Master Thesis
Electrical Engineering
November 2020

Master Thesis
In Electrical Engineering with
Emphasis on Signal Processing
November 2020

Traffic Sign Recognition

Srikanth Reddy Chiluka

Department of Mathematics and Natural Sciences
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

This thesis is submitted to the Department of Mathematics and Natural Sciences at Blekinge Institute of Technology in partial fulfilment of the requirements for the degree of Master of Science in Electrical Engineering with Emphasis on Signal Processing.

Contact Information:
Author:
Srikanth Reddy Chiluka
E-mail: chilukasrikanth978@gmail.com

Supervisor:
Irina Gertsovich
irina.gertsovich@bth.se

University Examiner:
Dr.Sven Johansson
sven.johansson@bth.se

Dept. of Mathematics and Natural Sciences
Blekinge Institute of Technology
371 79 Karlskrona, Sweden

Internet: www.bth.se
Phone: +46 455 38 5000 SE
Fax: +46 455 38 50 57

Abstract

Smart vehicles with capabilities of autonomous driving are a big revolution in automobile industry. The vehicles can sense their environment and react based on it. It replaces the manual driver. Recognition of traffic sign is an important enabler for autonomous driving. Camera installed in the vehicle captures the traffic sign on the road and they must be recognized accurately for triggering the suitable action. In this thesis both image processing and Euclidean distance matching are used to pre-process and classify the traffic signs and thresholding and thinning are applied on image for feature extraction. In this work, a simple, efficient traffic sign recognition system with low computational time and to achieve good accuracy is proposed. Time to classify the traffic sign is achieved in milliseconds and accuracy is maintained using the proposed system.

Keywords: Autonomous Driving, Image processing, Thresholding, Thinning, Euclidean distance matching.

Acknowledgements

Foremost I would like to express my gratitude to my supervisor Irina Gertsovich for her motivation. She gave me the confidence to work what I believed in. Even with the little conversations, suggestions she used to boost my confidence.

I would like to express indebtedness to my parents for their belief in me and supporting me throughout the thesis, both morally and financially. Without their support and motivation in my tough times I would not have finished this thesis.

Table of Contents

Chapter 1 Introduction 1
1.1 Automated Traffic Recognition 1
1.2 History 1
1.3 Introduction to Traffic Sign Recognition 2
1.4 Introduction to Thresholding and Thinning 2
1.5 Aim and Research Questions 2
Chapter 2 Related Work 3
2.1 Survey 3
Chapter 3 Methodology 6
3.1 Proposed Solution 6
3.2 OOPS Design 10
3.3 Data Flow Diagram 13
4.1 Functional Requirement 15
4.2 Non Functional Requirement 15
4.3 Resource Requirements 16
Chapter 5 Implementation 17
5.1 Language Used 17
5.2 Create Reference Database 17
5.3 Feature Extraction and Matching 20
5.4 GUI 21
Chapter 6 Discussion 25
6.1 Comparison of Matching Distances 25
6.2 Comparison to Existing Works 27
Chapter 7 Conclusion 28
7.1 Conclusion 28

List of Figures

Figure 1 Architecture Diagram 6
Figure 2 Result of a Thinning Operation on a Binary Image 8
Figure 3 Templates for Thinning 9
Figure 4 Use Case Diagram 10
Figure 5 Class Diagram 11
Figure 6 Loading Sequence 12
Figure 7 Classification Sequence 12
Figure 8 Level 0 Data Flow Diagram 13
Figure 9 Level 1 Classification Flow 14
Figure 10 Creating Reference Database 18
Figure 11 Intersection Pattern 19
Figure 12 Classification 20
Figure 13 Graphical User Interface 21
Figure 14 Loading of Image Completed 22
Figure 15 Browse Image to Classify 22
Figure 16 Pre-processing Results 23
Figure 17 Classification Results 24
Figure 18 Wrongly Detected Image 25
Figure 19 Matching Distance of Every Image 26
Figure 20 Correctly Classified Images 26
Figure 21 Incorrectly Classified Images 26

Chapter 1 Introduction

1.1 Automated Traffic Recognition

Automatic detection and recognition of traffic signs is an important addition to any smart car in this modern day of life. Having such a tool in a car would alert the driver to possible obstacles or changes in the road and therefore reduce the possibility of accidents. Previous systems for real time detection of traffic signs are limited to a small set of circular or triangular signs. A system capable of recognizing all traffic signs within an acceptable amount of time for a moving car is highly desirable. With the growing technology much research has been going on in the field of traffic signs recognition to improve the safety of drivers.

Many researchers studied the detection and recognition of traffic signs. Most of the approaches to traffic sign recognition follows a two-step process. The first step is the detection of a traffic sign in the image and the second step is to recognize the sign. The second step may involve classification of the sign first into predefined classes.

In the detection stage, colour is commonly used as a factor, since colour contains much information about traffic signs and reflects messages of the signs. When using colour to detect the traffic signs, the most common problem is the illumination changes. Usually, colour segmentation depends on the colour information and the implemented colour space.

Another approach is using the shape detection technique from Gray scale images. The difficulties of this approach are the signs appear in cluttered scenes, which increases the number of candidates since many objects typical of urban environments can be confounded with road signs from a “shape” point of view (building windows, commercial signs, cars, etc); the signs do not always have a perfect shape; the signs taken by different view angles results in different shape; the variance in scale, signs get bigger as a vehicle moves toward them.

In this project, we propose a shape detection-based approach for traffic sign recognition which is simple and easy to implement. The proposed solution can be paralleled to reduce the classification time.

1.2 History

Generally, traffic signs can be analysed using forward-facing cameras in any modern cars, vehicles, and trucks in the world. The most basic use cases of a traffic-sign recognition system are mainly for speed limits. Most of the GPS data would procure speed information, but additional speed limit traffic signs may also be useful to extract information and display it in

the dashboard of the car to make the driver alert about the road signs. Generally, this system is an advanced driver-assistance feature available in all high-end cars, mainly in European vehicles.

1.3 Introduction to Traffic Sign Recognition

Automatic detection and recognition of traffic signs is an important addition to any smart car in this modern day of life. Having such a tool in a car would alert the driver to possible obstacles or changes in the road and therefore reduce the possibility of accidents. In current traffic management systems, there is a high probability that the driver may miss some of the traffic signs on the road because of overcrowding due to many vehicles. With increase in number of vehicles around the world, this problem is expected to grow worse. To solve the transportation safety, automatic traffic sign detection and recognition system has been introduced.

1.4 Introduction to Thresholding and Thinning

Thresholding is a type of image segmentation, where we change the pixels of an image to make the image easier to analyse. In thresholding, we convert an image from colour or grayscale into a binary image, i.e., one that is simply black and white.

Thinning is a morphological operation that is used to remove selected foreground pixels from binary images, somewhat like erosion or opening. It is commonly used to tidy up the output of edge detectors by reducing all lines to single pixel thickness.

1.5 Aim and Research Questions

In this thesis I mainly focus on recognition of a wide set of traffic signs using thresholding and thinning, Euclidean distance matching. Once a traffic sign is detected, three horizontal lines (T, H, B) and three vertical lines (R, V, L) across the image are used to recognize the sign. The number of crossings from a black pixel to a white pixel (peak) on each line is calculated.

Research Questions

- Classify the traffic sign.
- Develop a GUI where we can perform traffic sign detection, classification and measure performance.
- Increase the accuracy of traffic sign classification.

Chapter 2 Related Work

2.1 Survey

These days' recognition and detection of traffic signs is very important for drivers. For the safety of drivers, it is important to have advanced driver assistance system in every modern high-end car. These systems detect, recognize, and classify the traffic sign. But this is not so easy due to various illumination problems. In [1] the author proposed the recognition and detection of traffic signs by dividing it into two stages. Firstly, in the detection stage they localize signs from a whole image. In the classification stage the detected traffic sign is classified into one of the reference signs. In the detection module traffic sign is detected based on the colour, shape, and size. The classification module is done using a multi-layer perceptron neural network. By using neural networks, the classification has achieved a high rate with a good commutation-time.

Advanced driver assistance systems in modern high-end cars helps to save lives. Generally, accidents happen when the driver misses or misunderstands traffic sign. As the surveys tell most of the accidents occur when the driver is distracted. With the increase of driver's attention to the road signs many accidents can be averted. This happens when we try to detect the traffic sign correctly and help the driver in assisting it properly. For this in [2] authors proposed a real time system which integrates single view detection with region-based 3D tracking of traffic signs. By using this system, they are able track multiple traffic signs in 3D, from a single view.

Road signs or traffic signs are an important part of driver assistance system. These signs are designed to regulate the flow of vehicles. These signs give specific information to the drivers regarding any warnings against unexpected road circumstances. Traffic sign information resources are useful for the drivers to take precautions in avoiding accidents. In [3] authors proposed an approach for recognizing the broken area of traffic signs. It is based on recognition system of traffic signs. In this paper they used SIFT matching to adjust the captured image to a standard position. After that histogram bin compared the pre-processed image with the reference image and displayed the final output of the location, percent values of the broken area of the traffic signs. But with the help of this they can detect only broken are of the traffic sign. With the pre-processing they can achieve it more accurately.

Traffic signs information is very important for the safety of drivers. Much research has been done to detect and classify the traffic signs. In [4] authors dealt with the detection and classification of traffic signs especially in outdoor environments. In illumination conditions the

detection of traffic signs is very difficult due to loss of the vision blocked by other objects and the position, orientation cannot be predicted in this situation. Generally, in artificial vision system the important factor to recognize traffic sign is how they detect them and identify their shape. For this work authors proposed a technique based on support vector machines for classifying. Many of the works show the partial solutions in detection and recognition of traffic signs with the other methods. But proposed solution results indicate that the system is robust in numerous conditions. In future they plan to reduce high number of false alarms by using other methods.

Advanced driver assistance system is an important feature in a robotic vehicle that drives automatically on roads. For the detection of traffic sign, in [5] authors proposed an efficient novel approach, histogram-oriented gradient and support vector machine for the classification. The main purpose of detecting traffic signs is improving the safety of drivers on the roads. In the proposed solution authors used an algorithm which integrates top-down and bottom-up visual mechanism to recognize them and pay attention to the traffic signs. With this approach authors achieved high detection rate of 98% and false positive rate of 5% for each traffic sign.

For the regulation of traffic, warning and guiding drivers and pedestrians on roads, traffic sign detection and recognition is very essential feature in advanced driver assistance system in high-end cars especially in European countries. In [6] an automatic method for the detection and recognition of traffic sign in natural scenes is proposed which consists mainly three stages. In the first stage based on brightness and colour features authors detect road signs. In the second stage by using two shape classification criteria authors detect road signs. In the final stage by employing a fringe-adjusted joint transform correlation technique for the recognition of traffic sign. This method shows a new way to detect a traffic sign by integrating image features with the shape information. This paper developed a new methodology for classification of different shapes. This solution also offers a good false alarm rejection rate.

In this generation a vehicle with enormous features becomes an important research issue. In [7] this work, for speed limit sign authors use two types of images, they are colour image and grey image. For colour images, authors can utilize colour information to differentiate specialized patterns. But there can be a misdetection by using speed limit sign because the image may appear rather differently when they try to display the same colour image under different lightning conditions. For speed limit signs detection, authors used both AdaBoost and circular

Hough transform. With the help of support vector machine, a very high performance of recognition of speed-limit signs is achieved.

In this modern era the recognition and detection of traffic signs is an important test for an Advanced Driver Assistance System (ADAS). There are several distinguishing features that traffic signs possess for the detection and recognition – like contrast, shape, and colour. Authors in [8] present a novel system for the real-time detection and recognition of traffic symbols. The important features to detect a traffic sign are based on size, colour, and contrast of the image. In this paper authors used traffic sign dataset provided by UK government, but it is different for every country in the detection and recognition of traffic signs. In [8] they worked on the classification and accuracy by using maximally stable external regions and histogram of gradient.

Chapter 3 Methodology

3.1 Proposed Solution

The architecture of the proposed solution is shown in figure 1.

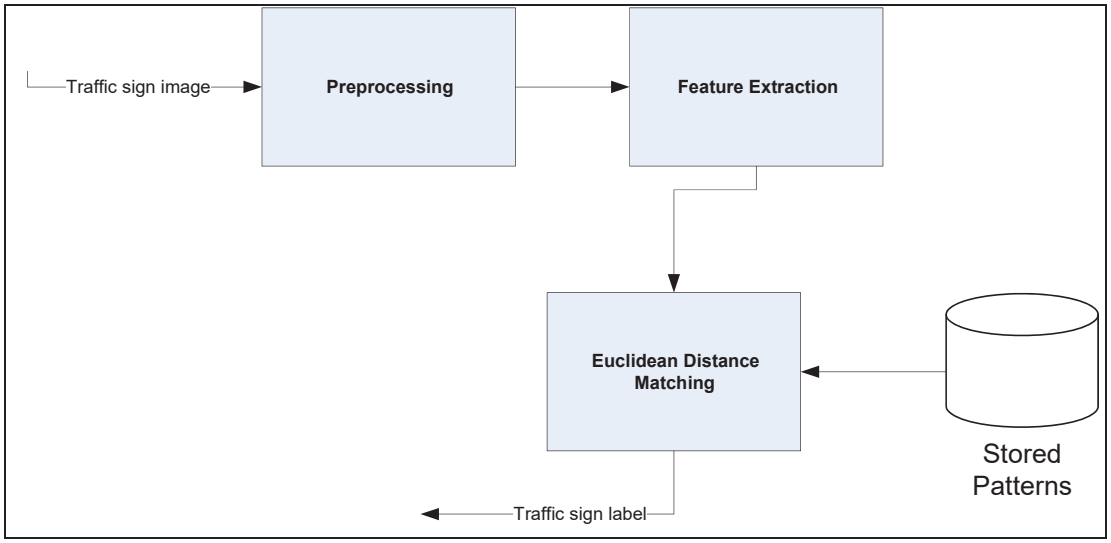


Figure 1 Architecture Diagram

Following are the modules in the above architecture diagram.

Pre-processing: This module processes the image to enhance it and apply thresholding and thinning on image to make suitable for feature extraction.

Image enhancement techniques have been widely used in many applications of image processing where the subjective quality of images is important for human interpretation. Contrast is an important factor in any subjective evaluation of image quality. Contrast is created by the difference in luminance reflected from two adjacent surfaces. In other words, contrast is the difference in visual properties that makes an object distinguishable from other objects and the background. In visual perception, contrast is determined by the difference in the colour and brightness of the object with other objects. Our visual system is more sensitive to contrast than absolute luminance; therefore, we can perceive the world similarly regardless of the considerable changes in illumination conditions. Many algorithms for accomplishing contrast enhancement have been developed and applied to problems in image processing.
Adaptive histogram equalization [9] is applied in this project for image enhancement.

Adaptive histogram equalization is a computer image processing and contrast enhancement technique used to improve contrast and demonstrated effectiveness in images. It differs from ordinary histogram equalization. Generally, adaptive histogram equalization has two problems,

slow speed and over enhancement of noise it produces in homogeneous regions to avoid any noise present in the image we can use adaptive histogram equalization optional parameters especially in homogeneous regions to limit the contrast. While histogram equalization works on entire image, adaptive histogram equalization works on small regions in the image called tiles. By using adaptive histogram equalization, we can enhance the contrast of each tile, so that the histogram of the output region approximately matches a specified histogram. After performing the equalization, adaptive histogram equalization combines neighbouring tiles (regions) to eliminate artificially induced boundaries so now it is suitable for improving the local contrast and enhancing the definitions of edges in each region of an image.

The basic form of adaptive histogram equalization (AHE) method was invented independently by Ketcham, Hummel and Pizer. In this form the method involves by applying to each pixel the histogram equalization based on the pixels in a region surrounding by that pixel or neighbouring region pixels.it means that each pixel is mapped to an intensity proportional to its rank in pixels in the surrounding(regions) it. Adaptive histogram requires time for an image with a range of intensity levels and a contextual regional size. The derivation of the transformation functions from the histograms is same as for ordinary histogram equalization:

The output image pixel at location (x, y) is generated in histogram equalization as

$$g[x,y] = \frac{CDF[f[x,y]] - CDF_{min}}{(N \text{x} M) - CDF_{min}} (L-1).$$

Eq. (1)

CDF is the cumulative distribution function calculated as

$$CDF[j] = \sum_{i=1}^{j} h[i] \ .$$

Eq. (2)

In the above Eq. (2), the h[i] is the histogram bucket. It is calculated as

$$h[i] = \sum_{x=1}^{N} \sum_{y=1}^{M} \begin{cases} 1, if\ f[x,y] = i \\ 0, \quad otherwise \end{cases}.$$

Eq. (3)

The histogram places the value of each pixel f(x, y) into one of the L uniformly spaced buckets. M x N is the size of the image and L=256. CDF_{min} is the smallest non-zero value of the cumulative distribution function.

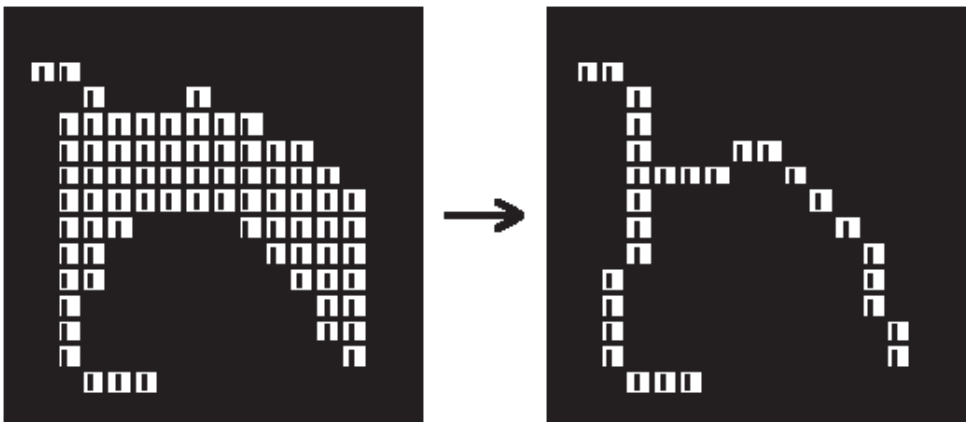
In adaptive histogram equalization the image is divided into small blocks, and each of these blocks is histogram equalized.

Pixels in the borders of the image outside of the sample pixels or near the image boundary must be treated specially, because their neighbourhood would not lie completely within the image.

This applies for example to the pixels to the left or above the blue pixel in the figure. This can be solved by extending the image by mirroring pixel lines and columns with respect to the image boundary. Copying the pixel lines on the border is not appropriate because it would lead to a highly peaked neighbourhood histogram.

Thinning operation [10] is like erosion or opening which is used to remove selected foreground pixels from binary images. Thinning is only applied to binary images and it produces another binary image as output. Thinning is a morphological operation which is determined by structuring element. The binary structuring elements used for thinning are of the extended type described under the hit-and-miss transform (i.e., they can contain both ones and zeros).

Thinning algorithm is a morphological operation that is used to remove selected foreground pixels from binary images. It preserves the topology (extent and connectivity) of the original region while throwing away most of the original foreground pixels. The result of a thinning operation on a simple binary image is given in figure 2.



Original Image Thinned Image

Figure 2 Result of a Thinning Operation on a Binary Image

Thinning operation uses a set of four 3 x 3 templates to scan the image. Figure 3 shows these four templates.

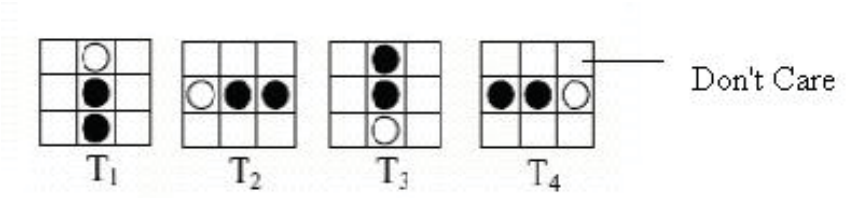


Figure 3 Templates for Thinning

The algorithm for thinning is given below:

1. Find a pixel location (i, j) where the pixels in the image match those in template T1. With this template all pixels along the top of the image are removed moving from left to right and from top to bottom.
2. If the central pixel is not an endpoint, and has connectivity number = 1, then mark this pixel for deletion.

Endpoint pixel: A pixel is considered an endpoint if it is connected to just one other pixel. That is, if a black pixel has only one black neighbour out of the eight possible neighbours.

3. Repeat steps 1 and 2 for all pixel locations matching T1.
4. Repeat steps 1-3 for the rest of the templates: T2, T3, and T4.

T2 will match pixels on the left side of the object, moving from bottom to top and from left to right. T3 will select pixels along the bottom of the image and move from right to left and from bottom to top. T4 locates pixels on the right side of the object, moving from top to bottom and right to left.

5. Set to white the pixels marked for deletion.

The connectivity number is a measure of how many objects are connected with a particular pixel. The following is the equation to calculate connectivity number:

$$C_n = \sum_{k \in S} N_k - (N_k N_{k+1} N_{k+2}) \text{ ,}$$

Eq. (4)

where N_k is the colour of the eight neighbours of the pixel analysed. N_0 is the centre pixel. N_1 is the colour value of the pixel to the right of the central pixel and the rest are numbered in counter clockwise order around the centre.

Feature Extraction: This module draws three horizontal lines and three vertical lines and then find the intersection points of traffic sign on these lines and use those intersection as features.

Matching: This module matches the features of traffic sign image with stored pattern and gives the matching label of the stored patter to whom the features of traffic sign is maximally matching.

The matching is based on maximal matching criteria. In this work it is the smallest distance between two feature vectors. Say, a feature vector of traffic sign image A is $<x_1, x_2, x_3, x_4, \ldots, x_n>$ and the feature vector of traffic sign image B is $<y_1, y_2, y_3, y_4 \ldots . y_n>$. The matching distance between A and B is calculated as

$$m = \sum_{i=1}^{N} F(x_i, y_i).$$

Eq. (5)

The function $F(x_i, y_i)$ is calculated as

$$F(x_i, y_i) = \begin{cases} 1, & if\ x_i == y_i \\ 0, & if\ x_i \neq y_i \end{cases}.$$

Eq. (6)

3.2 OOPS Design

Object oriented design of the project is described below. A use case diagram shown in figure 4 is a type of behavioural diagram created from a use case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

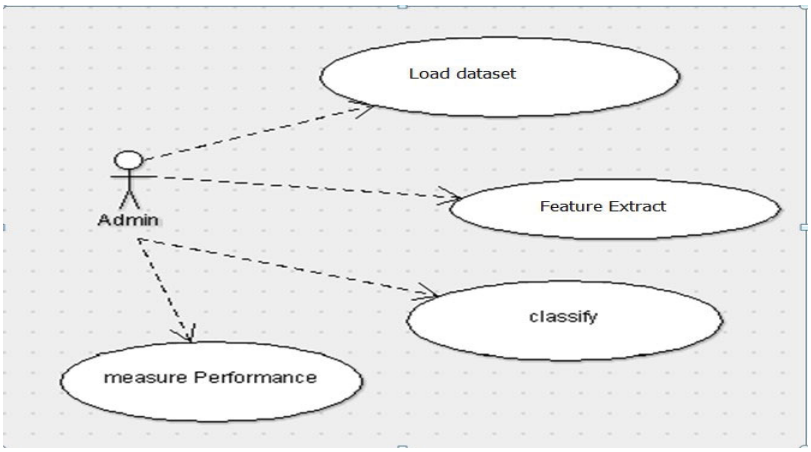


Figure 4 Use Case Diagram

Admin is the user who can do the following functions:

1. Load data set
2. Extract features
3. Classify any given input symbol.

4. Measure the performance.

A class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes.

The class diagram is shown in figure 5.

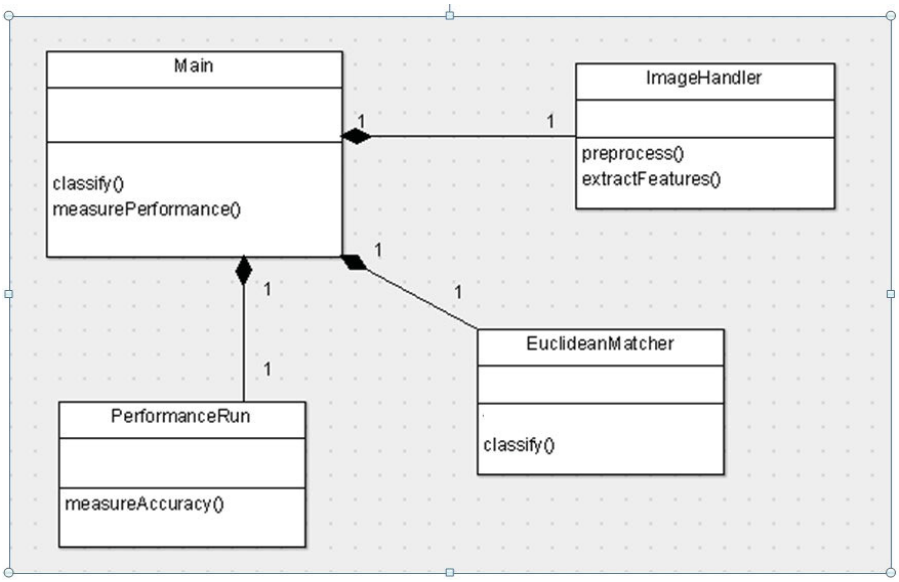


Figure 5 Class Diagram

Following are the classes:

1. Main
2. Image Handler
3. Euclidean Matcher
4. Performance Run

Main is the user interface class. Image Handler implements functionalities of pre-processing the image and extracting the features. Euclidean Matches implements the functionality of classifying the symbol to one of the traffic symbols labels. Performance Run class implements the functionality of performance measurement. A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a chart. The sequence diagrams show below.

Loading flow is given in figure 6.

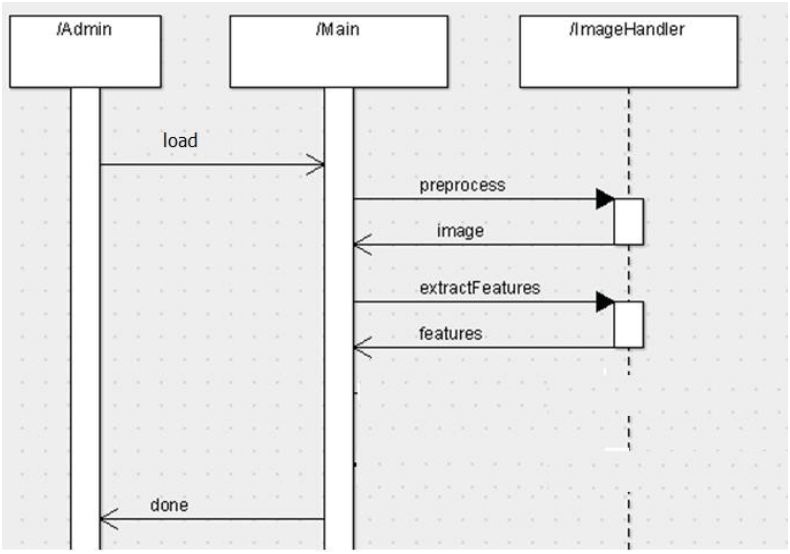


Figure 6 Loading Sequence

Admin calls load with the traffic sign images stored in a folder. Each of the traffic sign images are pre-processed and from its feature is extracted.

The classification flow is given in figure 7.

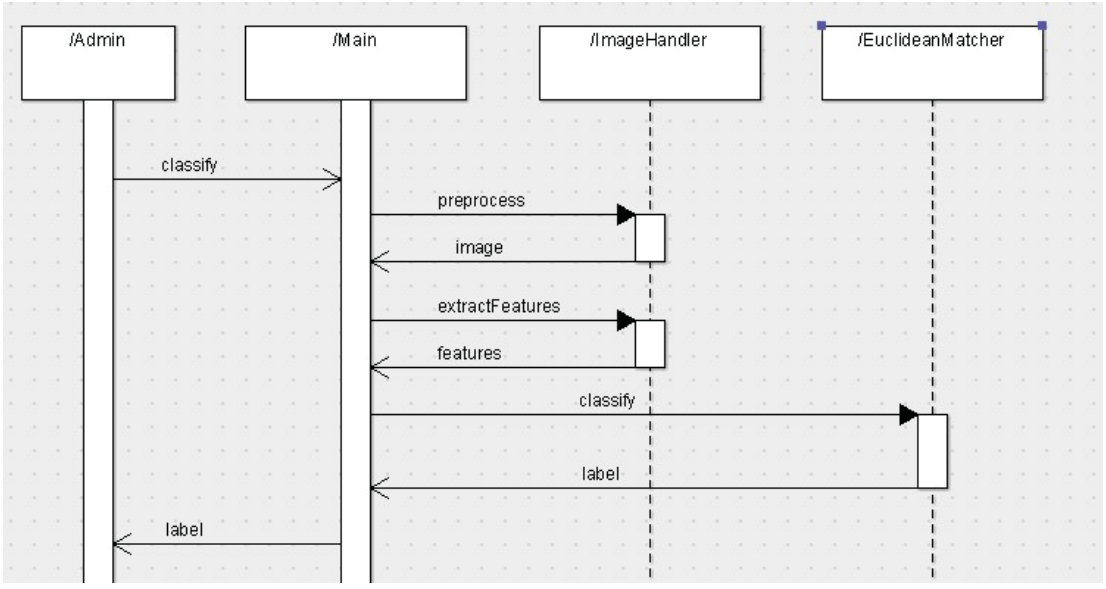


Figure 7 Classification Sequence

Admin calls classify function () giving an unknown traffic symbol image. The image is pre-processed, features are extracted, and it is matched to features of stored image to find the closest match. The label of the closest matching image in the stored repository is given as output.

3.3 Data Flow Diagram

The input, output and the process flow in the system is shown in figure 8.

Level 0 Data Flow Diagram

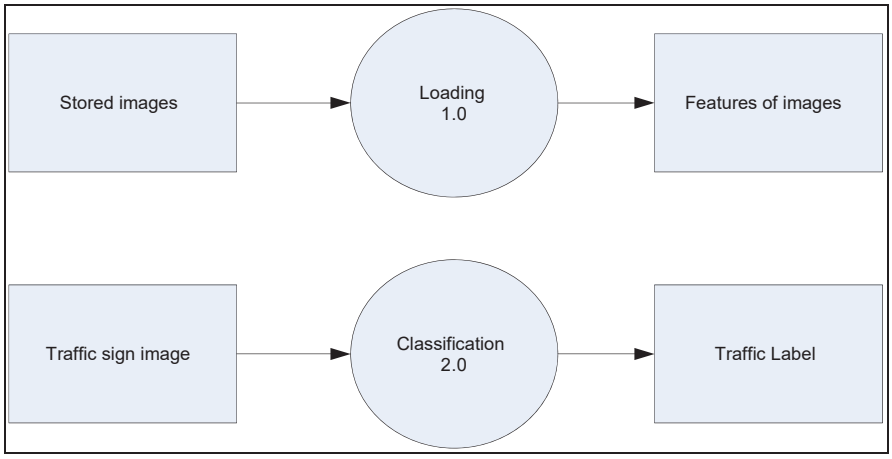


Figure 8 Level 0 Data Flow Diagram

Loading and classification are the two important processes in the system.

Loading process takes the labelled images as input and saves the features of these labelled images as output which is shown in figure 8.

Classification process takes any traffic sign image as input and provides the traffic label as the output which is shown in figure 8.

Level 1 Data Flow Diagram

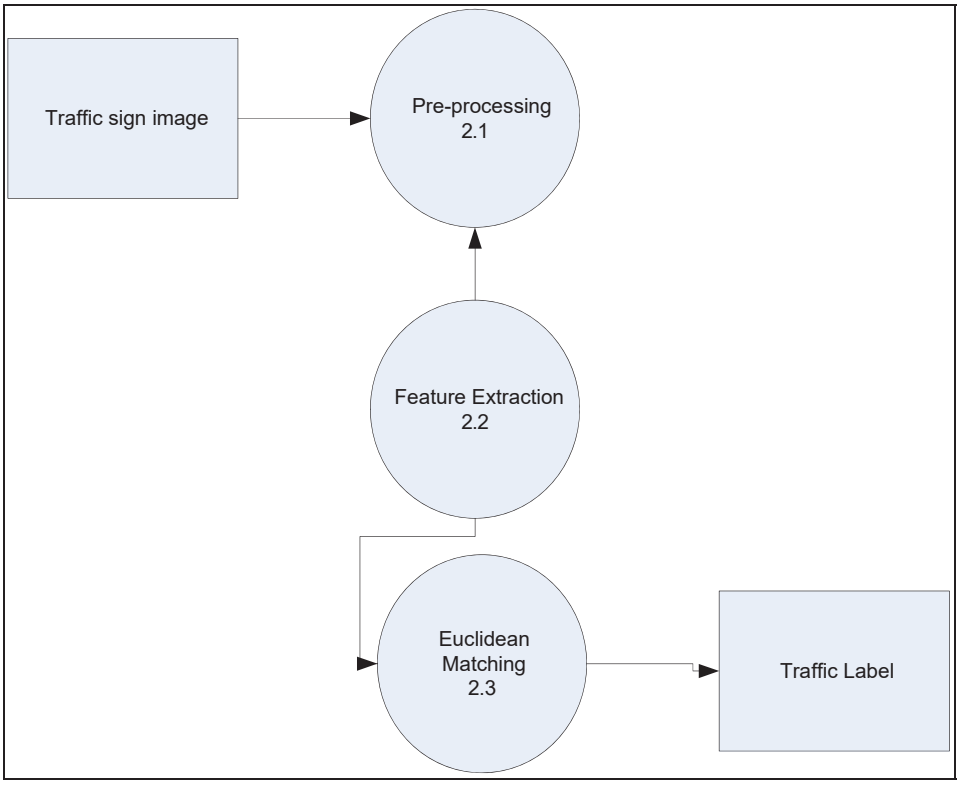


Figure 9 Level 1 Classification Flow

The classification process is split to sub process as shown in figure 9. The input image is pre-processed, features are extracted from it and matching to features of image stored in repository is done. The closest matching image‘s label is given as output.

Chapter 4 Software Requirements

4.1 Functional Requirement

The functionalities to be implemented are.

1. The system is loaded with image for traffic sign and their label.
2. User can browse and choose images.
3. Traffic sign image will be pre-processed to thin the image.
4. The intersection points of traffic sign along three horizontal and three vertical lines is marked and learnt as feature.
5. The feature is matched to the trained images using Euclidean distance to find the label for the traffic sign image.
6. The accuracy of the proposed solution is measured and plotted as graph.

4.2 Non Functional Requirement

The non-functional comes under following category,

1. Product Requirements
2. Organizational Requirements
3. User Requirements
4. Basic Operational Requirements

Product Requirements

Portability: The software developed should work for all kind of traffic sign

Correctness: Traffic sign must be recognized correctly.

Ease of Use: The system is designed in such a way that it provides a user to interact in an easy manner.

Modularity: The system must be modular. Any module can be added or removed.

Robustness: The software is being developed in such a way that the overall performance is optimized, and the user can expect the results within a limited time with almost relevancy and correctness.

Usability: Visual alert should be in an eye-catching colour.

Organizational Requirements

Process Standards: IEEE models are utilized to build up the application which is the standard utilized by the greater part of the standard programming designers everywhere throughout the world.

Design Methods: Modular design approach is used to design the project.

Basic Operational Requirements

Mission profile or situation: The mission of the project is recognizing the traffic sign.

Use situations: It can be used for any traffic sign images.

Operational life cycle: The system must be first loaded with labelled images and then it can be used for classification any number of times.

4.3 Resource Requirements

MATLAB: MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

- Math and computation
- Algorithm development
- Modelling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including Graphical User Interface building

Chapter 5 Implementation

5.1 Language Used.

Implementation phase should perfectly map the design document in a suitable programming language to achieve the necessary final and correct product. Often the product contains flaws and gets ruined due to incorrect programming language chosen for implementation.

In this project, for implementation purpose MATLAB is chosen as the programming language. MATLAB is an interpreted language for numerical computation. It allows one to perform numerical calculations and visualize the results without the need for complicated and time-consuming programming. MATLAB allows its users to accurately solve problems, produce graphics easily and produce code efficiently.

5.2 Create Reference Database

For the experimental part in this work the subset of images from German Traffic Sign Recognition Benchmark (GTSRB) database was used [11]. For creating a reference database, we followed the below flowchart procedure in figure 8. Firstly, when we click on the load button in the GUI interface, “do load” call-back function is called. This function creates a variable in computer memory that will hold the traffic sign images of the reference classes. In the loading process we show a wait bar for informing the user to know how much time it is taking for loading the images into to our database (DB). So initially it will be 0% when we start but it will be 100% when we finish loading the images into a variable. To do this, from the mail folder with the image database we first define the paths to the subfolders with distinct image classes. Now we are going into the subfolder directory, thus choosing a class for a possible template classification. So, after choosing a subdirectory (class), we are going to read each file (image) in that class. “Im” function will read the image from all the classes in the subfolder. After reading all the images, we need to bring all the images to a standard size of [50x50] pixels for feature extraction, because each image can be of different size (resolution). We have six different classes, they are named as 12, 14, 17, 21, 34 and 37. Let us say we have five objects in class 12 that means we need to read all those objects and resize them. All the images are in RGB format. So, once we have processed all the subfolders, all the images will be saved into DB file. In this way we create our DB file before we browse the image from the database.

After choosing the image we need to do pre-processing in which we need to convert the image to grey scale and after that we need to apply adaptive histogram equalization using Eq.(1) to improve contrast of the image. After improving the contrast of the image, we are detecting the

edges of the traffic sign using thinning operation in X and Y directions, for that we are applying norm of the gradient (combining the X and Y directional derivatives). Then we are applying thresholding to convert the image to binary i.e., in thresholding, we select a threshold value and then all the grey level value which is below the selected threshold value is classified as 0 (black i.e., background) and all the grey level which is equal to or greater than the threshold value is classified as 1 (white i.e., foreground). We calculate threshold value by dividing the sum of all the grey level pixel's value of grey scale image by total number of pixels. We can see in figure 10 how a DB file is created, and this entire process runs when we perform the load operation in GUI.

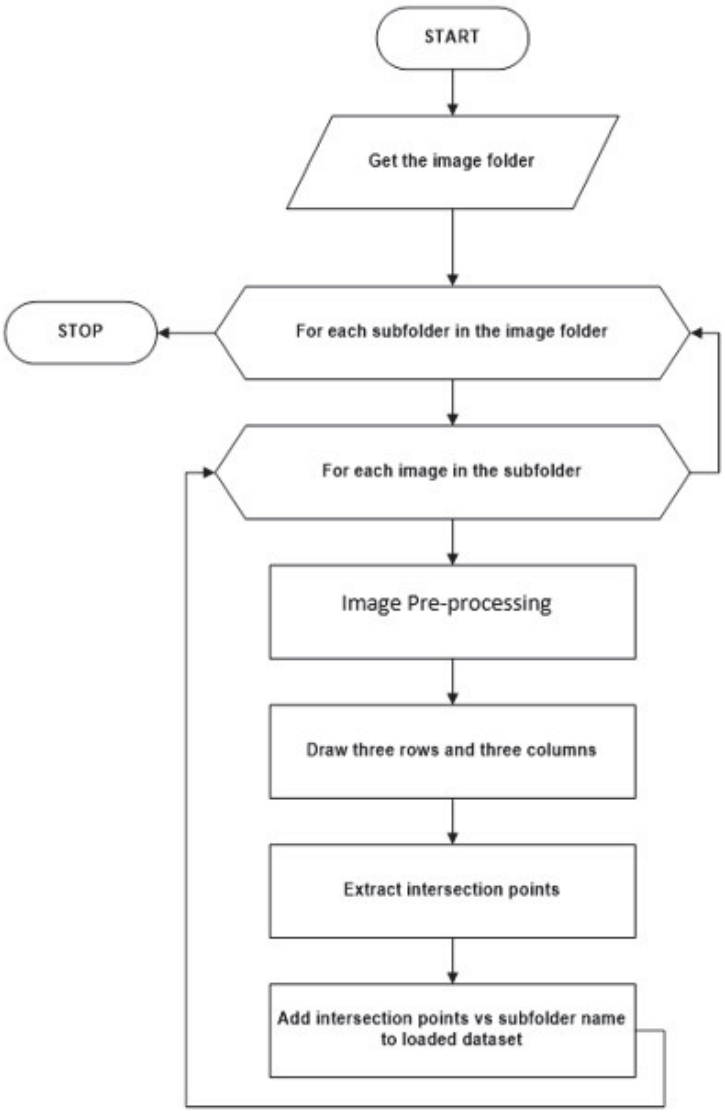


Figure 10 Creating Reference Database

So now the image matrix is form of 0 or 1 at pixel value. Then the traffic sign image is represented in MATLAB by a matrix with M rows and N columns pixels, where rows with index 1, M/2 and M are chosen to represent the horizontal lines and columns with index 1, N/2 and N are chosen to represent vertical lines. When the pixel value of 0 or 1 at the intersection of these lines alone are kept as features. The coordinates of intersection points of these horizontal lines and vertical lines with the detected edges of the traffic signs are chosen as the features for traffic sign identification. In figure 9 two examples of the intersection points (IP) between the edges detected in the traffic sign image and drawn horizontal and vertical lines in that image are shown. The figure below shows three horizontal and three vertical lines in the grid area which we take as intersection pattern. Then we extract intersection points from the intersection of traffic sign image three horizontal and vertical lines as shown in figure 11.

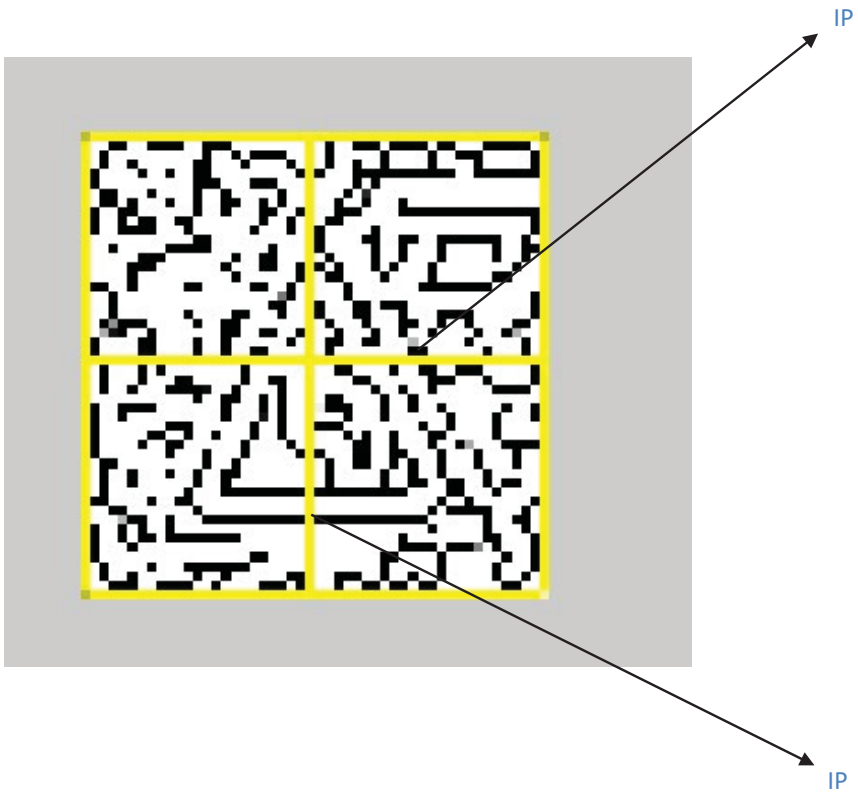


Figure 11 Intersection Pattern

5.3 Feature Extraction and Matching

The image is pre-processed and the intersection of detected edges of the traffic sign in the image with three horizontal and three vertical lines are the features. The intersection is calculated in terms of vector with values of 1 and 0. Where the edges in a traffic image intersect the line, the value is set as 1 and where there is no intersection, value is set as 0. Between the features of two images, the distance is calculated according to Eq.(5) as number of matching intersections between the detected edges of the traffic sign in the image and three horizontal and three vertical lines as described previously. Figure 12 illustrates the procedure of how a traffic sign image is classified.

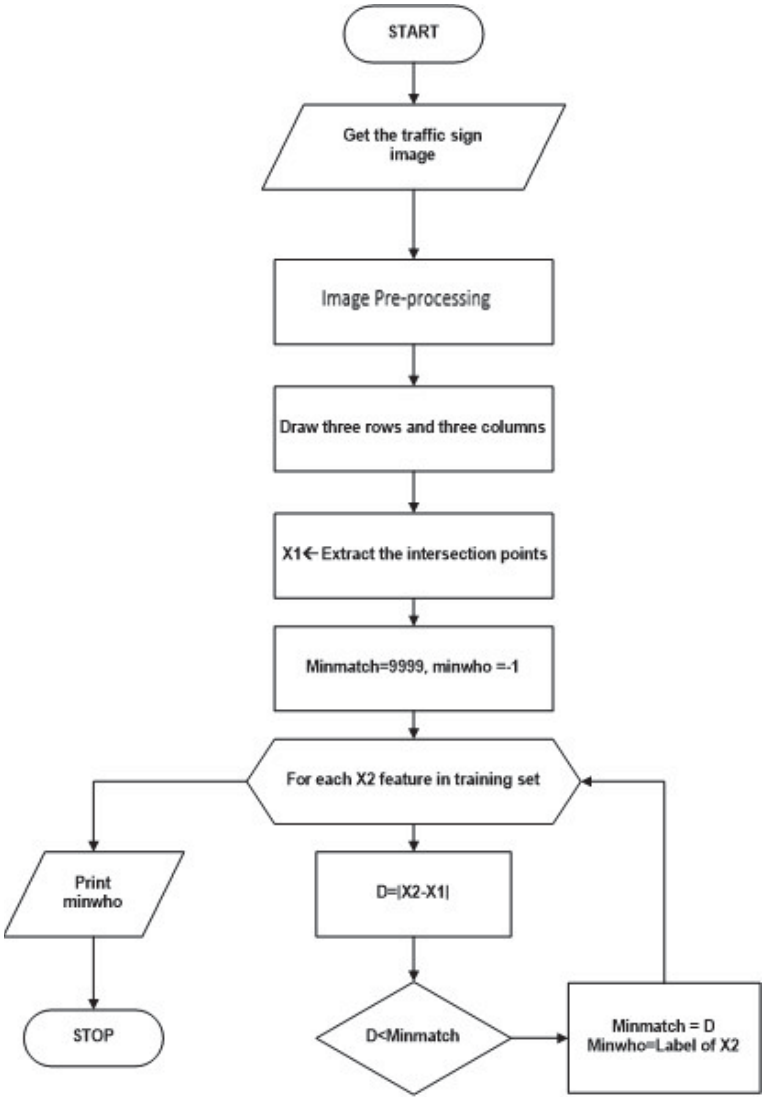


Figure 12 Classification

Firstly, we choose a traffic sign image and do pre-processing and extract the intersection points which we take as features, then we match those features with the features of the images in the repository. The label of the image in the repository to which the features of test image is having maximal match is given as the output label. Therefore, the class of the traffic sign in the test image is chosen based on the closest match to the reference image.

5.4 GUI

The following snapshots show the outputs that we will get after step-by-step execution of all the modules of the system.

Interpretation: The GUI of the project is developed using guide tool of MATLAB. Event handling model is followed. For each button in the foreground, there is a background function called which implements the functionality. When the traffic sign classification is started the user GUI, shown in figure 13, is presented to a user. Load button is handled by loading the labelled image from the folder and saving it. Browse image is handled by displaying a file selection dialog box to allow choosing an image to classify. Classify button is handled by implementing the functionality of matching the given image images to stored image features and finding the best match. Do Performance is handled by implementing the functionality to measure the performance of the proposed classification algorithm.

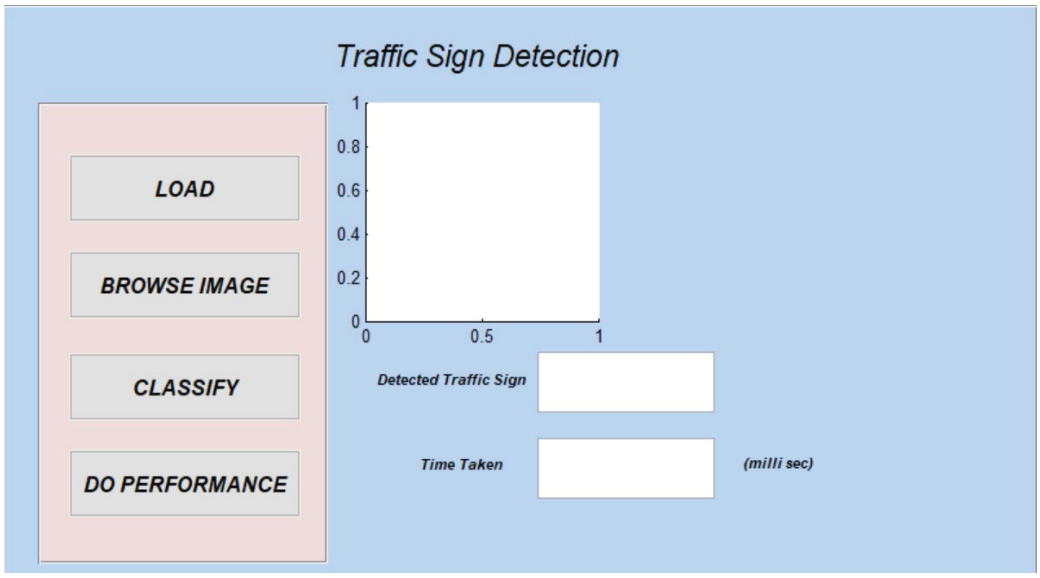


Figure 13 Graphical User Interface

Once load button is pressed, each image in the repository folder is read and number of images are stored into a MATLAB variable. In figure 14 we can see a dialog box is displayed when loading is completed.

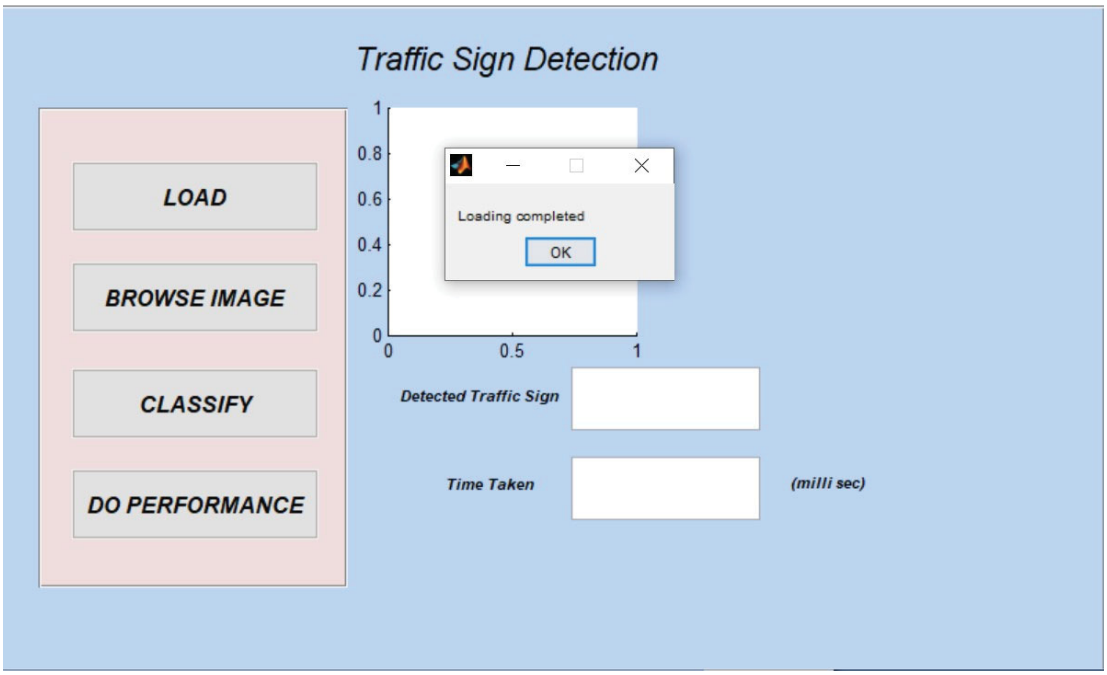


Figure 14 Loading of Image Completed

Figure 15 Browse Image to Classify

As we can see in figure 15, user can browse and choose any traffic sign image to be classified. Once after image is selected, classify button is pressed.

Figure 16 explains how pre-processing of the image is done, and how the features are extracted from the both the given image and the images stored in the repository. The given image after each step of pre-processing is shown figure 16. The image is enhanced and then thinned applying morphological operations.

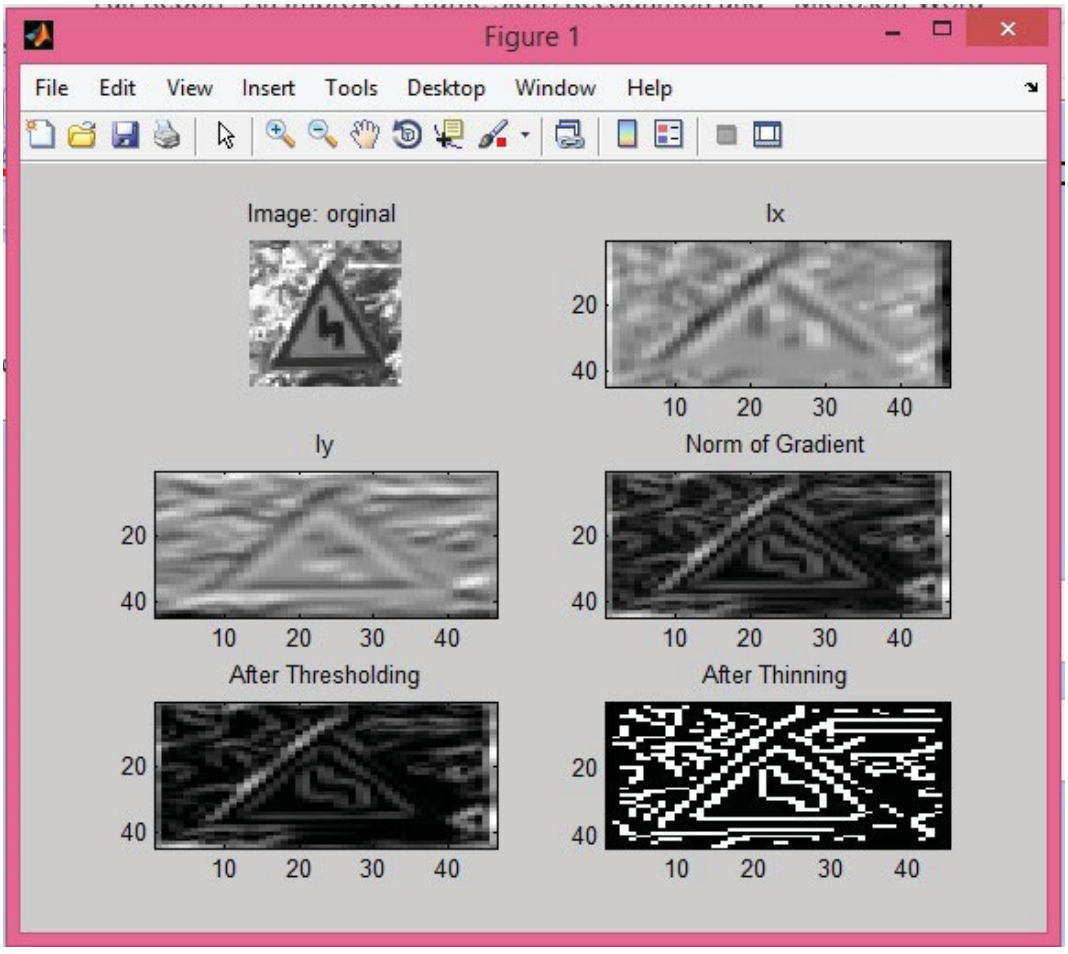


Figure 16 Pre-processing Results

Figure 17 Classification Results

Once the feature is extracted, matching is done against features of each of the labelled images in the folder to find the best match. The classified label and the time taken for classification is shown in the figure 17 of the GUI.

Chapter 6 Discussion

In this thesis, we have developed a GUI where we can perform traffic sign detection, classification, and measure performance. We used 72 test images for performance testing. For each test image in the test folder and their subdirectories, we have used 38 reference images from the load folder for classifying the traffic sign correctly. From the command window in MATLAB, we can see results of accuracy.

Total number of images used for performance testing= 72

Number of images classified correctly= 52

Accuracy of classification

= (Number of images classified correctly/ Total number of images used for performance testing)*100% = 72%

Number of images classified wrongly= 20

Example of wrong classified image:

Figure 18 Wrongly Detected Image

Figure 18 is from class 38, but when we try to classify it, it is classified as 21, it means it is classified wrongly. The reasons for wrong classification could be as following. Incorrect conversion from colour image to grayscale or to binary image, wrong thresholds for edge detection, or some other traffic sign image having close matching intersection on three horizontal and three vertical lines. In this thesis we can detect and classify the traffic sign achieved 72%.

6.1 Comparison of Matching Distances

The matching distance of each of the 72 image is given in figure 19.

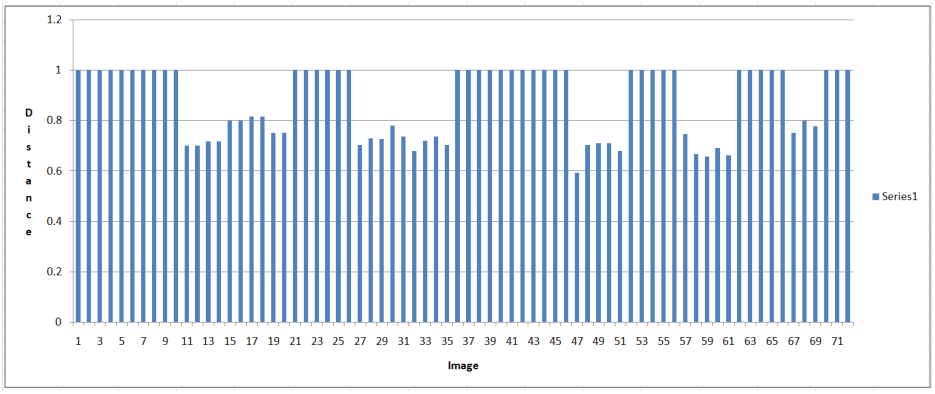


Figure 19 Matching Distance of Every Image

The matching distance to correctly classified image is given in figure 20.

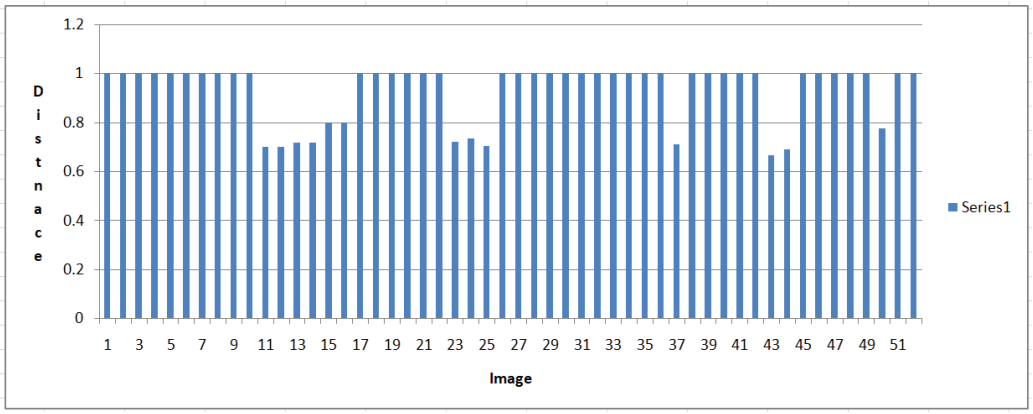


Figure 20 Correctly Classified Images

The matching distance to incorrectly classified image is given in figure 21.

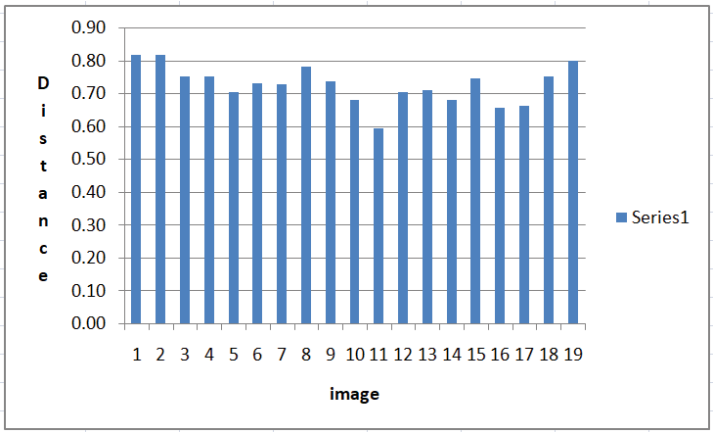


Figure 21 Incorrectly Classified Images

Figure 19 to 21 shows the normalized Euclidean distances between feature vectors of correctly and incorrectly classified traffic sign images. In figure 20 we see that more than 50% of correctly classified images has the distance equal to one. On the other hand, the Euclidean distances of correctly classified images are in the range from 0.6 to 1. The distances for incorrectly classified images are in the range from 0.6 to 0.8 as we see it in figure 21. So in the developed solution we cannot classify correctly all traffic sign images using only a threshold for Euclidean distances. Because the range of the distances for correctly and incorrectly classified images has a 50% overlap. So we need to consider using more features or some other criteria to classified images.

6.2 Comparison to Existing Works

The most recent trend in Traffic sign recognition is using deep learning models. Deep learning models with CNN can achieve about 98% accuracy. But there are following problems in deep learning models.

1. The computational complexity is high and requires costly resources.
2. Cannot be parallelized.
3. Classification time is high.

In comparison to deep learning models, the proposed solution in this work has low accuracy but the proposed solution has following advantages.

1. The computational complexity is very low, can be implemented in resource constrained devices.
2. The algorithm can be parallelized.
3. The classification time is less, as involves only simple distance computation operation.

Even though the accuracy is only 72% in the proposed solution, it can be improved by creating a robust mesh for extracting the features. Through this way, accuracy can be improved. A promising part in the proposed solution is its ability to implementation in parallel processing mode with low computational need.

Chapter 7 Conclusion

7.1 Conclusion

In this project, we introduced a new method for recognition and tracking of traffic signs dedicated for an automatic traffic assistance system. The proposed system is based on intersection of traffic sign on a known grid pattern. It is simple and easy to implement with low computational complexity. The proposed system was able to achieve more than 70% accuracy and able to detect traffic sign between 30 to 40 milliseconds. In future, we can achieve more accuracy by adding more no of images or by increasing the size of the database. We can have more classes. In this thesis, generally we created our own DB file by taking the images from German Traffic Sign (GTS), but in future we can apply this method directly to large datasets like the German Traffic Sign (GTS) dataset and the Belgium Traffic Sign (BTS) dataset to achieve faster time and greater accuracy.

References

[1] A.Hechri and A. Mtibaa, "Automatic detection and recognition of road sign for driver assistance system," *2012 16th IEEE Mediterranean Electrotechnical Conference*, YasmineHammamet, 2012, pp. 888-891, doi: 10.1109/MELCON.2012.6196571.

[2] V. A. Prisacariu, R. Timofte, K. Zimmermann, I. Reid and L. V. Gool, "Integrating Object Detection with 3D Tracking Towards a Better Driver Assistance System," 2010 20th International Conference on Pattern Recognition, Istanbul, 2010, pp. 3344-3347, doi: 10.1109/ICPR.2010.816.

[3] L. Yang, K. Kwon, K. Moon, S. Lee and S. Kwon, "Broken traffic sign recognition based on local histogram matching," 2012 Computing, Communications and Applications Conference, Hong Kong, 2012, pp. 415-419, doi: 10.1109/ComComAp.2012.6154884.

[4] S. Lafuente-Arroyo, P. Gil-Jimenez, R. Maldonado-Bascon, F. Lopez-Ferreras and S. Maldonado-Bascon, "Traffic sign shape classification evaluation I: SVM using distance to borders," IEEE Proceedings. Intelligent Vehicles Symposium, 2005., Las Vegas, NV, USA, 2005, pp. 557-562, doi: 10.1109/IVS.2005.1505162

[5] Yuan Xie, Li-Feng Liu, Cui-Hua Li and Yan-Yun Qu, "Unifying visual saliency with HOG feature learning for traffic sign detection," 2009 IEEE Intelligent Vehicles Symposium, Xi'an, 2009, pp. 24-29, doi: 10.1109/IVS.2009.5164247.

[6] J. F. Khan, R. R. Adhami and S. M. A. Bhuiyan, "Image segmentation based road sign detection," IEEE Southeastcon 2009, Atlanta, GA, 2009, pp. 24-29, doi: 10.1109/SECON.2009.5174040.

[7] Y. Huang, Y. Le and F. Cheng, "A Method of Detecting and Recognizing Speed-limit Signs," 2012 Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Piraeus, 2012, pp. 371-374, doi: 10.1109/IIH-MSP.2012.96.

[8] J. Greenhalgh and M. Mirmehdi, "Traffic sign recognition using MSER and Random Forests," 2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO), Bucharest, 2012, pp. 1935-1939.

[9] Stephen M. Pizer, E. Philip Amburn, John D. Austin, Robert Cromartie, Ari Geselowitz, Trey Greer, Bart terHaarRomeny, John B. Zimmerman, KarelZuiderveld, Adaptive histogram equalization and its variations, Computer Vision, Graphics, and Image Processing, Volume 39, Issue 3, 1987,

[10] Abhishek&LakshmeshaK.N,"THINNING APPROACH IN DIGITAL IMAGEPROCESSING",International Journal of Latest Trends in Engineeringand Technology,Special Issue SACAIM 2017, pp. 326-330

[11] “German traffic sign recognition benchmark.” https://benchmark.ini.rub.de/ (accessed Mar. 10, 2021).