Road Sign Recognition based on Invariant Features using Support Vector Machine

Syed Hassan Gilani

Master Thesis
No. E3465D

2007

DEGREE PROJECT
Computer Engineering

Programme **Master in Computer Engineering**	Reg number **E 3465D**	Extent **30 ECTS**
Name of student **Syed Hassan Gilani**	Year-Month-Day **2007-02-14**	
Supervisor **Hasan Fleyeh**	Examiner **Mark Dougherty**	
Company/Department **Faculty of Electrical and Computer Engineering**	Supervisor at the Company/Department **Hasan Fleyeh**	
Title **Road Sign Recognition based on Invariant Features Using Support Vector Machine**		
Keywords **Speed-limit Recognition, Shape Recognition, Support Vector Machines, Kernel Functions, Invariant Features, Feature space.**		

Abstract

Since last two decades researches have been working on developing systems that can assists drivers in the best way possible and make driving safe. Computer vision has played a crucial part in design of these systems. With the introduction of vision techniques various autonomous and robust real-time traffic automation systems have been designed such as Traffic monitoring, Traffic related parameter estimation and intelligent vehicles. Among these automatic detection and recognition of road signs has became an interesting research topic. The system can assist drivers about signs they don't recognize before passing them.

Aim of this research project is to present an Intelligent Road Sign Recognition System based on state-of-the-art technique, the Support Vector Machine. The project is an extension to the work done at ITS research Platform at Dalarna University [25]. Focus of this research work is on the recognition of road signs under analysis. When classifying an image its location, size and orientation in the image plane are its irrelevant features and one way to get rid of this ambiguity is to extract those features which are invariant under the above mentioned transformation. These invariant features are then used in Support Vector Machine for classification. Support Vector Machine is a supervised learning machine that solves problem in higher dimension with the help of Kernel functions and is best know for classification problems.

DEDICATION

I dedicate this endeavour to my parents and to my brothers, who have been the constant driving force in my pursuit of knowledge.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Mr. Hasan Fleyeh. The decision he made in Fall, last year, to take me as one of his graduate assistants gave me the opportunity to do the research I am interested in. I am grateful to his support and advice ever since then. He creates a wonderful and dynamic environment for me to learn and gives me the freedom to explore the interesting problems in field of Computer Vision and Digital Image Processing.

I would also like to thank Professor Mark Dougherty, Dr. Pascal Rebreyend and Dr. Ernst Nordstrom for kindly serving on the committee and reviewing the report.

Finally, I would like to express my gratitude to my parents and my brothers, specially my elder brother (Syed Omer Gilani) for their love and support anytime, anywhere on demand or without demand at all.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLE

CHAPTER ONE

Introduction

Background

In an environment with all kinds of traffic, Road signs play an important role in regulating the traffic and warns the driver to prohibit certain actions for there safety and for the safety of there passengers.

Road signs use colours, shapes, and markings to communicate message to on road drivers. Without such information the motion of traffic would be disorderly and unpredictable. It's very crucial for drivers to identify road signs at right time, at right place but at times when everything is expected to be perfect, from others off course, we tend to forget the inherent imperfection of mankind [1, 2]. Noticing these safety precaution signs on the road greatly depends on the physical and mental health of the drivers. There visual perception ability can be affected by stress, tension and physical illness [3] and some times it's the lack of knowledge about road signs. According to a recent poll conducted by motoring website, **New Car Net**, one in three motorists fail to recognize even the most basic Road Signs [4].

It's because of these reasons an autonomous robust real time road sign recognition system has gained interest since last two decades. The very first paper appeared in 1984 which aimed on testing various *computer vision* methods for detection of objects in outdoor scenes. Since then many research groups and companies have been interested and have conducted research in the field [3, 5].

Computer vision has been applied to a wide variety of intelligent transport systems (ITS) such as traffic monitoring system, traffic related parameter estimation and intelligent vehicles, and an important part of intelligent vehicles is the detection and recognition of Road signs [1, 6]. A robust real time and automatic road sign detection and recognition system can really support and disburden drivers by giving information at good time; it can increase driving efficiency, save lives and can provide driving comfort.

1.1 Types of Road Signs:

There are two basic categories of Road Signs:

- The ones that give precautionary information like prohibitory and warning signs used for, speed limit, no-entry, men at work, children crossing etc. as shown in figure 1.1(a) [7]

- Other with general information like destinations signs and services, as shown in figure 1.1(b) [7]

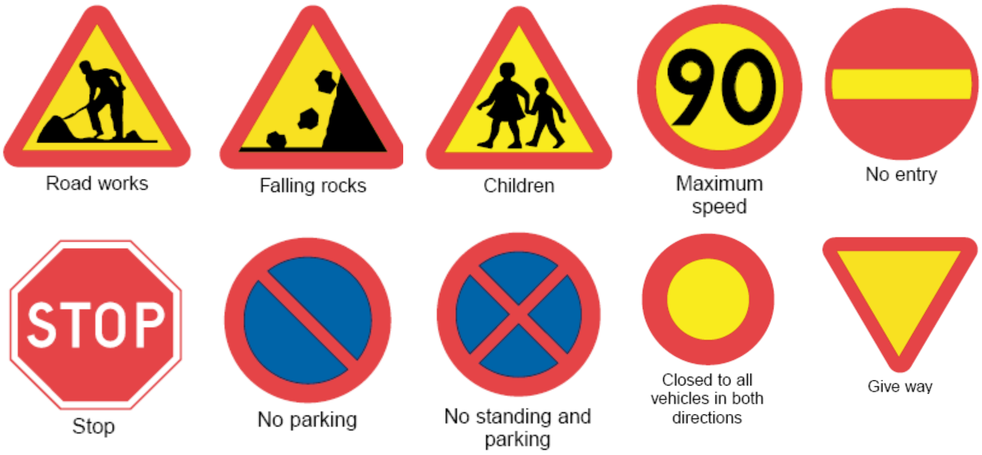


Figure 1.1(a): Prohibitory and Warning signs [8]

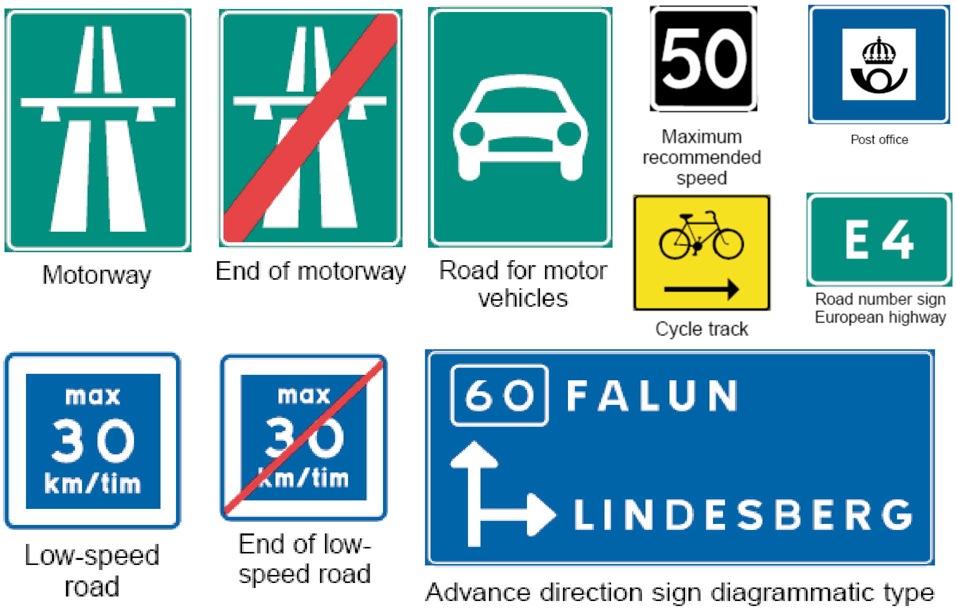


Figure 1.1(b): Destination and Services signs [8]

Road signs are designed and positioned in a way to make it easy for the drivers to detect them and understand them. Colours and shapes can also provide effective information to drivers when the words or symbols on the sign are not clearly understandable [9]. Several different design styles and colours are used to convey the desired information to the driver as simply as possible. Table 1.1 and 1.2 [10] illustrate the meaning of different shapes and colours used in road signs.

Table 1.1: Shape Description [10]

Shape	Description
Circle	Prohibitory
Octagon	Stop
Diamond	Construction and maintenance
Rectangle	Regulation and guidance
Equilateral triangle, pointing up	Warning
Equilateral triangle, pointing down	Yield

Table 1.2: Colour Description [10]

Colour	Description
Red	Prohibitory and warning
Blue	Directive
Green	Guidance and mileage
Orange	Construction and maintenance
Yellow	Warning
Black	Information
White	Auxiliary

1.2 Previous Work

Several research projects have been conducted in the past for implementation of safety and efficiency on roads such as ***Intelligent Vehicle and Highway System (IVHS)*** and ***PROgraM*** for European traffic with highest efficiency and unprecedented safety.

The very first known attempt for making real time system was by Akatuska and Imai, in 1987. Since then many techniques have been proposed. In recent years, the performance of many object detection systems has received a boost by "**Viola Jones**" detector, work of *Paul Viola and Michael J. Jones (Cambridge Research Laboratory in 2001).* It's an approach that discriminates object from non-object image patches with help of machine learning techniques. Its main idea is to generate an over-complete set of (up to 100000) efficiently computable Haar wavelet features, combine them with simple threshold cascaded classifiers, and utilize AdaBoost [11] to select and weight the most discriminative subset of wavelet features and threshold classifiers. The framework was demonstrated on the task of face detection and has shown ability to solve many practical problems in real-time [1, 12].

Aryuanto and Koichi proposed a new technique for fast and robust detection of traffic signs using *Geometric Fragmentation*. The focus was on detection of red circular traffic signs. The idea was to detect the outer ellipse of the sign by combining the left and right fragments and then using geometric fragmentation to find the ellipse fragments [13].

Traffic sign recognition using Back-Propagation neural network showed success rate, generally, between 60% and 70%. Based on the results it was concluded that colour distortion caused by colour complexity is the key factor affecting the recognition rate. Zhu and Jiang presented a new solution to this problem by breaking the complex colour information to 5 standard colours and then using the same BP neural network for recognition [14].

Zhu, Zhany, and Lu used a method based on colour and shape for the detection of traffic signs in four steps. They used red colour down triangle shape sign to explain the method. First RGB colour space was converted to HIS colour space and pixels with red colour were extracted. Then LOG mask operation was performed on extracted pixels, for edge detection. Then they used neural network to determine the specific angle the pixels, and finally using shape information for detection of traffic sign [15].

And nevertheless, the success of last years DARPA (Defense Advance Research Project Agency) Grand Challenge [16] showed that computer vision can be used effectively in solving many problems regarding autonomous driving system. However they also showed some concerning issues such as, the system used in autonomous vehicles participating the challenge for processing the data were far superior then a normal PC and practically it would be impossible for majority of people to have such system installed in there vehicles.

1.3 Potential Challenges

Traffic sign recognition is a difficult task if aimed at detecting and recognizing sign image captured with unfavourable background. Complex backgrounds, weather conditions, lighting and shadows make the task more complicated and difficult. An intelligent transport system (ITS) has following concerns [3, 14, 17-19]:

- Damaged or disoriented signs making it hard for the system to detect and recognize.
- Poor visibility because of bad weather and lighting conditions.
- Images acquired are usually blurred because of car vibration and speed.
- Positing of traffic sign is also very important for the system to detect the sign. Signs placed near trees often have portions hided by tree branches.
- Colour fading because of constant exposure to sunlight.
- Presence of objects in the background with likely shape and colour.

.

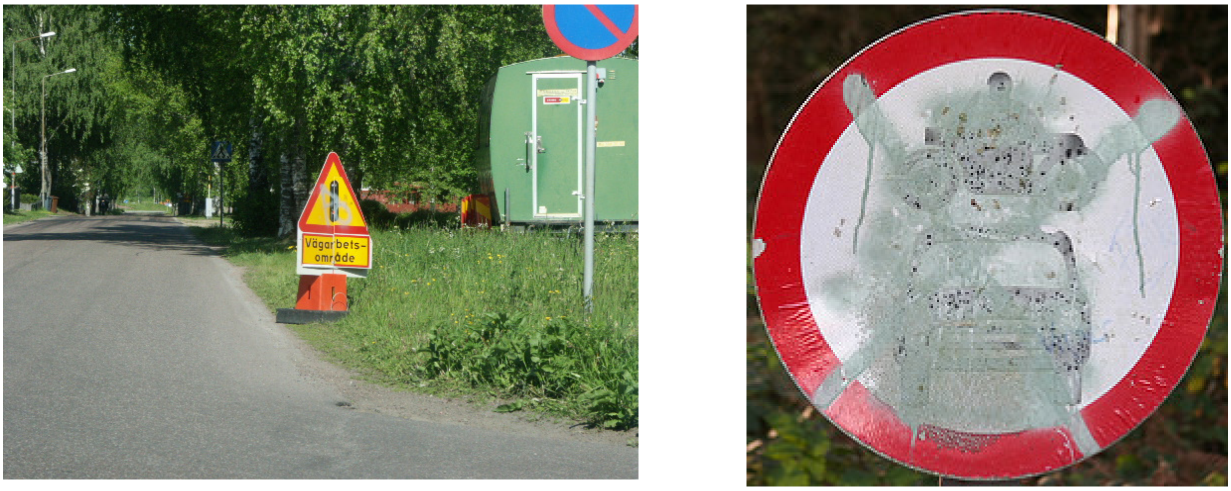


Figure 1.2: Damaged Signs [20, 21]

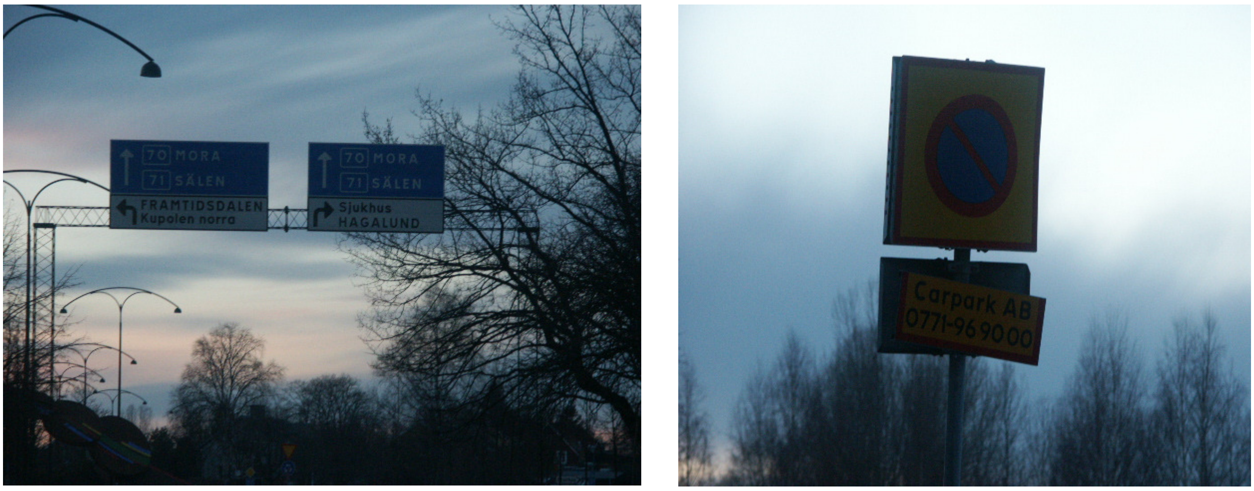


Figure 1.3: Bad Lighting Condition [20]

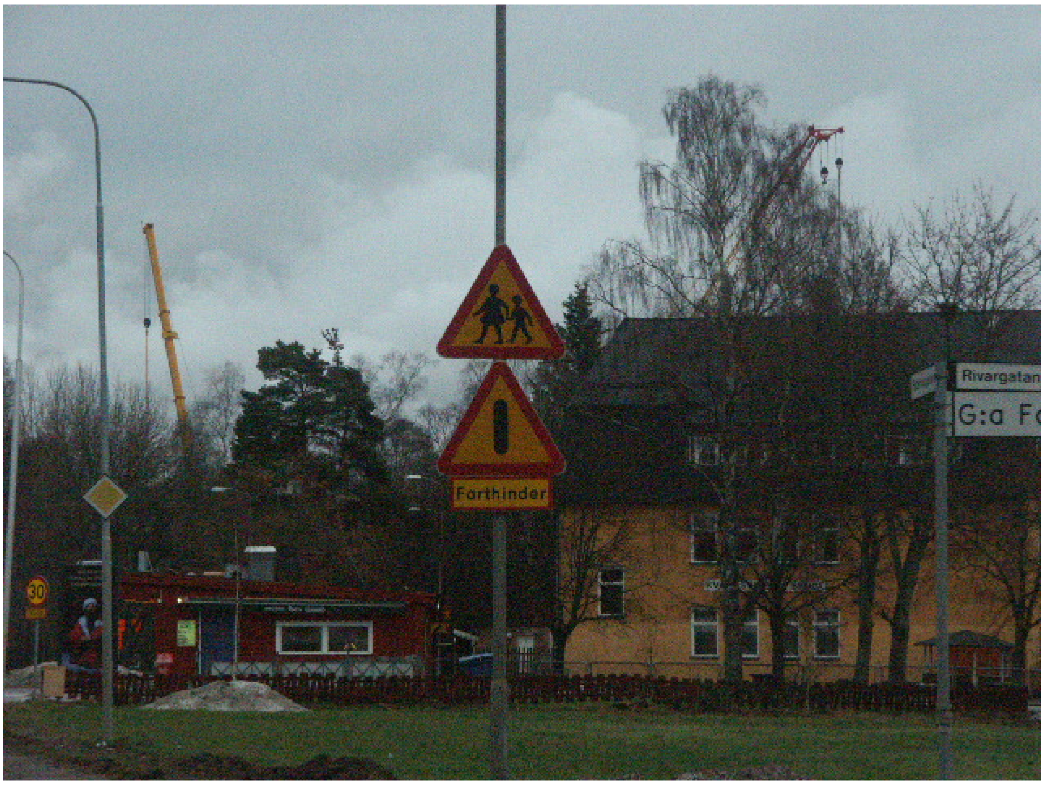


Figure 1.4: Bad Weather Condition [20]

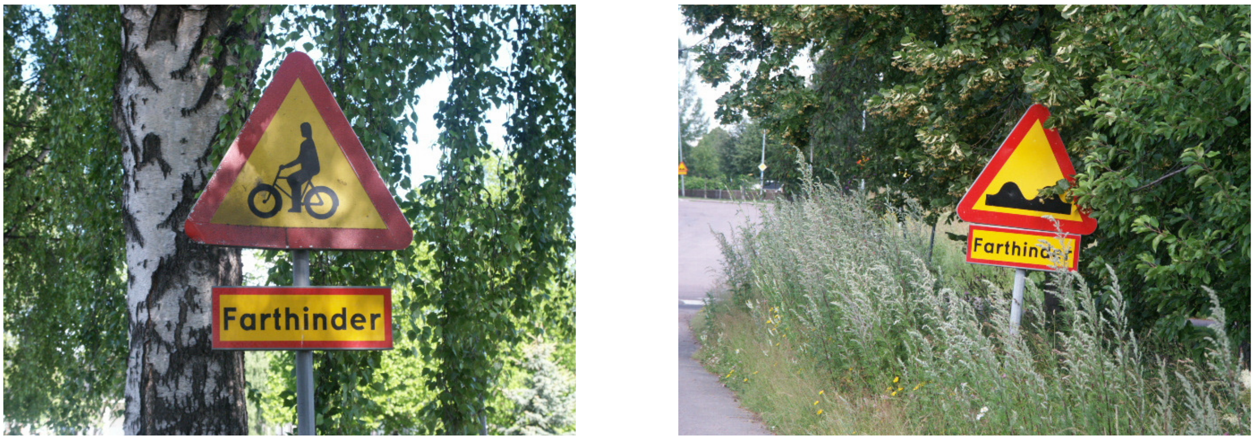


Figure 1.5: Bad Sign Positions [20]

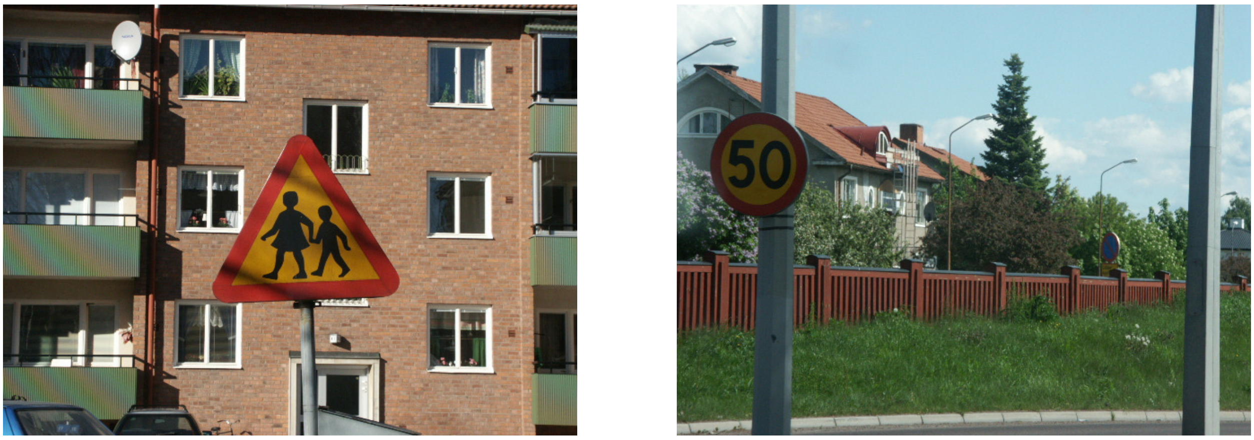


Figure 1.6: Signs with background having same color [20]

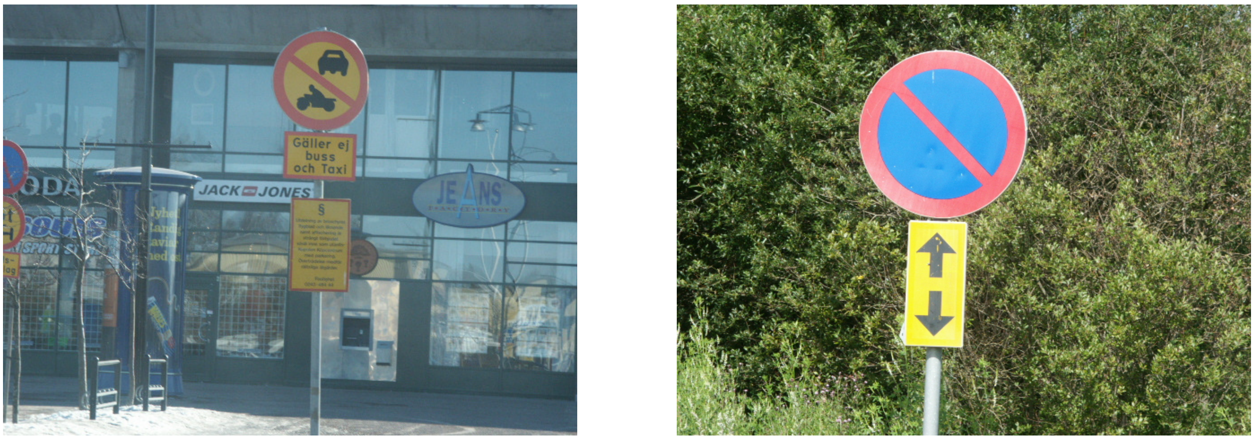


Figure 1.7: Faded Traffic Signs [20]

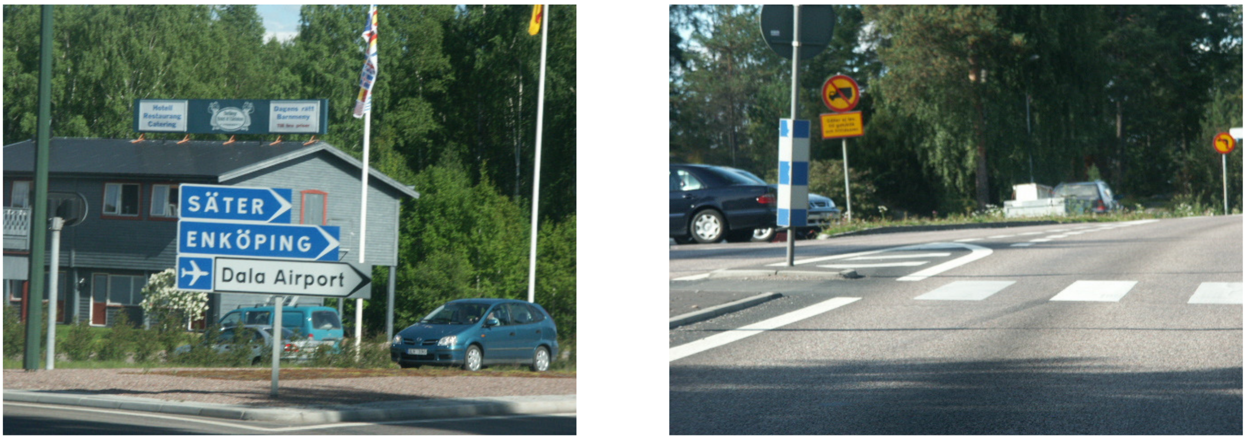


Figure 1.8: Blur Images caused by car vibration and motion [20]

1.4 Application of Road Sign Recognition System

The Road Sign Recognition is a field of applied computer vision research concerned with the automatic detection and classification of road signs in traffic scene images acquired from a moving car. The result of this research effort will be the subsystem of Driver Support System (DSS). The aim is to provide DSS with the ability to understand its neighbourhood environment and so permit advanced driver support such as collision prediction and avoidance.

Employing computer vision technology in smart vehicle design calls for consideration of all its advantages and disadvantages. Firstly, vision subsystem incorporated into the DSS may exploit all the information processed by human drivers without any requirements for new traffic infrastructure devices (a very hard and expensive task). Smart cars equipped with vision based systems will be able to adapt themselves *to* operate in different countries (with often quite dissimilar traffic devices).

As the integration of various technologies in the field of traffic engineering has been introduced (ITS) the convenience of computer vision usage has become more obvious. We may observe this trend e.g. in proceedings of annual IEEE International Conference on Intelligent Vehicles (IVS). More then 50% of papers are focused on Image Processing and Computer Vision methods [22].

Obviously, there exist even disadvantages of the vision-based approach. Smart vehicles will operate in real traffic conditions on the road. So, the algorithms must be robust enough to give good results even under adverse illumination and weather conditions. Although this system property may seem to be solved easily it is the real challenge for the algorithm developers. For example Fridtjof Stein, main project manager of Cleopatra project (Clusters of embedded parallel time-critical applications) said [23] that "*reliable optical detection is the biggest hurdle the project must overcome*".

There cannot be assured absolute system reliability and the system will not be "fail-safe" because of the definition of individual transportation system. The aim is to provide a level of safety similar to or higher than that of human drivers. For example it could assist drivers about signs they did not recognize before passing them. Specifically, speed limit sign recognition could provide driver the present speed limit as well as giving an alert if a car is driven faster than the speed limit.

In future, autonomous vehicles would have to be controlled by automatic road sign recognition. As with any vehicle, an autonomous vehicle driving on public roads must obey the rules of the road. Many of these rules are conveyed through the use of the road signs, so an autonomous vehicle must be able to detect and recognize sings and change its behaviour accordingly [24].

1.5 Aim

Aim of this research project is to present an Intelligent Road Sign Recognition System based on state-of-the-art technique, the Support Vector Machine. The project is an extension to the work done here at ITS research Platform at Dalarna University [25]. In this research work our focus is on extracting the Invariant features from the Road signs. These Invariant Features are then feed to Support Vector Machine as input which performs the categorization based on the similarity measure between the features.

The signs used in experiments are the Five Speed Limit Signs:

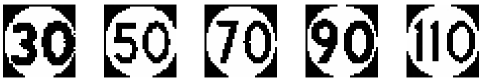


Figure 1.9: Speed Limit Signs

and the Seven Shape Signs:

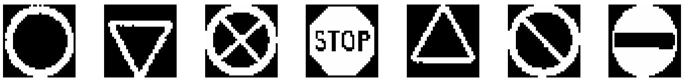


Figure 1.10: Different Shape Signs

already segmented and normalized.

Safety is the primary factor before making a commercial application of road sign detection and recognition system. The system should be able to identify the Region of Interest (ROI) and most importantly should to able to recognize the information in real time. There is no room for false predictions, if the system is unable to recognize the sign it should refrain from indicating the information rather than indicating wrong information.

This highlights the main concerns of our Road sign recognition system which in turn will be the performance criterion for our developed system.

- System's Response in Real Time.
- System's Reliability.

1.5.1 Use of Support Vector Machine

Support Vector Machine is infact the brain of our system. It translates the information provided by the road sign image and can help in performing decision making actions. The classification task is performed by mapping the data set nonlinearly to a higher dimensional space and then using a linear hyperplane for data separation.

Support Vector Machines models are close cousin to classical neural network with a very specific class of algorithm, characterized by the use of kernel and with the absence of local minima they deliver state-of-the-art performance in real world application like Text categorization, Hand written character recognition, **Image Classification** and Bioinformatics. An important part of SVM model is the mapping of input space data to a higher dimension. This operation is performed with the help of a very powerful technique, called the Kernel trick. The Kernel helps in mapping the data from input space, nonlinearly, to a higher dimensional feature space where it's easy to categorize the data using linear hyperplane.

Bernhard Schölkopf, in an introductory overview, points out that a particular advantage of SVMs over other learning algorithms is that it can be analyzed theoretically using concepts from computational learning theory, and at the same time can achieve good performance when applied to real problems [26].

Since its first inception in COLT-92, SVM has been used to solve several classification and regression problems. Several experiments have been performed to compare the performance of SVM with other types of neural networks. SVM has been used for crop classification using hyperspectral images, benchmarked to well known neural networks such as multilayer perceptrons (MLP), Radial basis function (RBF) and Co-Active Neural Fuzzy Inference System (CANFIS). The conclusions of the experiments showed that SVM yield better results than neural networks. It was unfeasible to train a neural model while working in high dimensional input space as compared to SVM which deals the problem in higher dimensional space [27].

As compared to every other model SVM also has some strengths and weakness. In the absence of local optimal training SVM is relatively easy compared to neural networks. SVM can scale well to higher dimensional data. The tradeoffs between classifier complexity and error can be controlled explicitly. Non traditional data like trees and strings and can use then as input to SVM, instead of feature vectors. However, there is one very important factor which is very crucial to SVMs performance, the Kernel Function. Selection of suitable kernel function for a problem can improve the optimal performance of the vector machine.

1.5.2 Approach

Our focused is on extracting those features in an image which are invariant to translation and rotation. The methods used are very effective and robust in calculating those invariant features. These invariant features are then feed to SVM (Support Vector Machine) as an input which performs the classification task very efficiently.

1.6 Detection and Recognition System

An intelligent Road Sign recognition system consists of Four basic stages having subparts. These basic stages are:

- Image Acquisition.
- Image Pre-processing.
- Image Normalization.
- Image Recognition.

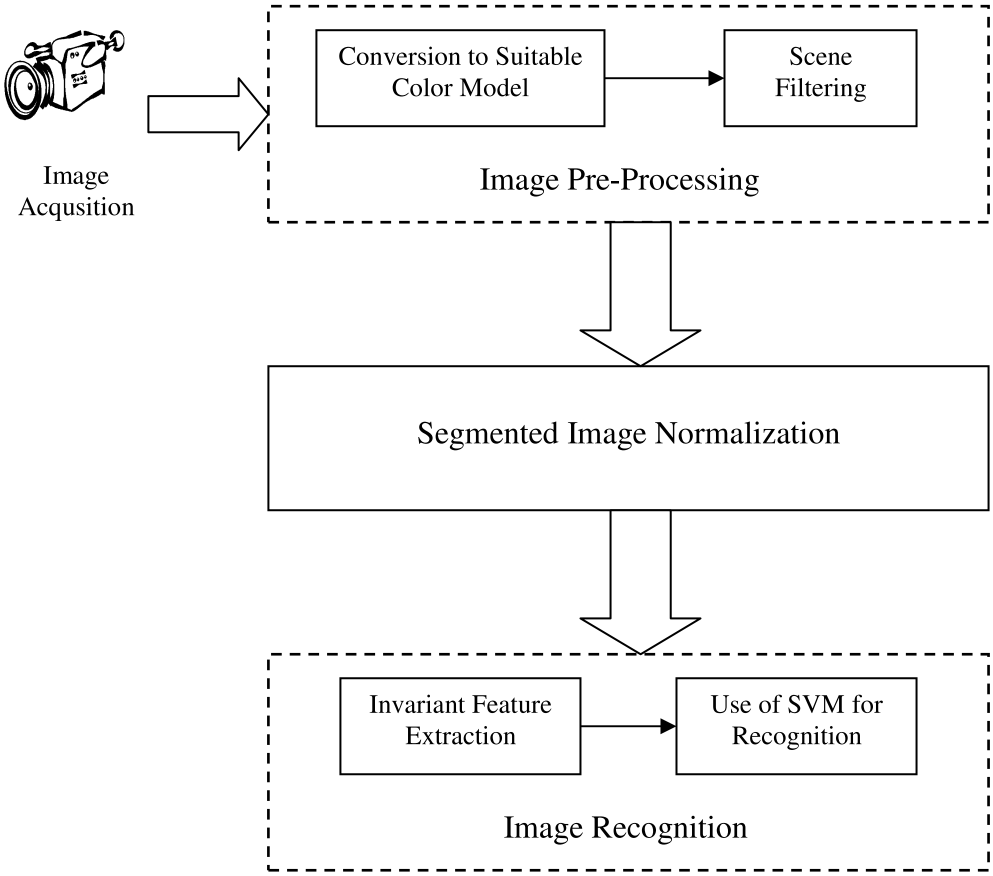
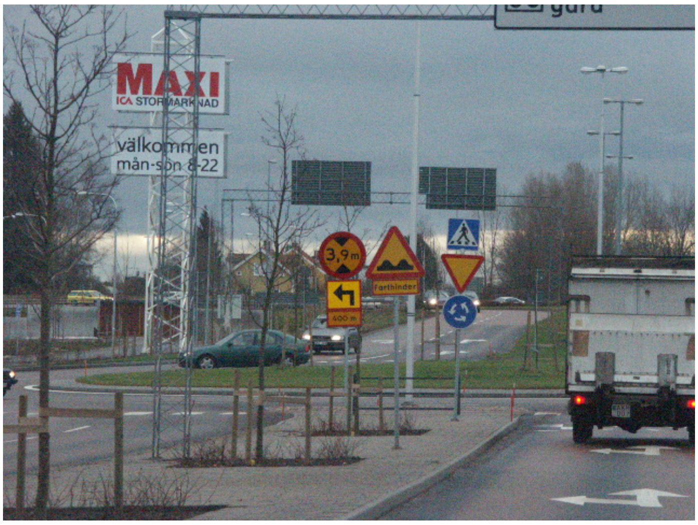


Image Acqusition
Conversion to Suitable Color Model
Scene Filtering
Image Pre-Processing
Segmented Image Normalization
Invariant Feature Extraction
Use of SVM for Recognition
Image Recognition

Figure 1.11: Block Diagram of a Road Sign Detection system

1.6.1 Image Acquisition

An RGB Image is captured using a digital camera mounted on a car deck. The dimension of captured image is 256x256 pixels. The Pictures taken in outdoor environment are usually much cluttered. Figure 1.12 shows such an example. So Image Pre-processing is necessary to simplify the image before further processing.



.

Figure 1.12: Original Digital Image [10]

1.6.2 Image Pre-Processing

1.6.2.1 Conversion to Suitable Colour Model

RGB is the best-known and widely used colour space. However it is greatly sensitive to chromatic variation of the daylight. The coordinate of three colours are highly correlated, and as a result of this any variation in the ambient light intensity affect the RGB system by shifting the cluster of colours towards the white or the black corners (Figure 1.13). As result of this it will be hard to recognize the object under consideration under different brightness condition

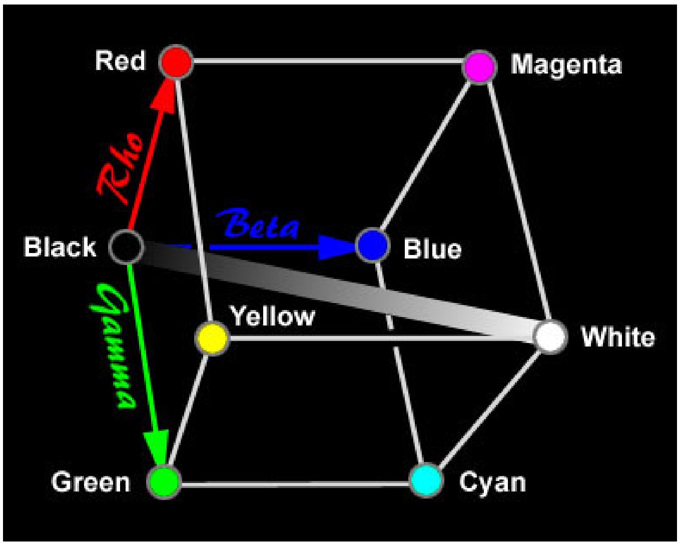


Figure 1.13: RGB Colour Cube [10]

To avoid this, a necessary colour conversion has to be done. HSV was the ideal colour model for our Speed limit recognition problem. Since it decouples the chromatic and achromatic notion of light. Every Colour in this space is represented by three components: the Hue (H): the apparent colour of the light (determined by dominant wavelength), the saturation (S) : the purity of light, and the value (V): the total light across all frequencies (Figure 1.14).

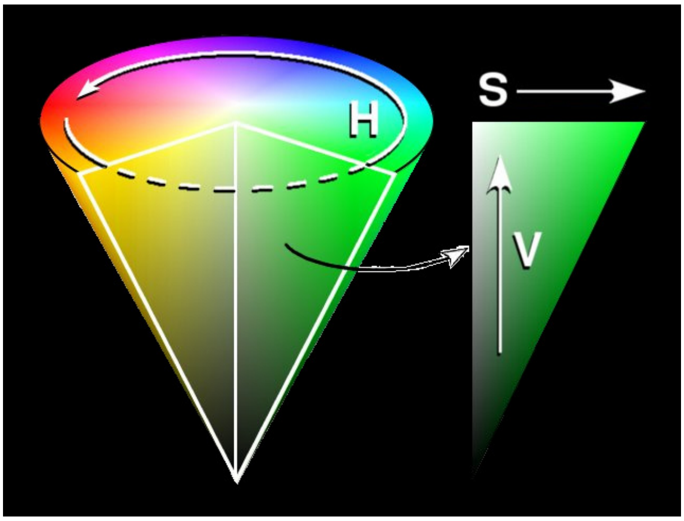


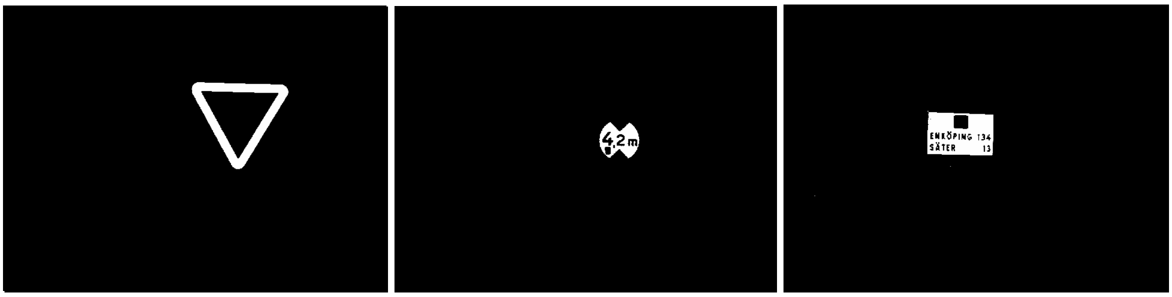
Figure 1.14: HSV Colour Cone [10]

It is very important and attractive colour space for the application of image processing specially road sign recognition because it represents the colours in a similar way by which human eye senses the colour.

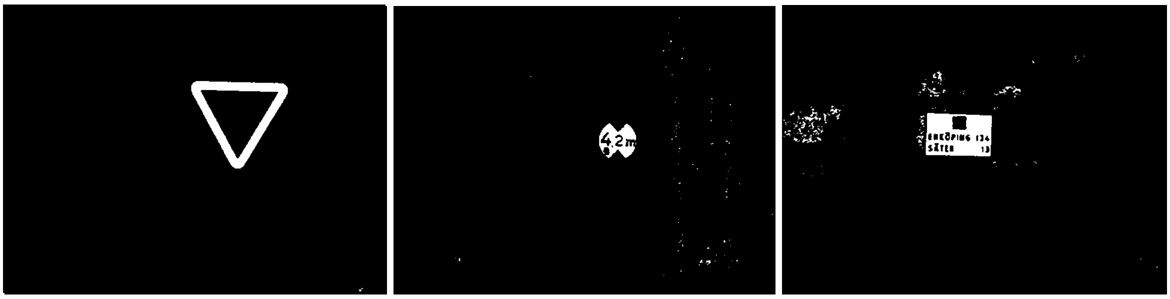
1.6.2.2 Scene Filtering

After conversion of image in to suitable colour model for processing, segmentation process is carried out to identify the possible regions of interest (ROIs). Several techniques such as Dynamic-threshold, seeded region growing and Minimum-Maximum method were applied for filtering of ROI from the noisy environment.

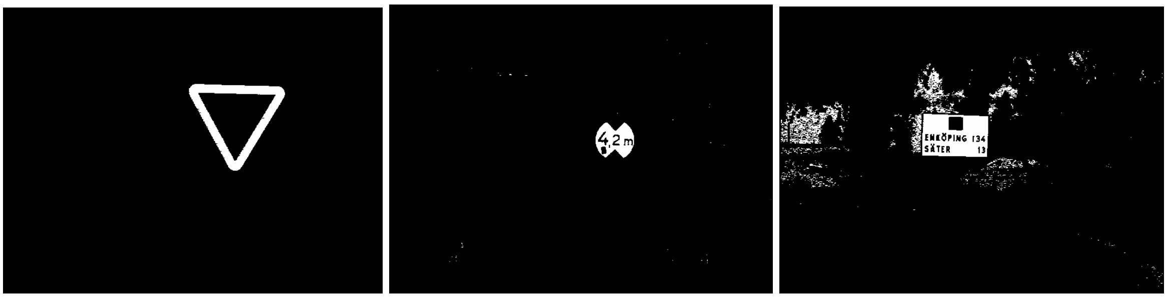
Original Images



Segmentation based on Dynamic-threshold method



Segmentation based on seeded region growing method



Segmentation based on Minimum-Maximum method (modified de la Escalera et al)

Figure 1.15: Results of applying Colour Segmentation Methods [10]

1.6.3 Image Normalization

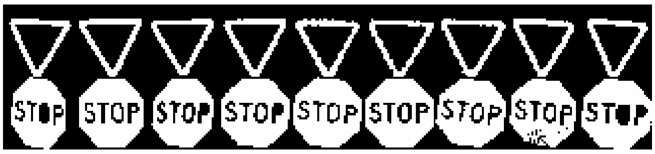
The Binary Image output of segmentation stage is normalized to 36x36 dimensions. This is done to standardize the size of ROI (Region of Interest) irrespective to its scale in original RGB image.

The image is normalized to a certain size; say NxN pixels by using the following formula:

$$x' = N \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$
$$y' = N \frac{y - y_{\min}}{y_{\max} - y_{\min}}$$

Where the coordinates values $x_{\min}, x_{\max}, y_{\min}, y_{\max}$ are the rectangle vertices containing the sign before normalization with sides parallel to the vertical and horizontal axes, and (x', y') are the coordinates of a generic point in the new NxN matrix corresponding to the (x, y) coordinates of the pixel of the original matrix.

Few samples of the segmented and normalized road sign images are shown in figures below.



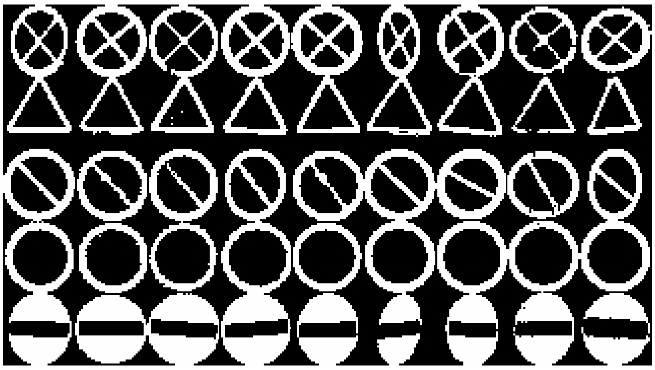


Figure 1.16: Normalized Samples of Different Shape Signs

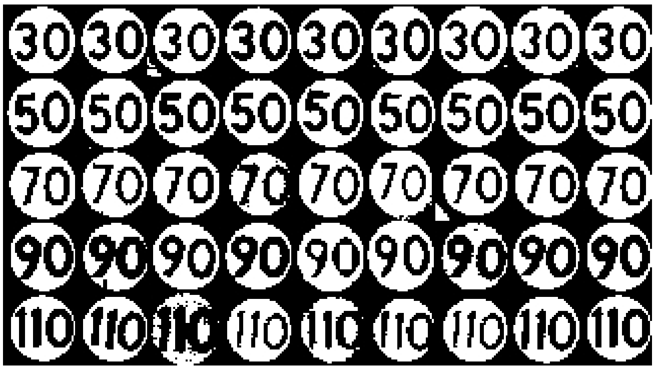


Figure 1.17: Normalized Samples of Different Speed Limit Signs

1.6.4 Image Recognition

Recognizing the road sign is the most important part of an Intelligent Transportation System (ITS). Single miss classification can be the difference between life and death of the people moving on the road. So its really very important for the system to either correctly classify the sign or refrain if not.

Several methods have been applied for the correct recognition of road signs using artificial neural networks. Based on its previous reputation the Support Vector Machine is used to perform the classification task. In this research project Invariant features of the image were used as the raw input data for the Support Vector Machine. By Invariant we mean features those are not affected by the translating or rotating the sign. Using these features is very effective since signs shape can deform with the passage of time and can be positioned and orientated at any angle. Five methods were selected for the extraction of invariant features. These are Haar Invariant Features, Effective Invariant FT coefficients, Geometric Moments, and Orthogonal Fourier-Mellin Moments.

CHAPTER TWO

Support Vector Machine

And

Kernel Method

Introduction

Support vector machines (SVMs) (Vapnik, 1995, 1998) are widely used for pattern classification because of their good generalization ability compared with conventional classifiers [28]. In support vector machines the input space is mapped to a higher dimensional space called the Feature Space. The aim is to find an optimal hyperplane in this higher dimension feature space that can separate the data in the best way possible. Since training of a support vector machine is formulated as a quadratic optimization problem with the number of variables being equal to the number of training data, a global optimal solution can be achieved. Among these training data set the instances necessary for the construction of the decision function are the ones closer to the class boundary. These are called the Support Vectors.

Kernel based techniques (such as Support vector machines SVM, Kernel fisher discriminant KFD, Bayes point machines, Kernel principal component analysis KPCA and Gaussian processes) have made major developments in machine learning algorithms and has shown practical relevance not only in classification and regression problems but also in unsupervised learning. This kernel technique is the backbone of Support Vector Machine.

2.1 Machine Learning

Being a broad field of Artificial intelligence, *Machine Learning* is concerned with the development of algorithms and techniques that allow computers to **Learn**. It has a wide spectrum of applications including *object recognition, medical diagnosis, speech and hand-writing recognition, robot locomotion, computer vision and many more.* To be more specific the goal of machine learning is to ensemble learning and adaptation abilities of living species in computers; more deeply to program computers to use past experience to solve a given problem. As also stated by Michalski:

> "Learning is constructing or modifying representations of what is being experienced." [Michalski & Georghe, 1993]

Machine learning under went a great deal of advancement in the late eighties and nineties with the active research done in the field of *Artificial Intelligence and Neural Networks*. These advancements in machine learning will lead researchers in understanding the learning behaviour in humans and animals and systems like I-Swarm Robots, that imitate the behaviour of ant colonies performing tasks which are much difficult and unsafe for humans to perform [29], and the success of DARPA grand challenge have shown the achievements and upcoming challenges in this field.

Learning can be categorized in various types [Haykin, 1999] some as follows:

- Supervised learning
  - Learning form examples.
  - Learning by taking advice.

- Unsupervised learning
  - Competitive learning.
  - Clustering.
  - Reinforcement learning.

In context of object recognition, machine learning aims on finding a pattern of similarity or structure in a data set that will lead to generalization of learning system and consequently identification of unknown data.

2.2 Statistical Learning Theory

Support vector algorithms are considered as the first practical spin-off of statistical learning theory [30]. Therefore, it's important to have a little insight about *statistical learning theory* before going into details of *Support Vector Machine.* Statistical learning theory addresses the fundamental issue of how to control the generalization ability of a neural network in mathematical terms. Since SVM is a set of supervised learning algorithms, so statistical theory is only reviewed in its context.

There are three basic components interrelated with each other in a supervised learning model. These are:

- *Environment:* providing input vector $\vec{x}$, generated with a fixed but unknown *probability distribution function* $\mathrm{P}(x)$, a standard assumption learning theory [31]. Data generated using probability distribution function is known as *Independent and Identical Distribution (IID).*
- *Supervisor or Teacher:* the teacher knows about the desired response *d* for every input $\vec{x}$ received from the surrounding environment. This desired response *d* is in accordance with the conditional probability distribution function $\mathrm{P}(x \mid d)$ which is also unknown but fixed.
- *Learning Machine:* now the task of learning machine is to map the input vector $\vec{x}$ and the desired response *d* such that to find a function $F(\vec{x}, \vec{w})$ that can approximate input and output in optimum way (statistically).

The feasibility of the system depends how much information does the training set has, generated by the joint probability distribution function of environment and supervisor $\mathrm{P}(x, d)$, for the learning system to have good generalization. Supervised learning problem can be viewed as an *approximation problem*, i.e. finding a function $F(\vec{x},\vec{w})$ that can give the best approximation to the desired function $f(x)$. To find this approximating function there are some tools pioneered by *Vapnik and Chervonenkis* in 1971. Here only few are described.

2.2.1 Loss Function

A popular definition of loss function is:

> $L(d, F(\vec{x},\vec{w}))$ *is the quadratic loss function defined by the squared distance between* $d = f(x)$ *and the approximation* $F(\vec{x},\vec{w})$ *shown as* [30]:
>
> $$L(d, F(\vec{x},\vec{w})) = (d - F(\vec{x},\vec{w}))^2$$

where:

$L(d, F(\vec{x},\vec{w}))$ denotes the loss function.
d the *desired response* for corresponding input vector $\vec{x}$.
and $F(\vec{x},\vec{w})$ the actual response generated by the learning system.

The loss is zero (0) if vector x is correctly classified and one (1) otherwise.

2.2.2 Empirical Risk

For a training set $\{(x_i, d_i)\}_{i=1}^{n}$ the empirical risk function is defined in terms of loss function as [30]:

$$R_{emp}[\vec{w}] = \frac{1}{n}\sum_{i=1}^{n} L(d_i, F(\vec{x},\vec{w}))$$

However it should be noted that *empirical risk* (training error) doesn't ensure that there will be small *risk* (testing error). The *risk function* can be defined for any *loss function* provided the integral exits.

$$R[\vec{w}] = \int L(d, F(\vec{x},\vec{w}))\, d\mathrm{P}(x,d)$$

$R_{emp}[\vec{w}]$: Error on Training data.
$R[\vec{w}]$: Error on Testing data.

In learning machines the goal is to minimize this *risk function* $R[\vec{w}]$ over the class of approximating function $F(\vec{x},\vec{w})$ but its evaluation is complicated because $\mathrm{P}(x,d)$ is usually unknown. The only known information is the training set. To overcome this mathematical difficulty, inductive principal of *Empirical Risk Minimization* is used. This principal relies completely on the availability of training data set, which makes it perfectly suitable for the problem.

2.2.3 VC-Dimension

VC for ”*Vapnik Chervonenkis dimension”* named in honour of its originators, can be defined as a class $F(\vec{x},\vec{w})$ with large number of points that can be *shattered* by the members of $F(\vec{x},\vec{w})$ [32]. The theory of uniform convergence of the *empirical risk* (training data) $R_{emp}[\vec{w}]$ to the *actual risk* (testing data) $R[\vec{w}]$ includes bounds on the rate of convergence and this bound is defined by the *VC-Dimensional* parameter called the *capacity concept of VC theory*.

2.3 Support Vector Machine

Support Vector Machine, invented by *"Vladimir Vapnik",* is a *linear classifier*, using the roots of *statistical learning theory* and the very powerful *kernel function*, and are more demandingly used for solving classification and regression problems. It's a linear machine closely related to classical *Neural Networks,* infact a support vector machine with a sigmoid kernel function acts as a two-layer feed forward neural network. SVM is based on the concept of decision planes that defines the decision boundaries. . To explain the main idea of a support vector machine perhaps the easiest way is to take the scenario of separating patterns that arises in context of pattern classification. In that case the role of support vector machine would be to draw a decision surface which will be called *Hyperplane*, Such that the distance between the closest samples and the hyperplane is maximized. This distance between the closest sample and the hyperplane is known as the *Margin* and the closest samples with respect to which we calculate the margin are called the *Support Vectors*.

Finding a hyperplane with maximum margin is very important. It helps prevent data over fitting problem and enables the system to classify unknown samples from testing set which come closer to hyperplane. A hyperplane with maximum margin is called the *Optimal Hyperplane*.

Any classification task consists of data instances divided into two sets:

- Training set: used to train the system.
- Testing set: used to test the learning of the system.

Now each instance in the training set has one "target value" called the *Class Label* along with several "attributes" called as *Features*. The task of selecting the most suitable features for learning and testing is called *Feature Selection*. It's these features that help the learning system define the hyperplane.

2.3.1: Optimal Margin Hyperplane:

Considering a finite set of input space

$$\mathrm{P}(x,d) = \{(x_1,d_1),(x_2,d_2),......,(x_n,d_n)\} = \{(x_i,d_i)\}_{i=1}^{n} \in \mathrm{X} \times \mathrm{D}$$

generated through probability distribution function.

Where,

x_i represent data instance from input space **X**.

and d_i represent the corresponding output of input space { -1, +1 }

Such that the data represented by $d_i = +1$ (data points inside blue region, i.e. rectangles) and the data represented by $d_i = -1$ (data points inside red region, i.e. stars) is linearly separable. The role of *support vector machine* would be to draw a decision surface that will linearly separate the data points.

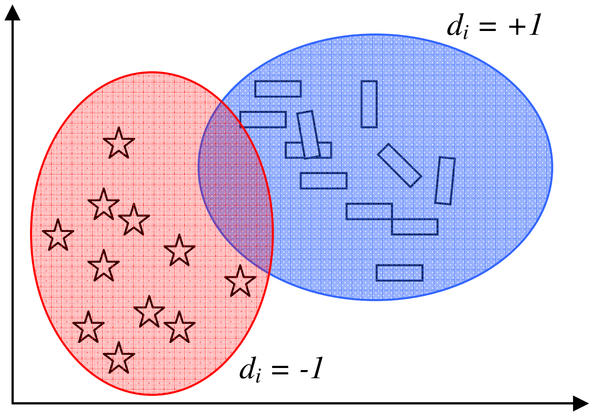


Figure 2.1: Data points belonging to two different classes.

In neural terms a hyperplane separating a linearly separable data is represented by following equation:

$$\sum_{i=1}^{N} w.x + b = \vec{w}.\vec{x} + b = 0$$

where.

$\vec{w}$ is the weight vector orthogonal to the hyperplane (decision surface), controlling the angular movement of the hyperplane.

and b is the bias controlling the movement of the hyperplane parallel to the origin.

This phenomenon is shown in the following figures:

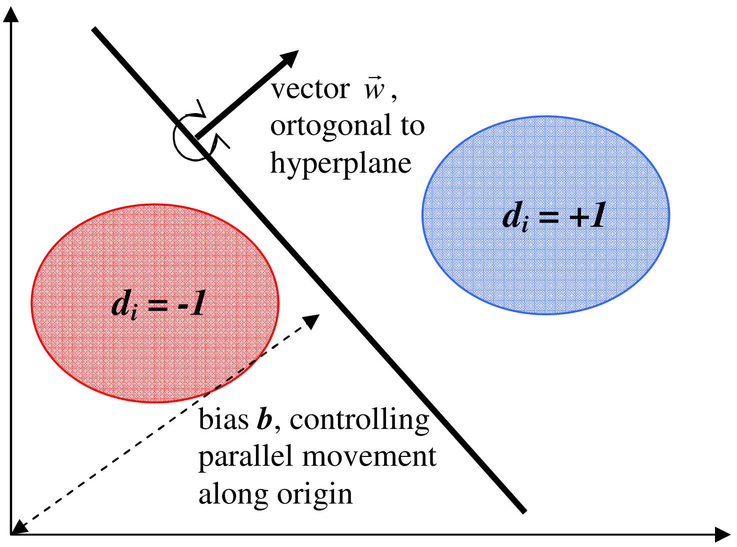


Figure 2.2 (a): Dependency of Hyperplane on weight vector and bias.

Re-arranging the above weight vector and bias equation:

$$\begin{aligned} \vec{w}.\vec{x}+b &\geq 0 \quad for \quad d_i = +1 \\ \vec{w}.\vec{x}+b &< 0 \quad for \quad d_i = -1 \end{aligned}$$

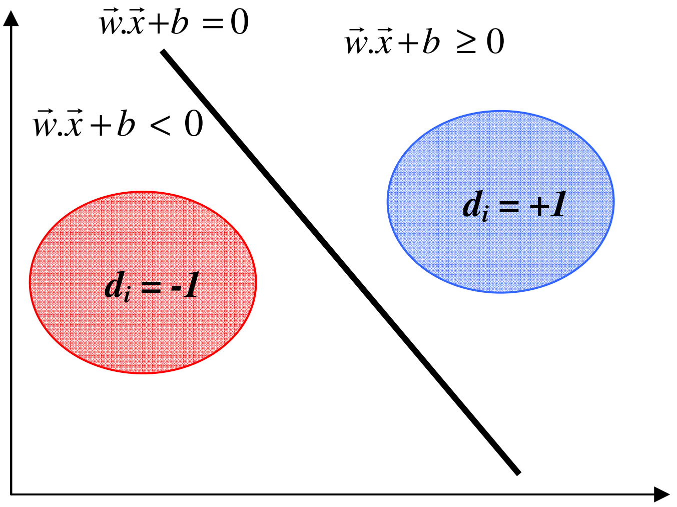
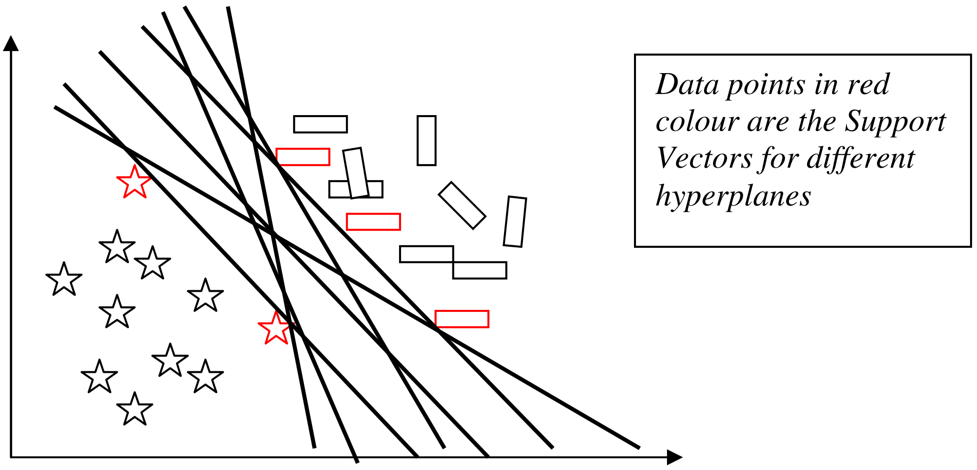


Figure 2.2 (b): Categorization with hyperplane

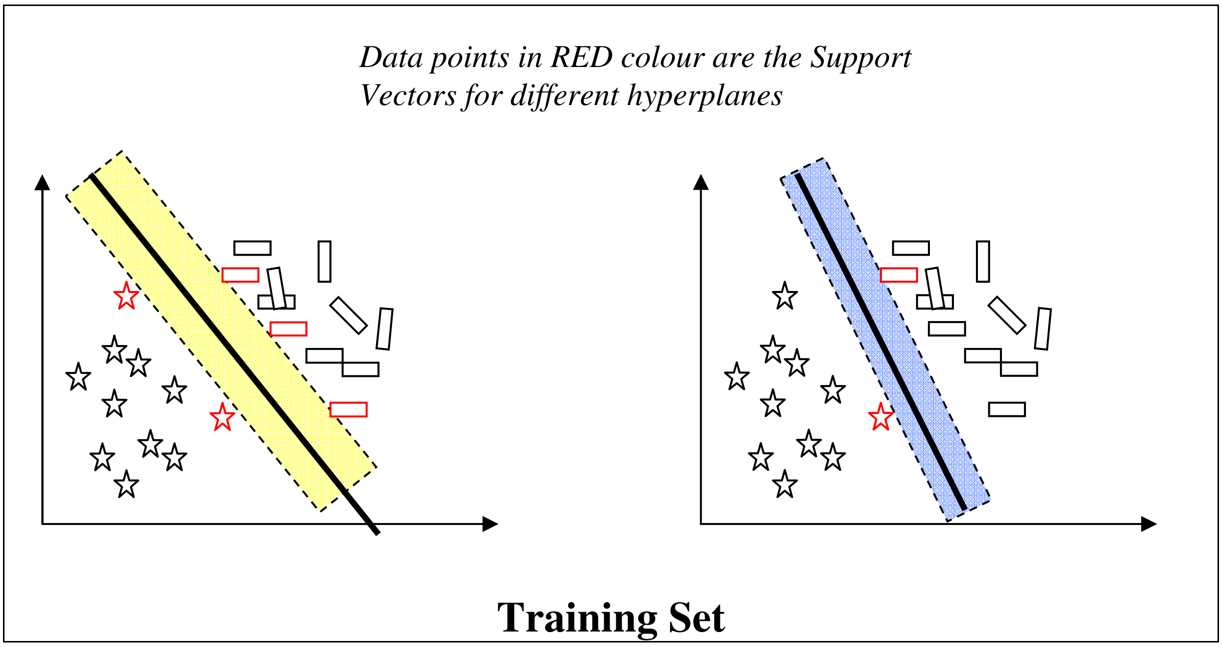
Now provided that the data set generated by i.i.d is linearly separable, an *optimal margin hyperplane* will be the one that can completely separate the data into two classes and has maximum *margin*. In the figure shown below it can be seen that there can be large, an infinite, number of hyperplanes, that can linearly separate the data instances into two classes and each one has a margin different from the other.



Data points in red colour are the Support Vectors for different hyperplanes

Figure 2.3: Set of linear hyperplanes categorizing data at different tangents.

To emphasis the effect of choosing the decision surface with maximum margin let's take two hyperplanes such that there orientation allows one to have greater margin then the other.

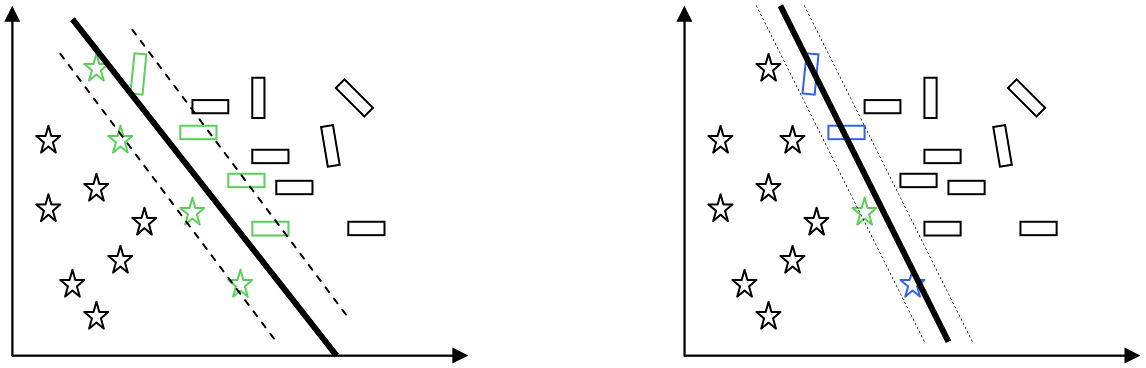


Data points in RED colour are the Support Vectors for different hyperplanes

Figure 2.4: Categorizing training set with hyperplane having different weight and bias

From the selected scenarios shown above it can be seen that the hyperplane on the left hand side has greater margin then the one on the right hand side but still both are able to completely separate the data into two classes. Now when both these hyperplanes are provided with testing set a possible result could be like the one shown in the figure below:

Data points in GREEN colour are the points that come inside the margin but still distinguished by the hyperplane but the data points in BLUE colour are the ones those are not recognized by the hyperplane



Testing Set

Figure 2.5: Categorizing test data with hyperplane having different weight and bias

In above example figure some data instances came too close to hyperplane but the left side hyperplane, the one with greater margin, was able to classify them because of its flexibility but the hyperplane with small margin, the one on the right side, wasn't able to classify some of the data instances as they lie on the hyperplane. Such flexible hyperplane is called the *Optimal Hyperplane* giving the optimal results on both the training and the testing set.

Having a data set that is linearly separable is unlikely to happen in real world problems. Most of the time the data is received in forms that require nonlinear dividing line for separating the instances into two classes such as the one shown in the figure below:

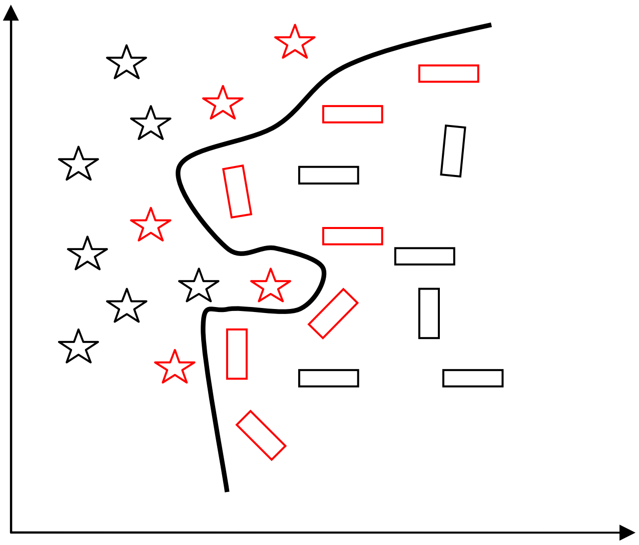


Figure 2.6: Data set that requires a non-linear dividing line

This is the point where some advance technique for handling the situation are required and this is where the concept of *Kernel* comes in handy. Rather then fitting a nonlinear curve to the data set the *Support Vector Machine* uses the *kernel function* to map the data into a different space where a linear hyperplane can be used as the dividing line.

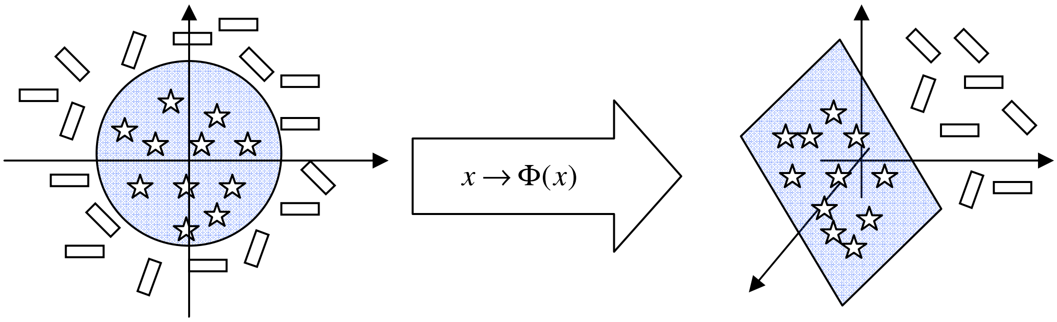


Figure 2.7: Mapping of data to a higher dimensional space

This higher dimensional mapping space is called the Feature Space. The functional concept of *kernel mapping* is very important and powerful. It allows SVM models to perform separations even on data set having very complex boundaries by using *N-dimensional* hyperplanes.

2.3.2 Quadratic Optimization for Optimal Hyperplane

Finding an Optimal Hyperplane using training set $\{(x_i, d_i)\}_{i=1}^{n}$ subjected to the some bounded constrain leads to a Quadratic Optimization Problem.

$$\vec{w}.\vec{x}+b \ \geq \ 0 \quad for \quad d_i = +1$$
$$\vec{w}.\vec{x}+b \ < \ 0 \quad for \quad d_i = -1$$

can be summerized in the form:

$$d_i(\vec{w}.\vec{x}+b) \geq 1$$

This bounded constrained can be stated as [5]:

Given a training set $\{(x_i, d_i)\}_{i=1}^{n}$ *search the optimal values of weight vector* $\vec{w}$ *and bias b such that following constrain is satisfied*

$$d_i(\vec{w}.\vec{x}+b) \geq 1$$

and the cost function is minimized by the weight vector $\vec{w}$:

$$C(\vec{w}) = \frac{1}{2}\vec{w}.\vec{w}^T$$

Where $\frac{1}{2}$ is included for convenience of presentation. This constrained optimization problem is called *Primal Problem* such that:

- *the cost function* $C(\vec{w})$ *is a convex function of* $\vec{w}$.
- *and constrains are linear to* $\vec{w}$

Finding minimum or maximum of a function subjected to fixed outside conditions or constrains is a difficult problem and can be solved by a powerful tool called the *Lagrange Multiplier.*

2.3.2.1 Lagrange Multiplier

The method reduces the problem with *N* variables and *M* constrains to a solvable *N+M variable* with no constrains. The method introduces a new unknown scalar variable called the *Lagrange multiplier* for each constrain and forms a linear combination of constrains along with the multiplier as coefficient. The Lagrange function is:

$$L(\alpha, w, b) = \frac{1}{2}\vec{w}.\vec{w}^T - \sum_{i=1}^{n}\alpha_i\left[d_i(\vec{w}.\vec{x}_i + b) - 1\right]$$

where the non-negative variable α is the *Lagrange multiplier* proportional to the number of times an instance is misclassified causing an update to weight vector.

Every instance has its corresponding α value with α being small if there are few mistakes in classifying the instance and large the other way. This α is also referred as the *embedded strength of instances* and gives *Dual Representation* (an alternate representation). Knowing the values of α_i optimum values of weight vector and bias can be calculated.

2.4 Types of SVM

In order to construct an optimal hyperplane, the support vector machine uses an iterative learning algorithm that minimizes the error function. According to the form of error function, SVM can be classified into four distinct groups [33]:

- C-SVM Classifier. (also called Classification SVM Type 1)
- nu-SVM Classifier. (also called Classification SVM Type 2)

2.5 Kernel Function

With lot of work and advancements being done in the field of machine learning in eighties and nineties, neural networks faced a great deal of difficulty in solving problems with large data set, high dimensionality, data overfitting and lack of interpretation, which were recently alleviated with the introduction to *Kernel* methods. Kernel methods are emerging and innovative techniques that are based on nonlinear data mapping from the original input feature space to a kernel feature space with higher dimensionality, and then solving a linear problem in that space. These methods allow us to interpret (and design) learning algorithms geometrically in the kernel space (which is nonlinearly related to the input space), thus combining statistics and geometry in an effective way.

In last few year kernel based learning techniques such as *support vector machines* (SVM), *kernel fisher discriminant* (KFD), *bayes point machines*, *kernel principal component analysis* (KPCA) and *gaussian processes,* have made major developments in machine learning algorithms. Successful application of kernel based algorithms have been reported in various field of engineering and medicine such as *audio and image processing, biomedical engineering, communication, bioinformatics, computational biology* and many more. In many cases kernel based techniques showed superiority to there competitors and revealed advantages that have theoretical and practical importance. It's because of there ability to solve complex problems (like pattern recognition and clustering) efficiently and been able to adapt to wide area of research (like biology, signal and image processing, communication), these methods have captured the attention of many researchers and practitioners in safety-related areas.

It is often difficult to solve problems like classification, regression and clustering in the same space from which the underlying observations have been made. Kernel methods provide the mapping power to transform the data from observational space to a higher dimensional feature space. This underlying justification is found in *Covers Theorem* on the *Separability of Patterns*, which in qualitative terms many be stated as follows (Cover, 1965) [30]:

> *A complex patterns classification problem cast in high-dimensional space non-linearly is more likely to be linearly separable that in a low-dimensional space.*

If the methods applied in the feature space are only based on dot or inner products the projection does not have to be carried out explicitly but only implicitly using kernel functions. This is often referred to as the "kernel trick" [31].

$$k = \mathrm{X} \times \mathrm{X} \in R^N$$

$$k(x, x') = \langle \vec{x}, \vec{x}' \rangle = \sum_{i=1}^{N} [\vec{x}]_i [\vec{x}']_i$$

Where:

x and x' are the two patterns.
i is the ith element of *x*.
and *k* is the *Kernel Function.*

Though there are different types of kernels one can always find in literatures these four basis types of kernel functions:

- Linear: $k(\vec{x}_i, \vec{x}_j) = \vec{x}_i^T \vec{x}_j$
- Polynomial: $k(\vec{x}_i, \vec{x}_j) = (\gamma \vec{x}_i^T \vec{x}_j + r)^d$, $\gamma > 0$
- Radial Basis Function (RBF): $k(\vec{x}_i, \vec{x}_j) = \exp(-\gamma \| \vec{x}_i - \vec{x}_j \|^2)$, $\gamma > 0$
- Sigmoid: $k(\vec{x}_i, \vec{x}_j) = \tanh(\gamma\, \vec{x}_i^T \vec{x}_j + r)$

Here, γ, r, and d are ***kernel parameters***. It is always recommended for starters to use Radial Basis Function as the primary kernel function from the four basic kernels because of following advantages:

- RBF kernel can non-linearly map the samples into higher dimensional space, unlike linear kernels.
- Linear kernel is a special case of RBF as linear kernel with penalty parameter $\tilde{C}$ has same performance as RBF with some parameters (C, γ).
- Hyperparameters affect the complexity of the model selection. Polynomial kernel has more hyperparameters than RBF.

- A Sigmoid kernel behaves as RBF for certain parameters, and also sigmoid function is not valid under some parameters (no inner product of two vectors).
- And last but not the least, RBF kernel has less numerical difficulties. One key point is $0 < k_{ij} \leq 1$ in contrast to polynomial kernels in which kernel values may go to infinity ($\gamma \vec{x}_i^T \vec{x}_j + r > 1$) or zero ($\gamma \vec{x}_i^T \vec{x}_j + r < 1$) while the degree is large.

2.5.1 Feature Space Learning

Complexity of learning system depends on the way data is represented to the system. An ideal way is to transform the target problem into a form that matches the specific learning problem. Taking the same training set, generated through probability distribution function a higher dimensional mapping of the data set will result in following equation:

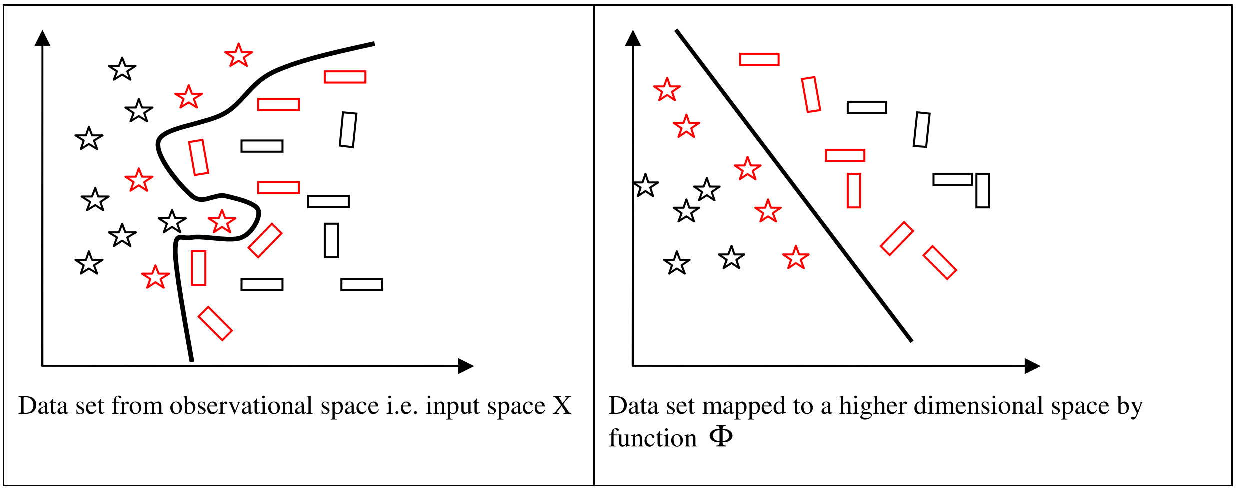
$$\mathrm{P}(x) = (x_1, x_2, \ldots\ldots, x_n) \in \mathrm{X}$$

$$\Phi : \mathrm{X} \to \mathrm{H}, \qquad x \mapsto \Phi(x)$$

$$(x_1, x_2, \ldots\ldots, x_n) \mapsto (\Phi(x_1), \Phi(x_2), \ldots\ldots, \Phi(x_n))$$

$$k(x, x') = \langle \vec{x}, \vec{x}' \rangle = \langle \Phi(\vec{x}), \Phi(\vec{x}') \rangle$$

This step is equivalent to mapping the input space X into a new feature space $\mathrm{H} = \{\Phi(x) \mid x \in \mathrm{X}\}$, where Φ is the mapping function that non-linearly maps the input to new dot product space. This phenomenon is shown in the figure below:



Data set from observational space i.e. input space X

Data set mapped to a higher dimensional space by function Φ

Figure 2.8: Division curve in Observational space and Higher Dimension space

Mapping of input space to higher dimensional space requires extracting features of data set for representation. The procedure of choosing the suitable features is called *Feature selection*. Different approaches for feature selection exits. Seeking for the smallest set of features that can still convey the essential information contained in the original attributes result in *Dimensional Reduction* and can be very beneficial as computational and generalization performance can degrade as the number of features grows and is called the *Curse of Dimensionality.*

$$(x_1,\ x_2, \ldots\ldots,\ x_n) \mapsto (\Phi(x_1),\ \Phi(x_2), \ldots\ldots,\ \Phi(x_m)) \quad with \quad m < n$$

One way could be the detection of irrelevant features and there subsequent elimination or use of variance. The use of principal component analysis provides a way of mapping attributes of input set to feature space with new features being linear function of the original attributes sorted in the order of variance that the data exhibits. Features corresponding to direction with low variance can be removed for reducing dimensionality; however there is no guarantee that if these features contain important information for performing target classification.

.

Building a non-linear machine with a linear classifier requires following two steps:

- A fixed non-linear mapping that transforms the data into feature space H or dot product space.
- Use of linear decision surface for classification in feature space: $\sum_{i=1}^{m} w.\Phi(x) + b$

Dual representation property of linear machines states that the hypothesis can be expressed as a linear combination of training points, that the decision rule can be evaluated using just inner product between the training points and the test points:

$$\sum_{i=1}^{m} \alpha_i\, d_i\, \langle \Phi(\vec{x}).\Phi(\vec{x}') \rangle + b$$

One advantage of kernel function is that it implicitly computes the inner-product, $\langle \Phi(\vec{x}).\Phi(\vec{x}') \rangle$, without explicitly knowing about the mapping Φ. So, kernels allow us to compute inner-products in spaces where one could hardly perform any computation: *every linear algorithm that only uses scalar products can implicitly be executed in H by using kernels, i.e. one can very elegantly construct a non-linear version of a linear algorithm* [34].

2.5.2 Kernel Features

By the use of kernel SVM is restricted to deal with only linear classification problem by transforming the too complex non-linear classification problem to a much simpler linear classification problem. Its main idea is to first pre-process the data by some non-linear mapping Φ and then apply the linear algorithm on the new mapped space.

2.5.2.1 Gram Matrix

Given a training set $(x_1, x_2, \ldots\ldots, x_n) \in \mathrm{X}$, a $\mathrm{M} \times \mathrm{M}$ matrix with elements $\mathrm{K}_{ij} = k(x_i, x_j)$ is called *Gram Matrix or Kernel Matrix.* This matrix is said to be positive definitive only if any quadratic over K is positive: i.e. for all z_i a set of real values for $i = 1, \ldots., \mathrm{M}$,

$$\sum_{i,j=1}^{\mathrm{M}} \mathrm{K}_{ij}.z_i.z_j \geq 0$$

For a kernel (and a matrix) to be positive definite, it is necessary to be symmetric and non-negative on the diagonal.

2.5.2.2 Mercer's Theorem

One of the fundamental mathematical results underlying learning theory with kernel is the *Mercer's Theorem.* The Mercer's Theorem presented in James Mercer, *Functions of positive and negative type and their connection with the theory of integral equations,* in 1909, is one of the most notable works of James Mercer. In mathematics and functional analysis *it is the representation of a symmetric positive-definitive function,* a Hermitian[1], *on a square as a sum of a convergent sequence of product function.*

Given a training set $(x_1, x_2, \ldots\ldots, x_n) \in \mathrm{X}$, and K be a symmetric function satisfying for any finite set of points $\{x_i\}_{i=1}^{n}$ in X and set of real values $\{z_i\}_{i=1}^{n}$: $\sum_{i,j=1}^{\mathrm{M}} \mathrm{K}_{ij}.z_i.z_j \geq 0$ with K being positive-definitive kernel over X. Since K is symmetric there is an Orthogonal matrix ***V*** such that:

$$\mathrm{K} = V \Lambda V'$$

[1] Hermitian is a square matrix with complex entities which is equal to its own conjugate transpose

Where:

Λ is a diagonal matrix containing *Eigenvalues* λ_k of K.

with corresponding *Eigenvectors* $v_k = (v_{ki})_{i=1}^{n}$ the columns of ***V***.

Now considering all non-negative eigenvalues[22] with following feature space mapping:

$$\Phi : \vec{x}_i \mapsto \left(\sqrt{\lambda_k}\;\; v_{ki}\right)_{i=1}^{n}$$

The dot product is:

$$\langle \Phi(\vec{x}_i).\Phi(\vec{x}_j)\rangle = \sum_{i=1}^{n} \lambda_k\, v_{ki} v_{kj} = (V\Lambda V')_{ij} = \mathrm{K}_{ij} = k(\vec{x}_i, \vec{x}_j)$$

For a square matrix *A,* λ will be the **eigenvalue** of *A*, if there exists a non-zero vector v such that $Av = \lambda v$. In this case, v is called an **eigenvector** (corresponding to λ), and the pair (λ,*x*) is called an **eigenpair** for *A*.

2.5.2.3 Hilbert Space

*A Hilbert space is a vector space **H** with an inner product $\langle f, g\rangle$ such that the norm defined by:*

$$| f | = \sqrt{\langle f, f\rangle}$$

*turns **H** into a complete Metric Space and if the metric defined by the norm is not complete, then **H** is instead know as the Inner Product Space.*

Proposition [35]**:** *let* X *be a finite input space with* $k(\vec{x}, \vec{x}')$ *a symmetric function on* X. *Then* $k(\vec{x}, \vec{x}')$ *is a kernel function if and only if the matrix* $\mathrm{K} = \{k(\vec{x}_i, \vec{x}_j)\}_{i,j=1}^{n}$ *is positive semi definite (has non-negative eigenvalues)*

Based on this, a slight generalization on inner product in Hilbert space is made by introducing weighting to eigenvalues λ_t for each dimension.

$$\langle \Phi(\vec{x}).\Phi(\vec{x}')\rangle = \sum_{i=1}^{\infty} \lambda_k \Phi(\vec{x}).\Phi(\vec{x}') = k(\vec{x},\, \vec{x}')$$

And sufficient and necessary conditions provided by Mercer's theorem for the continuous symmetric function $k(\vec{x}, \vec{x}')$ to accept such a representation:

[2] Eigenvalue has great importance in the field of engineering. They can help engineers build better designs and structures. An eigenvalue of a square matrix is a scalar that is usually represented by the Greek letter λ and an eigenvector is a vector.

$$k(\vec{x},\, \vec{x}') = \sum_{i=1}^{\infty} \lambda_k \Phi_i(\vec{x})\; \Phi_i(\vec{x}')$$

Theorem [35]: *(Mercer) let* X *be a compact subset of* $\Re^n$. *Suppose **k** is a continues symmetric function such that the integral operator:*

$$\mathrm{T}_k : L_2(\mathrm{X}) \to L_2(\mathrm{X})$$
$$(\mathrm{T}_k . f) = \int_{\mathrm{X}} k(\cdot, \vec{x}')\, f(\vec{x}')\, dx$$

is positive, that is

$$\int_{\mathrm{X}\times\mathrm{X}} k(\vec{x}, \vec{x}')\, f(\vec{x}) f(\vec{x}')\, dx\, dx' \ge 0$$

for all $f \in L_2(\mathrm{X})$. *Then* $k(\vec{x}, \vec{x}')$ *can be expanded in a uniformly convergent series (on X x X) in terms of* T_k’s *eigen-funcitons* $\Phi_i \in L_2(\mathrm{X})$ *normalized in such a way that* $\|\Phi_i\|_{L_2} = 1$, *and positive associated eigenvalues* $\lambda_i \ge 0$

$$k(\vec{x}, \vec{x}') = \sum_{i=1}^{\infty} \lambda_i\, \Phi_i(\vec{x}) \Phi_i(\vec{x}')$$

More details and comprehensive proof of these propositions and theorems could be found in [35, 36] and numerous other literatures.

CHAPTER THREE

Features

Introduction to Invariant Features

Using invariant features to identify a pattern is a very useful tool in machine vision applications like in aircraft identification, ship identification, and automatic inspection in an industrial environment and in our case for Recognition of Road signs. The performance of these applications critically depends upon methods used for extracting image features. An invariant features are those image characteristics which remain unaffected when a transformation group is applied over the image.

When classifying a pattern its location, size and orientation in the camera plane are irrelevant. One way to get rid of this ambiguity is to extract those features from the pattern which are invariant on translation, rotation and scaling. Since last decade numerous moment based methods have been proposed for the extraction of these invariant features. A moment can be described to be any set of parameter obtained by projecting a pattern on a 2 Dimensional polynomial basis function.

3.1 Haar Invariance

For a grey scale image G, the number $G(x, y)$ defines the grey value of the image at pixel coordinate (x, y). For an image with dimension $N \times N$, the range of pixel coordinates will be from $0 \le (x, y) \le N-1$. Considering a transformation group R that constitutes rotation and translation transformations. If a transformation element r such that $r \in R$, acts on this grey scale image G then the transformed image would be described as:

$$rG(x, y) = G(i, j)$$

with,

$$\begin{bmatrix} i \\ j \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} t_0 \\ t_1 \end{bmatrix}$$

The matrix $\begin{bmatrix} t_0 & t_1 \end{bmatrix}^T$ defines the translation matrix such that $\begin{bmatrix} t_0 & t_1 \end{bmatrix}^T = T \in \Re^2$. The range of this translation matrix can be restricted to corresponding image size. The square matrix $\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$ defines the rotational transformation matrix such that its range is from $\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} = \Theta \in \begin{bmatrix} 0 & 2\pi \end{bmatrix}$.

3.1.1 Invariant Transformation Features

Calculating invariant features results in a complex valued function i.e. $C(G)$. This complex valued function is invariant with respect to the action of transformation group on the image [37], i.e.

$$C(rG) = C(rG\,) \quad \forall \quad r \in R \qquad \text{.......} \ (3.1.1.a)$$

Such invariant features can be constructed by integrating $c(rG)$ over the transformation group *R*, where *c* is a complex valued function. This is elaborated by the equation shown below:

$$C(R) = \frac{1}{|R|} \int_R c(r\,G)\; dr \qquad \text{.......} \ (3.1.1.b)$$

3.1.1.1 Translation and Rotation Invariance

Cyclic boundary conditions can be used to find translation and rotation invariant features of the image such that the integral over the transformation group will be:

$$C(R) = \frac{1}{2\pi N^2} \int_{t_0=0}^{N} \int_{t_1=0}^{N} \int_{\phi=0}^{2\pi} c(r\,G)\, d\phi\, dt_1\, dt_0 \qquad \text{.......} \ (3.1.1.1.a)$$

For complex valued function $c(rG) = G[0,\ 0]G[0,\ 1]$ the above equation becomes:

$$C(R) = \frac{1}{2\pi N^2} \int_{t_0=0}^{N} \int_{t_1=0}^{N} \int_{\phi=0}^{2\pi} G[-t_0,\ -t_1]G[\sin\phi - t_0, \cos\phi - t_1]d\phi dt_1\, dt_0 \qquad \text{.......} \ (3.1.1.1.b)$$

With a close examination of this equation it can be seen that:

- $G[\sin\phi - t_0, \cos\phi - t_1]$ with range $0 \le \phi \le 2\pi$ defines a unit circle. The centre of this circle is at coordinate $G[-t_0,\ -t_1]$. The integral over the angle ϕ adds the grey values of all the pixels that lie inside the unit circle. Then it multiplies this sum with the grey value of the centre pixel of this unit circle.

- The integrals over t_0 *and* t_1 indicate that the first step should be performed for every pixel in the image

The above two steps compute the *Local Function* of every pixel whose value depends upon the grey value of the coordinate neighbouring pixels. Once the local function of every pixel is computed they are all added up giving us the features that are invariant to rotation and translation transformation. This procedure is particularly good for images containing multiple objects.

3.2 Effective Fourier Transform Coefficients

The concept of effective fourier transform coefficients is taken from the article published in 2004, written by *Shan Li, Chuen Lee and Donald Adjeroh* [38]. The article aimed on *Content Based Image Retrieval System (CBIR)* by using *Compound Image Descriptors (CID).* The CID incorporates the Fourier Transform, its magnitude and phase coefficients, along with Global Features, including the form factor (FF) and compactness (C), of the image. The FT magnitude and phase coefficients were analytically proved to be translation, rotation and scale invariant.

Our aim is to find the unique features of the image those are invariant on translation and rotation. The Global features do not provide us these unique image properties instead they focus on describing image shape contours which is more useful in filtering out a number of images from a stock of different images. Computation of these unique invariant features is done by calculating the magnitude and phase coefficients of the Fourier transform.

3.2.1 Properties of Fourier Transform

For an image function $f(x,y)$ having size $N \times M$, the discrete Fourier transform is given by the equation:

$$F(u,v) = \frac{1}{NM}\sum_{x=0}^{N-1}\sum_{y=0}^{M-1} f(x,y)\exp\left[-j2\pi\left(\frac{ux}{N}+\frac{vy}{M}\right)\right] \quad \text{....... (3.2.1.a)}$$

where,

u and v are called the *frequency variables*, ranging from $u = 0, 1, 2, \ldots, N-1$ and $v = 0, 1, 2, \ldots, M-1$ respectively.

Given the Fourier transform its inverse FT can be computed. This is given by the equation:

$$f(x,y)=\sum_{u=0}^{N-1}\sum_{v=0}^{M-1}F(u,v)\exp\left[j2\pi\left(\frac{ux}{N}+\frac{vy}{M}\right)\right] \qquad \text{....... (3.2.1.b)}$$

where,

x and y are the *spatial pair*, ranging from $x = 0, 1, 2, \ldots, N-1$ and $y = 0, 1, 2, \ldots, M-1$ respectively.

These two equations (3.2.1.a & 3.2.1.b) are called the *Discrete Fourier Transform Pair (DFT)*, also written as:

$$f(x,y) \Leftrightarrow F(u,v) \qquad \text{....... (3.2.1.c)}$$

The two important parts of Fourier transform are the phase angle and the magnitude also called as the Fourier Spectrum. These two quantities are defined respectively as:

$$\Phi(u,v)=\tan^{-1}\left(\frac{I(u,v)}{R(u,v)}\right) \qquad \text{....... (3.2.1.d)}$$

$$|F(u,v)|=\left[R^2(u,v)+I^2(u,v)\right]^{1/2} \qquad \text{....... (3.2.1.e)}$$

where,

$R(u,v)$ describes the real part.

and $I(u,v)$ describes the imaginary part.

3.2.2 Invariant Features

For a transformation group G constituting of translation and rotation, a transformation $g \in G$ applied on the image in Cartesian coordinates has its equivalent effect in Frequency domain.

3.2.2.1 Translation Invariance

Let x' *and* y' be the linear displacement in objects position from the origin, then the Fourier transform pair will be:

$$f(x-x',\ y-y') \Leftrightarrow F(u,v)\exp\left[-j2\pi\left(\frac{ux'}{N}+\frac{vy'}{M}\right)\right] \quad \text{....... (3.2.2.1.a)}$$

From equation 3.2.2.1 and 3.2.1.c it is clear that the magnitude of the Fourier transform is the same. This means a shift in image position in Cartesian coordinate doesn't affects its magnitude in frequency domain i.e. $|F(u,v)|$ is translation invariant.

3.2.2.2 Rotation Invariance

Polar coordinate is used to illustrate the effect of rotation. Let,

$$x = r\cos\theta \qquad u = \omega\cos\phi$$
$$y = r\sin\theta \qquad v = \omega\sin\phi$$

be the polar coordinate conversion parameters. Then equivalent representation of the image $f(x,y)$, existing in Cartesian coordinates, in polar coordinates will be $f(r,\ \theta)$. Similarly equivalent representation of the image $F(u,v)$, existing in frequency domain, in polar coordinates will be $F(\omega,\ \phi)$.

Let's consider that the original image is rotated by an angle β. Then the Fourier transform pair in polar coordinates will be:

$$f(r,\ \theta+\beta) \Leftrightarrow F(\omega,\ \phi+\beta) \quad \text{....... (3.2.2.2.a)}$$

This equation implies that the amount of rotation in original image results in equivalent amount of rotation in its Fourier transform. Similarly a rotation in frequency domain will result in an equivalent rotation in Cartesian space. However later on, in the detail of polar coordinate conversion it will be seen that a rotation in Cartesian space results a column shift in polar space.

The equation 3.2.2.2 is valid for rotation of the object about its origin. However an object can be rotated around any point. Let's suppose that there is an arbitrary point P in the image around which the object is rotated. If $(x_p,\ y_p)$ are the object coordinates before rotation and $(x'_p,\ y'_p)$ are the coordinates after rotation, then:

$$x' = x - x_p \qquad x'_p = x' \cos\beta + x_p$$
$$y' = y - y_p \qquad y'_p = y' \sin\beta + y_p$$

$$\Rightarrow\; x'_p = x\cos\beta - x_p \cos\beta + x_p$$
$$\Rightarrow\; y'_p = y\sin\beta - y_p \sin\beta + y_p$$

$$\Rightarrow\; x'_p = x\cos\beta + x_p(1-\cos\beta)$$
$$\Rightarrow\; y'_p = y\sin\beta + y_p(1-\sin\beta)$$

where,

the term $x_p(1-\cos\beta)$ *and* $y_p(1-\sin\beta) \Rightarrow R(x_p,\, y_p,\, \beta)$ implies that the object moved to the origin, rotated and then shifted back.

Rotation of image in Cartesian coordinate will result in a column-shift in it corresponding polar image $f(r,\theta) \rightarrow f(r,\theta+\beta)$, where β is the angle of rotation. A column shift in polar image affects its FT pair as:

$$f(r,\theta+\beta) \Leftrightarrow F(\omega,\phi)\exp\left[-\frac{j2\pi\phi\beta}{N}\right] \qquad \text{....... (3.2.2.2.b)}$$

and in exponential-form

$$F(\omega,\phi) = \left|F(\omega,\phi)\right| e^{j\alpha(\omega,\phi)} \qquad \text{....... (3.2.2.2.c)}$$

where,

$\left|F(\omega,\phi)\right|$ is the magnitude spectrum and $\alpha(\omega,\phi)$ is the phase angle.

Let $F_R(\omega,\phi)$ be the FT of the polar image after a shift of size β in the column and $\alpha_R(\omega,\phi)$ be the corresponding phase angle. From above two equations of FT pair for polar image:

$$\left|F_R(\omega,\phi)\right| = \left|F(\omega,\phi)\right| \qquad \text{....... (3.2.2.2.d)}$$

and

$$\alpha_R(\omega,\phi) = \alpha(\omega,\phi) - \frac{2\pi\phi\beta}{N} \qquad \text{....... (3.2.2.2.e)}$$

Rotation invariance can be achieved by substituting a phase of $F(1,1)$, weighted by ϕ, in all the FT phase coefficients, i.e.

$$F'(\omega,\phi) = F(\omega,\phi)\, e^{-j\alpha(1,1)\phi} \qquad \text{....... (3.2.2.2.f)}$$

The corresponding phase angle will be:

$$\alpha'(\omega,\phi) = \alpha(\omega,\phi) - \alpha(1,1)\phi \qquad \text{....... (3.3.2.2.g)}$$

where,

$F'(\omega,\phi)$ and $\alpha'(\omega,\phi)$ are the corrected FT and phase of the polar image. This correction enables the polar image to be invariant to column shifts. From equation_3.2.2.2.e:

$$\alpha_R(1,1) = \alpha(1,1) - \frac{2\pi\beta}{N}$$

and

$$\alpha_R(1,1)\phi = \alpha(1,1)\phi - \frac{2\pi\phi\beta}{N}$$

and from equation (3.2.2.2.g)

$$\alpha'_R(\omega,\phi) = \alpha_R(\omega,\phi) - \alpha_R(1,1)\phi$$

By substituting $\alpha_R(1,1)\phi$ and $\alpha_R(\varpi,\phi)$:

$$\alpha'_R(\omega,\phi) = \alpha(\omega,\phi) - \frac{2\pi\phi\beta}{N} - \left[\alpha(1,1)\phi - \frac{2\pi\phi\beta}{N}\right]$$

$$= \alpha(\omega,\phi) - \alpha(1,1)\phi$$

$$\Rightarrow \alpha'_R(\omega,\phi) = \alpha'(\omega,\phi) \qquad \ldots\ldots (3.2.2.2.f)$$

The above equation implies that by subtracting phase of $F_R(1,1)$ and its corresponding phase angle $\alpha_R(\omega,\phi)$, the feature of the image will become rotation invariant.

3.3 Geometric Moments

For a Two-Dimensional image the use of Geometric Moments (GM), some times also called *Standard Moments* [39], was first introduced by Hu [40]. He defined a set of invariant moments derived from GMs which were invariant to translation shifts, rotation and scale changing. Since then GMs had been widely used for calculating the invariant feature of the image under analysis it has been successfully applied in aircraft identification, texture classification and radar image to optical image matching [41]. Geometric moment is the simplest one among the moment functions. The main advantage of GM is that image coordinate transformation can be easily expressed and analyzed in terms of corresponding transformation in moment space [42]. Because of theorem uniqueness, every image is uniquely determined by its geometric moments. Different order geometric moments give different spatial characteristics of image intensity distribution [42]. A low order moment gives less information about the image while high order moments give more information but they are also more vulnerable to noise.

A Two-Dimensional geometric moment of order $(n+m)$ for an image with function $f(x,y)$ is described as:

$$g_{n,m} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} x^n\, y^m\, f(x,y)\,dx\,dy \qquad \ldots\ldots (3.3.a)$$

The Discrete form of this equation for an $(N \times N)$ image will be:

$$G_{n,m} = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} x^n\, y^m\, f(x,y)\,\Delta x\,\Delta y \qquad \ldots\ldots (3.3.b)$$

where

n and m are integer values $0, +1, +2, \ldots$

From the above two equations it can be seen that the monomial product $x^n\, y^m$ is the basis function indicating that geometric moments are not orthogonal since the basis function is not orthogonal.

3.3.1 Translation Invariant Features

By definition, the zero order moments i.e. m_{00} defines the total intensity of the image. The first order moment function i.e. m_{01} & m_{10} provide the intensity of the image about its X-Axis's and Y-Axis's respectively. The intensity centroid $(x_0,\, y_0)$ of the image is calculated using the zero order and first order moments:

$$\Rightarrow \; x_0 = \frac{m_{10}}{m_{00}} \;, \quad y_0 = \frac{m_{01}}{m_{00}}$$

Moments captured with respect to image intensity centroid are called *Central Moments*. These moments are translation invariant because the image centroid $(x_0,\, y_0)$ moves with the image under transformation group and the moments are defined with respect to this centroid as the origin.

$$\Psi_{nm} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} (x - x_0)^n\, (y - y_0)^m\, f(x, y)\, dx\, dy \qquad \text{....... (3.2.2.a)}$$

where,

n and m are integer values $0, +1, +2, \ldots$

3.3.2 Rotation Invariant Features

Considering a linear transformation of the image coordinates from $f(x,\, y)$ to $f(x',\, y')$:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

With the assumption that a_{ij} are constants and the determinant $\Delta = \left| a_{11}\, a_{22} - a_{12}\, a_{21} \right| \neq 0$
The GM of this transformed image $f(x',\, y')$ will be:

$$g'_{n,m} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} (x')^n \, (y')^m \, f(x', y') \, dx' \, dy'$$

$$g'_{n,m} = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} (a_{11}x + a_{12}y)^n \, (a_{21}x + a_{22}y)^m \, f(x, y) \, \Delta \, dx \, dy \qquad \text{.......} (3.3.3.a)$$

Now let's suppose that the transformation function applied on the image $f(x, y)$ is the rotation function, rotating the image at an angle ϕ. The transformed image pixel coordinates will be:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

By using equation 3.2.3.a different order of image moments are computed. The zero order moment m_{00} which gives the total intensity of the image is invariant under rotation transformation. The second order moment $(m_{20} + m_{02})$ of the image will be:

$$m'_{20} = \left[\frac{1+\cos\phi}{2}\right] m_{20} - (\sin 2\phi)\, m_{11} + \left[\frac{1-\cos\phi}{2}\right] m_{02} \qquad \text{.......} (3.3.3.b)$$

$$m'_{02} = \left[\frac{1-\cos\phi}{2}\right] m_{20} + (\sin 2\phi)\, m_{11} + \left[\frac{1+\cos\phi}{2}\right] m_{02} \qquad \text{.......} (3.3.3.c)$$

From equation 3.2.3.b & 3.2.3.c the second order moment, $(m_{20} + m_{02})$, of the image is invariant to rotation at any angle ϕ.

3.4 Orthogonal Fourier-Mellin Moments

Orthogonal Fourier-Mellin Moments(OFMM) also known as generalized Zernike Moments(ZM) were first introduced by Sheng and Shen in 1994 [43]. They showed that with the ability of having finite number of moments OFMM can produce better results in describing images of small size in terms of reconstruction error and signal-to-noise ratio (SNR) as compared to ZM. In 2000 Chao and Srinath [44] used OFMM with different scaling methodologies for the classification of alphanumeric characters selected from two different databases. They simulated the classification results using both OFMM and ZM and concluded that OFMM contains more local information about the character image under analysis as compared to ZM, therefore, OFMM can be used to describe small character image in the same way as large character images. OFMM are radial polynomials thus they require much lower order of moments as compared to ZM resulting in less vulnerability to character variation and noise.

Orthogonal Fourier-Mellin moments are based on radial polynomials. They are robust because the moment order required to describe the image is much lower than the moments required by Zernike. In polar coordinates the radial moments and the circular Fourier transform of OFMM are defined as:

$$\mathrm{O}_{k,h} = \frac{1}{2\pi\alpha_k}\int_0^{2\pi}\int_0^1 f(r,\varphi)\,Q_k(r)\exp(-jh\varphi)\,r\,dr\,d\varphi \quad \ldots\ldots (3.4.a)$$

where,

$f(r,\varphi)$ is the image function under analysis.

α_k is the normalization factor defined by $\frac{1}{2(n+1)}$

h is the circular harmonic order ranging from $0, \pm 1, \pm 2, \ldots$

and $Q_k(r)$ is the polynomial in *r* of degree *k*. This polynomial is defined as:

$$Q_k(r) = \sum_{i=0}^{k} \alpha_{k,i}\, r^i$$

where

$$\alpha_{k,i} = \frac{(-1)\ (k+i+1)!}{(k-i)!\ i!\ (i+1)!}$$

It is proved that the polynomial $Q_k(r)$ is orthogonal over the range $0 \le r \le 1$ provided the maximum size of the object is greater than 1; $r = 1$. The discrete form of OFMM is expressed in regular x, y coordinates. The above equation of OFMM in Cartesian coordinates will be:

$$\text{O}_{k,h} = \frac{k+1}{\pi} \sum_{x} \sum_{y} f(x,y) Q_k(r) \exp(-jh\varphi) \Delta x \Delta y \qquad \ldots\ldots \text{(3.4.b)}$$

The equation still satisfies the statement that $Q_k(r)$ is orthogonal in Cartesian coordinates as:

$$x^2 + y^2 \leq 1, \qquad x = r\cos\varphi, \qquad y = r\sin\varphi$$

indicating that the object is inside the unit circle.

3.4.1 Translation Invariance

In OFMM translation invariance can be achieved by moving the centre of the object, which is the central mass of the object, to the geometric centre of the image. For an image with $N \times N$ size placed in a two-dimensional space, the geometric centre is expressed by the equation [43-45]:

$$x' = \frac{n_{10}}{n_{00}}, \qquad y' = \frac{n_{01}}{n_{00}}$$

where,

$$n_{k,h} = \sum_{k=0}^{N-1} \sum_{h=0}^{N-1} x^k \, y^h \, f(x,y)$$

3.4.2 Rotation Invariance

For rotation invariance if the image $f(x,y)$ is rotated by an angle β, the relation between the OFMM of the original image $O'_{k,h}$ i.e. before rotation, and the OFMM of the image after rotation $O_{k,h}$ at an angle β will be [43-45]:

$$O'_{k,h} = O_{k,h} \exp(-\, j\, h\, \beta) \qquad \ldots\ldots \text{(3.4.2.a)}$$

From the above equation it is evident that the magnitude $|\, O_{k,h} \,|$ of OFMM for the underlying image is rotation invariant.

CHAPTER FOUR

Implementation

As explained in chapter one our intelligent Road Sign recognition system consists of four basic stages having subparts. These basic stages are:

- Image Acquisition.
- Image Pre-processing.
- Image Normalization.
- Image Recognition.

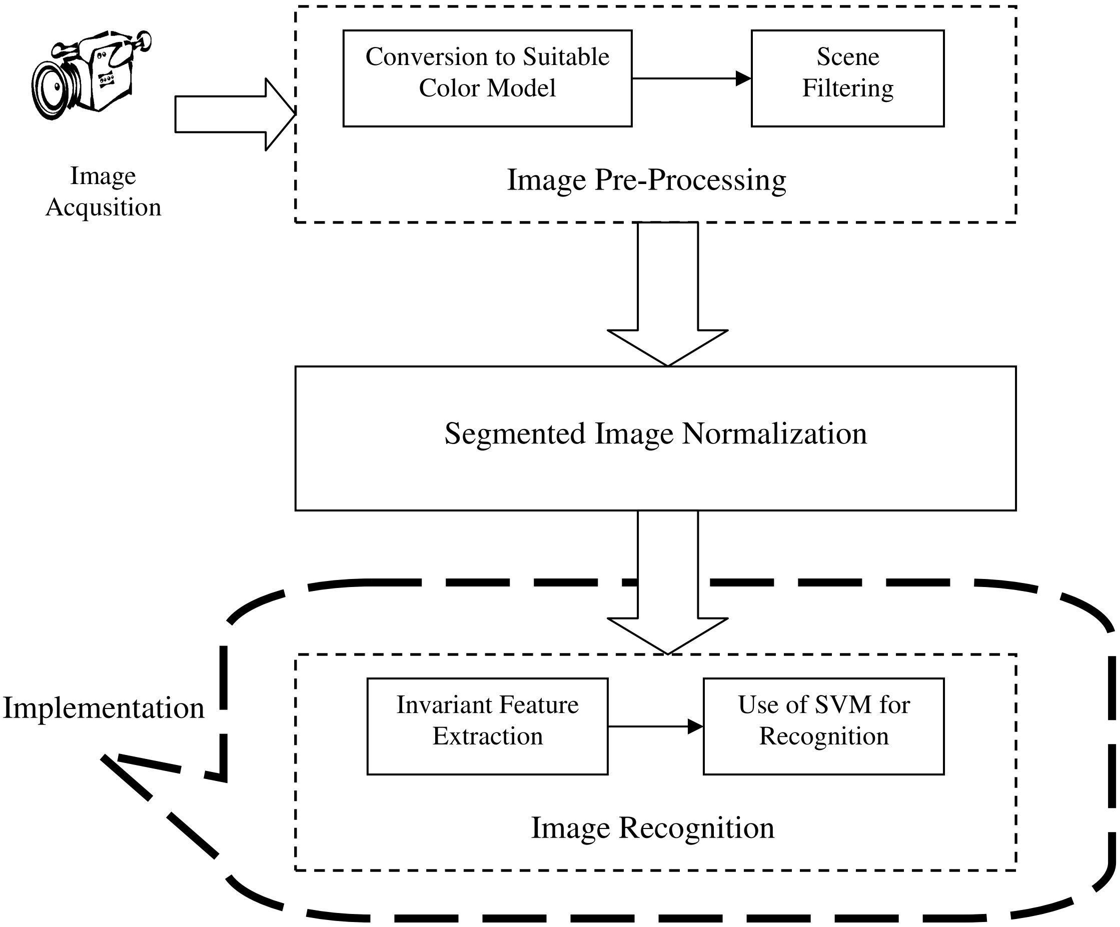


Image Acqusition
Conversion to Suitable Color Model
Scene Filtering
Image Pre-Processing
Segmented Image Normalization
Implementation
Invariant Feature Extraction
Use of SVM for Recognition
Image Recognition

Figure 4.1: Block Diagram of a Road Sign Detection system

Our aim is on formulating an efficient system that can produce best result in real time. In any vision system recognizing the object of interest from the captured image is a very important and complicated task, especially if the system is designed for safety purposes. In a Driver Support System (DSS) recognizing the captured road sign has the highest priority. False interpretation of sign could lead to devastating results so it's very important for the system to either correctly identify the sign or say nothing.
Several techniques have been used for object recognition using neural networks and fuzzy ART-map. Our focus is on extracting object features which are invariant upon transformations like translation, rotation and scaling and then using Support Vector machine to perform the classification task. Following four methods for extracting objects invariant features were used:

- Haar Transform.
- Effective Fourier Transform Coefficients.
- Geometric Descriptors.
- Orthogonal Fourier-Mellin Moments.

4.1 Haar Features

Calculating invariance using haar features we follow the technique proposed by *"Lokesh Setia"*. The technique is very simple and useful in extracting effect invariant features from a binary image. He proposed the use binary operators as the kernel function for calculating features based on neighbouring pixels.

A two-point kernel evaluated at point $(x,\ y)$ would be:

$$k(x,\ y) = G(x,\ y)\ \Theta\ G(x+\Delta x,\ y+\Delta y) \quad \text{....... (4.1.a)}$$

where,
Θ defines any binary operator
and $(\Delta x,\ \Delta y)$ defines the support of neighbouring pixels

The haar invariant features of binary road signs are calculated using the following equation:

$$C_i = \frac{1}{NN}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} XOR\left[G(x,\ y),\ G(x+\Delta^i x,\ y+\Delta^i y)\right] \quad \text{....... (4.1.b)}$$

where superscript *i* defines the range of neighbouring pixels along *x and y* directions. We restricted ourselves to 35 features, differing in the range of neighbouring pixel support. The scale for calculating these 35 features is from $0 \rightarrow 5$, with values (0, 1), (1, 0), (1, 1), (0, 2) and so on to (5, 5).

Pseudo Code:

```
- Prompting user for location of road sign image list.
- Prompting user for location to save output file.
- Sorting image samples from each category for further processing.
    - Thresholding image to remove noise.
- Calculating Haar Features for each image.
- Saving invariant features of each image in the output file specified by user.
```

Flow Chart:

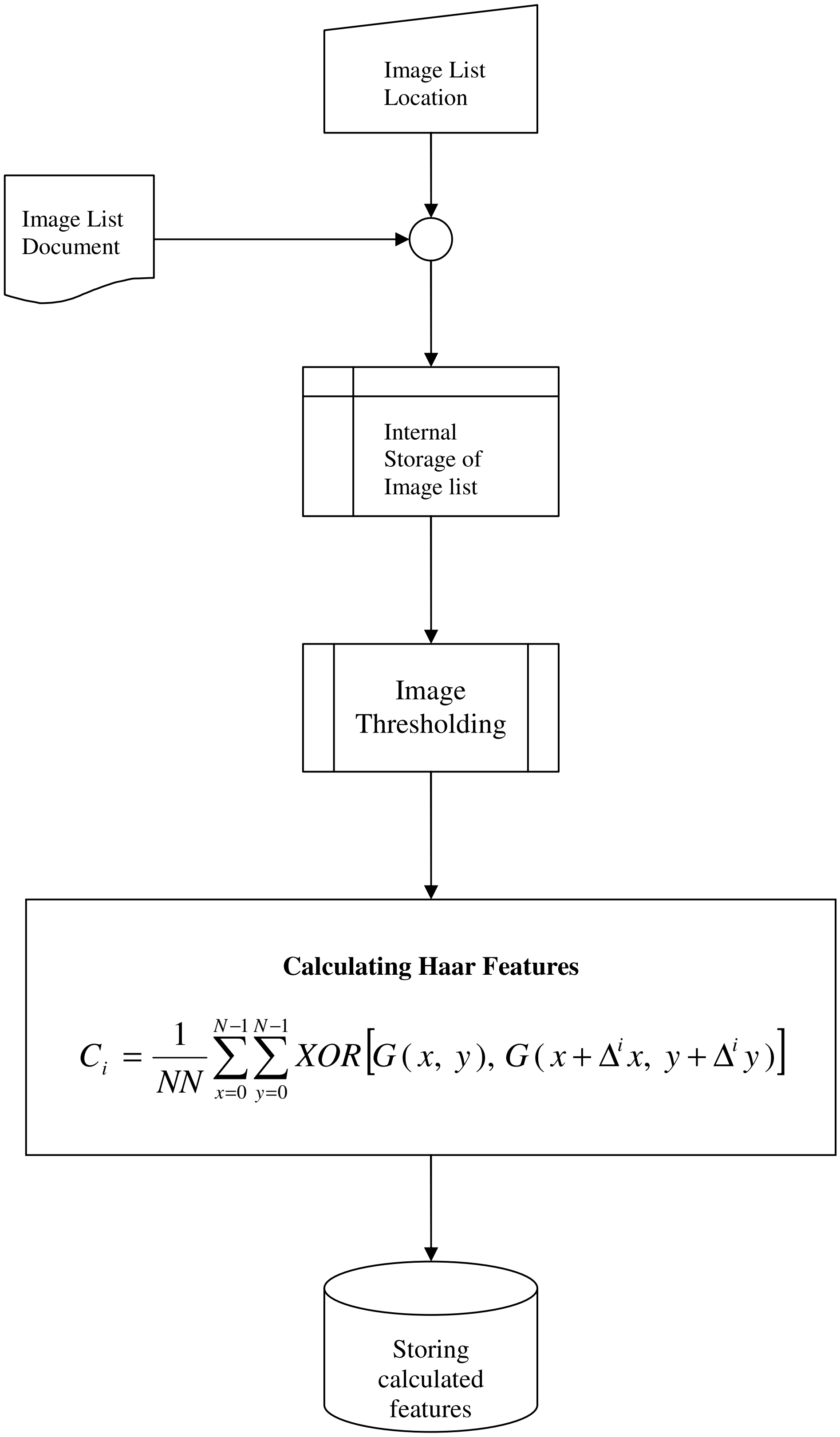
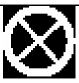
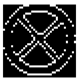
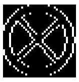
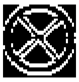
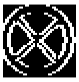
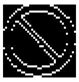
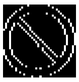
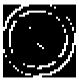
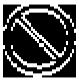
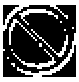
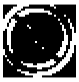
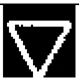
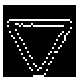
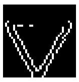
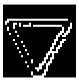
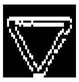
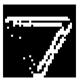
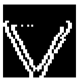
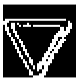
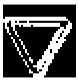
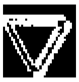
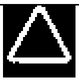
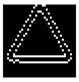
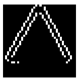
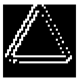
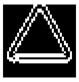
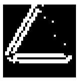
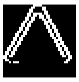
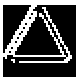
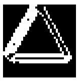
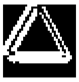
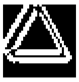
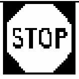
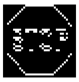
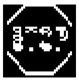
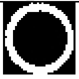
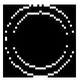
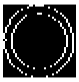
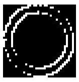
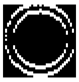
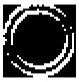
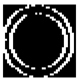
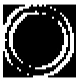
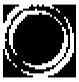
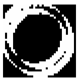
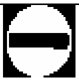
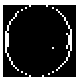
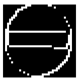
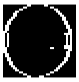
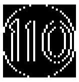
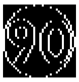
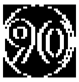
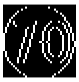
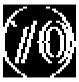
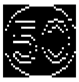
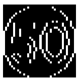
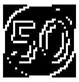
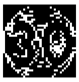
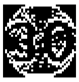
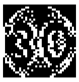


Image List Location
Image List Document
Internal Storage of Image list
Image Thresholding
Calculating Haar Features
$C_i = \frac{1}{NN}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} XOR\left[G(x,\ y),\ G(x+\Delta^i x,\ y+\Delta^i y)\right]$
Storing calculated features

Few of the calculated feature images are shown in the table below:

Original Image	(0, 1)	(1, 0)	(1, 1)	(0, 2)	(1, 2)	(2, 0)	(2, 1)	(2, 2)	(3, 3)	(4, 4)
 rcx_6										
 rcb_50										
 yield_40										
 warning_45										
 stop_49										
 rc_49										
 noentry_44										
 sl110_46										
 sl90_41										
 sl70_50										
 sl50_44										
 sl30_47										

4.2 Calculating Effective FT Coefficients

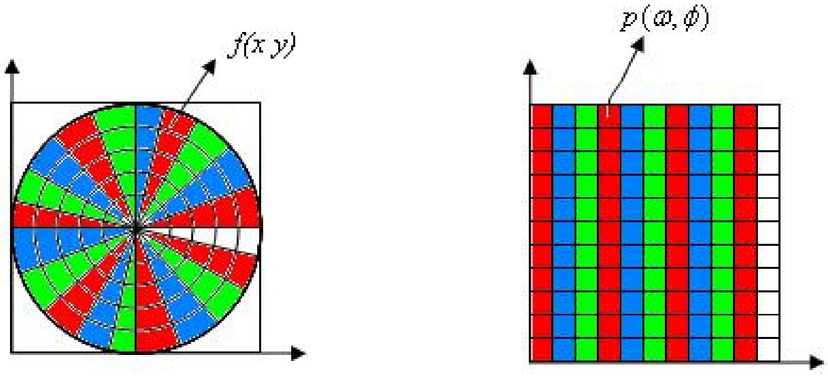
The robust method for calculating FT coefficients proposed by *Shan Li, Chuen Lee and Donald Adjeroh* [38] requires the image to be transformed into its corresponding polar coordinate image.

4.2.1 Polar Transformation

Constructing a polar image $p(\omega,\phi)$ for an image $f(x,y)$ with dimensions $M \times M$, following two steps are required:

- First scan the image N times along angular direction. The pixels are evenly distributed from $0^0 \rightarrow 360^0$. This distribution of pixels defines the resolution of the image. More resolution more detail and more memory space requirement for storing polar image values. Each angular scan gives a row value in polar coordinate.

- In second step scan the image N times along radial direction. The pixels in radial direction are evenly distributed from $0 \rightarrow M/2$ along x-axises and $0 \rightarrow M/2$ along y-axises. Each radial scan gives a column value in polar coordinate.



MxM Size cartesian coordinate with polar grid

NxN Size resulting image

Figure 4.2.1: Conversion of Cartesian coordinate to Polar coordinate

The polar image is computed by using the equation:

$$p(\omega,\phi) = f\left[\left\lfloor \frac{M}{2} + \omega\cos\left(\frac{2\pi\phi}{N}\right)\right\rfloor, \left\lfloor \frac{M}{2} - \omega ain\left(\frac{2\pi\phi}{N}\right)\right\rfloor\right]$$

where,

$\lfloor v \rfloor$ express the largest integer that is not larger than "v"

After transforming the image into its corresponding polar image, take the Fourier transform of the image. The invariant features descriptors of the image are based on both, the FT magnitude and phase. In our experiments we used 23 magnitude features and 22 phase features. The 23 magnitude features contain Four-Radial frequencies and Six-Angular frequencies.

$$Magnitude\,Descriptors = \left\{ \begin{array}{l} \frac{|F(0,1)|}{|F(0,0)|} \cdots \frac{|F(0,5)|}{|F(0,0)|}, \quad \frac{|F(1,0)|}{|F(0,0)|} \cdots \frac{|F(1,5)|}{|F(0,0)|} \\ \frac{|F(2,0)|}{|F(0,0)|} \cdots \frac{|F(2,5)|}{|F(0,0)|}, \quad \frac{|F(3,0)|}{|F(0,0)|} \cdots \frac{|F(3\ 5)|}{|F(0,0)|} \end{array} \right\}$$

where,

$|F(0,0)|$ is the dc component and is used for normalization of magnitude features.

In case of phase descriptors 22 values were selected. Among these 22 phase features there are Four-Radial frequencies and Six-Angular frequencies, just like in case of magnitude features but in phase descriptors we don't use $\phi(1, 1)$ it will be zero after phase correction, and $\phi(0, 0)$ is the dc component.

$$Phase\,Descriptors = \left\{ \begin{array}{ll} \phi(0,1)\ldots\ldots\phi(0,5), & \phi(1,0),\ \phi(1,2)\ldots\ldots\phi(1,5) \\ \phi(2,0)\ldots\ldots\phi(2,5), & \quad \phi(3,0)\ldots\ldots\phi(3,5) \end{array} \right\}$$

Pseudo Code:

```
    - Prompting user for location of road sign image list.
    - Prompting user for location to save output file.
    - Sorting image samples from each category for further processing.
        - Transformation of image from cartesian coordinate to corresponding polar
        coordinate.
        - Performing padding on new polar image.
                - Calculating Fourier Transform from polar image.
    - Calculating Magnitude Frequency Descriptors.
    - Calculating Phase Frequency Descriptors.
    - Saving invariant frequency features of each image in the output file specified by user.
```

Flow Chart:

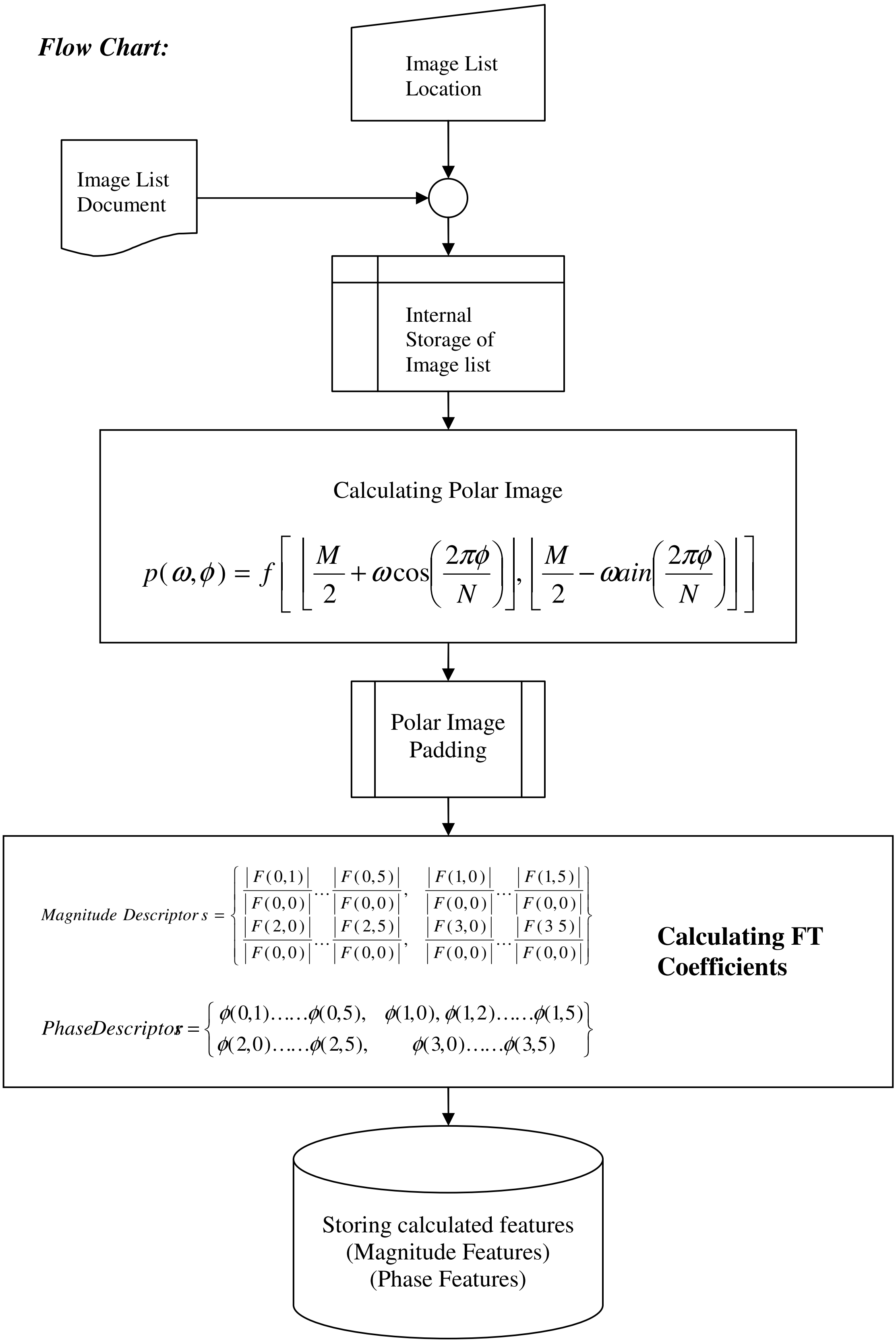


Image List Location
Image List Document
Internal Storage of Image list
Calculating Polar Image
$p(\omega,\phi) = f\left[\left\lfloor \frac{M}{2} + \omega\cos\left(\frac{2\pi\phi}{N}\right)\right\rfloor, \left\lfloor \frac{M}{2} - \omega ain\left(\frac{2\pi\phi}{N}\right)\right\rfloor\right]$
Polar Image Padding
$Magnitude\ Descriptor\,s = \left\{ \begin{matrix} \frac{|F(0,1)|}{|F(0,0)|} \cdots \frac{|F(0,5)|}{|F(0,0)|}, & \frac{|F(1,0)|}{|F(0,0)|} \cdots \frac{|F(1,5)|}{|F(0,0)|} \\ \frac{|F(2,0)|}{|F(0,0)|} \cdots \frac{|F(2,5)|}{|F(0,0)|}, & \frac{|F(3,0)|}{|F(0,0)|} \cdots \frac{|F(3\ 5)|}{|F(0,0)|} \end{matrix} \right\}$
$PhaseDescriptor\,s = \left\{ \begin{matrix} \phi(0,1)\ldots\ldots\phi(0,5), & \phi(1,0), \phi(1,2)\ldots\ldots\phi(1,5) \\ \phi(2,0)\ldots\ldots\phi(2,5), & \phi(3,0)\ldots\ldots\phi(3,5) \end{matrix} \right\}$
Calculating FT Coefficients
Storing calculated features (Magnitude Features) (Phase Features)

4.3 Calculating Geometric Moments

A Two-Dimensional geometric moment of order $(n+m)$ for an image with function $f(x, y)$ is described as:

$$g_{n,m} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^n \; y^m \; f(x, y)\, dx\, dy \qquad \ldots\ldots\ (4.3.a)$$

The Discrete form of this equation for an $(N \times N)$ image will be:

$$G_{n,m} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} x^n \; y^m \; f(x, y)\, \Delta x\, \Delta y \qquad \ldots\ldots\ (4.3.b)$$

where

n and m are integer values $0, +1, +2, \ldots$

From the above two equations it can be seen that the monomial product $x^n \; y^m$ is the basis function indicating that geometric moments are not orthogonal since the basis function is not orthogonal.

By definition, the zero order moments i.e. m_{00} defines the total intensity of the image. The first order moment functions i.e. m_{01} & m_{10} provide the intensity of the image about its X-Axis's and Y-Axis's respectively. The intensity centroid $(x_0,\ y_0)$ of the image is calculated using the zero order and first order moments:

$$\Rightarrow\ x_0 = \frac{m_{10}}{m_{00}}\ ,\quad y_0 = \frac{m_{01}}{m_{00}}$$

These central moments are translation invariant because the image centroid $(x_0,\ y_0)$ moves with the image under transformation group. So first objects centroid is computed and shifted to the origin of the reference system such that the intensity centroid coincides with the origin of the reference system.

Then to compute rotation invariant features second order moments m_{02} & m_{20} are used. This is shown in equation 3.2.3.a and 3.2.3.b that the two moments are invariant to any angle of rotation.

Pseudo Code:

```
- Prompting user for location of road sign image list.
- Prompting user for location to save output file.
- Sorting image samples from each category for further processing.
    - Calculating object centroid.
    - Shifting centroid
- Calculating Geometric moments.
- Saving invariant frequency features of each image in the output file specified by user.
```

Flow Chart:

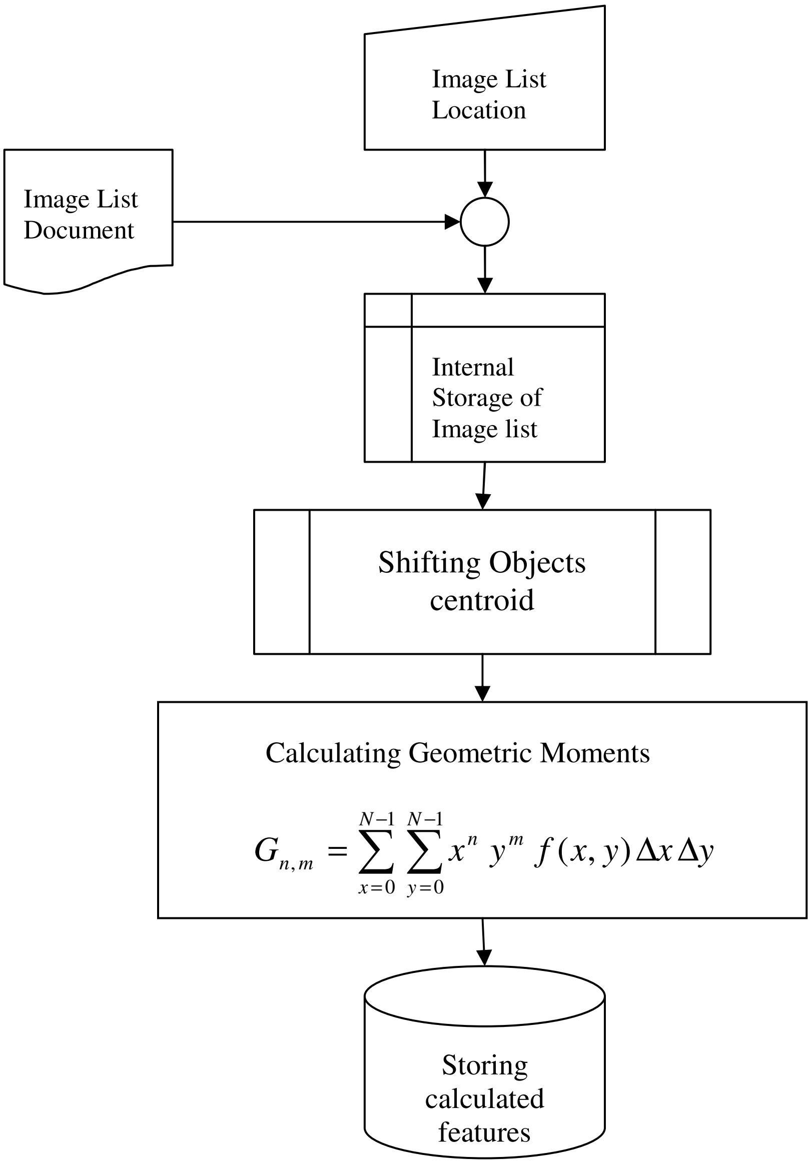


Image List Location
Image List Document
Internal Storage of Image list
Shifting Objects centroid
Calculating Geometric Moments
$G_{n,m} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} x^n \, y^m \, f(x,y) \, \Delta x \, \Delta y$
Storing calculated features

4.4 Calculating Orthogonal Fourier-Mellin Transform

Advantage of using orthogonal function for calculating invariant features yields additional independent object shape descriptors. They can increase content information and minimize information redundancy among a set of descriptors. For calculating orthogonal fourier-mellin transform following equation was used:

$$\mathrm{O}_{k,h} = \frac{k+1}{\pi} \sum_x \sum_y f(x,y) Q_k(r) \exp(-jh\varphi) \Delta x \Delta y \quad \ldots\ldots\ (4.4.b)$$

The equation still satisfies the statement that $Q_k(r)$ is orthogonal in Cartesian coordinates as:

$$x^2 + y^2 \leq 1, \qquad x = r\cos\varphi, \qquad y = r\sin\varphi$$

indicating that the object is inside the unit circle. $Q_k(r)$ is the polynomial in r of degree k. This polynomial is defined as:

$$Q_k(r) = \sum_{i=0}^{k} \alpha_{k,i}\, r^i$$

where

$$\alpha_{k,i} = \frac{(-1)\ (k+i+1)!}{(k-i)!\, i!\, (i+1)!}$$

Pseudo Code:

```
- Prompting user for location of road sign image list.
- Prompting user for location to save output file.
- Sorting image samples from each category for further processing.
    - Calculating object centroid.
    - Defining boundary circle.
    - Allocating temporary storage place for new object position.
    - Shifting objects centre to image plane centre.
            - Calculating corresponding polar image of new image.
- Calculating $Q_k(r)$.
- Calculating Orthogonal Fourier-Mellin Moments..
- Saving invariant frequency features of each image in the output file specified by user.
```

Flow Chart:

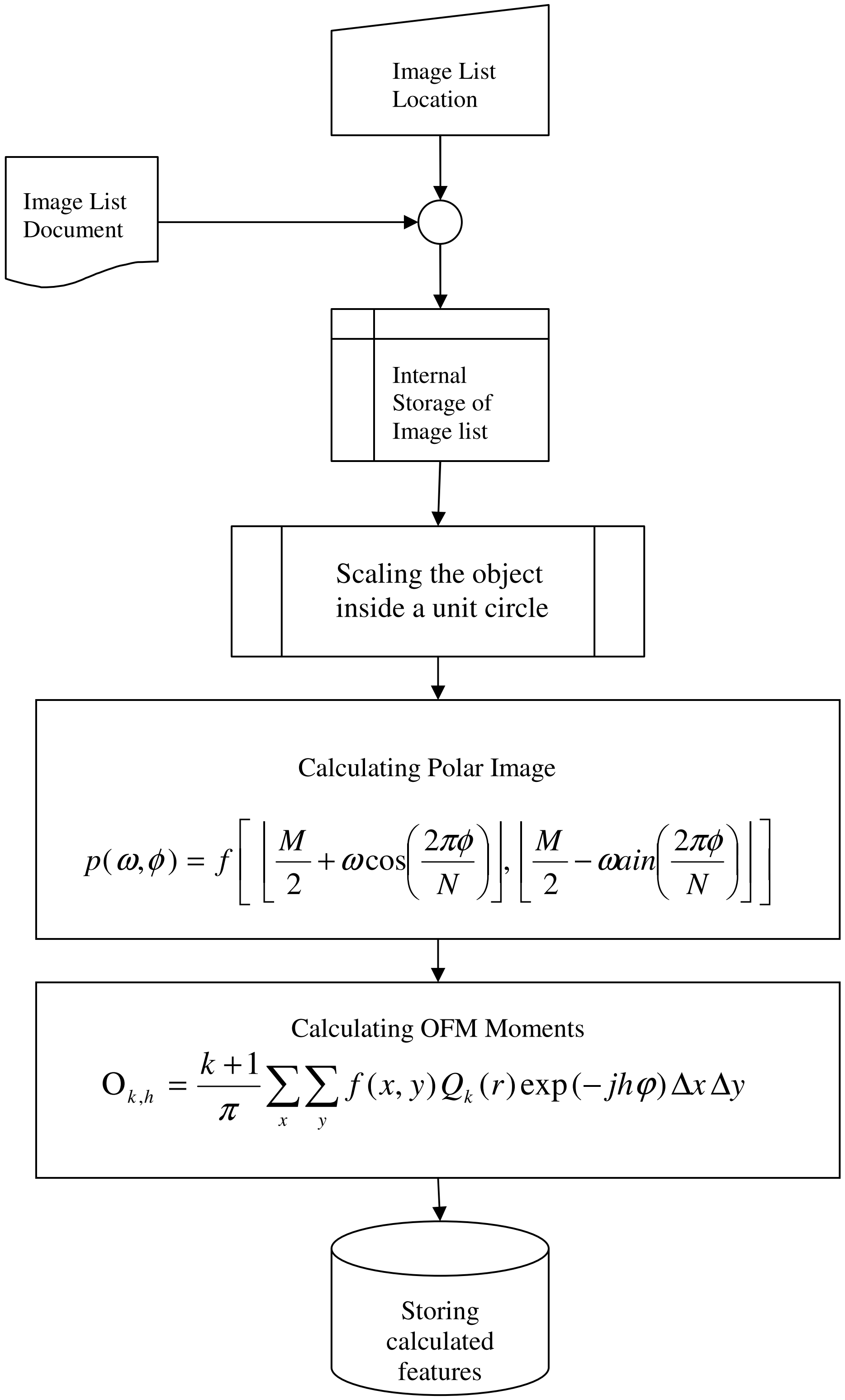


Image List Location
Image List Document
Internal Storage of Image list
Scaling the object inside a unit circle
Calculating Polar Image
$p(\omega,\phi) = f\left[\left\lfloor \frac{M}{2} + \omega\cos\left(\frac{2\pi\phi}{N}\right)\right\rfloor, \left\lfloor \frac{M}{2} - \omega ain\left(\frac{2\pi\phi}{N}\right)\right\rfloor\right]$
Calculating OFM Moments
$\mathrm{O}_{k,h} = \frac{k+1}{\pi}\sum_{x}\sum_{y} f(x,y)\,Q_k(r)\exp(-jh\varphi)\,\Delta x\,\Delta y$
Storing calculated features

CHAPTER FIVE

Result Analysis

Success of our method does not only depend upon the extraction of important invariant features but also on the selection of SVM parameters and kernel function. The chosen SVM type is C-SVM with Linear classifier as the kernel function.

The system is initiated by first normalizing the data to specific scale (range). The vector machine then separates the data into two sets:

- Training set.
- Testing set.

The random data distribution ratio is 7:3 with 70% data, 35 samples, used in training vector machine and 30%, 15 samples, as testing set. The C and gamma parameters were experimentally selected.

Following results were achieved with different feature extraction methods. A more analytical analysis is given in the following sections.

Feature Extraction Method	**Shape Recognition Accuracy**	**Speed-limit Recognition Accuracy**
Haar Features	97.77%	96.00%
Effective FT Coefficients	99.04%	90.67%
Orthogonal Fourier Mellin	92.22%	50.67%
Geometric Moments	92.22%	62.67%

A confusion matrix is used to tabulate the results. A template of confusion matrix is shown below:

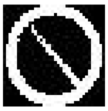
<table>
<tr><th rowspan="2">Desired Output</th><th colspan="6">Classified As</th><th rowspan="2">Total</th></tr>
<tr><td colspan="6">Number of Classes</td></tr>
<tr><td rowspan="6">Number of Classes</td><td></td><td></td><td></td><td></td><td></td><td></td><td rowspan="6">Total number of samples in each class</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td><b>Total</b></td><td colspan="6">Total number of images classified as without taking into account if its right or wrong</td><td></td></tr>
</table>

5.1: Haar Features

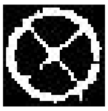
Haar features were easy to implement with no preliminary conversion to polar coordinates or computation of Fourier transform. Instead it only uses a binary operator for extracting invariant features with support form neighbouring pixels in the image. With only binary operation performed between pixels the process is also quite fast. The method produced some very promising results:

5.1.1: Analysis of borders with Haar features

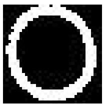
A confusion matrix table is constructed for easy interpretation of results. The table shows that only two images; RCB and RCX, were incorrectly classified as RC.



RCB



RCX



RC

NOTE: the images are presented for analysis and should not be considered as specifically the misclassified ones.

From the images it can be seen that they have same shape contour but different inner information. Since the shape contour is more prominent so probably that's why the two images were misclassified. Beside these two images all other images were correctly classified.

The classification accuracy was 97.77%

Table 5.1: Confusion matrix of test set of Borders with Haar features.

Desired Output	Classified As						Total
	NOE	STP	RC	TRI	RCB	RCX	
NOE	**15**						15
STP		**15**					15
RC			**15**				15
TRI				**15**			15
RCB			1		**14**		15
RCX			1			**14**	15
Total	15	15	17	15	14	14	

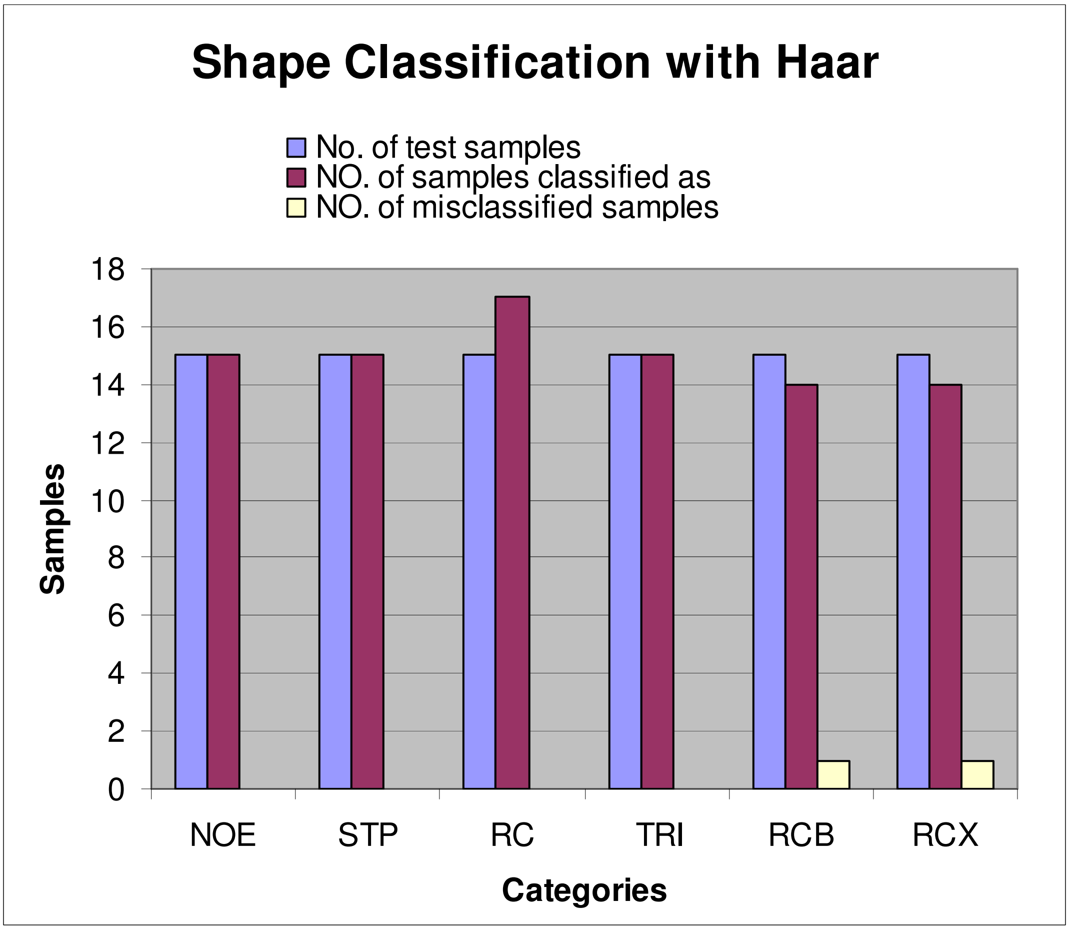


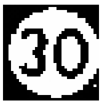
Figure 5.1(a): Shape classification with Haar features

5.1.2: Analysis of speed-limit with Haar features

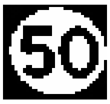
There were three misclassifications reported by SVM. The overall accuracy of the system over 15 testing samples (30% of total samples) was 96.0% which is very good. The misclassified signs were:

- 1 SL30 sign: classified as SL70.
- 2 SL50 sings: 1 classified as SL30 and 1 as SL110.

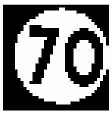
There is not much difference in speed-limit signs. All have circular shape with either two or three digits inside. There are also signs with one digit but they are not frequently used. In our experiments we have taken signs with two digits, i.e. 30, 50, 70, 90 and 110.



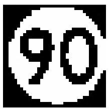
SL30



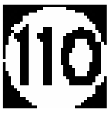
SL50



SL70



SL90



SL110

A look at these images shows that the only difference they have is 3, 5, 7, 9, and 11. These digits look more a like and with so much likeliness using invariant features for there classification could be not good enough.

Haar invariant features however came to be very successful method in extracting features which quite distinguishable. With only 3 misclassifications the system produced 96.0% accuracy.

Table 5.2: Confusion matrix of test set of Speed-Limit with Haar Features.

Desired Output	Classified As					Total
	SL30	SL50	SL70	SL90	SL110	
SL30	**14**		1			15
SL50	1	**13**			1	15
SL70			**15**			15
SL90				**15**		15
SL110					**15**	15
Total	15	13	15	15	16	

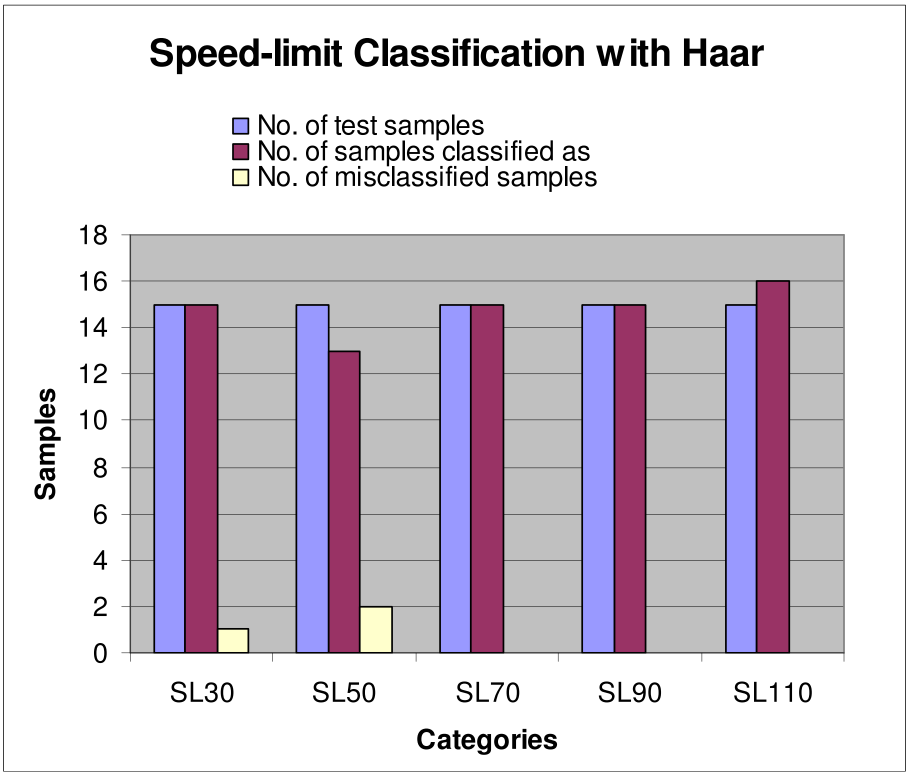


Figure 5.1(b): Speed-limit classification with Haar features

5.2: Effective Fourier Transform Features

The classification result with image features calculated using effective Fourier coefficients were very good with 99.04% accuracy on shape signs and 90.67% accuracy on speed-limit signs. The method is a bit time consuming with first conversion of image into polar image, then image padding, fourier transform and then extraction of magnitude and phase coefficients.

5.2.1: Analysis of borders with FT Coefficients

In shape signs images there was only one image that was misclassified by the SVM.

- STOP sign misclassified as RC.

Since it's the only sample that is misclassified we can't be sure what could be the reason. However one possible reason could be the edges of the STOP. Its possible because of bad conditions or damaged sign that the shape of STOP sign looks more like a circle then an Octagon. Under these conditions we need to use the information that is inside the circle.

Beside this sample the SVM correctly classified every other sample which is a very promising.

Table 5.3: Confusion matrix of test set of Borders with Effective FT Features.

Desired Output	Classified As						Total
	NOE	STP	RC	TRI	RCB	RCX	
NOE	**15**						15
STP		**14**	1				15
RC			**15**				15
TRI				**15**			15
RCB					**15**		15
RCX						**15**	15
Total	15	15	16	15	15	15	

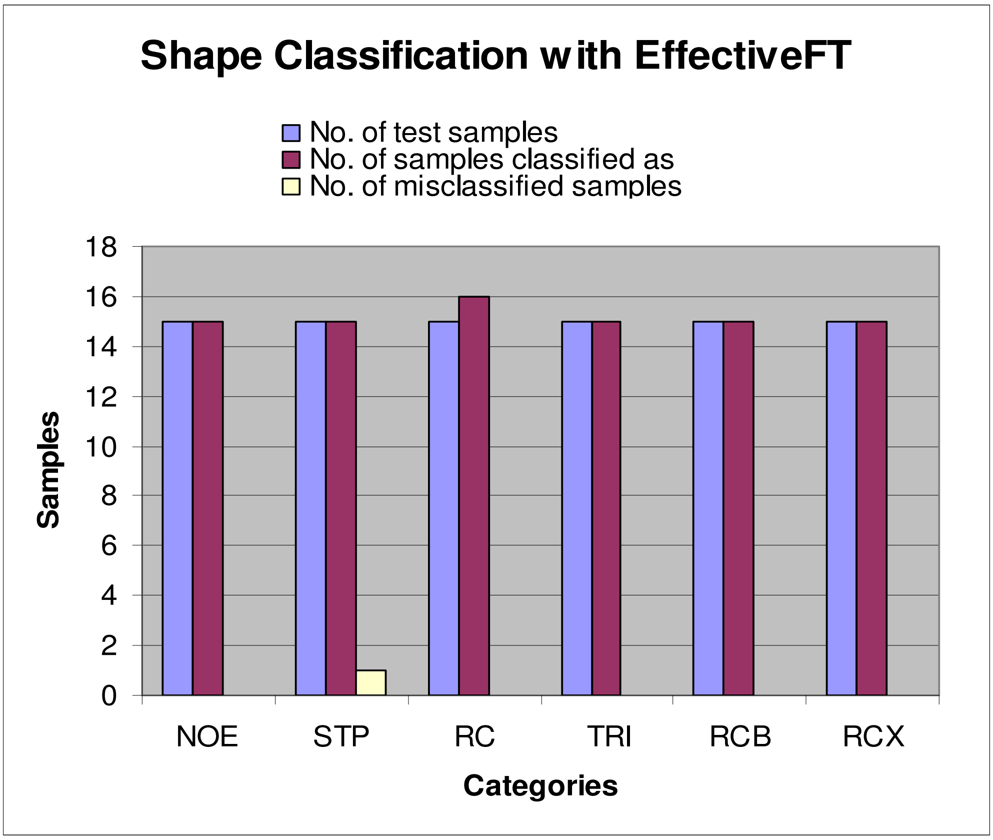


Figure 5.2(a): Shape classification with Effective FT Coefficients

5.2.2: Analysis of speed-limit with FT Coefficients

In case of recognizing speed-limit signs the SVM produced 90.67% accuracy with 7 misclassifications. Despite 7 misclassifications it's still a very good method for extracting invariant features as the images are very similar and also the misclassified images are uniformly distributed. This can be seen from the data in the last row. The misclassified samples are:

- 3 SL30 signs misclassified as SL110.
- 1 SL90 sign misclassified as SL110.
- 3 SL110 signs: 1 misclassified as SL70 and 2 as SL30.

Table 5.4: Confusion matrix of test set of Speed-Limit with Effective FT Features

Desired Output	Classified As					Total
	SL30	SL50	SL70	SL90	SL110	
SL30	**12**				3	15
SL50		**15**				15
SL70			**15**			15
SL90				**14**	1	15
SL110	2		1		**12**	15
Total	14	15	16	14	16	

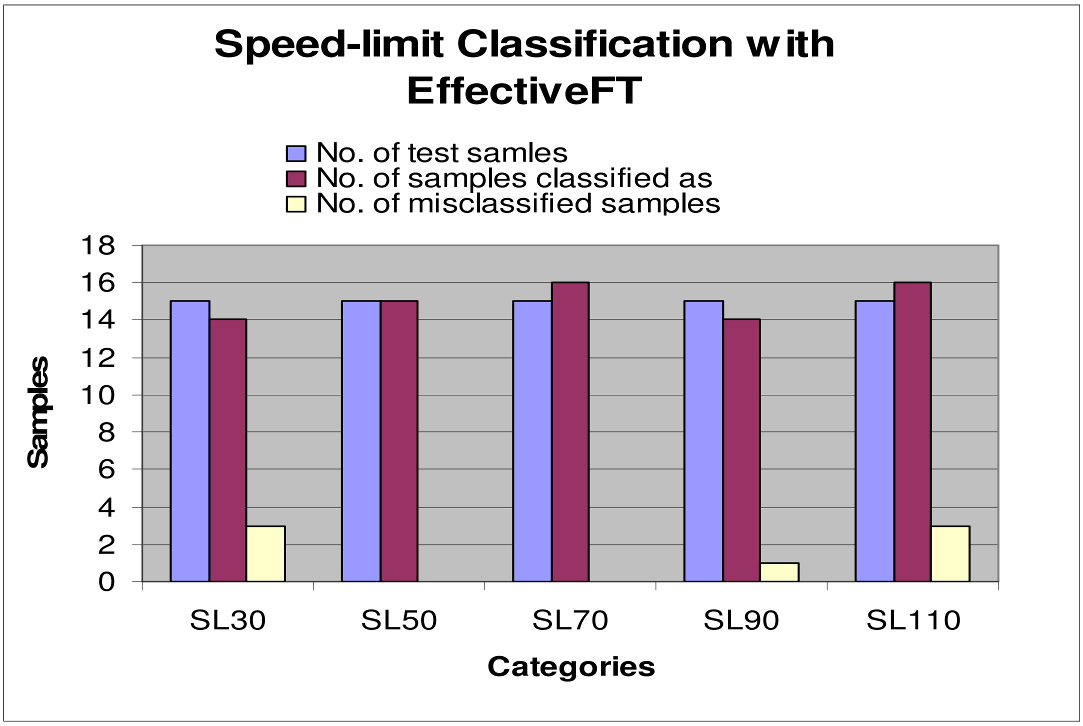


Figure 5.2(b): Speed-limit classification with Effective FT Coefficients

5.3: Geometric Moments

There are different types of geometric moments based on the condition they are used. Standard moments were used for extracting translation and rotation invariant features form the gray road sign images under analysis.

5.3.1: Analysis of borders with Geometric Moments

In shape classification, based on the features extracted from geometric moments, the SVM successfully classified 83 images out of 90 and misclassified only 7. The accuracy of the system is pretty much the same as provided with OFMM however the misclassified images are not the same.

Table 5.5: Confusion matrix of test set of Borders with Geometric moments.

Desired Output	Classified As						Total
	NOE	STP	RC	TRI	RCB	RCX	
NOE	**15**						15
STP	1	**14**					15
RC			**15**				15
TRI				**15**			15
RCB					**12**	3	15
RCX			1		2	**12**	15
Total	16	14	16	15	14	15	

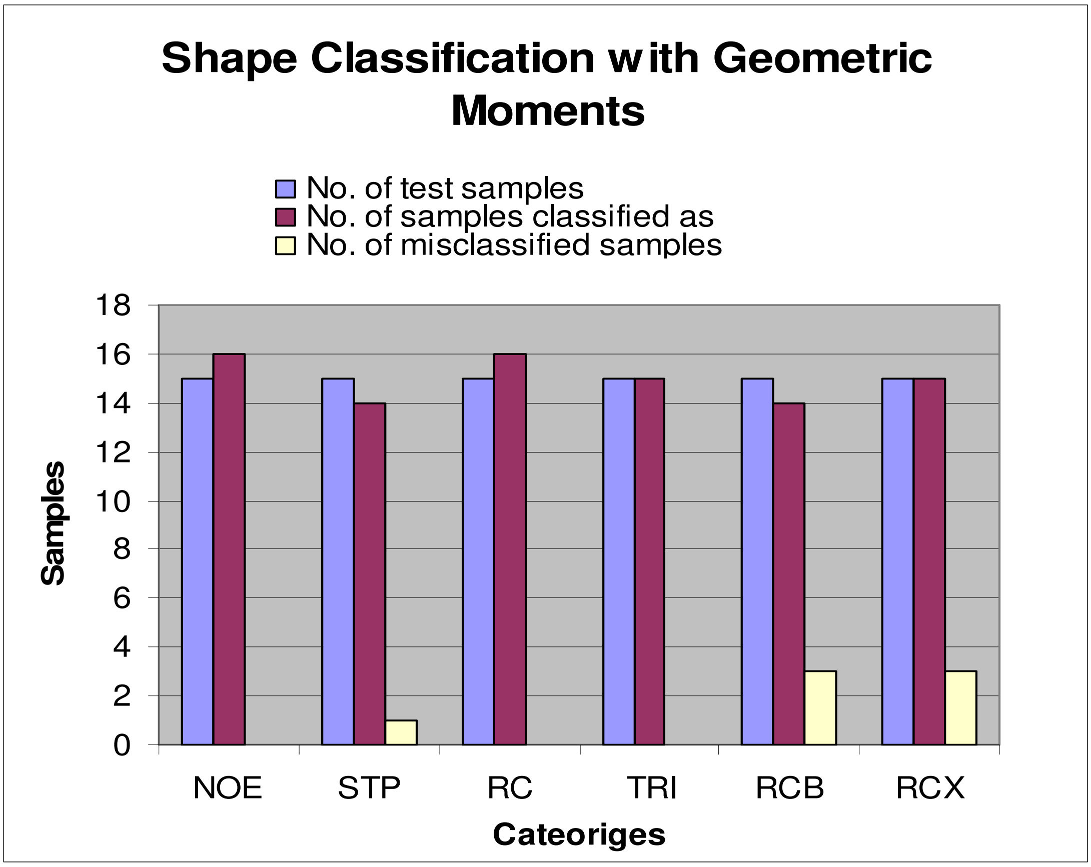


Figure 5.3(a): Shape classification with Geometric Moments

5.3.2: Analysis of speed-limit with Geometric Moments

The system classification accuracy was 62.67% with 28 images being misclassified. Most of the images were classified as SL50.

Table 5.6: Confusion matrix of test set of Speed-Limit with Geometric moments.

Desired Output	Classified As					Total
	SL30	SL50	SL70	SL90	SL110	
SL30	**11**	3			1	15
SL50		**13**		2		15
SL70	2		**12**	1		15
SL90	1	7		**7**		15
SL110	3	5	1	2	**4**	15
Total	17	28	13	12	5	

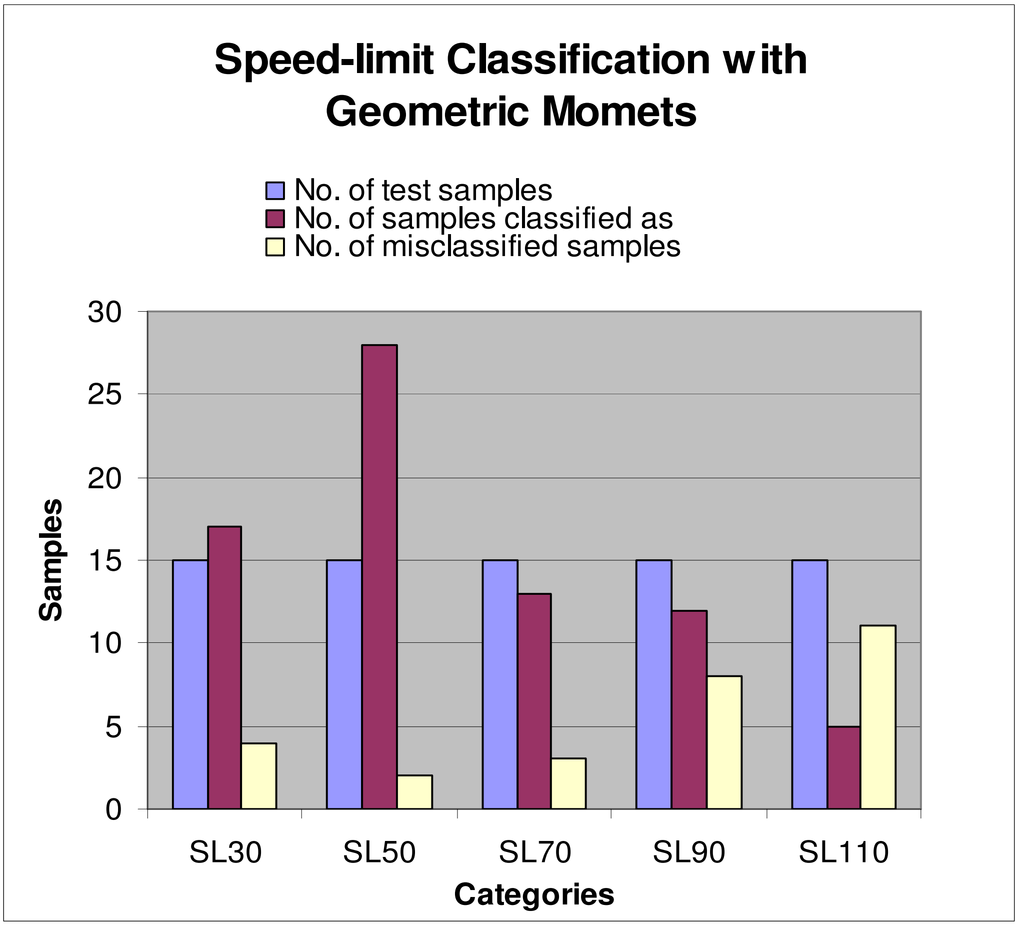


Figure 5.3(b): Speed-limit classification with Geometric Moments

5.4: Orthogonal Fourier Mellin Moments

This method produced good results on shape classification with 92.22% accuracy by the vector machine but in case of classifying speed-limit the vector machine was only able to classify 38 images out of 75 with an accuracy of almost 50%. The method also takes fair amount of time in computing the features as it requires all the steps that are used by Effective FT Coefficient along with some addition pre-processing tasks, i.e. calculating factorials, coordinate scaling such that the object lies inside the unit circle with its centre coinciding with the centre of unit circle.

5.4.1: Analysis of borders with OFM Moments

The vector machine classified 83 images correctly with 7 misclassifications. The misclassifications were uniform and were not concentrated to any specific image. The misclassified images are:

- 1 No-Entry sign misclassified as RCB.
- 1 RC sign misclassified as RCB.
- 2 RCB signs: 1 misclassified as RC and 1 as STOP.
- 3 RCX signs: 2 misclassified as RCB and 1 as RC.

Table 5.7: Confusion matrix of test set of Borders with OFMM moments.

Desired Output	Classified As						Total
	NOE	STP	RC	TRI	RCB	RCX	
NOE	**14**				1		15
STP		**15**					15
RC			**14**		1		15
TRI				**15**			15
RCB		1	1		**13**		15
RCX			1		2	**12**	15
Total	14	16	16	15	17	12	

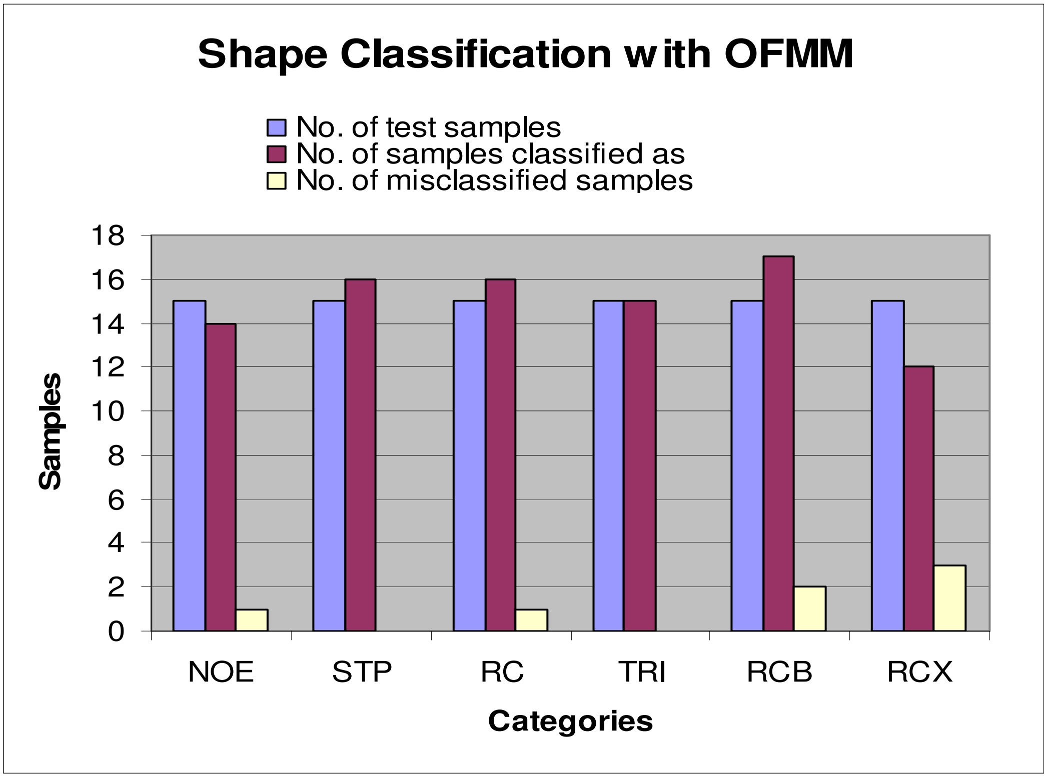


Figure 5.4(a): Shape classification with Orthogonal Fourier Mellin Moments

5.4.2: Analysis of speed-limit with OFM Moments

In case of speed-limit an accuracy of 50.67% was achieved with 37 misclassifications. Out of these 37 misclassified samples 89% were classified as SL70. Only two and one image was classified as SL30 and SL50, respectively. Not only the classification ratio is low but also the misclassified signs are concentrated to one sign i.e. classified as SL70.

Table 5.8: Confusion matrix of test set of Speed-Limit with OFMM moments.

Desired Output	Classified As					Total
	SL30	SL50	SL70	SL90	SL110	
SL30	**2**		11	2		15
SL50		**1**	13	1		15
SL70			**15**			15
SL90			5	**9**	1	15
SL110			4		**11**	15
Total	2	1	48	12	12	

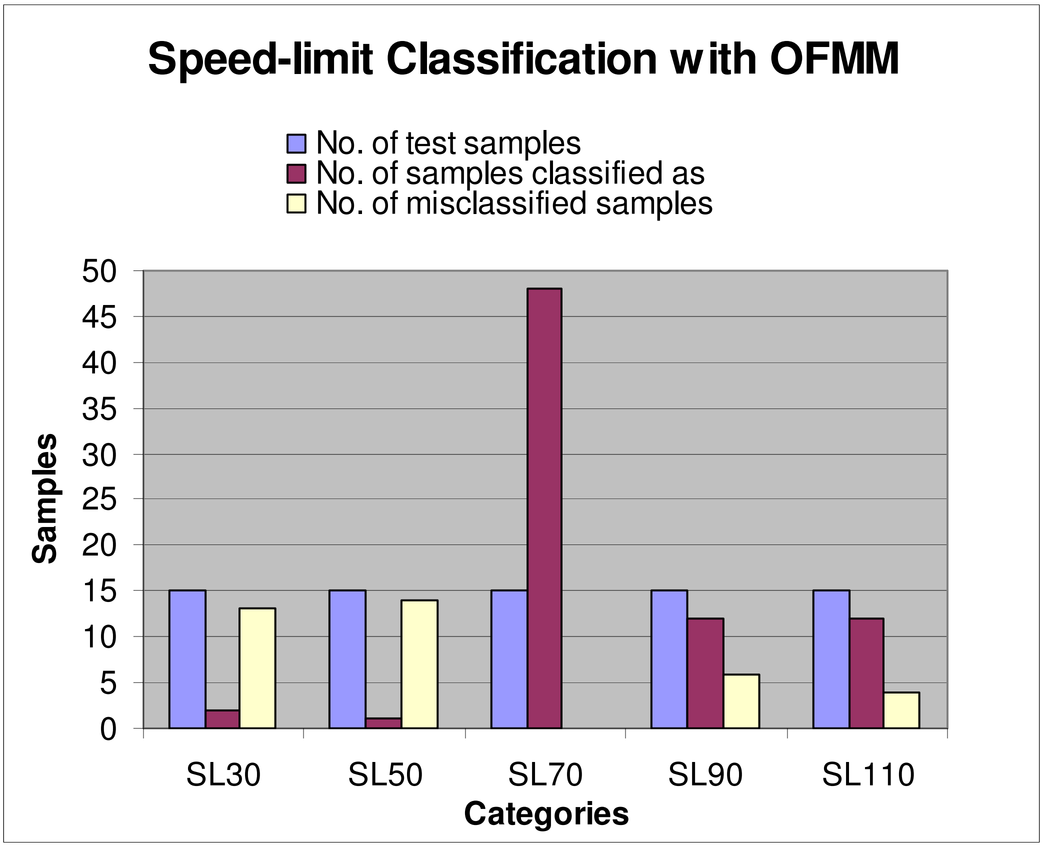
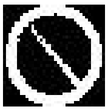


Figure 5.4(b): Speed-limit classification with Orthogonal Fourier Mellin Moments

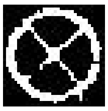
CHAPTER SIX

Future Work

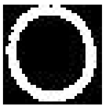
Most shape images that were misclassified had similarity on outer boundary circle with slight difference in inner contents, as evident from figures shown.



RCB



RCX

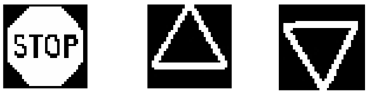


RC

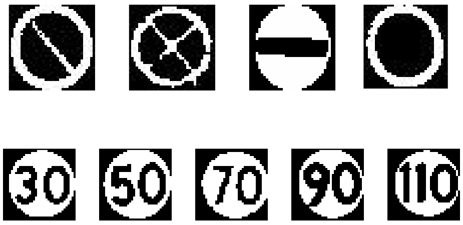
In future we will merged these images during pre-classification and then a cascaded subsystem will be used to perform further categorization based on information inside the boundary circle.

An increase in number of samples in each category is required. Currently only 50 samples per category were used with 35 samples used for training and 15 samples for testing. Increasing the number of training set would better the performance of vector machine and will result in much less false alarms.

Another direction would be to make the vector machine realize the difference between signs with information and sign with speed limit. If we look at the images used in our research work we will see that there are only 3 signs that have different boundary shape; Stop sign, Warning Sign and Yield Sign,



and 9 signs with similar shape. It will be interesting to see if the vector machine could categorize them efficiently.



Images under analysis had size 36x36 but in future we will increase the size of these images to 48x48. This increase in size will result in more information contained by the images. Although this increase in size will result in an increase in computation time but we will be able to extract more detail information that will help the vector machine to perform classification more efficiently.

REFERENCES

[1] Y. Z. Claw Bahlmann, Visvanathan Ramesh, Martin Pellkofert, and Thorstea Koehled, "A System for Traffic Sign Detection, Tracking, and Recognition Using Color, Shape, and Motion Information," *Intelligent Vehicles Symposium, Proceedings, IEEE*, pp. 255 - 260, 2005.

[2] L. P. a. D. S. X.W.Gao, "Application of Vision models to Traffic Sign Recognition."

[3] H. Fleyeh, "Road and Traffic Sign Color Detection and Segmentation - A Fuzzy Approach," 2005.

[4] "Motion Trends."

[5] P. Paclik, "Road Sign recognition group."

[6] C.-L. L. Hsiu-Ming Yang, Kun-Hao Liu, and Shang-Ming Huang, "Traffic Sign Recognition in Disturbing Environments."

[7] "Swedish National Road Administration."

[8] "Swedish road signs, signals, road markings and signals by policemen," in *Swedish National Road Administration*, 2003-05.

[9] "http://www.madehow.com/Volume-2/Road-Sign.html," in *How Products Are Made*.

[10] S. O. Gilani, "Swedish Speed-limit Sign Detection and Recognition Using Computer Vision Techniques and Fuzzy ARTMAP," August 2005.

[11] H. F. a. M. Dougherty, "Road Sign Detection and Recognition," presented at IEEE conference on Cybernetics and Intelligent Systems, 2004.

[12] P. V. a. M. J. Jones, "Robust Real Time Object Detection," Cambridge Research Laboratory February 2001.

[13] A. S. a. K. Yamada, "Fast and Robust Traffic Sign Detection," presented at IEEE International Conference on Systems, Man and Cybernetics, October 2005.

[14] Z. S. a. J. Tian-tian, "Intelligent approach of traffic sign recognition based on Color Standardization," presented at IEEE International Conference on Vehicular Electronics and Safety, October 2005.

[15] Z. Y. a. L. X.-f. Zhu Shuang-dong, "Detection for Triangle Traffic Sign Based on Neural network," presented at IEEE International Conference on Vehicular Electronics and Safety, October 2005.

[16] "DARPA Grand Challenge," 2005.

[17] T. K. J. Miura, and Y. Shirai, "An Active Vision System for Real-Time Traffic Sign Recognition," presented at Proceeding IEEE Conference on Intelligent Transportation System, October 2000.

[18] C. S. F. C.Y. Fang, S.W. Chen, P.S. Yen, "A Road Sign Recognition System based on Dynamic Visual Model," presented at IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003.

[19] G. P. S. Vitabile, G. Pilato, E. Sorbello, "Road Sign Recognition using a Dynamic pixel aggregation technique in the HSV color space," presented at Proceeding, 11th International Conference on Image Analysis and Processing Palermo, Italy, September 2001.

[20] H. Fleyeh, "Traffic Signs."

[21] Flickr.com, "Damaged Signs."

[22] M. Aoki, "Image Processing in ITS," presented at IEEE International Conference on Intelligent Vehicles, 1998.

[23] "CLEOPATRRA project."

[24] C.-Y. F. S.-W. C. C.-S. Fuh, "Road-sign detection and tracking," *Vehicular Technology, IEEE Transactions*, vol. 52, pp. 1329, September 2003.

[25] P. M. Doughtery, "ITS Research Platform at Högskolan Dalarna."

[26] M. A. Hearst, "Support Vector Machines," *IEEE INTELLIGENT SYSTEMS*, July/August 1998.

[27] L. G.-C. G. Camps-Valls, J. Calpe-Maravilla, E. Soria-Olivas, J. D. Martín-Guerrero and J. Moreno, *Support Vector Machines for Crop Classification Using Hyperspectral Data*: Springer Berlin / Heidelberg, 2003.

[28] T. Mitchell, "Machine Learning," McGraw Hill, 1997.

[29] T. S. Brain, "I-Swarm Robots," September 2005.

[30] S. Haykin, *Neural Networks: A Comprehensive Foundation*: Prentice Hall, 1999.

[31] B. S. a. A. Smola, "Support Vector Machines, Regularization, Optimization and Beyond," in *Learning with Kernels*: MIT Press, 2001.

[32] M. T. T. s. presentation, "Chapter 7.2."

[33] "Support Vector Machines," Statsoft, 2004.

[34] S. M. Klaus-Robert Muller, Gunnar Rätsch, Koji Tsuda, and Bernhard Schölkopf, "An Introduction to Kernel Based Learning Algorithms," *IEEE Transactions on Neural Networks*, vol. 12, March 2001.

[35] N. C. a. J. Shawe-Taylor, *An Introduction to Support Vector Machines*.

[36] "Non Linear SVM."

[37] H. Schulz-Mirbach, "Invariant Features for Gray Scale Images." Bielefeld: Springer, 1995, pp. 1-14.

[38] D. A. Shan Li, "Effective Invariant Features for Shape-Based Image Retrieval," *Journal American Soc. for Information science and technology*, pp. 729-740, 2005.

[39] J. M. a. F. Thomas, "Efficient Computation of Local Geometric Moments," *IEEE Transactions on Image Processing*, September 2002.

[40] M. K. Hu, "Visual Pattern Recognition by moment Invariants," *IRE Transaction Information Theory*, pp. 179-187, 1962.

[41] H. Y. M. Rizon, P. Saad, Ali, Abdul Rehman, M. Rozailan, Sazali, Hazri, M. Karthigayan, "Object Detection using Geometric Invariant Moment," *American Journal of applied sciences*, pp. 1876-1878, 2006.

[42] "Geometric Moments."

[43] Y. S. a. L. Shen, "Orthogonal Fourier-Mellin moments for Invariant Pattern Recognition," *JOSA*, vol. 11, pp. 1748.

[44] M. D. S. Chao Kan, "Invariant character recognition with Zernike and orthogonal Fourier-Mellin moments," pp. 143-154, 2002.

[45] Y. S. J. Christophe, S. Akamatsu, "Invariant Neural-Network based Face Detection with Orthogonal Fourier-Mellin Moments," ATR Human Information Processing Research Lab. and Center d'Optique, Photonique et Laser, Dep. de Physique.