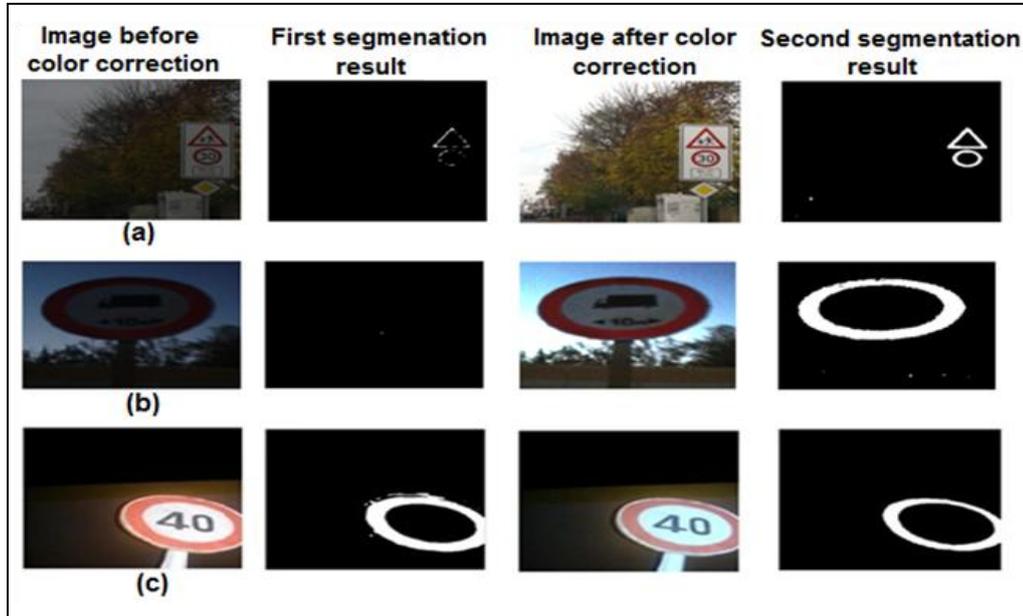


Figure 3. Comparison of segmentation results before and after adjustment. For (a) image of class 2, (b) image of class 3, (c) image of class 4

Figure 3 shows that lightening conditions can affect the color constancy. To achieve successful red color segmentation task it is necessary to adjust and correct the coloration of images that was captured under different lightening conditions. To overcome the problem and to be able to use same thresholds for all images of same class an automatic color adjustment algorithm was used. This algorithm is simple and accurate and contributes effectively in the enhancement of the image quality.

Many adjustment algorithms were developed to enhance and correct colors intensities of images taken under different illumination conditions. The performance is related to the desired task to be accomplished. Different algorithms were developed in the literature and several ones have been tested in our work to choose the one that best fits the developed segmentation task. Three algorithms were tested that are Grey World, White Patch and Modified White Patch algorithms (Edmund et al., 2009). These approaches are based on the same concept consisting in the calculation of a scaling factor which is used to generate the new corrected image. In the presented work the aim is to generate a new adjusted image that will be used in the segmentation stage in place of the original image. The purpose is to reach a high level segmentation with same thresholding values for images of same class. Tests carried on various images of different classes have shown that the Modified White Patch algorithm shows the more accurate results when varying the threshold value 'th' for each image class. The Modified White Patch algorithm is used to generate a new corrected image after color correction of an original RGB image, using equation 3.

$$(3) \quad \begin{cases} R_{\text{new}} = \alpha_r \cdot R \\ G_{\text{new}} = \alpha_g \cdot G \\ B_{\text{new}} = \alpha_b \cdot B \end{cases}$$

With  $\alpha_r$ ,  $\alpha_g$  and  $\alpha_b$  presents the scaling factors calculated as follows:

$$(4) \quad \begin{cases} \alpha_r = \frac{255}{\text{mean}(\text{mean}(R > \text{th}))} \\ \alpha_g = \frac{255}{\text{mean}(\text{mean}(G > \text{th}))} \\ \alpha_b = \frac{255}{\text{mean}(\text{mean}(B > \text{th}))} \end{cases}$$

Where ‘**th**’ is a threshold value fixed for each image class differently.

Images of class 1 they shouldn’t be adjusted since they have been captured under good lightening conditions. The different thresholds were fixed after experimental analysis of a big image dataset for each class.

### 2.3 Thresholding and Segmentation

This task is the final step in the whole red color segmentation process. The image classification and color adjustment tasks are used respectively to choose automatically the suitable thresholds according to the image class and to correct the color intensities which enhance the segmentation result. A simple red color segmentation method based on the RGB color space thresholding as the one presented in (Benallal et al., 2003) was used. Our contribution consists in the use of the classification task respectively the color adjustment task as pretreatments for segmentation, in order to choose the suitable thresholds that best fit the processed image and to obtain a new image as it is captured under standard illumination condition. The use of different thresholds helps to overcome the deficiency caused by the sensibility of colors to lightening conditions and illumination variation. If we dispose of an image of size 320x240, the pseudo-code of the segmentation is presented as follows:

For all image pixels P (i, j) where i varies from 1 to 320, respectively j varies from 1 to 240.

- 1- Extract the three components R (i, j), G (i, j), B (i, j)
- 2- Compute difference values:
  - D1= R (i, j)-G (i, j)
  - D2= B (i, j)-G (i, j)
- 3- Verify conditions

$$(5) \quad \text{Out\_image}(i,j) = \begin{cases} 1 & \text{if } \text{Th2} \leq \text{D1} \leq \text{Th1} \ \& \ \text{Th4} \leq \text{D2} \leq \text{Th3} \\ 0 & \text{Otherwise} \end{cases}$$

The Out\_image presents the result of the segmentation task. This image is a binary image where pixels of red objects take the value 1 and others take the value 0. Th1, Th2 ,Th3 and th4

are the thresholding values. For each image class a different set of 4 values are fixed. These values are fixed after many tests carried on images of all classes. The color adjustment algorithm were used to enhance the efficacy of the segmentation with same thresholds for one image class. Indeed, by testing this algorithm without classification and adjustment tasks using same set of thresholds for different images, it was noted that the efficiency of results varies due to illumination changes. The proposed solution has shown effective results by using different thresholds for each image class. These values are fixed after testing the algorithm in a big number of images and comparing the influence of illumination variation on red color.

The flowchart of the whole red color segmentation process is shown in Figure 4.

Table 1 exhibits the time elapsed for the execution of the three steps separately as well as for the whole segmentation process for several images. These results are obtained using the software implementation on the MATLAB Tool.

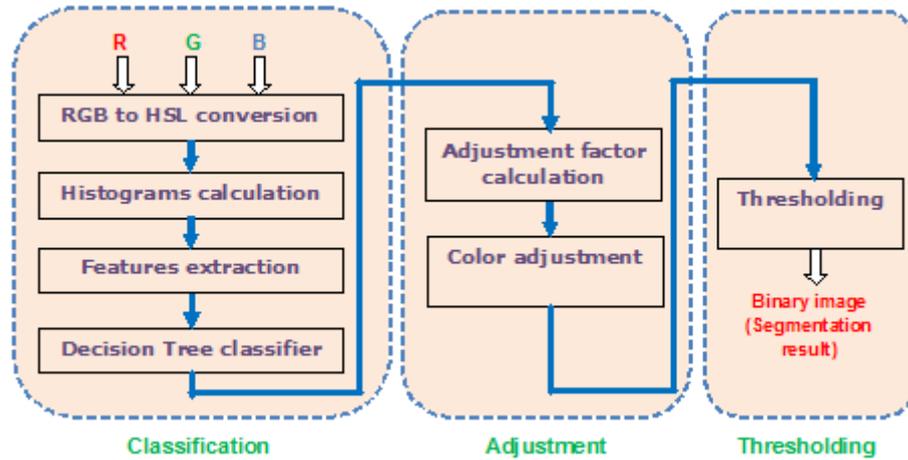


Figure 4. Flowchart of the red color segmentation task

Table 1. Elapsed time comparison of the different segmentation steps

Image Class	Class1	Class2	Class3	Class4	Class5
<b>Time for Classification</b>	0.057662	0.062788	0.049106	0.047446	0.048401
<b>Time for Adjustment</b>	0.004765	0.004911	0.005371	0.004580	0.004729
<b>Time for Thresholding</b>	0.008467	0.008556	0.008410	0.008464	0.008434
<b>Time for Segmentation</b>	0.070641	0.080702	0.066835	0.063729	0.065180

In order to evaluate our proposed method, it was compared to some existing approaches. The one proposed in (Khanal et al., 2012), in which authors have used the log-chromaticity color space. Their experimental results have shown advantages compared to other existing approaches mainly the robustness against illumination variation. But the use of the logarithmic function needs a lot of hardware resources occupation and an important execution time. Authors in (Zumra et al., 2014) have presented a thresholding based segmentation method carried on the HSV color space. They have used more consistent thresholds than similar approaches, but they have involved only images taken in day and evening times. Authors in (Sheng et al., 2012) have also considered that the Hue component is invariant to lightness

variation, for this reason their method is carried on the HSI color space. The approach is based on thresholding the three components H, S and I. Nevertheless, thresholding HSI color space gives more accurate results than RGB color space, the use of same thresholds for images taken in different illumination conditions affect the performance of the method. In (Qin et al., 2010) a color based segmentation approach was proposed based on color distance thresholding. Its problem consists also in how to dynamically fix the suitable threshold. In our work we have tried to give a robust method that show accurate segmentation results for images taken in different weather and geometric conditions. In the same time give a low computation time and simple hardware calculations. Table 3 shows the comparison of our proposed method with those cited as above for some image samples. It is strongly noted in these examples that our method shows robustness and accuracy even for faded and night time images. Table 2 exhibits results comparison of red signs detection carried on a dataset of 300 road images. This dataset contains images captured from road scenes in different conditions and some other images from the dataset in (Hasan Fleyeh, 2015).

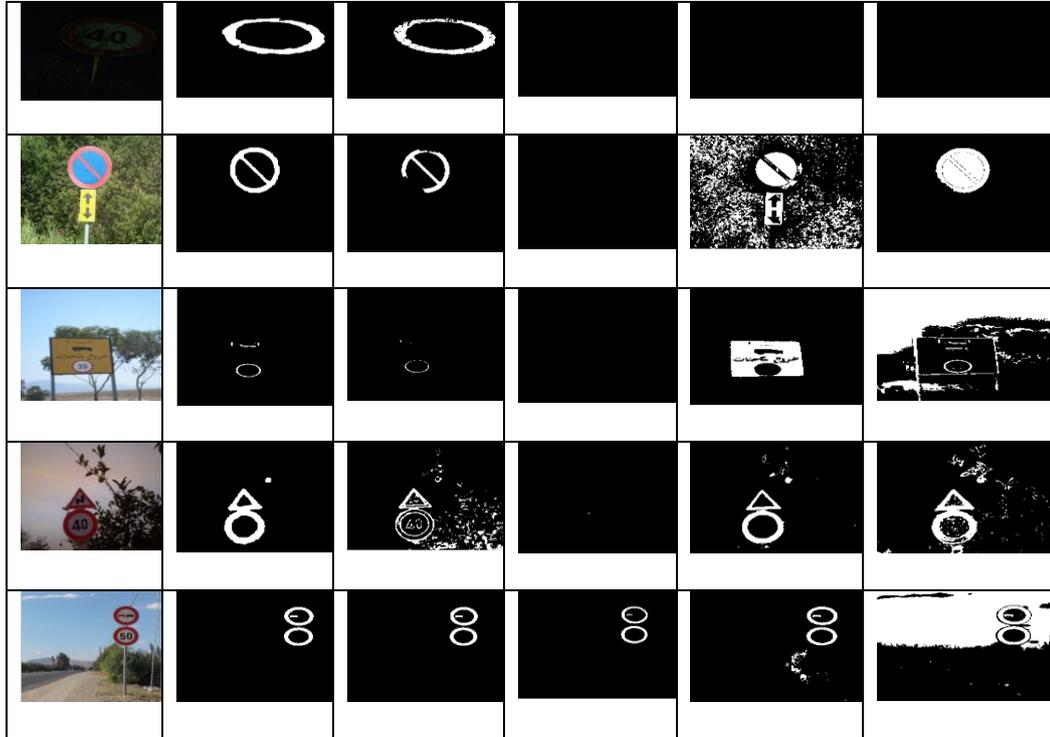
Table 2. Results comparison

Method	This work	Log-chromaticity (Khanal et al., 2012)	HSV thresholding (Zumra et al., 2014)	HSI thresholding (Sheng et al., 2012)	Color distance (Qin et al., 2010)
Missed signs	7	27	116	144	159
Detection rate	97.66%	91%	61.33%	52%	47%

As exhibited in Table 2 and Table 3, it is clearly noted that our proposed method gives more robust and efficient results than other methods. Good segmentation results are obtained for images captured under night or day time, bad weather conditions, varying geometry and also for faded and blurred images. Nevertheless, the log-chromaticity method shows the best results among other methods some weak segmentation results were obtained. In addition to that the hardware implementation of the logarithmic function requires more hardware resources.

Table 3. Proposed method segmentation results Vs other methods results

Image	Proposed method	Method in (Khanal et al., 2012)	Method in (Qin et al., 2010)	Method in (Zumra et al., 2014)	Method in (Sheng et al., 2012)
					
					



### 3. VHDL IMPLEMENTATION OF THE PROPOSED SEGMENTATION METHOD

The developed segmentation task described by the flowchart of Figure 4 was coded using the VHDL Language in purpose to be implanted on an FPGA card. The VHDL design was simulated using ModelSim Altera tool. The R, G and B image channels are stored in a Read Access Memory (RAM). The overall module is composed of five blocks. Each one is charged to realize a specific level in the segmentation algorithm.

#### 3.1 RGB to HSL Convertor

This block reads the three image components R, G and B, and then calculates the corresponding H, S and L components using equations in (Hanene et al, 2016). The RGB444 format was used. On this Format color data per pixel has the red component, respectively green and blue ones corresponding to take values between 0 and 15.

As depicted in Figure 5. This block diagram has three inputs that are R, G and B values and gives as output the H, S and L components. The R, G and B values are stored in the RAM. This RAM is composed of 40000 (Number of total image pixels) std logic vector. Each vector is of size 12 bits representing R component (4bits), G component (4 bits) and B component (4 bits). The 'st' signal is used to enable reading from the memory.

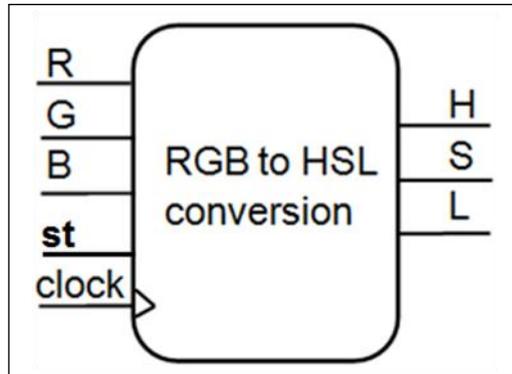


Figure 5. Block diagram of the conversion module



Figure 6. Image of class 2 used for VHDL simulation

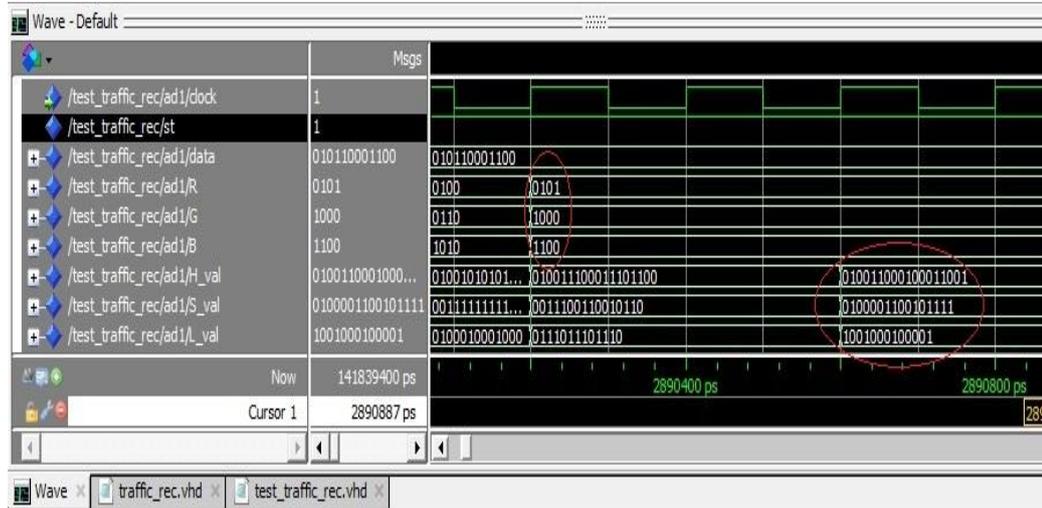


Figure 7. Simulation results of the conversion module

To avoid using fixed point operations the values of the three components R,G and B are multiplied by  $2^{12}/15$  which gives the values in  $[0;1]$  multiplied by  $2^{12}$  of these components. So that the calculations give the value of the Hue multiplied by  $2^{17}$ , the value of the Saturation multiplied by  $2^{15}$  and the value of the lightness multiplied by  $2^{13}$ . For each pixel the H\_val, L\_val are S\_val are generated after two clocks. Figure 7 shows testing results. For the underlined example for which: R= 0101; G=1000 and B=1100 the correspondent H, S and L values are represented as follow:

H\_val = 010011000100011001, converted to decimal gives 78105, and then divided by  $2^{17}$  gives 0.5959.

S\_val =100001100101111, converted decimal gives the value 17199, and then divided by  $2^{15}$  gives 0.5249.

Similarly L\_val = 1001000100001, converted decimal gives the value 4641, and then divided by  $2^{13}$  gives 0.5665.

These values are close to those given using Matlab software that shows effectiveness and accuracy.

### 3.2 Feature Extraction Block

This block is charged to calculate the different features to be used by the classifier. Figure 8 shows its block diagram and Figure 9 shows the VHDL simulation results. The inputs are H\_val, S\_val and L\_val calculated in the previous block and the R, G and B components used to calculate their mean values that will be used in the color adjustment block. The 'en' signal is used to enable the calculation of the mean values if the sum of H values of all image pixels is calculated as well the sum of S and L. At each rising edge clock the N01, N02 and N03 values which represent respectively N1, N2 and N3 cited in sub section 2.1 are updated according to conditions as well as the sum values are also updated according to the new values of each component. When all components are processed, these values and the different thresholds are ready to be used. So the 'en' signal enables the calculation of different sum

values. The outputs are Mean\_H, Mean\_V, som\_R, som\_G, som\_B, nb\_R, nb\_G, nb\_B and the final values of N01, N02 and N03. Where som\_R, som\_G and som\_B are arrays containing each one 4 values presenting the means calculated using the 4 thresholds 'th' and nb\_R, nb\_G and nb\_B presenting number of pixels satisfying thresholding task for each class and each component.

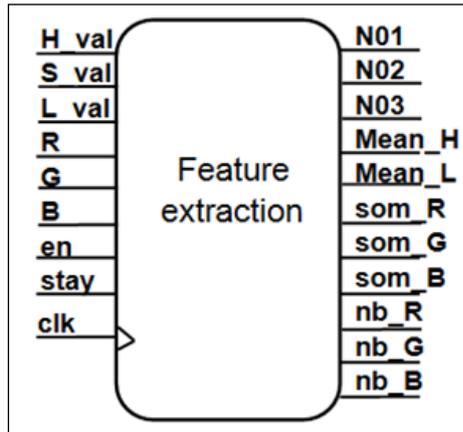


Figure 8. Block diagram of the feature extraction module

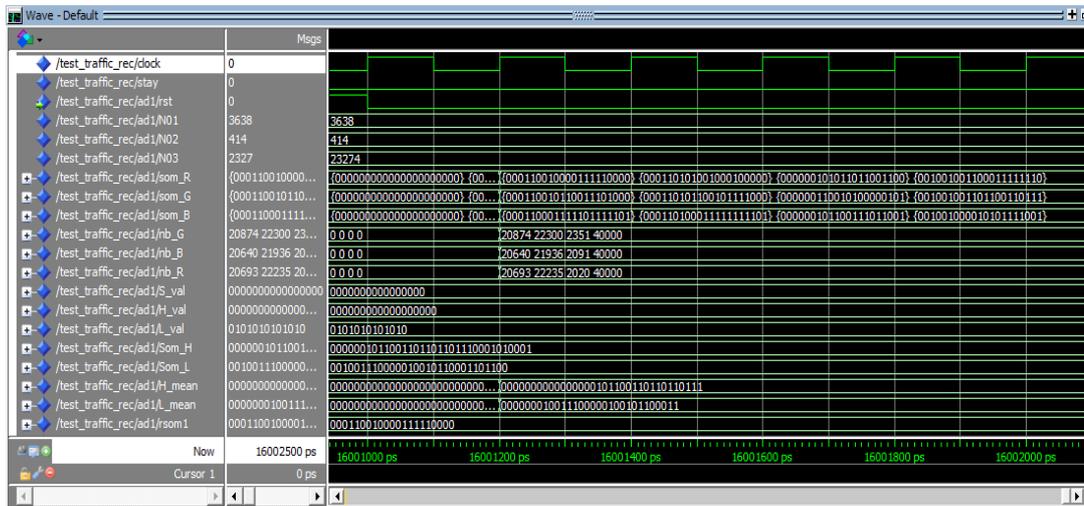


Figure 9. Simulation results of the feature extraction block

Table 4 describes some of the different values generated using the VHDL simulation. These results show efficiency for different variables calculation.

Table 4. VHDL generated values

Variable	VHDL values
Sum of Lightness	$1001110000010010110001101100 / 2^{13} = 19977$
N01	3638
N02	414
N03	23274
Mean_H	$101100110110110111 / (40000 \times 2^4) = 0.2871$
Mean_L	$1111000010111111100111 / (40000 \times 2^8) = 0.4994$

### 3.3 Decision Tree Classifier

In this step the Decision Tree classifier developed in (Hanene et al, 2014) is used to determine the image class according to the five features as explained in sub section 2.1. The VHDL implementation of this classifier was done using a finite state machine composed of 67 states. This number presents the number of total nodes in the Decision Tree. The inputs of the block are Mean\_H, Mean\_L, N01, N02 and N03. The outputs are the image class and the different thresholds for segmentation Th1, Th2, Th3 and Th4. ‘we’ signal enables the generation of outputs after 67 clocks. The starting state is state 1 that represents node 1 in the Decision Tree and at each rising clock the next state that presents the next node would be decided according to conditions. If it is a leaf node that presents the class of the image, the next state would keep the same state as the current one and the class of the image would take its new value. In the case of a branch node the class of the image would keep its value as 0 and the next state would be decided according to conditions. After 67 clocks the class of the image is the final value of the image\_class which is used to generate automatically the three thresholds. The block diagram is presented in Figure 10.

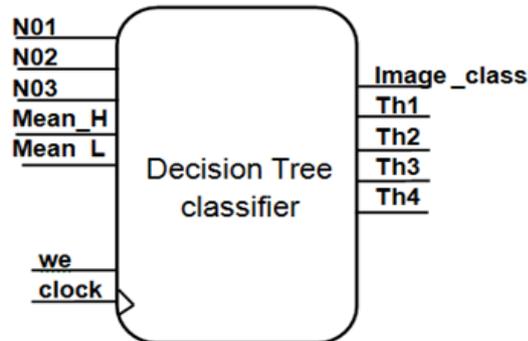


Figure 10. Block diagram of the Classifier

### 3.4 Adjustment Factor Calculation

This block is charged to calculate the three scaling factors mentioned in sub section 2.2; the inputs are the image class, tables containing sums of R, G and B components calculated using each ‘th’ value for all classes, and the corresponding number of pixels for each sum. These values are used to calculate the three outputs  $\mathbf{a}_r$ ,  $\mathbf{a}_g$  and  $\mathbf{a}_b$ .

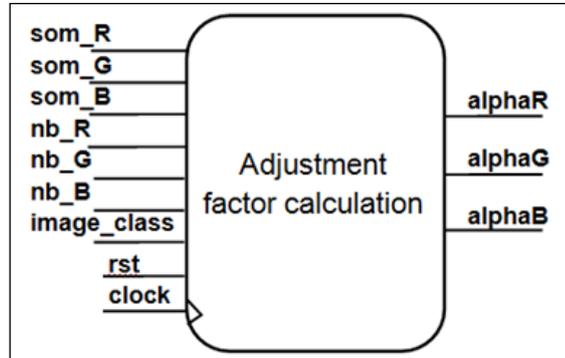


Figure 11. Block diagram of the adjustment factors calculation module

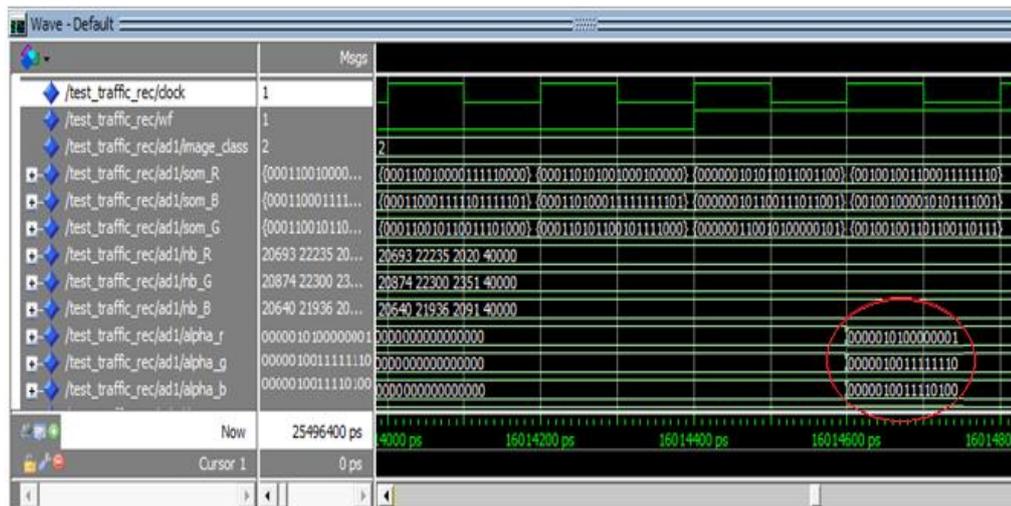


Figure 12. Simulation results of the adjustment factors calculation

Table 5 presents the VHDL values of adjustment factors where  $\alpha_r$ ,  $\alpha_g$  and  $\alpha_b$  in Figure 12 presents respectively  $\alpha_r$ ,  $\alpha_g$  and  $\alpha_b$ . The obtained values are multiplied by  $2^{10}$ . As noted in the table, values of the scaling factors generated using VHDL circuit are very close to those generated using Matlab software. This will increase the efficacy of the color adjustment and thresholding tasks.

Table 5. VHDL Results for adjustment factors

	VHDL value	Converted value
<b>alphaR</b>	1010000001	$1281 / 2^{10} = 1.2510$
<b>alphaG</b>	10011111110	$1278 / 2^{10} = 1.2480$
<b>alphaB</b>	10011110100	$1268 / 2^{10} = 1.2383$

### 3.5 Color Adjustment and Thresholding

This block is charged firstly to calculate the  $R_{new}$ ,  $G_{new}$  and  $B_{new}$  components of the new corrected image and then to realize the thresholding task on these components in order to detect and segment red pixels. The inputs are thresholding values,  $\alpha_r$ ,  $\alpha_g$ ,  $\alpha_b$  and R, G and B values that were stored in the memory. Difference values are calculated and used to verify conditions. If conditions are satisfied, the pixel is decided as red pixel, so the segmented image pixel is set to 1 and to 0 if not. Figure 13 presents the VHDL simulation.

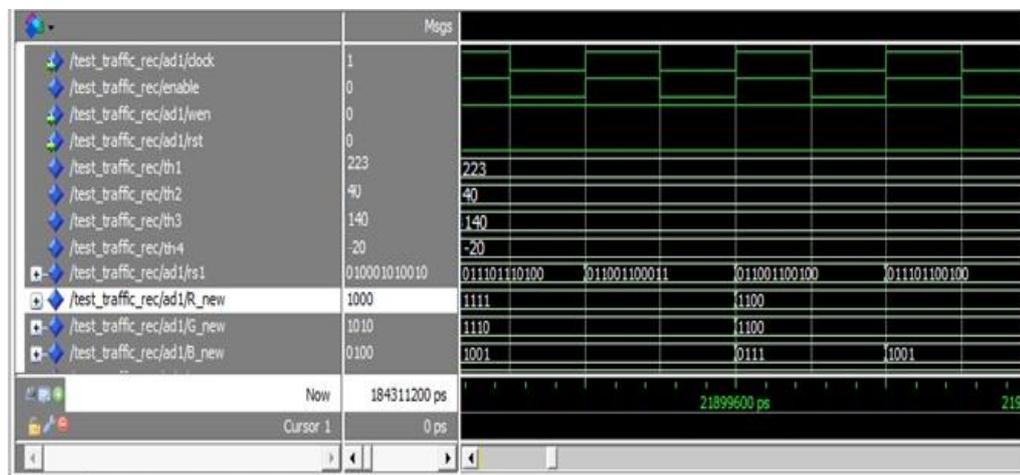


Figure 13. Simulation results of the color adjustment and thresholding tasks

In order to calculate  $R_{new}$ ,  $G_{new}$  and  $B_{new}$  the original values of R, G and B in the RGB444 format are read from the RAM memory then multiplied respectively by  $\alpha_R$ ,  $\alpha_G$  and  $\alpha_B$ . Obtained values that exceed the value 15 are replaced by the value 15 and values in the range 0 to 15 are rounded to the nearest integer. The signal 'wen' is used to enable reading from memory. 'enable' and 'rst' signals are used to control the different calculations and the 'wr' signal is used to enable writing pixel\_out,  $R_{new}$ ,  $G_{new}$  and  $B_{new}$  values to text files to be visualized and compared to Matlab values to check the validity of the developed algorithm. As shown in Figure 15, 'rs1' is a 12 bit std logic vector presenting respectively R (4 bit), G (4 bit) and B (4 bit). Data is read from the memory then each component is extracted and multiplied by the scaling factors. Obtained values are mapped to the range 0 to 15 then difference values are calculated. The 'pixel\_out' that present the pixel in the binary segmented image is decided according to conditions.

The obtained results approve the high precision and accuracy achieved with low computational time cost and low resource utilization. Table 6 summarizes the hardware resources utilization of the developed red color segmentation design synthesized for a Cyclone III FPGA platform (EP3C120F780I7). Results show that this architecture has used a reduced amount of hardware resources.

Table 6. Hardware resource occupation

Element	Occupation
<b>Total logic elements</b>	2,384 / 119,088 (2%)
<b>Total registers</b>	304
<b>Total memory bits</b>	960,000 / 3,981,312 (24%)
<b>Embedded Multiplier 9-bit elements</b>	14 / 576 (2%)
<b>Total PLLs</b>	0/4 (0 %)

#### 4. CONCLUSION

In this paper we have proposed an improved road sign segmentation technique using color information. The developed approach is specially designed to segment road signs with red borders in order to be used as a detection stage. The VHDL circuit of this task is presented in the paper that shows the simplicity of the proposed method that doesn't require complex hardware calculations. Compared to approaches in (Khanal et al., 2012), (Zumra et al., 2014), (Sheng et al., 2012) and (Qin et al., 2010), our method shows a high accuracy and robustness in segmentation for images captured under different day times and different weather and illumination condition. Compared to the method in (Khanal et al., 2012) which is the best one among other methods our method correctly segment faded and blurred signs and gives good results in dark conditions. The automatic classification task of the captured images into 5 classes has solved the problem of the dynamic selection of the adequate thresholds. Furthermore the color adjustment applied on classified images improves the segmentation results. The proposed method not only shows a high detection rate but also low execution time is achieved with an average time of 0.07 seconds among the testing dataset using a 2.40 GHz CPU. The simple calculation operations require simple hardware calculations that make the method well suitable for embedded and real-time applications. Implementation and simulation results of the VHDL circuit of the proposed method have shown high accuracy and precision with low hardware resource occupation.

Our future work will be focused on the implementation of the whole speed limit road sign recognition system that will be addressed for advanced driver assistance applications.

#### REFERENCES

- Agarwal, C. and Sharma, A., 2011. Image understanding using decision tree based machine learning. *International Conference on Information Technology and Multimedia*.
- Ajay Raghavan, Juan Liu and Bhaskar Saha, and Robert Price, 2012. Reference image-independent fault detection in transportation camera systems for nighttime scenes. *15th International IEEE Conference on Intelligent Transportation Systems Anchorage, A Alaska, USA, September 16-19*.
- Benallal, M. and Meunier, J. , 2003. Real-time color segmentation of road signs. *IEEE CCECE Canadian Conference on Electrical and Computer Engineering.(Volume: 3)*
- Bishesh Khanal, Sharib Ali, Desire Sidibe.,2012. Robust Road Signs Segmentation in Color Images. *International Conference on Computer Vision Theory and Applications, Rome, Italy. pp.307-310. <hal-00658086>*
- Edmund Y. Lam and George S. K. Fung, 2009. Automatic White Balancing in Digital Photography. *Single-Sensor Imaging: Methods and Applications for Digital Cameras, Taylor & Francis Group*

- Fei Qin, Bin Fang and Hengjun Zhao, 2010. Traffic Sign Segmentation and Recognition in Scene Images. *IEEE Chinese Conference on Pattern Recognition (CCPR)*
- Hanene Rouabeh, Chokri Abdelmoula and Mohamed Masmoudi, 2014. Performance Evaluation of Decision Tree and Neural Network Techniques for Road Scene Image Classification Task. *1st International Image Processing, Applications and Systems conference (IEEE-IPAS'14), Hammamet (Tunisia)*
- Hanene Rouabeh, Chokri Abdelmoula and Mohamed Masmoudi, 2016. An improved Red Color Segmentation Algorithm as Part of an Automatic Speed Limit Sign Detection. *Proceedings of the 13<sup>th</sup> International Conference on Applied Computing, Mannheim, Germany, pp. 131-138, ISBN: 9978-989-8533-56-2.*
- Hasan Flayeh : <http://users.du.se/~hfl/>
- Khodadadzadeh, M., Sarrafzade, O. and Ghasseman, H., 2010. A fast traffic sign detection and classification system based on fusion of colour and morphological information. *6th Iranian Machine Vision and Image Processing.*
- Madokoro, H. Utsumi, Y. and Sato, K. , 2012. Scene classification using unsupervised neural networks for mobile robot vision. *Proceedings of SICE Annual Conference (SICE).*
- Majed Chambah, 2006. More than Color Constancy: Non Uniform Color Cast Correction Techniques & Applications. *Computer Vision and Graphics Computational Imaging and Vision Volume 32, pp 780-786*
- Malik, R., Khurshid, J. and Ahmad, S.N. , 2007. Road Sign Detection and Recognition using Colour Segmentation, Shape Analysis and Template Matching. *International Conference on Machine Learning and Cybernetics. (Volume:6 )*
- Manthira Moorthi, S. , Misra, I. , Kaur, R, Darji, N.P. and Ramakrishnan, R., 2011. Kernel based learning approach for satellite image classification using support vector MACHINE. *Recent Advances in Intelligent Computational Systems (RAICS)*
- Sallah, S.S.M., Hussin, F.A. and Yusoff, M.Z., 2011. Road sign detection and recognition system for real-time embedded applications” International Conference on Electrical, Control and Computer Engineering (INECCE), Pahang.
- Sheng Huang, Chenqiang Gao, Shang Meng, Qiang Li, Changchuan Chen and Chenyang Zhang, 2012. Circular Road Sign Detection and Recognition based on Hough Transform. *IEEE 5th International Congress on Image and Signal Processing.*
- Sinan Mutlu, Samuel Rota Bulò and Oswald Lanz , 2014. Exploiting Color Constancy for Robust Tracking Under Non-uniform Illumination. *Image Analysis and Recognition Lecture Notes in Computer Science, pp 403-410*
- Thongchai Surinwarangkoon, Supot Nitsuwat, and Elvin J. Moore, 2013. Traffic Sign Recognition System for Roadside Images in Poor Condition. *International Journal of Machine Learning and Computing, Vol. 3, No. 1, February 2013*
- Usman Zakir, Eran A. Edirisinghe and Amir Hussain, 2012. Road Sign Detection and Recognition from Video Stream Using HSV, Contourlet Transform and Local Energy Based Shape Histogram. *Advances in Brain Inspired Cognitive Systems Lecture Notes in Computer Science Volume 7366, pp 411-419*
- Xiaojuan BAN, Xiaolong Lv and Jie Chen, 2009. Color image retrieval and classification using fuzzy similarity measure and fuzzy clustering method. *Proceedings of the 48th IEEE Conference on Decision and Control.*
- Xiaozhao Xu, Li Zhuo, Jing Zhang and Lansun Shen, 2009. Research on color constancy under open illumination conditions. *Journal of Electronics (China) September 2009, Volume 26, Issue 5, pp 681-686*
- Xu Qingsong, Su Juan and Liu Tiantian, 2010. A detection and recognition method for prohibition traffic signs. *International Conference on Image Analysis and Signal Processing (IASP)*
- Zumra Malik and Imran Siddiqi, 2014. Detection and Recognition of Traffic Signs from Road Scene Images. *IEEE 12th International Conference on Frontiers of Information Technology.*