Identification of Vehicle Design and Transition of Traffic Signs with Image Processing Method

Metin BILGIN[1], Zekeriya ZEYBEK[2]

Abstract - Automobiles which are the most intensive and preferred means of transportation in all time, have many innovations and features based on the comfort, security and satisfaction of the users. It would be an ideal solution to develop a system that can alert drivers and intervene when needed given the sudden illness seizures that drivers experience in case of an accident. With the object recognition method, traffic signs, warning flashers and passages on the route are detected and the back panel is processed in milliseconds to warn the driver.

In this study, image processing methods are applied in order to create a prototype of an autonomous vehicle by establishing a fast and stable decision making mechanism.

Keywords: *Traffic sign recognition and detection, color segmentation, HSV color filtering, image processing*

1.Introduction

Traffic signs are the symbols and flashers that are designed to alert and direct the vehicle drivers in motion. It is possible to collect traffic signs under four main categories which are prohibition, warning, information and obligation. Signs are specified according to their types, colors and shapes. Prohibition signs are formed on a white background within a red frame. Symbols are formed within a white frame on a blue background or within a red frame on a triangular white background. The information plaques are generally rectangular and formed on a white background in thick blue frames with black symbols and shapes. There are two signs that are different from the rest of the signs, these are "Stop” and “Cannot be entered" signs. The unmarked sign consists of a white horizontal line on a red background. The stop sign, on the other hand, consists of the word "STOP" written in white with capital letters within the shape of a hexagon on a red background.

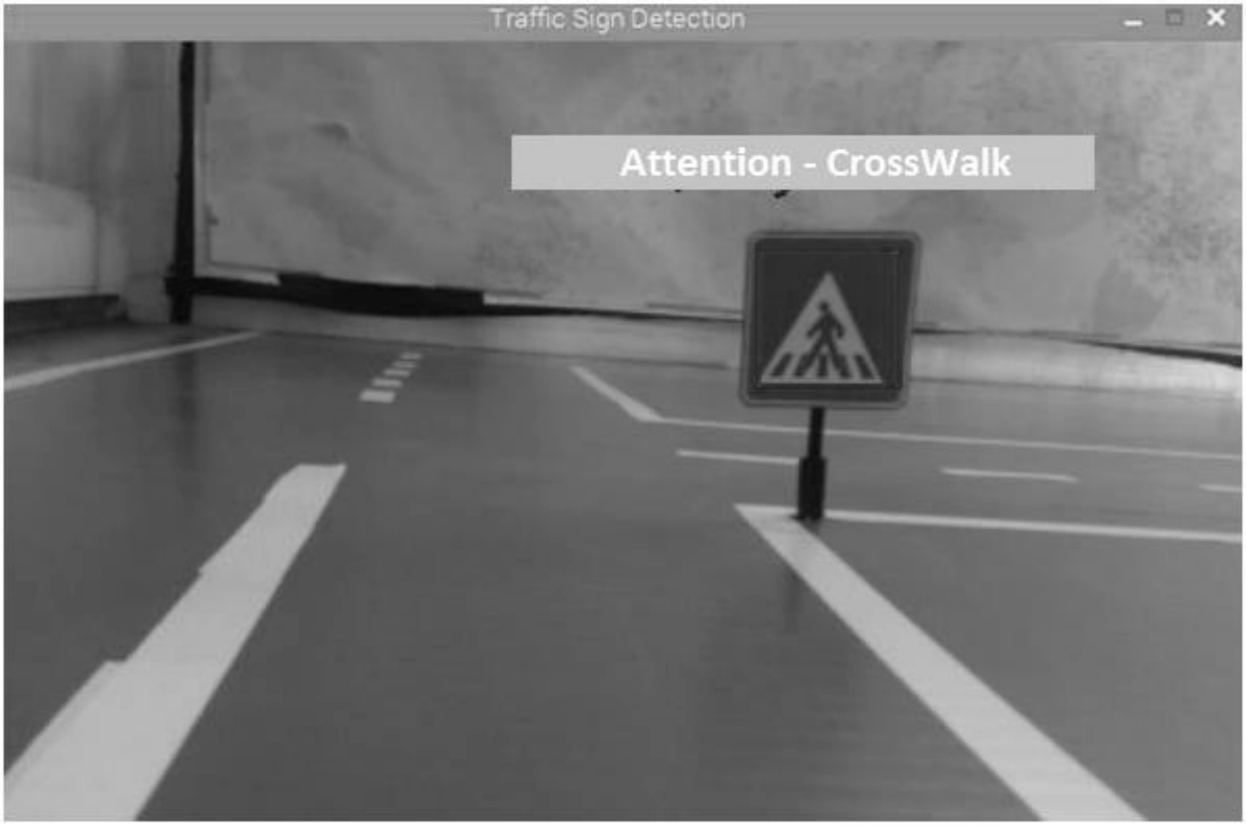
Traffic sign recognition has become popular in recent years and accordingly the number of methods has also increased. In this project, first, a snapshot will be taken using a camera. This image will then be converted to grayscale in order the desired shape and color to be determined. Then, the designated symbol will be identified, and the vehicle will perform the necessary physical operation. All these scan and recognition operations will protect the vehicle's course by defining the signs in the order of operation.

[1] *Bursa Uludağ University, metinbilgin@uludag.edu.tr*
[2] *Bursa Technial University, zekeriyazeybekk@gmail.com*

People are constantly faced with recognition problems in image processing based operations that they use in everyday life. Many recognition situations such as face recognition, finger recognition, disease diagnosis, etc. are needed in different forms and areas. In each case, there are characteristics to be examined due to obligation and examination techniques are available depending on these references.

When the previously written studies are examined, the image processing method and the sign detection and recognition systems have been applied using different methods. There are very few applications available as a tool capable of deciding what to do by recognizing traffic signs. Current study aims to create a tool that can move autonomously utilizing image processing techniques with the aid of a camera.

The main purpose of recognizing and determining the traffic signs is to warn and intervene in cases of loss of sensitivity due to environmental or personal problems, where the driver's reflexes are weakened. The camera that forms the entrance to the system is located on the front of the vehicle. The camera captures live images and continuously generates instantaneous input. These inputs are processed in the processor in order to perform detection, recognition and classification operations. As can be seen in Figure 1, the detected object is determined. As an output, control is provided by sending necessary signals to the related components of the vehicle.



Attention - CrossWalk

Figure 1. Signal Recognition System

2. Method

Techniques in related fields such as pattern detection, machine language, artificial intelligence, and others are used in the processing of the data that the camera captures. The basic technical pattern used in this project is the process itself.

2.1. Visual Object Detection

The input of the instant video viewing program from the test camera that is located on the ceiling of the vehicle is created. The snapshots taken from the videotape have significant blurriness. This is due to the fact that the vehicle is in motion and pixel shifts occurring while images are being taken. The snapshots that are taken for minimizing this blur are used in rows and columns. Signs are detected after multi-stage operations.

Then, a color range is determined, and then the image with the RGB color space captured from the camera device is converted to the related hue, saturation and value (HSV) color space.
On the image, the thresholding, grinding and expansion operators are applied. Using methods such as edge detection, colors are distinguished and objects are marked.

HSV space will be used for color detection; shaving in RGB color space is insufficient compared to shaving in HSV color space. Since the value of H (HUE) in HSV is changed more discernibly, the detection of different colored objects is much easier.

2.1.1. Histogram Synchronization

The images taken from the camera do not always look very good. The image quality can be distorted due to various factors, such as light reflections. Therefore, the method of histogram equalization is a very suitable method of healing [1,2]. Histogram equalization is used for improving low visibility in images where color values are not uniformly distributed. This method is applied locally, only to a specific region. Global histogram equalization, on the other hand, is applied when the entire image is at stake whereas local histogram equalization can be applied to a certain region of the image. Non-linear transformation is performed depending on the probability distribution. On this count, when the probability of high probability is high, the low probability levels become closer to each other.

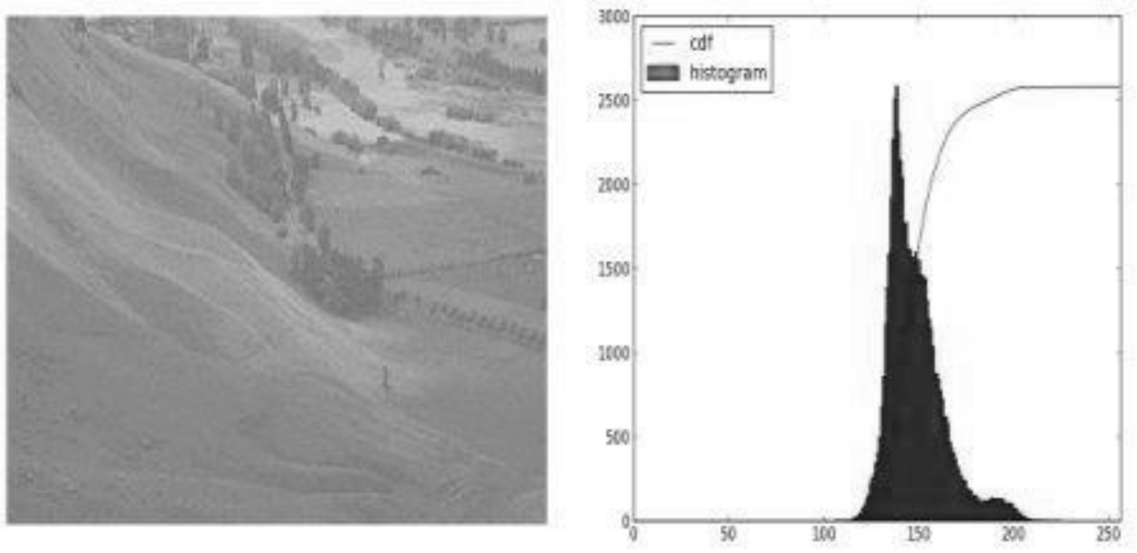


Figure 2. Before Histogram Equalization [3]

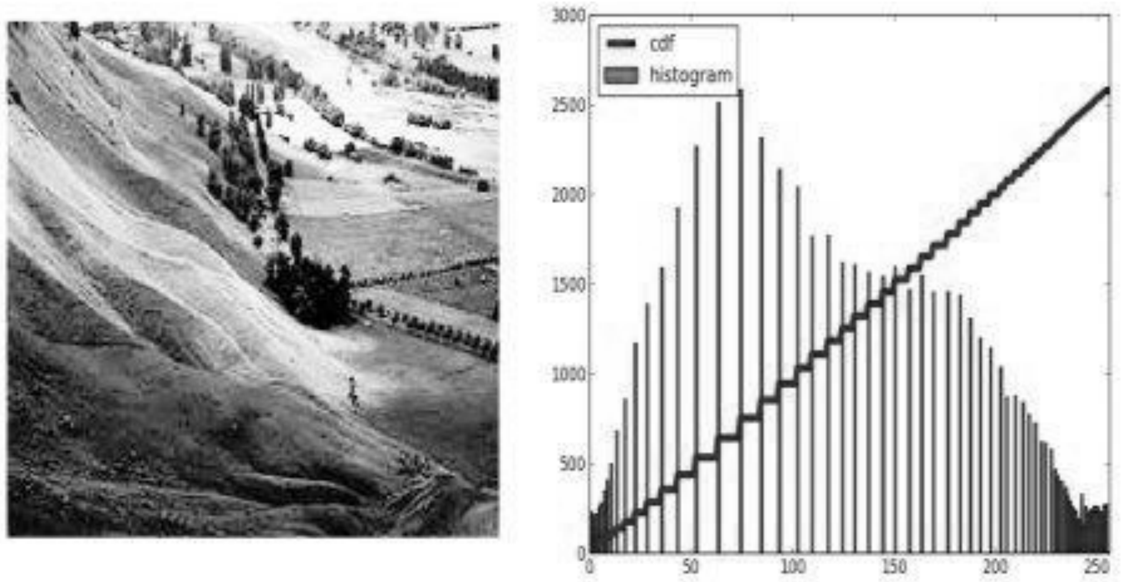


Figure 3. After Histogram Equalization [3]

2.1.2. HSV Color Filtering Method

The R (Red), G (Green) and B (Blue) color space is defined by a mixture of three main colors (red, green, blue). The HSV color space defines the color Hue, Saturation and Value [4]. Color, saturation, and brightness values are used in the HSV color space, although the mixture of colors is used in the RGB color space. Saturation refers to the brightness of color when determining the vitality of color. The brightness value is zero while the color and saturation values for the black color in HSV space can take anywhere from 0 to 255. For the color white, the brightness value is 255.

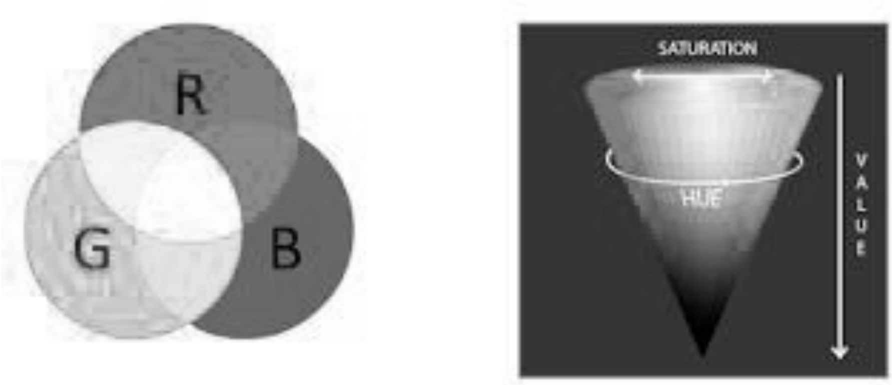


Figure 4. RGB - HSV scales [4]

Accordingly, in any image processing application, it is more convenient to use the HSV color space when we want to distinguish a particular color object. Due to contrasting with RGB, only the hue value can be used for clearly distinguishing colors by applying a threshold value.

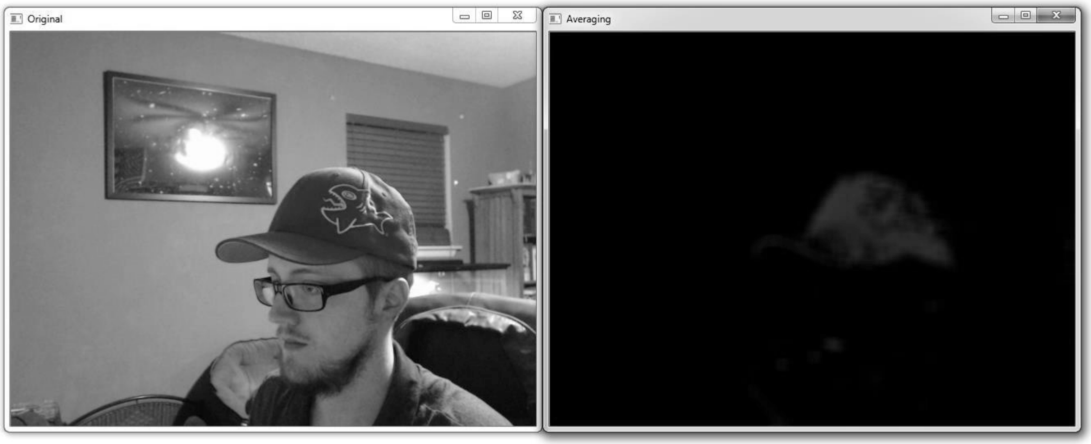


Figure 5. HSV Filter Application [5]

2.2. Object Recognition

Object recognition is very important for image processing. Different algorithms have been developed for this subject, which have been working for a long time. The guideline for this field was the Viola Jones algorithm developed by Paul Viola and Michael Jones in 2001. This algorithm has been announced in the article titled "Rapid Object Detection using a Boosted Cascade of Simple Features" [6]. In the following process, many algorithms have been developed, and many developed methods are available in OpenCV.

There are four different methods for object recognition and detection:

- Template Matching [7] [8]
- HAAR Cascade [9]
- LBP - Local Binary Pattern [10]
- HOG - Histogram of Oriented Gradients [11]

2.2.1. HAAR Cascade Classifier

The HAAR Cascade Classifier is a very functional feature. The classifier is primarily trained with examples and then it recognizes the object. The training consists of the steps of forming positive and negative samples of any object to be perceived, marking positive samples, educating the classifier in this direction and perceiving the object using positive images.

Positive and negative pictures are needed for training the classifier. Positive pictures are the images that contain the target object to be detected whereas the negative images are the images in which the object to be detected is absent. In classifier training, objects in positive pictures are scanned with frames set at certain sizes, as shown in Fig. 6, Fig. 7 and Fig. 8, and the dark luminance values are summed up by adding the pixel values in the black region and the white region in the frame.

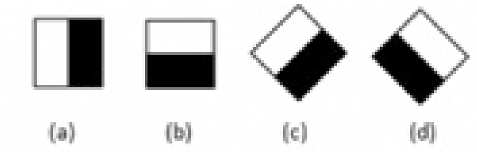


Figure 6. HAAR Cascade - Edge Classifier

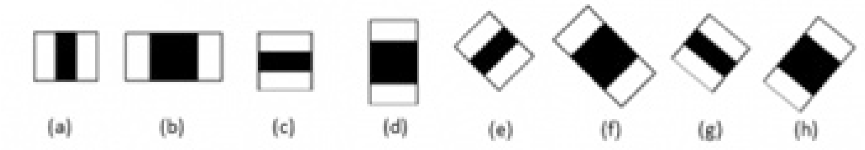


Figure 7. HAAR Cascade - Line Classifier

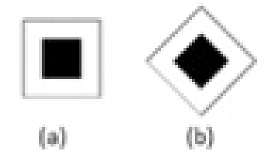


Figure 8. HAAR Cascade - Center and Environment Classifier

These frames, named Feature, are called weak classifiers. Since a weak classifier cannot provide a correct result on its own, there are many weak classifiers found in an object. The object that is searched with a great accuracy will be detected at the point where the classifiers are gathered. For the areas of reference, dark spots are created on the object to be detected. Target values are generated from each of these regions.

When these frames (weak classifiers) are considered for each image size; hundreds of thousands of nuclei will form. Negative images will be scanned so that the majority of the frames that are not going to be used are eliminated as there are no objects in these images.

In positive images, objects will be selected and the frames to be used in the object will be determined. Therefore, the desired object to be detected in the positive images is truncated in millimetric order, and full values are entered in order to facilitate the operation. The large number of positive and negative image samples is important to obtain better results regarding the desired object.

If there are too many pictures, it may cause the processor to think that it will spend a lot of time. One of the most important aspects of image processing is that it is real-time. Therefore, as the number of transactions increases, the latency also increases. However, in the haar cascade classifier, the integrals of the pictures are taken first. Thus, rather than calculating individual sums of pixel values, it is calculated with an integral. At this point, the processor gets faster by getting rid of the individual account.

Instead of being repeatedly scanned by each frame at the stage of object detection, only the parts that match the previous step are scanned in the next step, reducing the processing load. Although the Haar cascade method is very fast in terms of these aspects, it slows down at certain stages in the application phase, hence it appears as if there are slight deviations in real time. The real-time delays vary depending on parameters such as the classifier's training pattern, sample number, and number of repetitions.

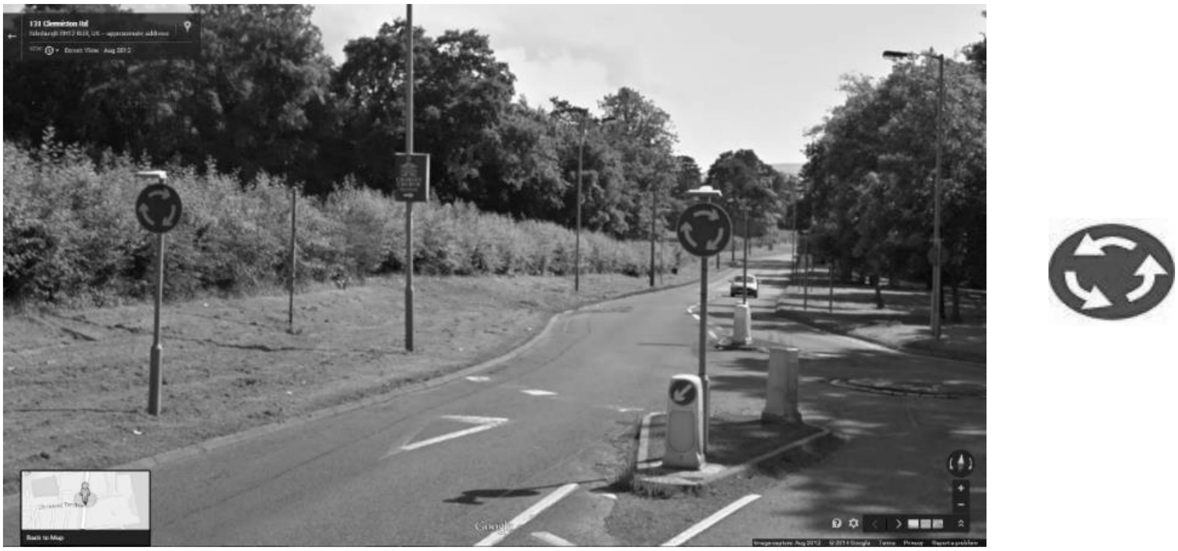
The minimum hit rate, maximum false rate, and alarm rate values that the classifier takes as a reference for the detection of the desired objects are available. The classifier tries to reach these values with the algorithm determined at each stage of the training.

When the training is completed, a file with an ".xml" extension is created which will be used as the source. With this source file created, the desired object is detected using OpenCV library.

2.2.2. Template Matching

Other methods, apart from Template Matching, are the Machine Learning algorithms which are defined as classifiers. These algorithms are used to recognize a learned object, and these algorithms are the most commonly used methods for object recognition. Objects for classifiers can be taught with Deep Learning methods. The template mapping method carries out the matching process by searching the image template in all the pixels on the source image.
There are two components required for this method:

a) Source Image: A snapshot of the video.
b) Template Image: The template to search for in the source image.



(a) **(b)**

Figure 9. Template Matching Example: (a) Source Image, (b) Template

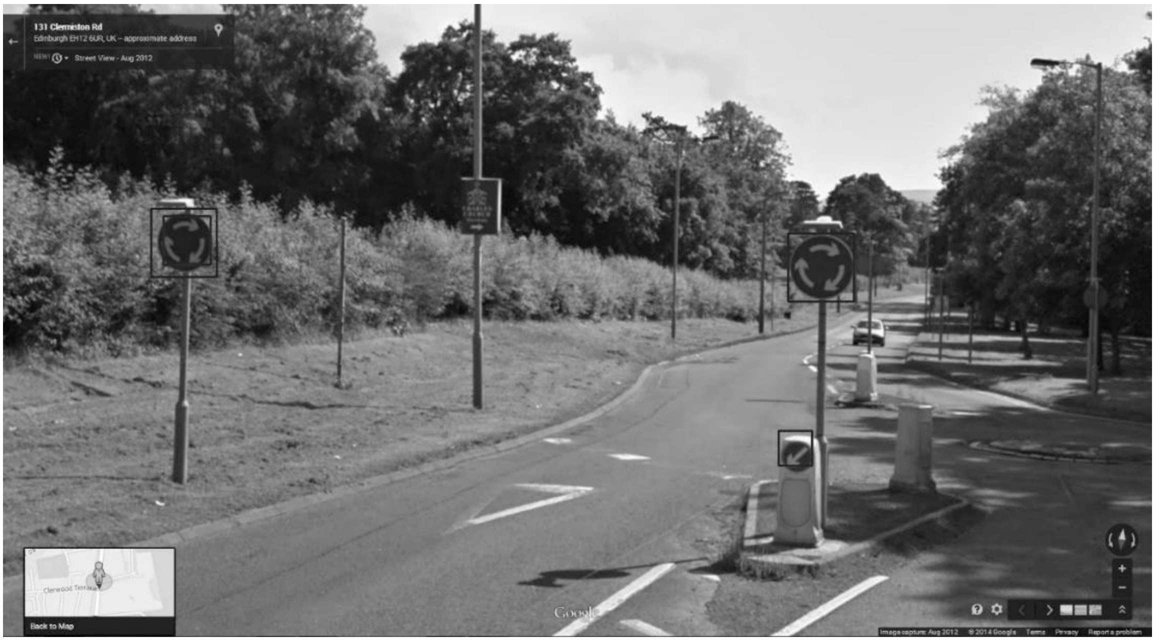

131 Clermiston Rd
Edinburgh EH12 6UR, UK – approximate address
NEW! Street View - Aug 2012
Clerwood Terrace
Back to Map
Google
Image capture: Aug 2012 ©2014 Google Terms Privacy Report a problem

Figure 10. Template Matching Example Result

To match the template image, the source image is checked pixel by pixel from left to right and from top to bottom. I matrix is stored in R matrix for each T position on the matrix. Each position (x, y) in the R matrix contains the matching components.

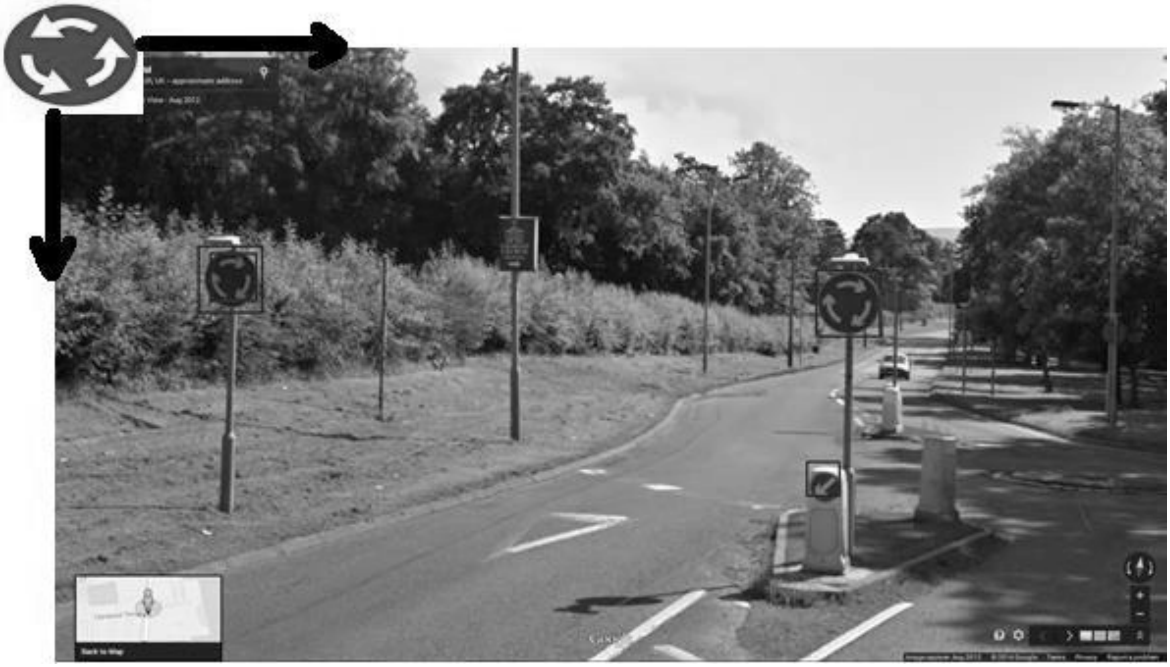


Figure 11. Template Source Searching in the Image

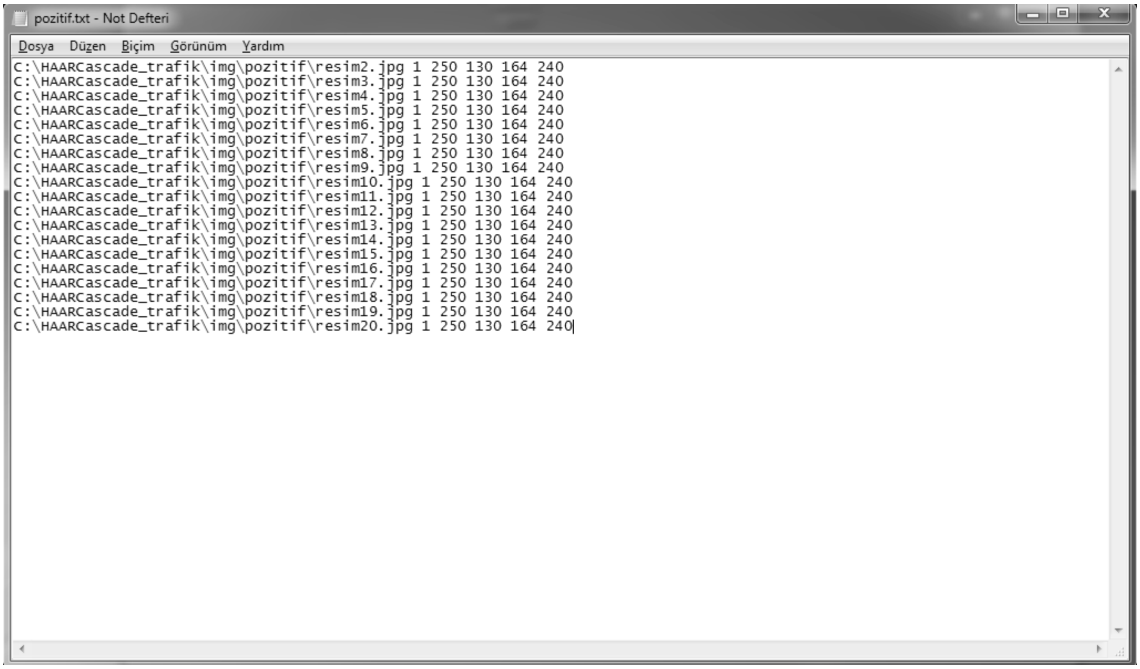
3. Method Used

Positive images are the example images of the objects that we want our classifier to find. In case these samples have different brightness values and different angles, a more effective classifier can be formed. A good classifier needs to be trained with thousands of examples, though with fewer examples a classifier can be created.

In this project, 250 positive and 100 negative pictures were used for each sign. The total number of used positive samples is 1500 and the number of used negative samples is 600.

3.1. Haar Cascade Training

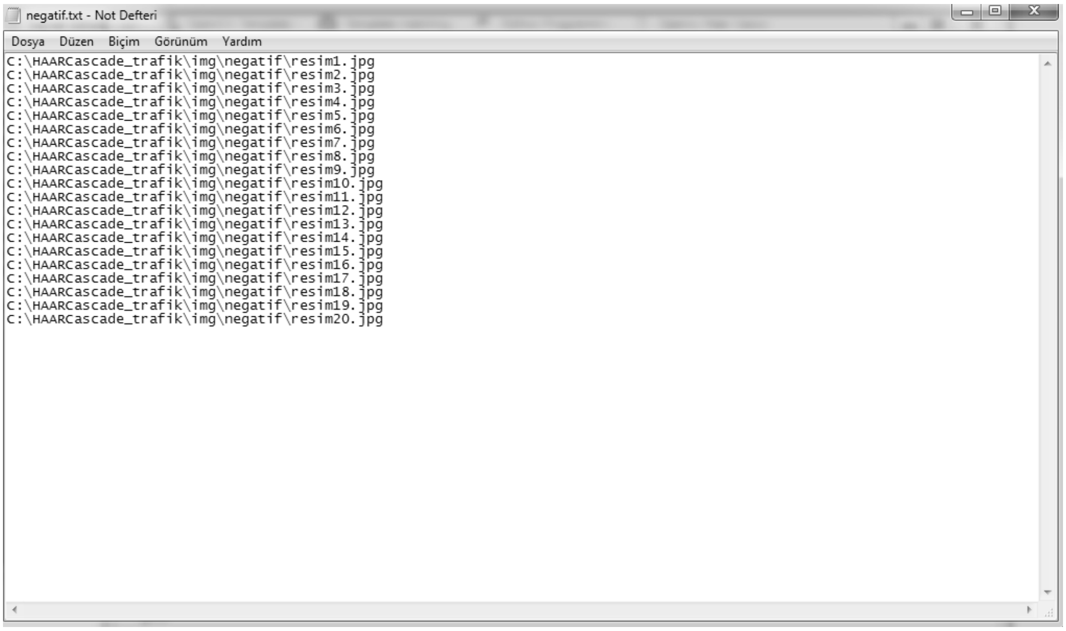
Cascade training requires utmost care [12]. Otherwise, the detection process fails. In order to avoid complicated training, related folders are created in the C directory and different files are created in this folder including the positive and negative images. Positive images should be placed in the positive folder, and negative images should be placed in the negative folder. Within the main folder, a text document is created for both positive and negative images and picture extensions, coordinates and pixels are written in it.



```
C:\HAARCascade_trafik\img\pozitif\resim2.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim3.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim4.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim5.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim6.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim7.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim8.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim9.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim10.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim11.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim12.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim13.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim14.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim15.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim16.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim17.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim18.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim19.jpg 1 250 130 164 240
C:\HAARCascade_trafik\img\pozitif\resim20.jpg 1 250 130 164 240
```

Figure 12. Cascade Training Positive Pictures

The images will be clipped and processed by the "createSamples" function in the OpenCv library according to the values indicated on their sides.



```
C:\HAARCascade_trafik\img\negatif\resim1.jpg
C:\HAARCascade_trafik\img\negatif\resim2.jpg
C:\HAARCascade_trafik\img\negatif\resim3.jpg
C:\HAARCascade_trafik\img\negatif\resim4.jpg
C:\HAARCascade_trafik\img\negatif\resim5.jpg
C:\HAARCascade_trafik\img\negatif\resim6.jpg
C:\HAARCascade_trafik\img\negatif\resim7.jpg
C:\HAARCascade_trafik\img\negatif\resim8.jpg
C:\HAARCascade_trafik\img\negatif\resim9.jpg
C:\HAARCascade_trafik\img\negatif\resim10.jpg
C:\HAARCascade_trafik\img\negatif\resim11.jpg
C:\HAARCascade_trafik\img\negatif\resim12.jpg
C:\HAARCascade_trafik\img\negatif\resim13.jpg
C:\HAARCascade_trafik\img\negatif\resim14.jpg
C:\HAARCascade_trafik\img\negatif\resim15.jpg
C:\HAARCascade_trafik\img\negatif\resim16.jpg
C:\HAARCascade_trafik\img\negatif\resim17.jpg
C:\HAARCascade_trafik\img\negatif\resim18.jpg
C:\HAARCascade_trafik\img\negatif\resim19.jpg
C:\HAARCascade_trafik\img\negatif\resim20.jpg
```

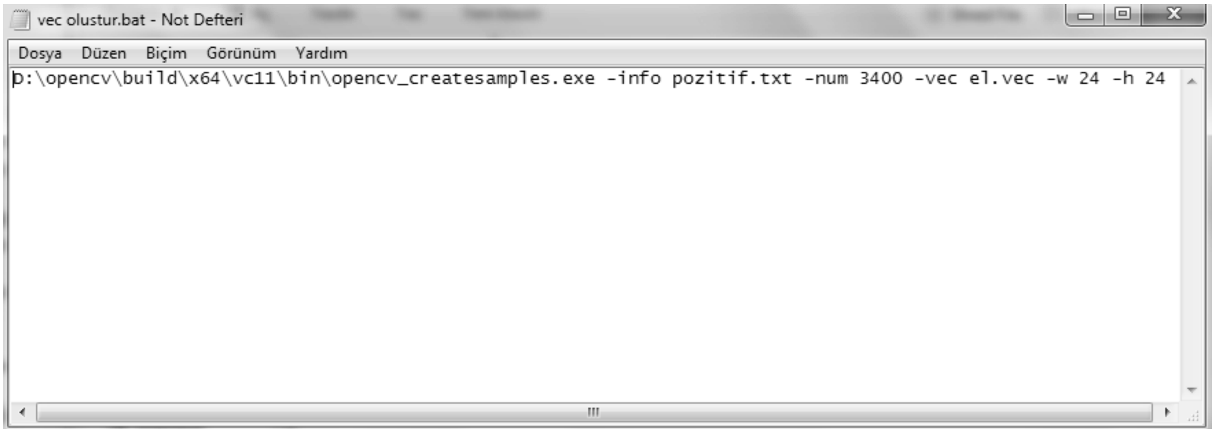
Figure 13. Cascade Training Negative Pictures

The images of the environments that look similar to the environment we want our classifier to detect and the objects we are looking for are also recorded as negative images. The image extensions valid for positive images are also valid for negative images and one must be careful of the samples are of the same sizes.

For the classifier to work effectively, the number of positive images must be very large. The number of negative images should be less than the number of positive images, not exceeding the half-time if possible. The cause will be understood in the form that we will calculate in the progressive lines.

After filling the Positive and Negative folders with examples and saving the names and paths of these examples in txt files, we need to gray scale them for use in the cascading classifier of the positive images, and we need to create a vector file by trimming the object part.

A text document with .bat extension is created for sending the vector file to "OpenCV_CreateSamples.exe" file. "OpenCV_CreateSamples.exe" file path is written in the edited file with the necessary parameters.



```
D:\opencv\build\x64\vc11\bin\opencv_createsamples.exe -info pozitif.txt -num 3400 -vec el.vec -w 24 -h 24
```

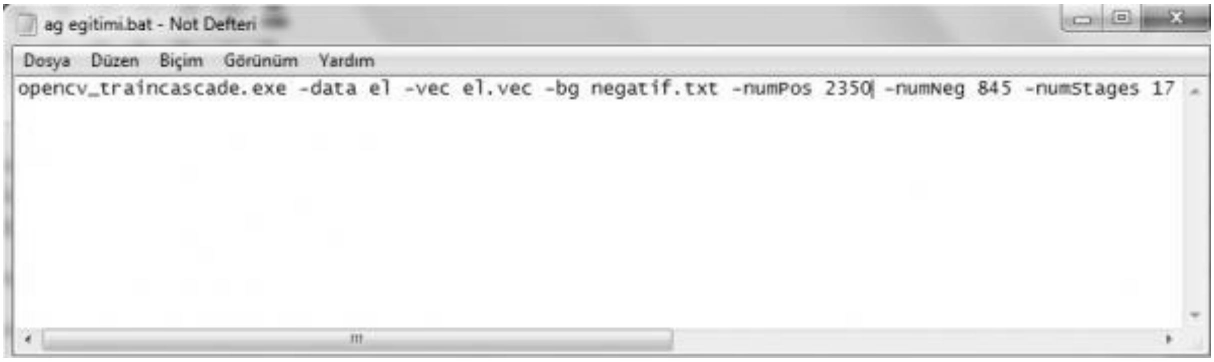
Figure 14. Cascade Training vector file

"OpenCV \ build \ x86 \ vc11 \ bin \ opencv_createsamples.exe" section of your OpenCv directory is the path to the file "OpenCV_Createamples.exe". When the command window sees this command, it executes the "exe" file located in the specified path.

The "-info positive.txt" section is used for taking the txt file that we created as a parameter, which holds the information that the file "OpenCV_CreateSamples.exe" will find the required images and the way to crop them.
In the "-num 3400" section, the "positive.txt" file contains the number of sample images that has been saved. The vector file contains information on how many images are created. The picture number from 2000 to 4000 is ideal for an effective detection.

In the "-vec el.vec" part, the name of the created vector file is entered. "-w 24 -h 24" -w specifies the width, and -h specifies the height. This section resizes the picture width and height so that the picture is cropped in the specified coordinates. This classifier makes it easy to work if you need to work with thousands of pixels; working on 24 * 24 pixels will be faster and easier. The vector file with ".bat" extension runs automatically and it is automatically closed after the necessary data is collected.

Training can be done with two exe files located in OpenCv library. These files are "OpenCV_Haartraining.exe" and “OpenCV_Traincascade.exe” files. "OpenCV_Haartraining.exe" only uses old algorithms that use the haar method. "OpenCV_Traincascade.exe" is the most up-to-date classifier training file and can be trained with both haar method and local binary method. In this project, we are training with "OpenCV_Traincascade.exe". This exe file tries to process the positive images on the negative images until the specified minhitrate and maxfalsealarmrate values are reached. In each step, positive images are scanned with different frames and values are created. When the specified stage value is reached, an xml file is created which can be used to find the object to be detected.


ag egitimi.bat - Not Defteri
Dosya Düzen Biçim Görünüm Yardım

```
opencv_traincascade.exe -data el -vec el.vec -bg negatif.txt -numPos 2350 -numNeg 845 -numStages 17
```

Figure 15. Creating a Cascade Training xml file

With the bezier steps of the previous process, the above file is created and executed, and our library with xml extension is created.

4. Identification of Traffic Signs

Since the created xml file only detects the object, a separate xml file is created for each sign in order to classify the detected object.

traffic_cascade = cv2.CascadeClassifier ('haarcascade_trafic.xml') is written in the form of codes. Then a snapshot is taken from the camera and a Gray Scale filter is applied.

Signs = traffic_cascade. detectMultiScale (gray, 1.3, 5)

The plates are determined by command. Then, rectangle parameters are entered in order to get the plates detected in the for loop into a frame.

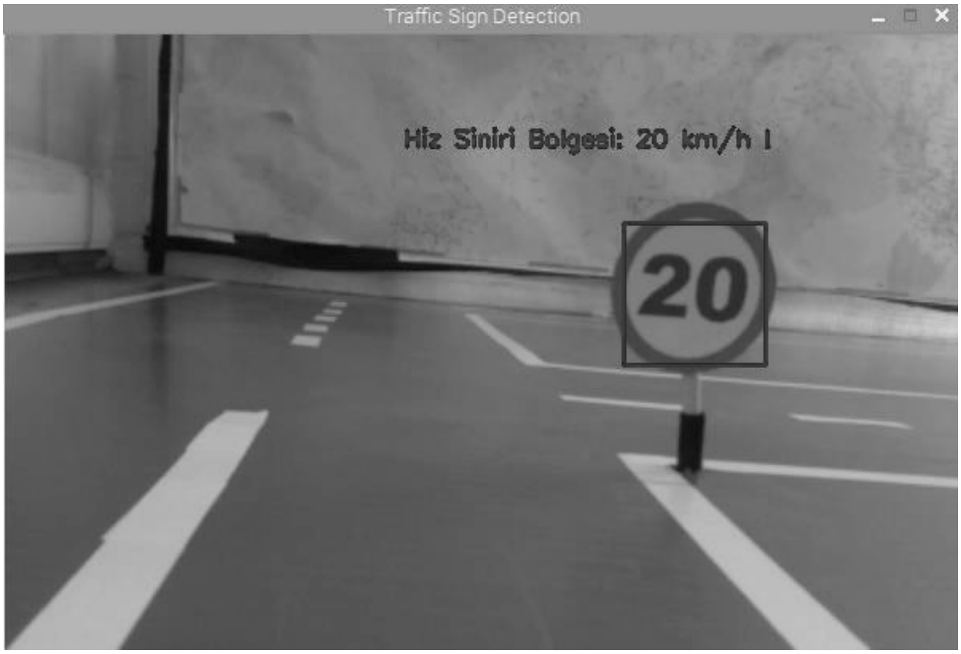


Figure 16. Sign Detection

5.Designed Vehicle

The vehicle to be used in the project is a toy car model. It also checks whether pedestrians are pedestrian crossing, whether there is a vehicle in front of them, or whether there is an external stationary object that could put traffic flow on the road in danger. Ultrasonic distance sensors are used for this control.

RasberryPi 3 motherboard is used as a control mechanism. The camera module is connected to the motherboard and the output generated to control the vehicle as a result of the processes are sent to the motors via the "output" on the card. Continuous input values are read from sensors, such as cameras. All control of the vehicle is provided by this card. Reducing dc motor is preferred for the vehicle to be able to move steadily even at low speeds. Thus, driving stability is increased by increasing vehicle traction. A total of two reducer dc motors are used, one for the rear and one for the wheels.

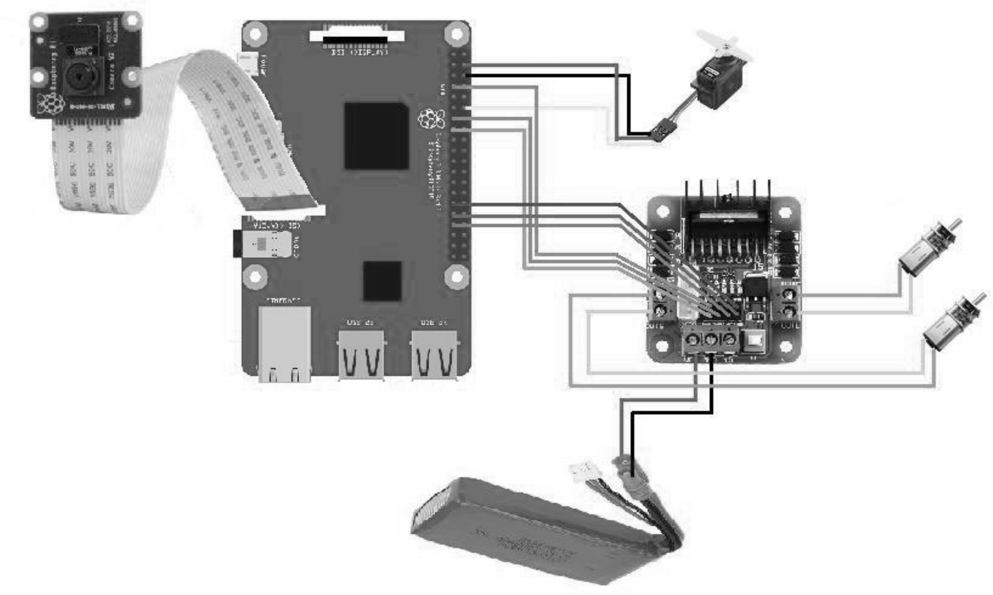


Figure 17. Vehicle connection diagram

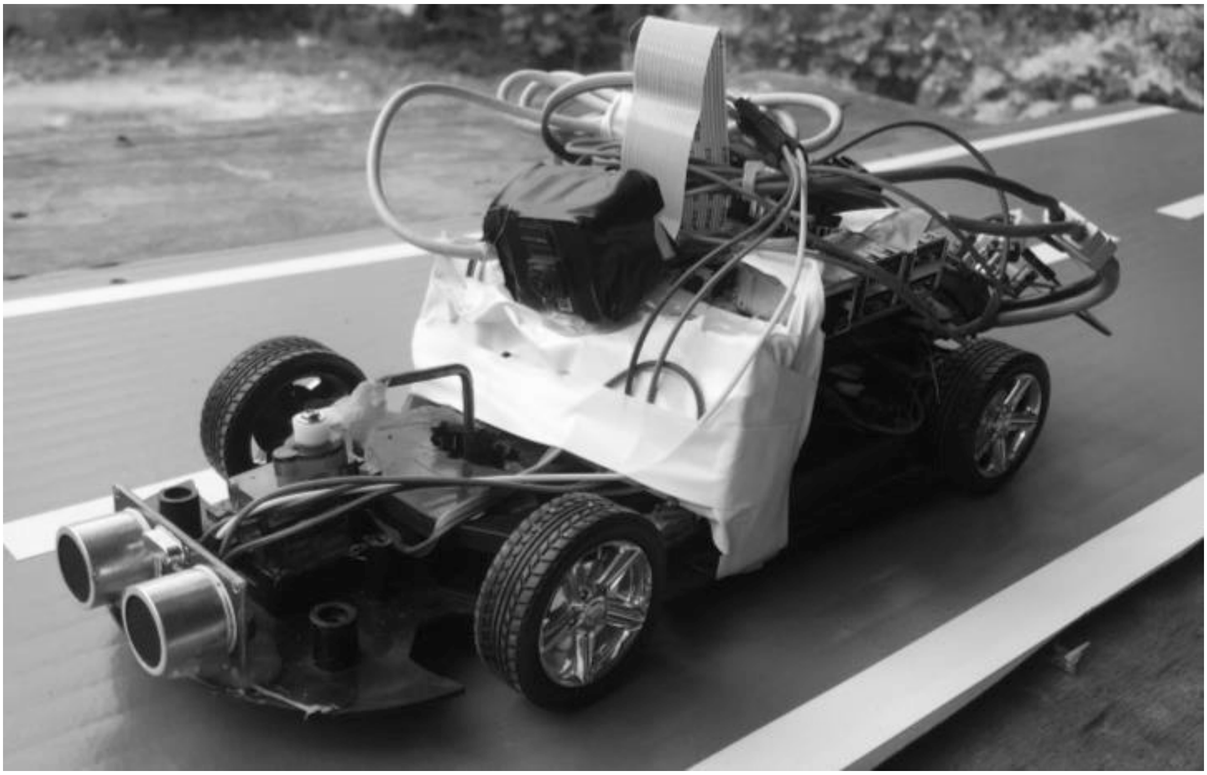


Figure 18. Vehicle View

6.Results and Discussion

Template Matching and Haar Cascade methods that are described in the study are both used. It has been determined that the efficiency of the template matching method is very low as it is very sensitive. In this method, the sample template sought in the environment must have the same maturity. A tolerance value can be given, however this value is very limited. False objects are detected more as the value increases.

The Haar Cascade method has been found to be much more efficient. There are many examples of tablets in terms of different environment, light intensity and position. Thus, the system becomes more stable. Image processing operations are 90-95% successful with the usual positions of the labels.

Since the motherboard used in the project did not have enough capacity, the intensive processes deviate from real time. More efficient algorithms and quality enhancements should be used in order to avoid these deviations and to reduce the worst case scenario. The project can be developed with the improvements in the required calculations and adapted to today's vehicles. In addition, the project can be passed on, not only as a car, but also as an autonomous application for production plants and transport systems in factories.

References

[1] Catalbas, Mehmet Cem, Didem Issever, and Arif Gulten. "Morphological feature extraction with local histogram equalization." *Signal Processing and Communications Applications Conference (SIU), 2015 23th*. IEEE, 2015.

[2] A Medium Corporation, Python ile Görüntü İşleme: Histogram, Normalleştirilmiş Histogram ve Histogram Eşitleme, Erişim: 10.05.2018. (https://medium.com/@sddkal/python-ile-g%C3%B6r%C3%BCnt%C3%BC -i%CC%87%C5%9Fleme-histogram-normalle%C5%9Ftirilmi%C5%9F-histogram-ve-histogram-e%C5%9Fitleme-3d0052174f1f)

[3] OpenCV-Python Tutorials, Histograms - 2: Histogram Equalization Erişim: 10.05.2018. (http://opencv-python tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_histograms/py_histogram_equalization/py_histogram_ equalization.html#histograms-2-histogram-equalization)

[4]Delibaşoğlu, İbrahim, "HSV renk uzayı, Renk filtreleme", Erişim: 10.05.2018. (http://ibrahimdelibasoglu.blogspot.com.tr/2016/11/ hsv-renk-uzay.html)

[5] Python Programming Tutorials, Blurring and Smoothing OpenCV Python Tutorial, Erişim: 10.05.2018. (https://pythonprogramming.net/blurring-smoothing-python-opencv-tutorial/)

[6] Viola, Paul, and Michael Jones. "Rapid object detection using a boosted cascade of simple features." *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. Vol. 1. IEEE, 2001.

[7] Nguyen-Dinh, Long-Van, et al. "Improving online gesture recognition with template matching methods in accelerometer data." *Proc. Int. Conf. Intelligent Systems Design and Applications (ISDA)*. 2012.

[8] Open Source Computer Vision, Template Matching, Erişim: 10.05.2018. (https://docs.opencv.org/trunk/d4/dc6/tutorial_py_template_matching.html)

[9] Python Programming, "Creating your own Haar Cascade OpenCV Python Tutorial", Erişim:10.05.2018. (https://pythonprogramming.net/haar-cascade-object-detection-python-opencv-tutorial/)

[10] Nabiyev, Vasif V., and Asuman Günay. "LBP yardımıyla görüntüdeki kişinin yaşının bulunması." *Cankaya University Journal of Science and Engineering* 8.1 (2010).

[11] Emrullah, A. C. A. R., and Mehmet Siraç Özerdem. "Tarımsal İmge Dokularından HOG Algoritması ile Öznitelik Çıkarımı ve Öznitelik Tabanlı Toprak Neminin Tahmini." *Anatolian Science-Bilgisayar Bilimleri Dergisi* 1.1: 1-7.

[12] Talha Koç, Opencv Haar Cascade Sınıflandırıcısı Eğitimi, Erişim: 10.05.2018.(http://talhakoc.net/opencv-haarcascade-siniflandirici-egitimi/)