MOVING VEHICLE CLASSIFICATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

DEMET DUMAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2013

Approval of the thesis:

MOVING VEHICLE CLASSIFICATION

submitted by **DEMET DUMAN** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** ___________

Prof. Dr. Gönül Turhan Sayan
Head of Department, **Electrical and Electronics Engineering** ___________

Prof. Dr. Gözde Bozdağı Akar
Supervisor, **Electrical and Electronics Engineering Dept., METU** ___________

Examining Committee Members:

Prof. Dr. A. Aydın Alatan ______________________
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Gözde Bozdağı Akar ______________________
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Tolga Çiloğlu ______________________
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. İlkay Ulusoy ______________________
Electrical and Electronics Engineering Dept., METU

Dr. Cevahir Çığla ______________________
SST, ASELSAN Inc.

Date: **04/09/2013**

Name, Last name	: Demet DUMAN
Signature	:

ABSTRACT

MOVING VEHICLE CLASSIFICATION

Duman, Demet
M.Sc., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Gözde Bozdağı Akar

September 2013, 78 pages

In recent years intelligent transportation systems have been an active research area in computer vision. The aim of this study is to classify moving vehicles from highway videos taken by stationary uncalibrated cameras. For this study, three types of vehicle classes with different scales are chosen to classify: car, van and truck. The proposed algorithm is composed of foreground/background segmentation, feature extraction and classification steps. In order to classify each vehicle, histogram of oriented gradients (HOG) features which are shape-based descriptors and blob features which are dimension-based descriptors are used in the algorithm. The effects of these features on the classification performance are also evaluated and simulation results are given on different highway videos.

Keywords: Vehicle classification, foreground/background segmentation, histogram of oriented gradients

ÖZ

HAREKETLİ ARAÇLARIN SINIFLANDIRILMASI

Duman, Demet
Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Gözde Bozdağı Akar

Eylül 2013, 78 sayfa

Son yıllarda akıllı ulaşım sistemleri görüntü işleme alanında aktif bir araştırma alanı olmuştur. Bu çalışmanın amacı, ayarsız bir video kamera ile alınan otoyol video görüntülerini kullanarak hareket halindeki araçları sınıflandırmaktır. Bu çalışmada, sınıflandırmak üzere farklı ölçülerdeki üç çeşit araba sınıfı seçilmiştir: araba, orta sınıf araç ve büyük sınıf araç. Önerilen algoritma ön plan/arka plan ayırımı, özellik çıkarımı ve sınıflandırma aşamalarından oluşmaktadır. Algoritmada her aracı sınıflandırmak için, şekle dayalı bir tanımlayıcı olan gradyan yönelim histogramı özellikleri ve boyuta dayalı bir tanımlayıcı olan imge bölgesi özellikleri kullanılmıştır. Bu özelliklerin sınıflandırma performansı üzerine etkileri incelenmiştir ve farklı otoyol videolarına ait deney sonuçları verilmiştir.

Anahtar Kelimeler: Araç sınıflandırma, ön plan/arka plan çıkarımı, gradyan yönelim histogramı

To my family...

ACKNOWLEDGEMENTS

First of all I would like to thank to my supervisor Prof. Dr. Gözde Bozdağı Akar very much for her endless help, guidance, encouragement and support during this study.

I would like to thank to my employer, ASELSAN for supporting me.

I also express my gratitude to TÜBİTAK BİDEB "National Scholarship Program for MSc Students".

Finally, I would like to thank my family, my mother (Aysel Duman), my father (Kazım Duman) and my brother (Egemen Duman) for their love, support and patience over the years. This thesis is dedicated to them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ITS	Intelligent Transportation System
HOG	Histogram of Oriented Gradients
MoG	Mixture of Gaussians
FAST	Features from Accelerated Segment Test
ORB	Oriented Features from Accelerated Segment Test
LoG	Laplacian of Gaussians
DoG	Difference of Gaussians
MSER	Maximally Stable Extremal Regions
GLOG	Gradient Location and Orientation Histogram
SIFT	Scale Invariant Feature Transformation
SURF	Speeded Up Robust Features
SVM	Support Vector Machines
PCA	Principle Component Analysis
LDA	Linear Discriminant Analysis

CHAPTER 1

INTRODUCTION

The intelligent transportation systems (ITS) have been developed to manage the traffic flow. These systems use a suite of sensors for obtaining the traffic parameters. The magnetic loop detectors are mostly used sensors which measure the length and the number of axles of vehicles. They are installed under the road to provide real time statistics. Unfortunately, they have some disadvantages; they are expensive, they stop and damage the traffic during installation and they are only able to sense the presence of a vehicle.

In recent years, video camera starts to be a promising traffic sensor and vision-based traffic monitoring becomes popular in ITS, because monitoring the traffic based on video cameras has some advantages. First, video cameras are easy to use and not disruptive to traffic during installation compared to the magnetic loop detectors. Second, large number of areas can be covered with a small number of video cameras. Third, they are cheaper compared to the magnetic loop detectors. Forth, video cameras allow collecting rich information about the traffic and provide analyses of the traffic flow at the level of the vehicle. By using video cameras many traffic parameters can be obtained like congestion, vehicle counts, lane changes, vehicle velocity and vehicle classes.

In ITS, vehicle classification is an important area in order to obtain the percentage of the vehicle classes on the streets and highways. Obtaining vehicle classes is important, because the geometric design of a road, like horizontal alignment and curb heights, depends on the vehicle types which use it, due to their heavy weights, inferior braking, and large turning radius. The heavy weight of the vehicles also affects the pavement design. In current situation human operators manually obtain the vehicle classes on some highways. In order to classify the vehicles using an automated system based on video cameras will also be cost effective by eliminating the need for human operators.

1.1 STATE-OF-THE-ART IN VEHICLE CLASSIFICATION

The literature on vehicle classification is mainly divided into 2D approaches, which means that operation in the camera view and 3D approaches which includes 3D modeling. This section covers the latest studies and the related work in the literature on these approaches.

1.1.1 2D Approaches

Systems which work in the camera coordinate domain are in the scope of 2D approaches. 2D approaches on vehicle classification vary according to the features used to discriminate the vehicles and classified vehicle types. Generally size based features like length, height and area are used in 2D approaches.

Lipton et al. [1] use a classification metric dispersedness which corresponds to ratio between perimeter and area of the vehicle blob in order to classify into three categories: human, vehicle or background clutter. In this work, classification accuracy is reported as around 85%.

Gupte et al. [2], [3], and Zhang et al. [18] use height and length to classify vehicles on a highway, while Avery, Wang and Rutherford [6] use only length. Length and height of the vehicles are obtained by using the 2D projections of the vehicles. This stage uses information about the camera's location and the camera parameters. In [2], classification is done into two categories as trucks or other vehicles and accuracy is reported as 90%. In [3], classification is done into two categories as cars or non-cars and accuracy is reported as 70%. In [18] and [6], it is aimed to discriminate between long and short vehicles in order to detect the long vehicles and the detection accuracy of long vehicles is reported as 97% and 91%, respectively while there is no direct information about classification accuracy.

Huang and Liao [7] use area, size and length to classify vehicles on a highway by using some rules. Seven vehicle classes which are pickup, sedan, van, van truck, truck, trailer or bus are classified from side view of a highway scene. Performance of the algorithm is reported as 91% overall classification rate.

Rad and Jamzad [8] propose a system to classify and count the vehicles. Also, they find out the lane-changes through tracking on the highways, while Veeraraghavan et al. [5] monitor road intersections. In both work, classification is done based on the size of the bounding box of the blobs and velocity. In [8], vehicles are classified into three classes, motorcycle/bicycle, car or bus/minibus; while in [5] classification is done into vehicle or pedestrian. In [8], bounding boxes which are classified are tracked by using a Kalman filter. The tracking error rate is reported as 5.4%. However, there is no quantitative information about the performance of the systems for classification.

Morris and Trivedi [9] and [10] utilize blob features like breadth, area, compactness, perimeter, elongation, roughness, area, length, long and short axis of fitted ellipse, centroid and 5 image moments. A comparison between image based features like pixels and image measurement features like region size is presented. These feature types are used with Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA) as dimensionality reduction techniques. Image measurement features with LDA which gives the best performance is selected as the final algorithm. Their system works on the videos captured from the side view of a highway. In [9], classification accuracy is given as 82.9% for classifying into three classes as sedan, semi or truck/SUV/van while in [10], classification combined with tracking is done into seven classes as sedan, truck, SUV, semi, van, truck/SUV/van or moving trucks and classification accuracy is given as 74.4%.

Hsieh et al. [11] use linearity and size features for the classification of the vehicles. The linearity feature is a measure for the roughness of the vehicle silhouette. In their work, classification is done into four classes, car, minivan, van-truck (including bus) and truck by assuming that the camera is in axis with the highway to see the lane division lines. Classification accuracy is given as 88%.

Zhang et al. [12], Arrospide and Salgado [13] and Chen and Zhang [15] use vehicle images as features to make a classification. [12] and [13] utilizes PCA while [15] utilizes ICA to reduce the dimension of the feature space. In [12] and [15], classification precision is given as 50% and 75%, respectively for three classes: passenger car, van or pick-up. In [13] classification is done into vehicle or non-vehicle with a classification rate 93.04%.

Gandhi and Trivedi [16] and Arrospide et al. [14] utilize Histogram of Oriented Gradients (HOG) features for in-vehicle classification systems. In [16], a classification into four classes as car, van, pickup-truck or no vehicle is done; while in [14] classification is performed as vehicle or non-vehicle. In [16] and [14] classification accuracies are reported as 64.3 % and 90%, respectively.

Alonso et al. [17] propose an vehicle detection system through classification as vehicle or non-vehicle. The bounding boxes of the vehicles are created based on edges. The bounding boxes are verified by corner detection and symmetry inside this region. In this work, classification rate is given as 90%.

Thi et al. [19] and Robert [20] propose a vehicle classification system for night time videos. Vehicle images are used as the features by applying PCA to reduce the dimension. Classification is done as vehicle or non-vehicle. In [19], classification accuracy is reported as around 94%. In [20], classification accuracy is given as around 95% by combining tracking with the classification.

1.1.2 3D Approaches

3D approaches are generally based on 3D modeling of the vehicle types which achieve high accuracy in the expense of the higher computational complexity and these algorithms need camera parameters in order to use in the projection of vehicle parameters on 2D. Sullivan et al. [21], Messelodi et al. [22] and Buch et al. [23] perform vehicle detection and classification through 3D models of the vehicles.3D models of vehicle types is created and according to a match measure, classification is done. In [21], classification is done into two classes as car or van and accuracy is given as around 96%. In [22], classification is done into bicycle, lorry, motorcycle, van car, extra-urban bus, urban bus or unknown. The reported classification rate is 91.5%. In [23], classification performance is given as a recall of 90.4% at a precision of 87.9% for four classes: bus/lorry, van, car/taxi and motorbike/bicycle.

In the literature, most of the works require camera parameters like height of the camera, angular direction of the camera and camera calibration or knowledge about the highways like the place of the lane division lines and the viewing side of the highway. In this study, it is aimed to classify moving vehicles into three classes as car, van or truck without needing specifically any camera parameters and the knowledge about the highway.

1.2 SCOPE OF THE THESIS

This thesis focuses on classification of the moving vehicles on highways. Our goal is to classify vehicles into three classes as car, van or truck with the proposed algorithm. For that purpose, initially vehicles are detected. In the detection step, a background subtraction of algorithm based on mixture of Gaussians (MoG) is used due to its capability of coping with the changes in the scene (i.e., adaptable) in order to isolate moving vehicle blobs. In order to move undesired components which are part of foreground, morphological operations are utilized.

Next step after detection of the vehicles is the feature extraction. Blob features which are dimension-based descriptors and histogram of oriented gradients (HOG) features which are shape-based descriptors are extracted to represent the vehicle classes. In the study, effects of using HOG and blob features together and separately are examined.

After features of each vehicle blob are extracted and classified with a robust supervised classifier support vector machines (SVM) in a hierarchical manner, each vehicle class is obtained.

1.3 OUTLINE OF THE THESIS

In Chapter 2, basic building blocks of a vehicle classification system is explained in detail.

In Chapter 3, proposed algorithm is presented by explaining each algorithm in the building blocks of the system which are detection of vehicles, feature extraction and classification.

After explaining the algorithm, experimental results on different videos are given in Chapter 4. In addition, a comparison with the literature is included.

The thesis is summarized and concluded in Chapter 5. It presents conclusions and observations made throughout the thesis study. In addition, some future works are presented in this chapter.

CHAPTER 2

BACKGROUND TOPICS

2.1 BASIC BUILDING BLOCKS OF VEHICLE CLASSIFICATION

In this section, basic elements required for vehicle classification will be explained. Generally, vehicle classification is divided into two stages. First stage is detection stage while second stage is classification. Typical vehicle classification system uses foreground segmentation in vehicle detection stage and then classification part comes. As it is seen in Figure 2.1 , a statistical model estimates foreground pixels and then those foreground pixels are grouped so that connected regions are obtained and those regions are propagated through the classification stage. A priori information about vehicle classes which is previously learned or preprogrammed is used to assign class label in classification stage.

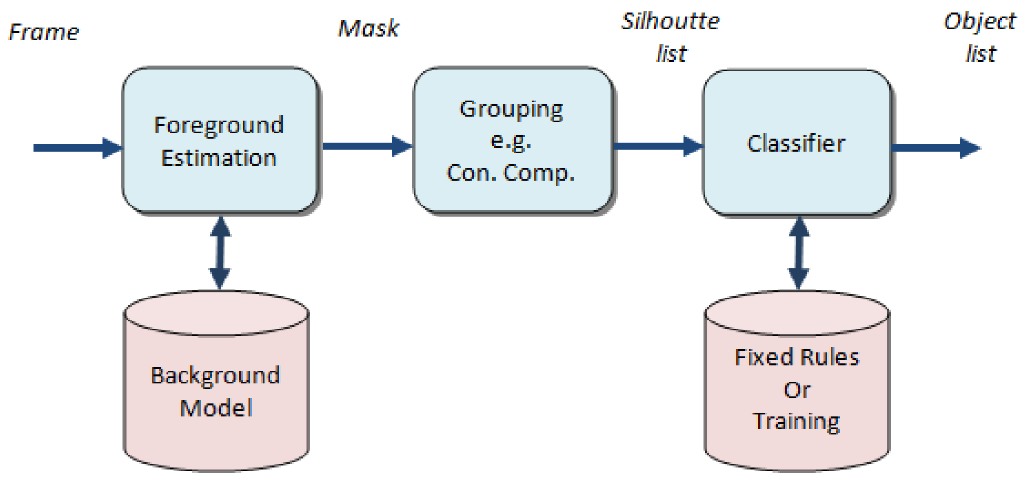


Figure 2.1. Block diagram of typical vehicle classification system

2.1.1 Foreground Segmentation

In an automated visual surveillance system, foreground segmentation is the first stage. Generally performance of the system is affected by the success in the obtaining the foreground regions correctly.

There are two basic different approaches used in vehicle classification systems to estimate the foreground (vehicle) regions. First approach is an obtaining background model. Providing that the camera is stationary, a comparison is done between this model and current frame to find out the differences which refer to the foreground regions. With this approach stationary objects are missed out because of the lack of motions. This approach is suitable for implementation on the computer; however, in slow moving traffic it has problems. Second approach segments foreground regions based on object appearances. This approach can be used for both stationary and moving cameras to obtain the vehicle (foreground) regions. However, it requires prior information for foreground object appearances and it is computationally costly. In the next section, generally used algorithms in vehicle classification systems will be introduced.

2.1.1.1 Frame Differencing

The easiest and simplest way of foreground segmentation is frame differencing. In this method, the model for the background is equal to the previous frame. The difference between the previous and current frame and is thresholded and used as foreground mask.

$$M(x,y,t) = \begin{cases} 1, & |I(x,y,t) \ - \ I(x,y,t-1)| > Threshold \\ 0, & |I(x,y,t) \ - \ I(x,y,t-1)| < Threshold \end{cases} \qquad (2.1)$$

In the above formula, I (x,y,t) is the intensity value at pixel location (x,y) at time t and I (x,y,t-1) is the intensity value at pixel location (x,y) at time t-1. M (x,y,t) is the mask image obtained thorough differencing and thresholding.

The algorithm is very easy and fast to implement. However, in dynamic scene conditions it has a low performance. It cannot cope with multi-modal distributions, abrupt illumination changes, periodic movements in the backgrounds like trees and noise. Also, its results are very sensitive to the threshold value. In the literature, [1], [2] and [6] use frame differencing to detect the vehicles.

2.1.1.2 Moving Average Filtering

In this method, by calculating the mean value of the previous N frames a reference background frame I_{ref} is generated and a mask image is obtained as follows:

$$M(x,y,t) = \begin{cases} 1, & |I(x,y,t) - I_{ref}| > Threshold \\ 0, & |I(x,y,t) - I_{ref}| < Threshold \end{cases} \tag{2.2}$$

The update equation of the background model is given as below:

$$I_{ref,t} = \alpha\, I(x,y,t-1) + (1-\alpha) I_{ref,t-1} \tag{2.3}$$

In (2.3), α is the learning parameter and must be chosen as considering the features (size, speed, etc.) of the video and moving objects. Learning parameter α determines how the background model adapts to changes in the scene.

This algorithm has little computational cost and superior to the frame differencing method. However it is sensitive to the threshold value. If the threshold value is too high, then foreground regions can be marked as background; if the threshold value is too low, some background regions become foreground. Also, this algorithm produces tails at the back of the moving objects because of the contamination of the background by the appearance of the moving objects. In addition, it cannot cope with multi-modal distributions. In the literature, [3], [7], [12] and [15] use moving average filtering to detect the vehicles.

2.1.1.3 Single Gaussian

Single Gaussian model [24] which has a dynamically changing threshold for each pixel improves robustness in background modeling. This method tries to fit a Gaussian distribution (μ, σ) to each pixel. By that way, background model is generated for each pixel. In this algorithm, if current pixel value x_t satisfies (2.4), it is matched with the corresponding Gaussian distribution.

$$|x_t - \mu_t| \leq 2.5\, \sigma_t \tag{2.4}$$

Where μ_t is the updated mean and σ_t is the updated variance of the corresponding Gaussian distribution. If current pixel value x_t is matched with a Gaussian distribution, parameters of that distribution are updated as below and this pixel is labeled as background.

$$\mu_t = (1 - \alpha)\mu_{t-1} + \alpha x_t \tag{2.5}$$

$$\sigma_t^2 = (1 - \alpha)\, \sigma_{t-1}^2 + \alpha\, (x_t - \mu_t)^2 \tag{2.6}$$

Where α is the learning rate. If α is high, recent pixel values have more influence on the background model. If α is low, the influence of recent pixel value is not much.

If current pixel is not matched the background model, it is labeled as a foreground pixel. Single Gaussian method has better performance than first two methods which is mentioned, since it has a dynamic threshold. However, it cannot cope with multi-model distributions and abrupt illumination changes since it has a one Gaussian distribution. In the literature, [9] and [10] use single Gaussian method for detection of the vehicles.

2.1.1.4 Mixture of Gaussians

In mixture of Gaussians (MoG) method [25], recent history of each pixel $\{X_1, \ldots, X_t\}$ is modeled by a mixture of K Gaussian distributions. The probability of observing the current pixel value in that model is given in (2.7) and a modeling of a pixel by a MoG is given in Figure 2.2.

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t}\, \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{2.7}$$

In (2.7), K is the number of distributions, $\omega_{i,t}$ is an estimate of the weight of i^th^ Gaussian in the mixture at time t and shows what portion of the data is accounted for by this Gaussian. $\mu_{i,t}$ and $\sum_{i,t}$ are the mean value and covariance matrix of the i^th^ Gaussian in the mixture at time t, and η is a Gaussian probability density function is given below

$$\eta(X_t, \mu, \sum) = \frac{1}{(2\pi)^{\frac{n}{2}} \, |\sum|^{\frac{1}{2}}} \, e^{-\frac{1}{2}(x_t - \mu_t)^T \, \Sigma^{-1}(x_t - \mu_t)} \tag{2.8}$$

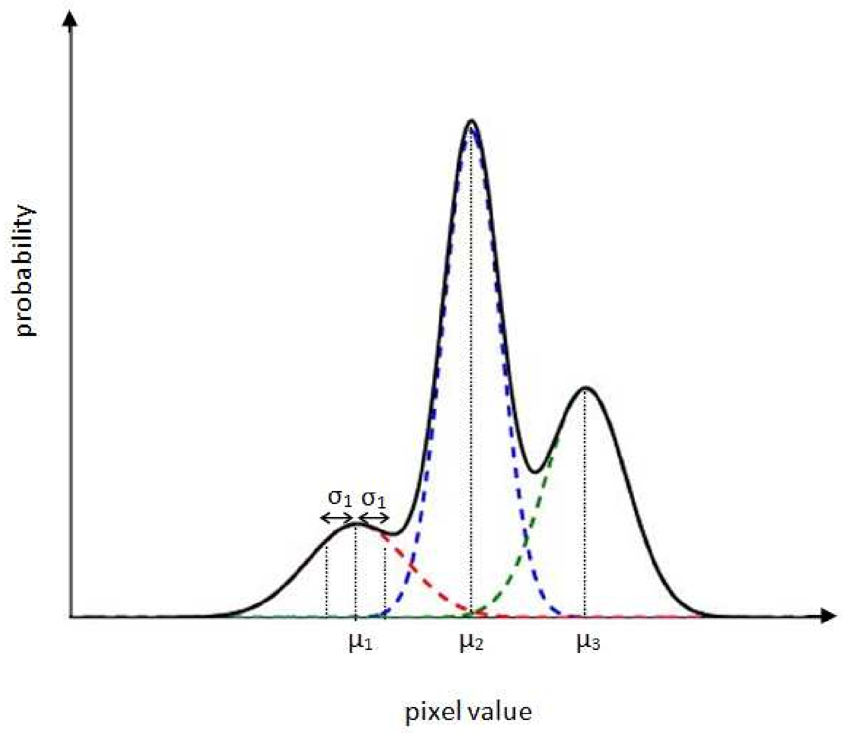


Figure 2.2. Model of a pixel as a mixture of Gaussians

Generally, from 3 to 5 number of distributions K is used and this is determined by the available memory and computational power. In addition, for computational reasons, covariance matrix is assumed to be of the form:

$$\sum_{k,t} = \sigma_k^2 I \tag{2.9}$$

This covariance matrix assumes that red, green and blue pixel values are independent ad have same variances, although this is not the ideal case. This assumption helps to avoid a costly matrix inversion at the expense of some accuracy.

So, the distribution of recently observed values of each pixel is characterized by a mixture of Gaussians. Every new pixel value will be represented by one of the major components in the model and used to update the mixture model. Each pixel value X_t is checked against K Gaussian distributions, until a match is obtained. A match is defined as a pixel value within 2.5 standard deviations of a distribution. If none of the K distributions match the current pixel value, the least probable distribution is replaced with a distribution which has the current pixel value as its mean, an initially high variance and low prior weight. Weights of K distributions at time t are updates as follows

$$\omega_{k,t} = (1 - \alpha)\omega_{k,t-1} + \alpha M_{k,t} \tag{2.10}$$

Where α is the learning rate and $M_{k,t}$ which is called ownership is 1 for the matched models and 0 for the remaining models. Also, weights are normalized after this approximation so that they add up to 1. If the current pixel value is matched, the parameters of the corresponding Gaussian distribution are updated as in (2.11) and (2.12). The mean μ and the variance σ parameters for unmatched distributions remain the same.

$$\mu_t = (1 - \rho)\mu_{t-1} + \rho X_t \tag{2.11}$$

$$\sigma_t^2 = (1 - \rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T(X_t - \mu_t) \tag{2.12}$$

where $\rho = \alpha\, \eta(X_t | \mu_k, \sigma_k)$.

As the parameters of the mixture model of each pixel change, it should be determined which of the Gaussians in the mixture are most likely to be produced by the background processes. In order to obtain the background model, Gaussian distributions of a pixel are ordered by the value of $\frac{w}{\alpha}$ from highest to lowest. This value increases both as distribution gains more evidence and as the variance decreases. With this ordering, the most likely background distributions remain on top and the less probable transient background distributions gravitate towards the bottom and eventually replaced by new distributions. Then, the first B distributions satisfying (2.13) are chosen as the background model.

$$B = argmin_b\left(\sum_{k=1}^{b} \omega_k > T\right) \tag{2.13}$$

where T is a measure of the minimum portion of the data that should be accounted by the background. If T is small, the background model is usually unimodal. If this is the case, using only the most probable distribution will save processing. If T is higher, a multi-modal distribution caused by a repetitive background motion (e.g. leaves on a tree, a flag in the wind etc.) could result in more than one color being included in the background model. This results in a transparency effect which allows the background to accept two or more separate colors.

This algorithm can deal with repetitive clutter and lighting changes. However, it is computationally complex comparing to the other three algorithms mentioned before. In the literature, MoG is used in [5], [18] and [23] to detect the vehicles.

2.1.1.5 Object-based Segmentation

Object-based segmentation based on detection of the objects in order to identify the foreground. In object-based segmentation, methods are considered which detect objects in a holistic way by searching for full objects. Generally, 3D models of objects are used in order to identify the objects. Because of using 3D modeling, methods relying on object-based segmentation have higher computational complexity. [21] and [22] use object based segmentation in order to obtain the vehicle blobs.

2.1.2 Classification

Classification is the assignment of a new instance to a group of previously known instances named as the class. The classifier wants information about a new instance which is called as features. After features of classes are extracted from the object, a machine learning algorithm is trained by using the instances of known classes to generate discriminative information from the features. Then, the classifier uses the learned information in order to assign a label to a new instance.

2.1.2.1 Feature Extraction

Feature extraction process produces similar values which come together to create a feature vector for the instances belonging to the same class and this is an important step for the classification part. In order to obtain the feature vectors, key points (features) are detected. In computer vision there are lots of key point detectors like Harris [38], Hessian [39], Laplacian of Gaussians (LoG) [40], Difference of Gaussians (DoG) [41], Canny [42], Sobel [43], Prewitt [44], Shi and Tomasi [45] , Features from Accelerated Segment Test (FAST) [46], and Maximally Stable Extremal Regions (MSER) [47] detectors, Haar wavelets [48], Hough transform [49]. Once features are detected, a local image patch around the feature can be extracted. For that purpose feature descriptors like MPEG-7 [33], Scale Invariant Feature Transformation (SIFT) [34], Speeded Up Robust Features (SURF) [36]., Histogram of Oriented Gradients (HOG) [31], Oriented FAST (ORB) [50], Gradient Location and Orientation Histogram (GLOH) [51] and Gabor filters [52]. In the following section, an overview of mostly used feature descriptors in vehicle classification is presented.

2.1.2.1.1 Region-based

Region-based features are generally extracted from the whole image region of an object. Image region of an object is generally the foreground area extracted by the foreground segmentation algorithm. Image moments, length, height, size and area of the image region, SIFT, SURF and HOG features are often used to generate a feature vector.

SIFT is introduced in [34]. The local features generated are invariant to rotation, image scaling and translation. Also, they are partially invariant to affine projection changes and illumination changes. Generally, a SIFT feature describes the appearance of salient points in the image uniquely, which will remain salient even if the image is rotated, resized or the illumination is changed. The SIFT features find point to point correspondences in two different images of the same object. Modified SIFT descriptors are used in [35] to create a rich representation of vehicle images.

The SURF descriptors are introduced in [36]. The descriptor is used for finding correspondence between images, like SIFT. However, the design focuses on computational speed by allowing loss of performance. The use of box filters instead of Gaussian filters in the case of [34] reduces computational complexity.

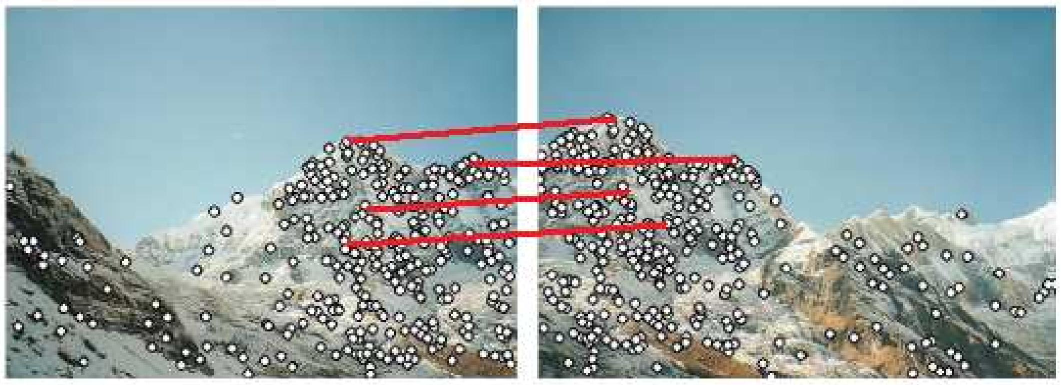


Figure 2.3. SIFT and SURF descriptors for matching in object recognition

The concept of HOG is introduced in [31]. To calculate the feature vector, the gradient of the input image is divided into cells. A histogram of the gradient orientation in pixels is calculated for every cell. The vectors of all cells are concatenated in order to create one global feature vector for the image. In [31], HOG is utilized for the detection of pedestrians.

In this thesis study, HOG concept is used to classify the vehicles. In Chapter 3, HOG will be introduced in more detail.

2.1.2.1.2 Contour-based

Contour based features only take the edge of a silhouette into account. They rely on the contour information of the object instead of the whole set of pixels inside the object region. The distance between contour points is used as a similarity measure. Processing is performed on closed contours as extracted from the video. These features are generated through several edge and corner detection algorithms like Canny, Sobel, Harris and Prewitt edge detectors. The contour which includes edges is used in [17] and [37] for vehicle classification.

2.1.2.2 Classifiers

Classifiers assign an unknown object instance to a known class by using the extracted feature vector. This assignment relies on learned information from the training data. Machine learning algorithms generate classifiers by using training data. An important property of the learning algorithms is the supervision which provides the labels for the training data. Ground truth is required for the evaluation of the classifier. The classifier output is compared to the manually generated ground truth. In the following section classifiers generally used for vehicle classification will be explained.

2.1.2.2.1 Nearest Neighbour Classifier

The nearest neighbour classifier is the easiest non parametric classifier for a feature vector. The distance between every vector of the training set and a new feature vector is calculated. Any distance measure can be chosen. The most common distance measure is Euclidean distance given in (2.14) where x and y are points in R^m.

$$d(x,y) = \|x-y\| = = \left(\sqrt{\sum_{i=1}^{m}(x_i - y_i)^2}\right) \tag{2.14}$$

The class label of the closest training vector is assigned to the new vector. K-nearest neighbour algorithm which improves robustness is the extension of the nearest neighbour algorithm. This algorithm classifies a feature vector by assigning it to the label most frequently represented among the k nearest samples. A decision is made by examining the labels on the k nearest neighbors and taking a vote as it is seen in Figure 2.4.

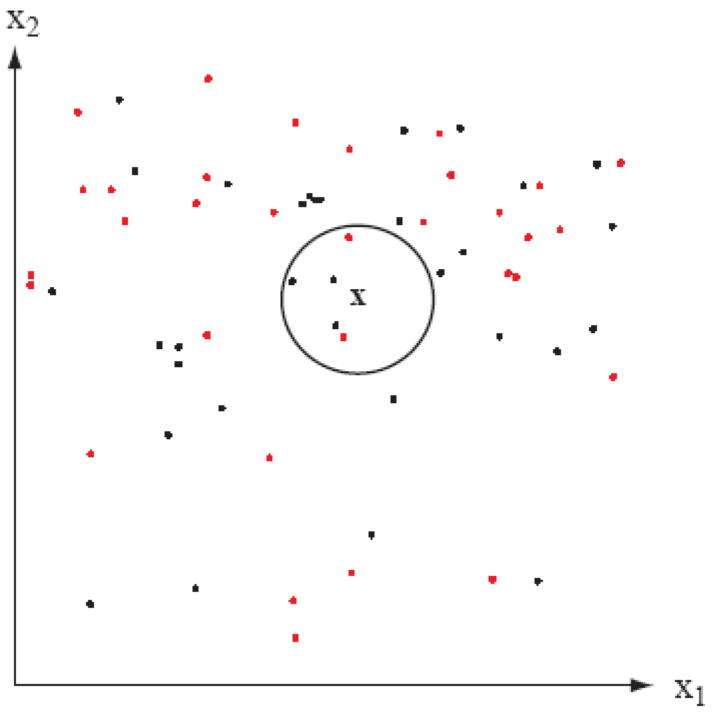


Figure 2.4. The k-nearest-neighbor classification

In terms of memory requirements and computational complexity nearest neighbour classifiers do not scale very well for large training sets and they need many distance calculations. There is no time requirement for training. However, the classification time increases with the training size.

2.1.2.2.2 Support Vector Machines

Support Vector Machines (SVM) perform classification using linear decision hyperplanes in the feature space [26]. SVM constructs a decision hyperplane between samples of two classes based on the most informative points of the training set which are called support vectors. The aim of SVM is to find the optimal hyperplane between samples of two classes with the largest margin (Figure 2.5). As seen in Figure 2.6, support vectors are equally close to the hyperplane and nearest patterns, a distance b from the hyperplane. The three support vectors are shown in solid square.

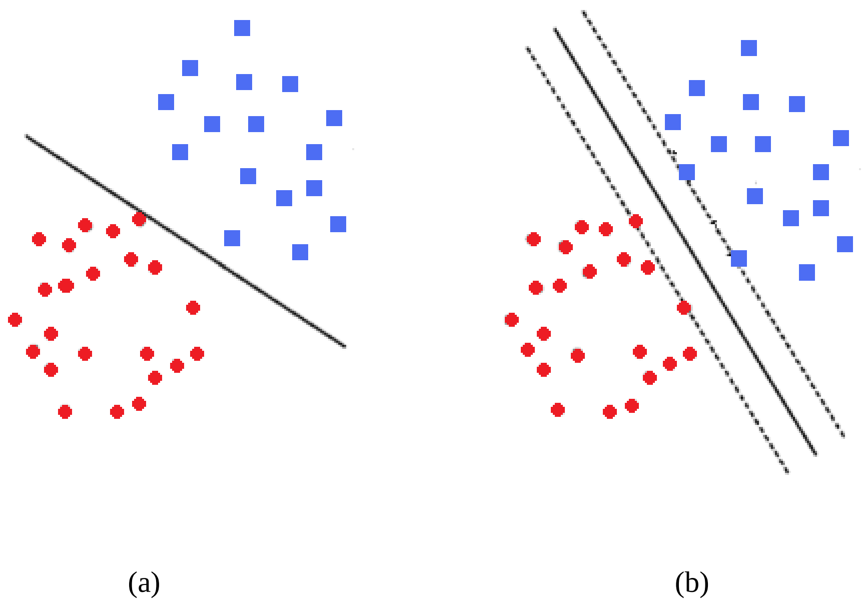


Figure 2.5. Decision boundary obtained by (a) an ordinary classifier and (b) SVM

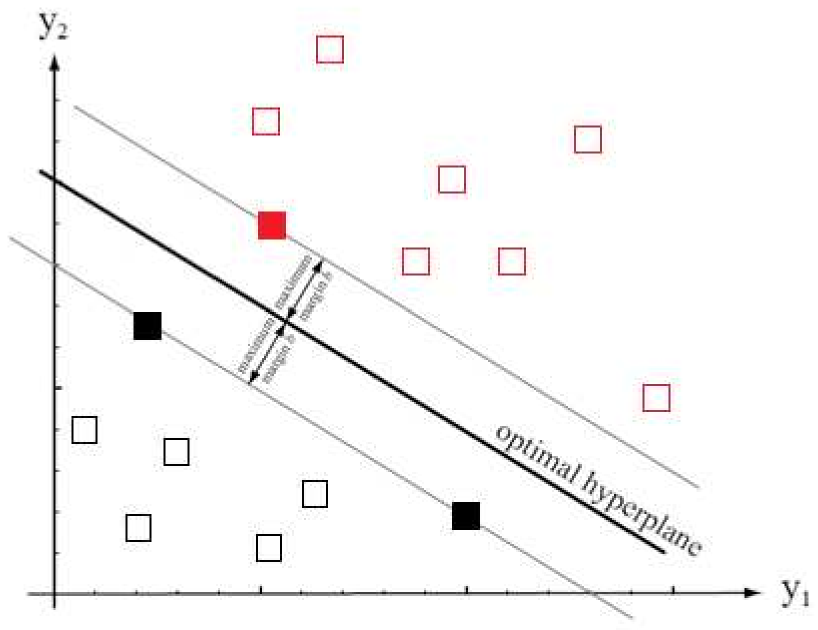


Figure 2.6. Finding the optimal hyperplane

A separating hyperplane which can also be called as Linear Discriminant Function is defined as in (2.15) given that a training data set $\{x_1, \dots, x_n\}$ and their corresponding labels $\{y_1, \dots, y_n\}$ taking values +1 and -1. The goal is to obtain $\overrightarrow{w^t}$ and w_0 which are weight vectors.

$$g(\vec{x}) = \overrightarrow{w^t}\vec{x} + w_0 \qquad (2.15)$$

It is decided that

$$\begin{aligned} y_i &= +1 \quad if\ g(\vec{x}_i) \geq +1 \\ y_i &= -1 \quad if\ g(\vec{x}_i) \leq -1 \qquad i = 1, \dots\dots, n \\ &either\ class\ otherwise \end{aligned} \qquad (2.16)$$

It is known that the distance of point $\vec{x}_i$ from the decision boundary as it is seen Figure 2.7 is given as (2.17)

$$r = \frac{\overrightarrow{w^t}\vec{x}_i + w_0}{\|\vec{w}\|} \qquad i = 1, \ldots\ldots, n \tag{2.17}$$

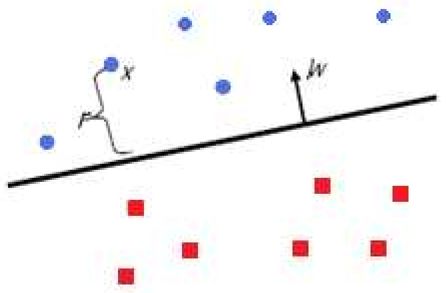


Figure 2.7. Distance (r) of a point from the decision boundary

Distance is normalized so that for the nearest point it becomes $\frac{1}{\|\vec{w}'\|}$ as it is seen in Figure 2.8. x is the nearest point to the boundary plane and called as support vector. Blue line shows $g(\vec{x}_i) = +1$ plane and green line shows $g(\vec{x}_i) = -1$ plane.

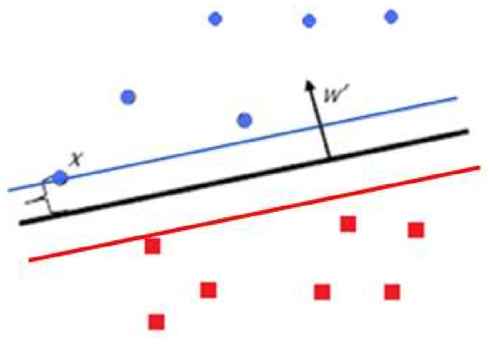


Figure 2.8. Distance of nearest point to the boundary plane

For a linearly separable problem, optimal hyperplane which separates feature space while maximizing the distance from the support vectors can be obtained as a result of an optimization process in (2.18).

$$\max \frac{1}{\|\overrightarrow{w'}\|^{2}} \quad \mathit{subject\ to}\ \ y_i\left(\overrightarrow{w'}^{t}\vec{x}_i + w'_0\right) \geq +1 \qquad (2.18)$$

$$i = 1, \dots, n$$

This problem can be solved by using the Lagrange multipliers in (2.19) where α_i nonzero for only support vectors.

$$min_{\vec{w},w_0,\alpha} \{\overrightarrow{w'}\ \overrightarrow{w'} - \sum_{i=1}^{n} \alpha_i\, (\, y_i\left(\overrightarrow{w'}^{t}\vec{x}_i + w'_0\right) - 1)\}$$

$$\Rightarrow \overrightarrow{w'} = \sum_{i=1}^{n} \alpha_i\, y_i \vec{x}_i \qquad (2.19)$$

If the problem is non-separable, equations (2.18) and (2.19) are modified as in (2.20) and (2.21)

$$\min\left\{ \|\overrightarrow{w'}\|^{2} + C\sum_{i} \xi_i \right\} \qquad (2.20)$$

$$\xi_i \geq 0\ \ \mathit{subject\ to}\ \ y_i\left(\overrightarrow{w'}^{t}\vec{x}_i + w'_0\right) \geq 1 - \xi_i \quad i = 1, \dots, n$$

$$\min_{\vec{w},w_0,\alpha} \{\overrightarrow{w'}\ \overrightarrow{w'} + C\sum_{i} \xi_i - \sum_{i=1}^{n} \alpha_i\,(\,y_i\left(\overrightarrow{w'}^t \vec{x}_i + w'_0\right) - 1 + \xi_i)\} \tag{2.21}$$

where ξ_i is used to compensate for misclassified samples and C gives a compromise between distance of nearest point and data.

In addition, if the training data is not linearly separable, a kernel function can be used to transform the data into a new vector space. The data has to be linearly separable in the new space. Mostly used kernel functions are given in (2.22).

$$k(\vec{x}_i, \vec{x}_j) = \begin{cases} \vec{x}_i\ \vec{x}_j & linear \\ \left(\vec{x}_i\ \vec{x}_j + 1\right)^d & polynomial \\ e^{\frac{(\vec{x}_i - \vec{x}_j)(\vec{x}_i - \vec{x}_j)}{2\sigma^2}} & radial\ basis \end{cases} \tag{2.22}$$

Support vector machines scale well for large training sets. The complexity for training increases with the number of training samples; however, the classification is independent of it.

CHAPTER 3

PROPOSED ALGORITHM FOR VEHICLE CLASSIFICATION

This chapter presents the work done to detect and classify vehicles in the highway scenes. A system block diagram is shown in Figure 3.1, where each block will be explained individually in the next sections. In the system, detection stage generates foreground regions which are corresponds to the moving vehicles. After that for every foreground region, features which are mentioned later in this section are extracted and given to a supervised hierarchical classifier to classify the vehicles as

- Car
- Van
- Truck.

With the proposed algorithm, any parameter related to the highway like direction of the moving vehicles, lane division lines, viewing side of the highway; any camera parameter like height of the camera , angular direction of the camera; and camera calibration are not required to classify vehicles. Algorithm only wants the user to mark a detection region in order to see the full view of a vehicle. In this algorithm, following assumptions are made:

- Camera is stationary,
- Every silhouette contains exactly one vehicle being fully visible. This implies no occlusion in the scene and between vehicles,
- Classification is done in day time.

The rest of the chapter is organized as follows. The detector is introduced in section 3.1. Section 3.2 covers the features which are extracted for classification and in section 3.3, classifier is given.

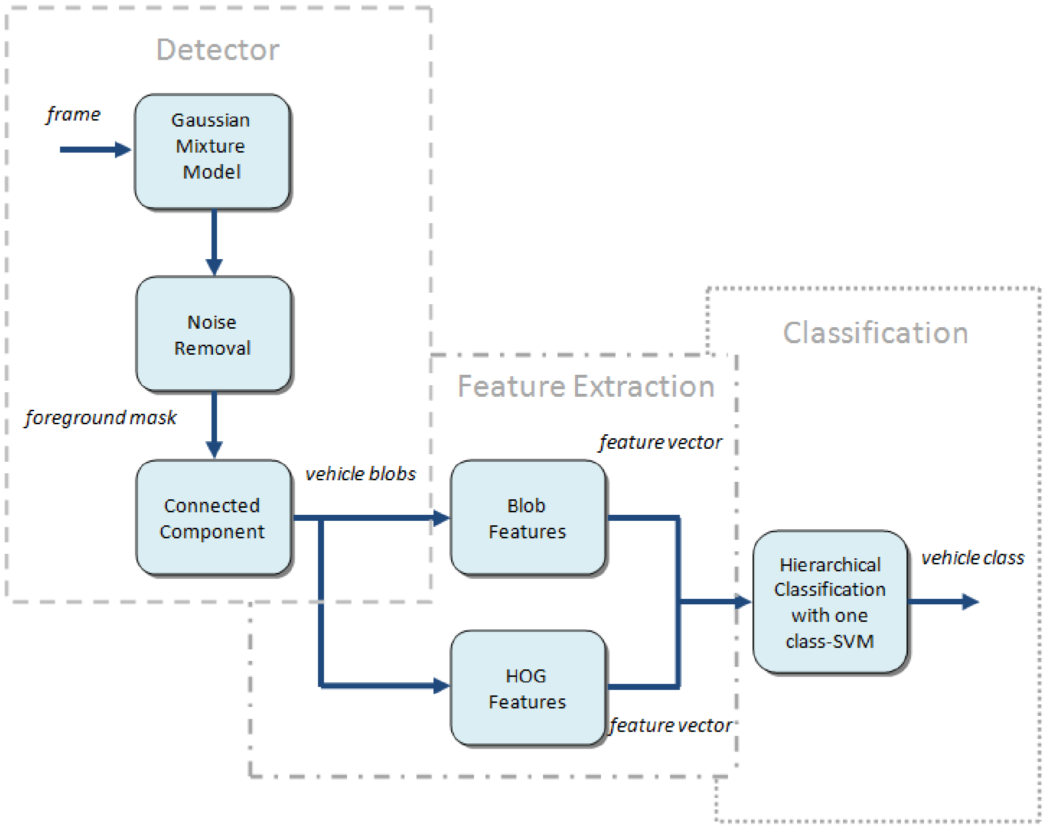


Figure 3.1. Block diagram of the detection and classification system

3.1 DETECTION OF VEHICLES

In visual surveillance systems, generally moving object detection, sometimes called motion segmentation or foreground extraction is the first step. The following operations, such as object tracking and object classification, take the output of moving object detection module as its input. Also, in the case of vehicle classification, detection of vehicles is the first step to create an input for classification part. Therefore, the performance of detection of vehicles affects the overall performance of the entire system. Every block in the detector part of Figure 3.1 is described in more detail in this section.

3.1.1 Foreground Segmentation

In Chapter 2, basic foreground segmentation algorithms are introduced. From those algorithms MoG is preferred in the proposed algorithm because it can deal with lighting changes, repetitive clutter and it is more stable compared to other foreground segmentation algorithms. In the proposed algorithm, adaptive MoG which is introduced in [27] was implemented. In [25] which MoG is firstly introduced in, number of Gaussian components K is fixed and constant over time. However, K fixed and the same for each pixel is not optimal in terms of detection and computational time. To solve this problem [27] proposes an online algorithm that estimates the parameters of the MoG and simultaneously selects the number of Gaussians using Dirichlet prior. The consequence is that K is dynamically adapted to the multimodality of each pixel.

Assume that *t* data samples exist and each of them belongs to one of the components of the MoG. Also assume that the number of samples that belong to the *m*-th component is $n_m = \sum_{i=1}^{t} M_m^i$ where M_m^i-s are defined in 2.1.1.4. Multinomial distribution for n_m-s gives likelihood function in (3.1) and by knowing the fact that the mixing weights are constrained to sum up to one, Lagrange multiplier λ is introduced and the Maximum Likelihood (ML) estimate follows from (3.2).

$$\mathcal{L} = \prod_{m=1}^{K} w_m^{n_m} \tag{3.1}$$

$$\frac{\partial}{\partial w_m}\left(log\mathcal{L} + \lambda\left(\sum_{m=1}^{K} w_m - 1\right)\right) = 0 \tag{3.2}$$

After eliminating λ, (3.3) is obtained from *t* samples and it can be rewritten in recursive form as function of $w_m^{(t-1)}$ for *t* - 1 samples and ownership M_m^t of the last sample as in (3.4).

$$w_m^{(t)} = \frac{n_m}{t} = \frac{1}{t}\sum_{i=1}^{t} M_m^i \tag{3.3}$$

$$w_m^{(t)} = w_m^{(t-1)} + \frac{1}{t}\left(M_m^t - w_m^{(t-1)}\right) \tag{3.4}$$

If the influence of the new samples is fixed by fixing $\frac{1}{t}$ to $\alpha = \frac{1}{T}$ where T is the time period in which the training set is updated by adding the new samples and discarding the old ones , update equation (2.10) is obtained. Fixed influence of the new samples means that we rely on the new samples and contribution from the old samples is downweighted.

Prior knowledge for multinomial distribution can be introduced by using Dirichlet prior $\mathcal{P} = \prod_{m=1}^{K} w_m^{c_m}$. c_m presents the prior evidence in the maximum a posteriori (MAP) sense for the class *m.* In other words, it shows the number of samples that belong to that class a priori. Negative prior evidence $c_m = -c$ is used. By that way, it is accepted that the class *m* exists only if there is enough evidence from the data for the existence of this class. The MAP solution that includes the mentioned prior follows from (3.5).

$$\frac{\partial}{\partial w_m}\left(log\mathcal{L} + log\mathcal{P} + \lambda\left(\sum\nolimits_{m=1}^{K} w_m - 1\right)\right) = 0 \tag{3.5}$$

where $\mathcal{P} = \prod_{m=1}^{K} w_m^{-c}$, then (3.6) is obtained.

$$w_m^{(t)} = \frac{1}{S}\left(\sum_{i=1}^{t} M_m^i - c\right) \tag{3.6}$$

where $S = \sum_{m=1}^{K}(\sum_{i=1}^{t} M_m^t - c) = t - Kc$. By rewriting (3.6) we get,

$$w_m^{(t)} = \frac{W_m - c/t}{1 - Kc/t} \tag{3.7}$$

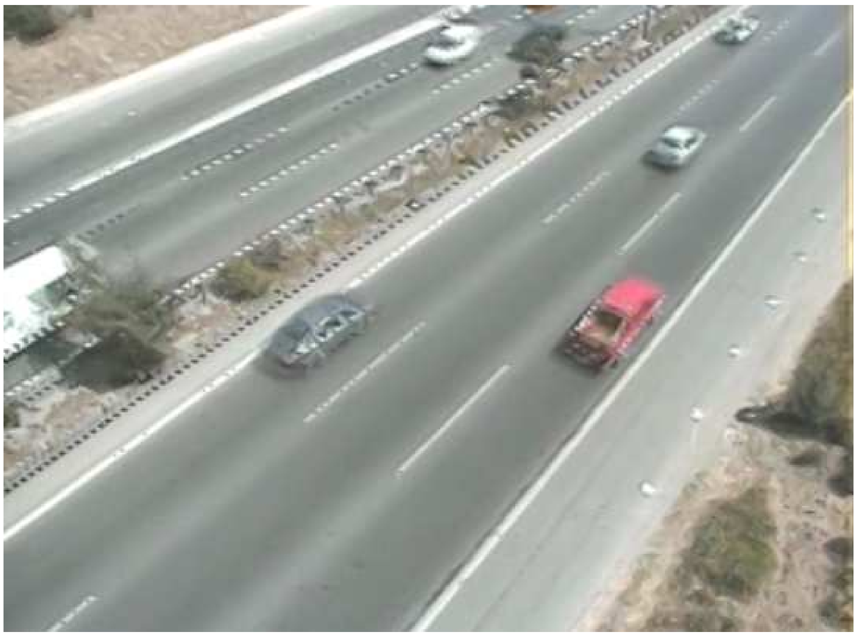
where $W_m = \frac{1}{t}\sum_{i=1}^{t} M_m^t$ is the ML estimate from (3.3) and the bias from prior is introduced through *c/t*. Bias decreases for larger data sets. If a small bias acceptable it can be kept constant by fixing *c/t* to $c_T = c/T$ with a large T value. This means that the bias will always be the same as if it would have been for a data set with T samples. With fixed bias, the recursive version of (3.6) can be obtained as in (3.8).

$$w_m^{(t)} = w_m^{(t-1)} + \frac{1}{t}\left(\frac{M_m^t}{1 - Kc_T} - w_m^{(t-1)}\right) - \frac{1}{t}\left(\frac{c_T}{1 - Kc_T}\right) \tag{3.8}$$

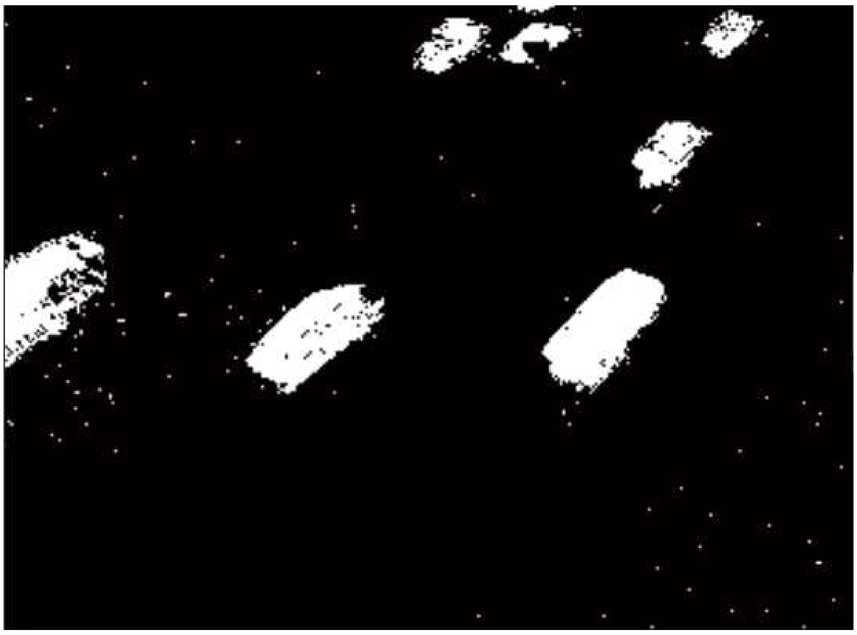
Generally, only a few components K exist and c_T is small, so $1 - Kc_T \approx 1$. Since *1/t* is set to α, final modified adaptive update equation becomes:

$$w_m^{(t)} = w_m^{(t-1)} + \alpha\left(M_m^t - w_m^{(t-1)}\right) - \alpha c_T \tag{3.9}$$

(3.9) is used instead of (2.10). After each update w_m-s should be normalized so that they add up to 1. In the implementation of this MoG, initially it is started with one component centered on the first sample and new components are added as mentioned in 2.1.1.4 while weight update is done according to (3.9). The Dirichlet prior with negative weights will suppress the components that are not supported by the data and the component m is discarded when its weight w_m becomes negative. For a chosen α = 1/T, it is required that at least c=0.01T samples support a component then, c_T becomes 0.01. In our implementation α has been chosen experimentally as 0.007 which gives better results in foreground segmentation.
Figure 3.2 shows the output of the foreground segmentation in proposed algorithm on a traffic video sequence.



(a)



(b)

Figure 3.2. Result of foreground segmentation with MoG. (a) Original frame (b) Segmented foreground mask

3.1.2 Noise Removal

Foreground segmentation produces silhouettes of vehicles. However, as it is seen in Figure 3.2 noises are also the part of the foreground segmentation output. These noises affect the performance of the other steps. In other to improve the overall system performance, noise removal is a crucial step. In order to remove these undesired noises some simple, but effective algorithms have been used in the proposed system. These algorithms are:

- Morphological operations: erosion and dilation
- Connected component labeling and area filtering

3.1.2.1 Morphological Operations

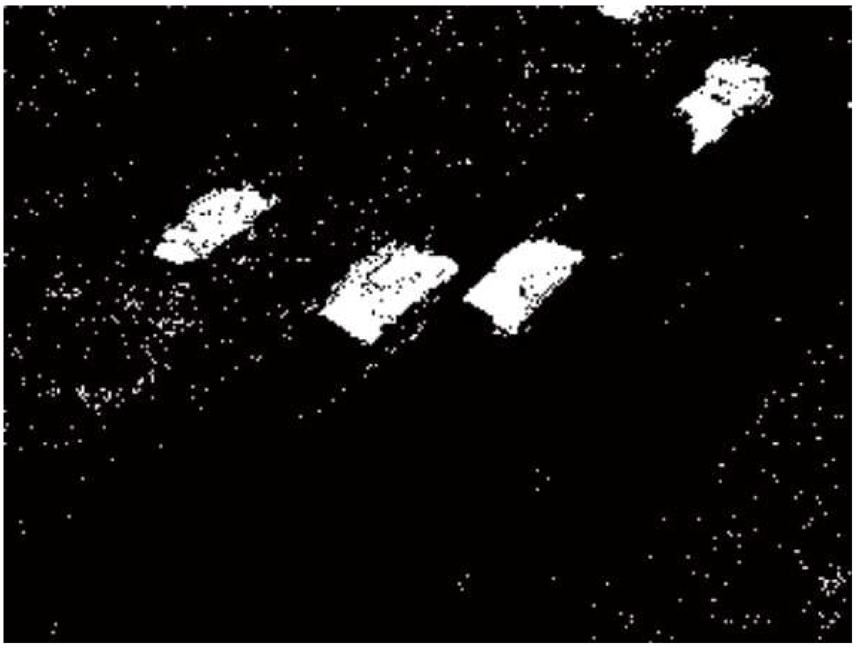
Morphological operations [29] are used for segmentation of objects, removing noise and complete the missing parts of the objects. They are generally applied in binary images by using a structuring element. Structuring element is a matrix which contains 0's and 1's and generally selected in size 3x3. The origin of the structuring element is at the center pixel. It is shifted over the image and at each pixel of the image its elements are compared with the ones on the image. If the two sets match the condition defined by the set operator (e.g. if element by element multiplication of two sets exceeds a certain value), the pixel underneath the origin of the structuring element is set to a pre-defined value (0 or 1 for binary images). In the proposed algorithm, two fundamental morphological operations erosion and dilation are applied to binary image which is obtained by foreground segmentation.

3.1.2.1.1 Erosion

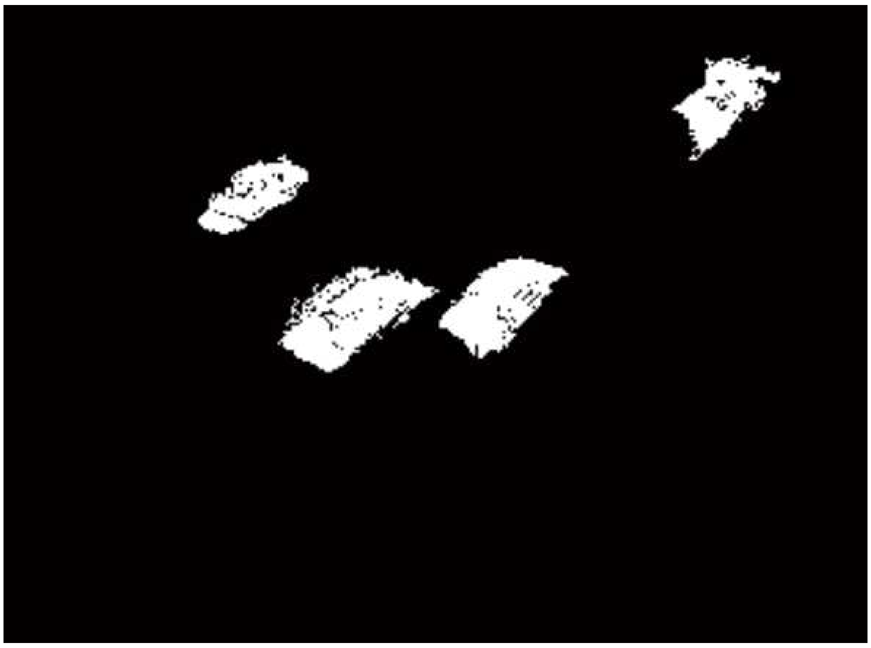
Erosion operator erodes away the region boundaries of the foreground pixels. A structuring element which has been utilized for this purpose is shown in (3.10). Each foreground pixel in the input image is aligned with the center of the structuring element.

$$SE_{erosion} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{3.10}$$

If, for each pixel having a value 1 in the structuring element, the corresponding pixel in the image is a foreground pixel, then input pixel is not changed. However, if any of the surrounding pixels (considering 8-connectedness) of the center pixel belong to background, the input pixel is also set to the background value. The effect of this operation is to remove any foreground pixel that is not completely surrounded by other white pixels as shown in Figure 3.3. As a result, small regions which correspond to noises are eliminated, foreground regions shrink and holes inside a region grow.



(a)



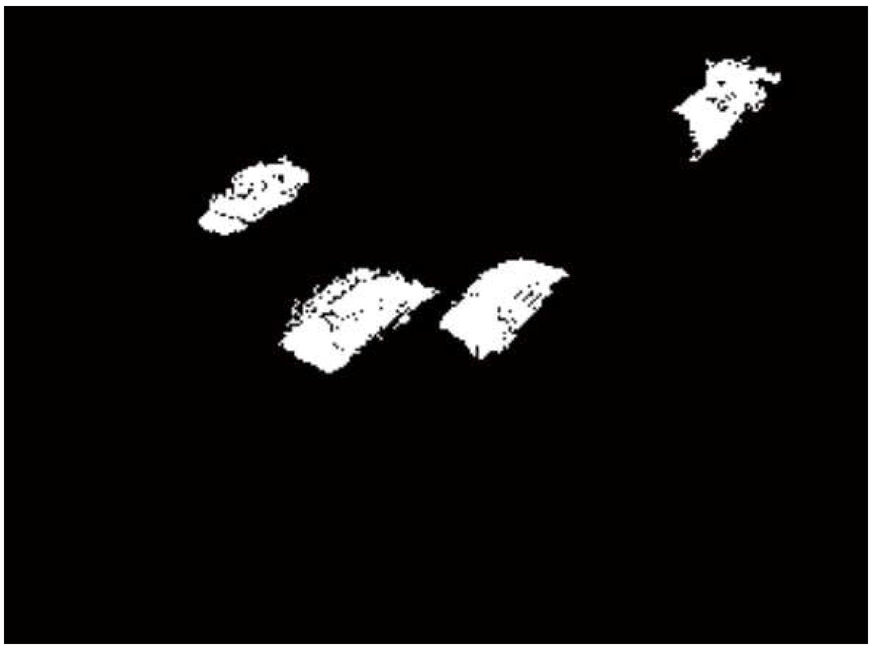
(b)

Figure 3.3. Effect of erosion operation (a) Output image of the foreground segmentation part (b) Eroded image

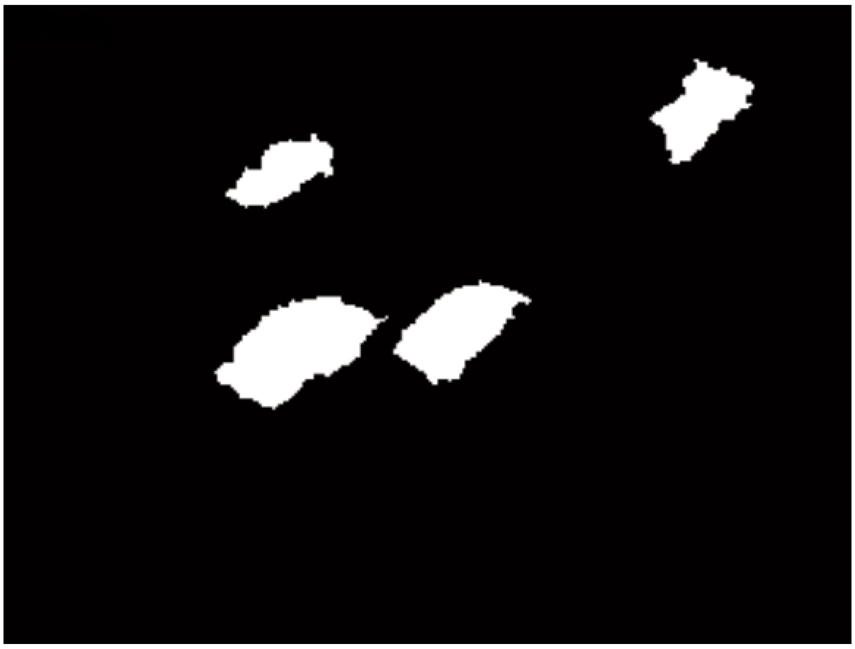
3.1.2.1.2 Dilation

Dilation is the dual operation of erosion. The same structuring element in (3.10) has been used. In this case, the structuring element works on background pixels instead of foreground pixels, with the same way defined in erosion. This time, foreground regions grow, while holes inside the regions shrink as it is depicted in Figure 3.4.

While removing the noise, erosion operation might disconnect the links between loosely connected regions. When the connectedness of a region is lost, it is possible that it is not treated as foreground anymore. In addition, strongly connected regions are affected by erosion from their boundaries. Dilation operation recovers that shrinkage caused by erosion.



(a)



(b)

Figure 3.4. Effect of dilation operation (b) Eroded image (a) Dilated image

3.1.2.2 Connected Component Labeling and Area Filter

One of the most common operations in computer vision is finding the connected components in an image. Connected components form a candidate region, in that way it represents an object. Connected component labeling segments objects by using the divisions between regions and labels them as different objects. Connected component labeling assigns a unique label to all points in the same object by grouping the pixels in an image into components based on pixel connectivity. The algorithm [30] adapted to the proposed system in this thesis works as described below:

1. Image is raster scanned.
2. If the foreground pixel (having value 1) is came across
    a. If one of the pixels on the left, on top, on the upper left or on the upper right (8-connectivity) is labeled, this label is copied as the label of the current pixel.
    b. If two or more of those neighbors have a label, one of the labels is assigned to the current pixel and all of the labels are marked as equal by forming an equivalence table.
    c. If none of the neighbors has a label, current pixel is assigned to a new label.
3. On the image all pixels are scanned and assigned label according to the rule in Step 2.
4. After scan is finished, labels representing the same group of pixels in the equivalence table are merged and given a single label.
5. Image is scanned once more to replace old labels with the new ones.

All isolated groups of pixels are given a distinct label at the end of this algorithm. After connected component labeling algorithm, the area of each distinct object region is obtained. By considering the average area of a vehicle in highway scene, a threshold value is determined experimentally. In our proposed algorithm, this threshold is selected as 20 pixels. Regions having an area below this value are not desired as moving vehicles and they are removed from the change mask. At the end, blobs which correspond to the vehicles are obtained as it is seen in Figure 3.5.

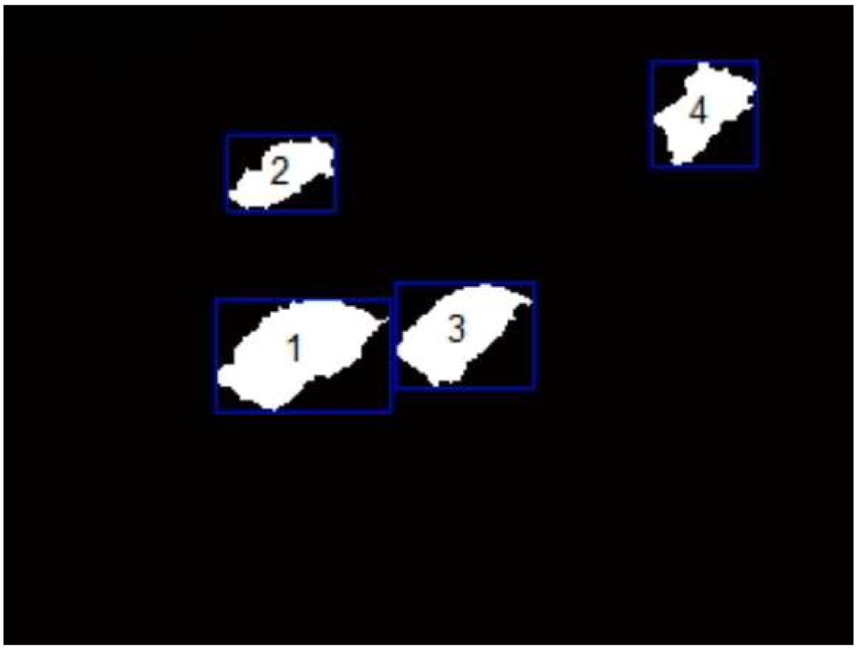


Figure 3.5. Connected component labelling and area filtering on the image

3.2 FEATURE EXTRACTION

After detection of vehicles and obtaining vehicle blobs, next step is the feature extraction which ideally produces similar values for the instances of a class. Feature extraction process is important part of the algorithm, since classification relies on it. In the classification step for both training of the classifier and classification of vehicles on test data, feature extraction is necessary. Features describe a large set of data with sufficient accuracy. In the proposed algorithm two region based features

- Blob features
- Histogram of oriented gradients (HOG) features

are used.

3.2.1 Blob Features

In the proposed algorithm it is desired to classify vehicles into car, van and truck. So, it is needed to select features to discriminate between these classes. One of the discriminative features between these vehicle classes is their sizes. For that purpose, for every vehicle blob obtained in connected component labeling part, following blob features which are also used in [9] and [10] are extracted in order to use as a feature vector in classification step:

- Area of the blob
- Ratio between area of the blob and bounding box of the blob(Figure 3.6)
- Major axis length which is a scalar specifying the length (in pixels) of the major axis of the ellipse that has the same normalized second central moments as the blob(Figure 3.7)
- Minor axis length which is a scalar specifying the length (in pixels) of the minor axis of the ellipse that has the same normalized second central moments as the blob(Figure 3.7)
- Ratio between number of white pixels and black pixels in the bounding box
- Width of the bounding box
- Height of the bounding box

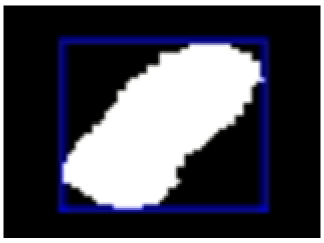


Figure 3.6. A vehicle blob and bounding box

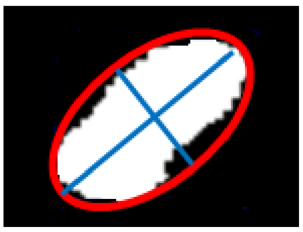


Figure 3.7. Major and minor axis

3.2.2 Histogram of Oriented Gradients (HOG) Features

Local object appearance and shape can often be characterized rather well by the distribution of local intensity gradients or edge directions even without precise knowledge of the corresponding gradient and edge positions. It is obvious that shapes and appearances of vehicle classes (car, van and truck) are discriminative features. So, in order to represent vehicles with their shapes and appearances a texture descriptor, Histogram of Oriented Gradients (HOG) features [31] are used in the proposed algorithm.

For a vehicle blob which is in grayscale obtained in detection part, grayscale vehicle image is filtered to obtain *x* and *y* derivatives of pixels with kernels in (3.11). After calculating *x* and *y* derivatives (I_x and I_y), the magnitude G and orientation θ of the gradient is also computed as in (3.12).

$$D_x = [-1 \quad 0 \quad 1] \; and \; D_y = [-1 \quad 0 \quad 1]^{\,T} \tag{3.11}$$

$$|G| = \sqrt{I_x^2 + I_y^2} \qquad \theta = \tan^{-1}\frac{I_x}{I_y} \tag{3.12}$$

Image is divided into NxN small sub-images which are called *cells* (Figure 3.9). Each pixel in a cell calculates a weighted vote for an edge orientation histogram channel based on the orientation of the gradient element centered on it, and the votes are accumulated into orientation bins over a cell. The vote is the gradient magnitude at the pixel. In other words, for each cell, the gradient directions are quantized in range of -180 to 180 degrees into K bins, each pixel votes for an orientation according to the closest bin in the range and by that way; histogram weighted by image intensity is obtained. To give less emphasis to gradients that are far from the center of the cell, a Gaussian weighting function with σ equal to one half width of the cell is used to assign a weight to the magnitude of each pixel.

Gradient is affected by illumination changes. That’s why, to provide better illumination invariance (lighting, shadows, etc.) the cells are normalized across larger regions incorporating multiple cells which are called *blocks* (Figure 3.9). Normalization within the blocks ensures that low contrast regions are stretched. In [31], histogram calculation is done by using overlapping blocks. In our implementation of HOG for the sake of simplicity, nonoverlapping blocks are used. Cells and blocks can be either rectangular (R-HOG) or radial (C-HOG) as it seen in Figure 3.8. After obtaining histograms of gradient directions vector for every cell which has a dimension *Kx1*, these vectors are concatenated to obtain one global feature vector for the vehicle image. This global feature vector which has a dimension *(number of cells) x K x 1* describes the shape of the vehicle.

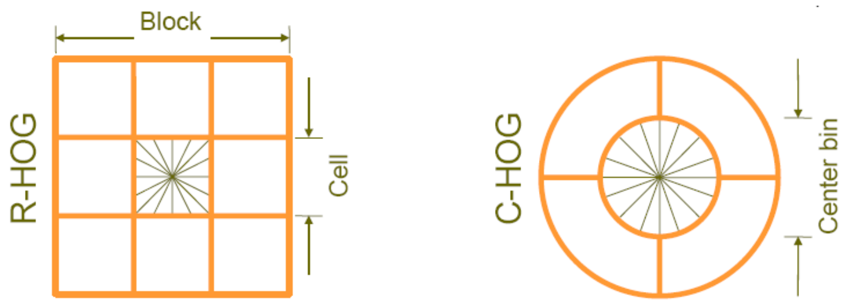


Figure 3.8. R-HOG and C-HOG geometry for cells and blocks

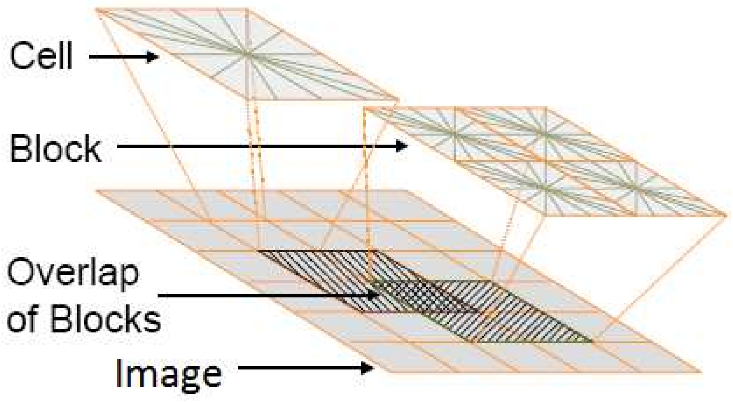


Figure 3.9. Visualization of HOG

In our proposed algorithm, according to our experimental observations which give us optimal results, HOG is computed by using 2x2 cell size blocks including 8x8 pixel size cells (N=8) with 9 orientation bins (K=9) for interval [-180^o, 180^o]. Our implementation uses R-HOG geometry. For the normalization of histograms within a block, L2-norm normalization is used as given in (3.13) where f is the normalized feature vector of a cell while v is initial feature vector of a cell, $v^{'}$ is feature vector of the all cells in a block and ε is a small constant (whose value will not influenced the results). Figure 3.10 shows example HOG representation of the vehicles shown by blue arrows which represent the gradient orientations.

$$f = \frac{v}{\sqrt{\|v'\|_2^2 + \varepsilon}} \tag{3.13}$$

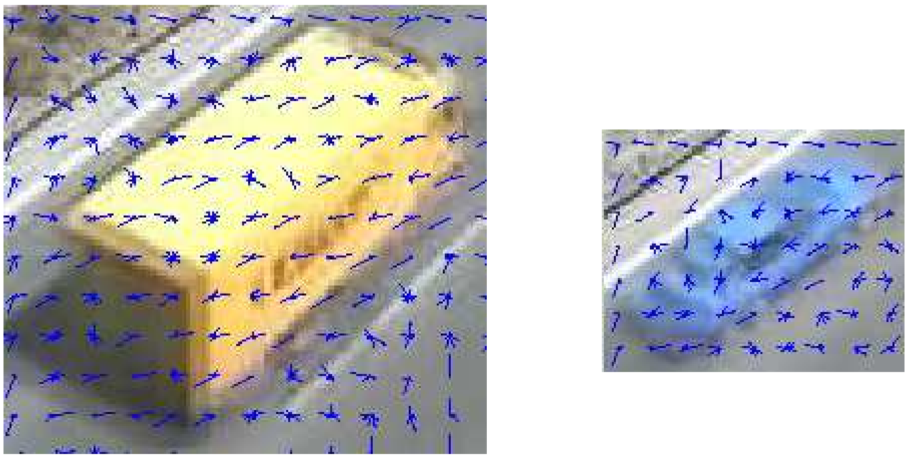


Figure 3.10. HOG representation of the vehicles

HOG algorithm is relatively invariant to local geometric and photometric transformations. Because within a cell rotations and translations do not affect HOG values and illumination invariance achieved through normalization. In addition, the spatial and orientation sampling densities can be tuned according to application.

3.3 CLASSIFICATION OF VEHICLES

The last part of the proposed algorithm is the classification step which will give the vehicle class as car, van or truck. In order to be able to discriminate between car, van and truck classes by using selected features mentioned in feature extraction part, a robust classifier should be used. For this purpose, a popular classifier also mentioned in section 2.1.2.2, Support Vector Machine (SVM) is selected. In our implementation, we have used one-class SVM with linear kernel and MATLAB SVM [32] library is utilized.

SVM is a supervised classifier so there should be a training phase in order to obtain a classifier. For that purpose, for each of the vehicle classes, car, van and truck, manually labelled training set is created. The details about training sets are given in Chapter 4. In the proposed algorithm, a hierarchical classification is performed as seen in Figure 3.11. Classifier-1 is trained with the set of feature vectors belonging to car class and other class which includes the feature vectors of van and truck classes. Classifier-2 is trained with the set of feature vectors belonging to van class and truck class. After the training phase, two classifiers are obtained where Classifier-1 discriminates between car and other (van+truck) classes, Classifier-2 discriminates between van and truck classes. In the hierarchical classification phase, by using these classifiers a new feature vector coming from a test data is firstly classified with Classifier-1 and labelled as car or other. If it is labelled as other, then the feature vector is classified again with Classifier-2 and it is decided that it is van or truck. In the proposed algorithm it is preferred to use a hierarchical classification with two classifiers, because one-class SVM is computationally simpler than multi-class SVM. In addition, cross validation is performed by splitting the training set into partitions, training on one and testing on another in order to optimize the classification accuracy. This procedure is repeated several times in order to find the optimal value of the parameter C. In our algorithm C is chosen 0.01.

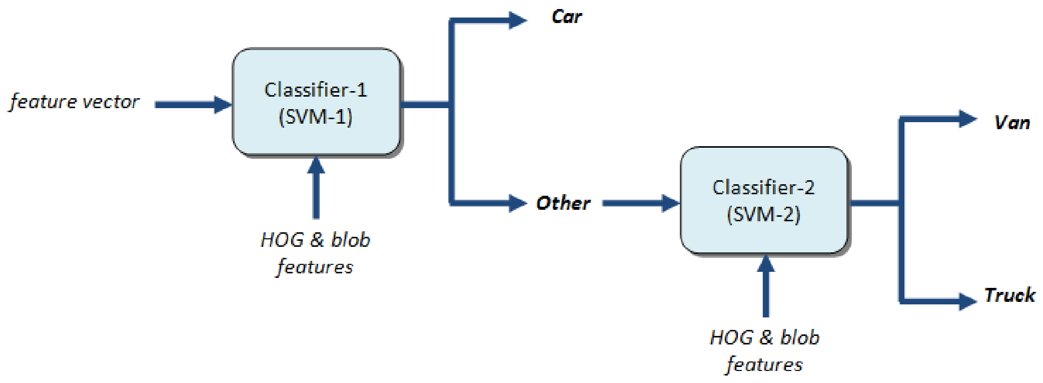


Figure 3.11. Proposed hierarchical classification

In our algorithm, it is proposed to use blob and HOG features for the classification (Figure 3.11). So, these features for the vehicle images in each training set are obtained and concatenated in order to create a global feature vector for every labelled vehicle image. Every feature vector has 7 blob features which are mentioned earlier. However, while obtaining HOG features, there exists a dimension problem in HOG feature vectors for each vehicle image, since their sizes are not same. In order to get rid of this problem, maximum sized vehicle image in the total training set (car+van+truck training sets) is obtained and before HOG features are extracted, every vehicle image is scaled to that maximum size with linear interpolation. By that way equal sized feature vectors are obtained for the classifiers. In Figure 3.14 and Figure 3.15, sample output of the proposed algorithm on two different test videos can be observed. Detailed information on these test videos are presented in Chapter 4.

In this study, in order to see the effectiveness of the proposed classification two experiments are conducted besides the experiments of proposed algorithm. One of them is classification into car, van and truck classes directly by using multi-class linear SVM trained with HOG and blob features together as it is seen in Figure 3.12. The other experiment is making a hierarchical classification by using blob features in Classifier-1 and using HOG features in Classifier-2 as it is seen in Figure 3.13. All experiments are presented in Chapter 4.

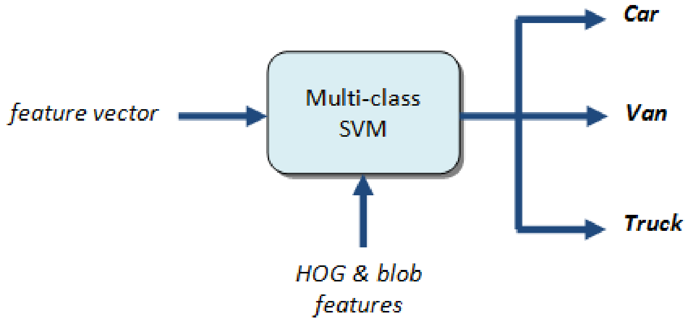


Figure 3.12. Classification with multi-class SVM

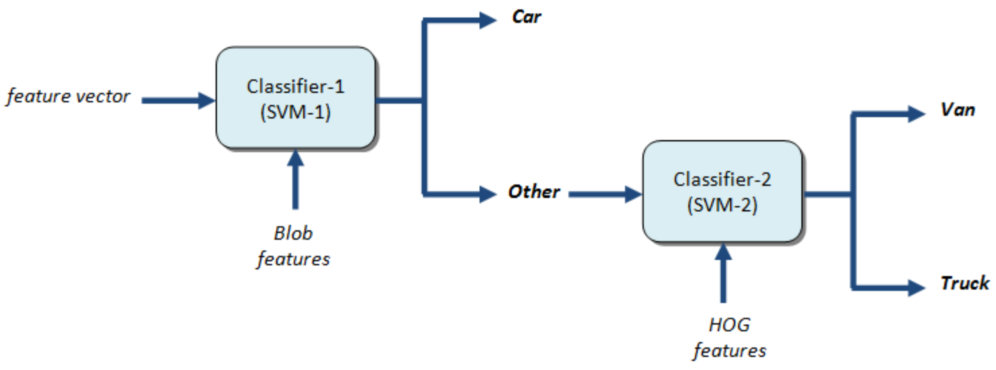


Figure 3.13. Hierarchical classification firstly using blob features then using HOG features

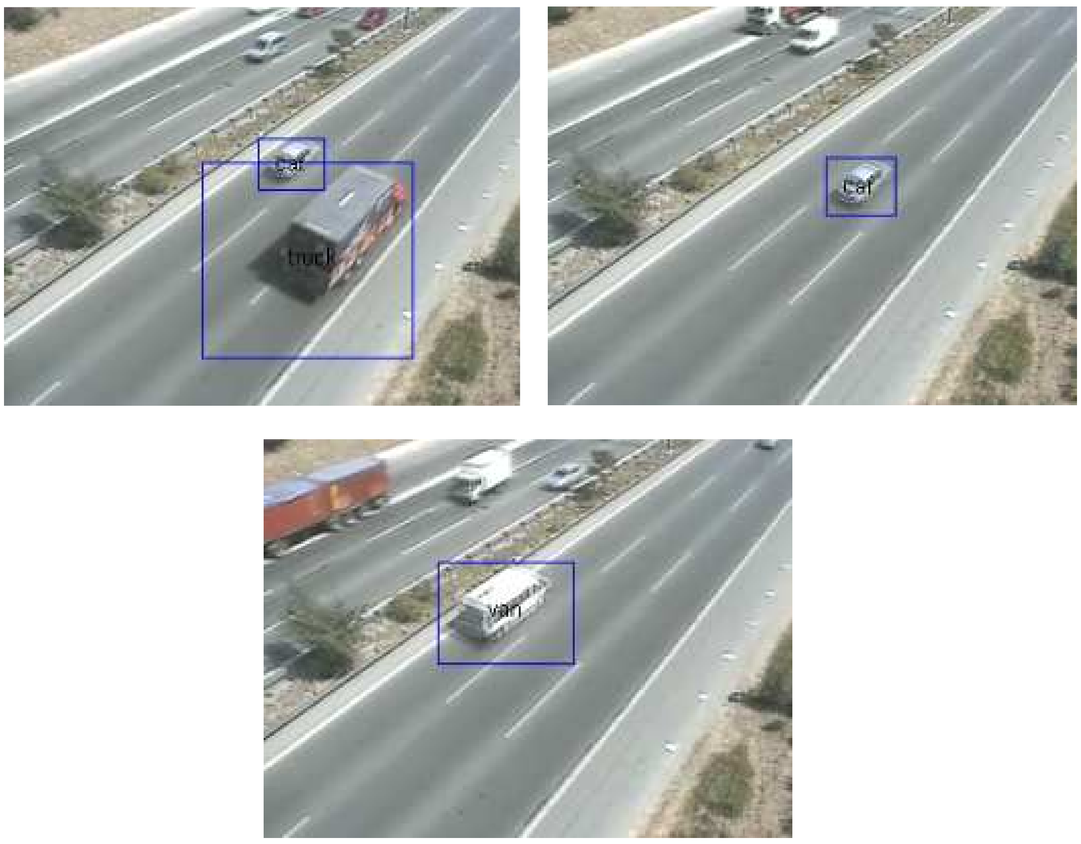


Figure 3.14. Sample output of the proposed algorithm on test video-1

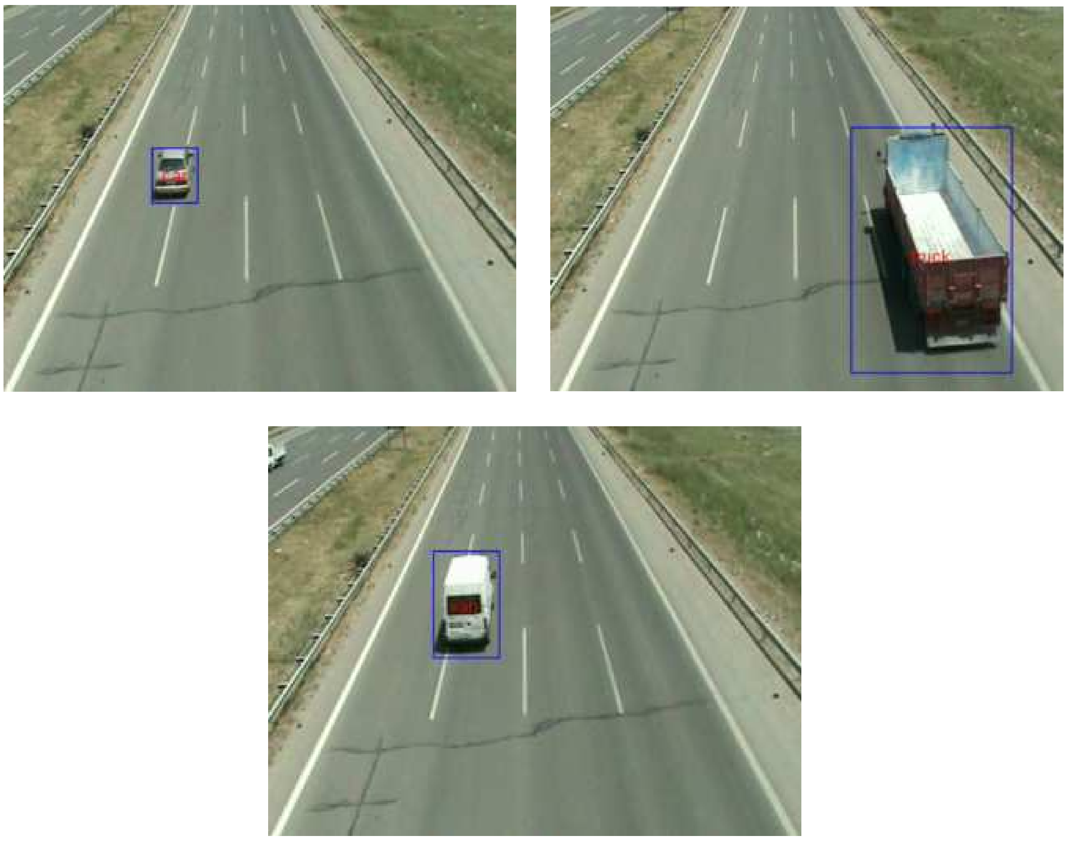


Figure 3.15. Sample output of the proposed algorithm on test video-2

CHAPTER 4

EXPERIMENTAL RESULTS

This chapter gives the evaluation of the proposed algorithm. Firstly, definition of metrics in order to evaluate the performance of the algorithm is given. Secondly, the results of the algorithm on different videos will be introduced and explained.

4.1 DEFINITION OF METRICS

The metrics used to evaluate the performance is defined as follows:

- **True Positive (TP):** is the test result that detects the condition when the condition is present.
- **False Positive (FP):** is the test result that detects the condition when the condition is absent.
- **True Negative (TN):** is the test result that does not detect the condition when the condition is absent.
- **False Negative (FN):** is the test result that does not detect the condition when the condition is present.

Table 4.1. Illustration of terminology for evaluation metrics

		Condition	
		Present	**Absent**
Test	**Positive**	True Positive	False Positive
	Negative	False Negative	True Negative

4.1.1 Recall

One of the evaluation metrics for a classifier performance is *recall*. Recall of a class is the percentage of labelled instances of that class that are predicted as this class.

$$Recall = \frac{TP}{(TP + FN)} \tag{4.1}$$

4.1.2 Precision

Precision of a class is a measure of the accuracy provided that this class is predicted.

$$Precision = \frac{TP}{(TP + FP)} \tag{4.2}$$

4.1.3 Accuracy

Accuracy is the overall correctness of the classifier and is calculated as the sum of correct classifications divided by the total number of classifications.

$$Accuracy = \frac{TP + TN}{(TP + FP + TN + FN)} \tag{4.3}$$

4.1.4 Confusion Matrix

A *confusion matrix* is a specific table that allows visualization of the performance of an algorithm. Each column of the matrix represents the number of instances in a predicted class, while each row represents the number of instances in an actual class.

4.2 EXPERIMENTAL RESULTS

The main difficulty in traffic analysis system is that there is no commonly used public data set. In this thesis study, we have conducted our experiments on the videos that have been captured on some highways in Ankara and İzmir. Experiments are conducted on four different highway videos as seen in Figure 4.1, Figure 4.2, Figure 4.3 and Figure 4.4.

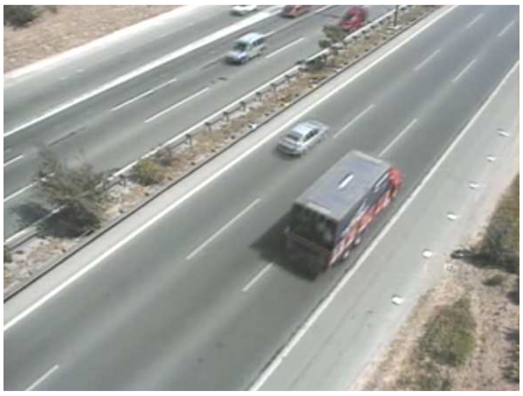


Figure 4.1. Video-1

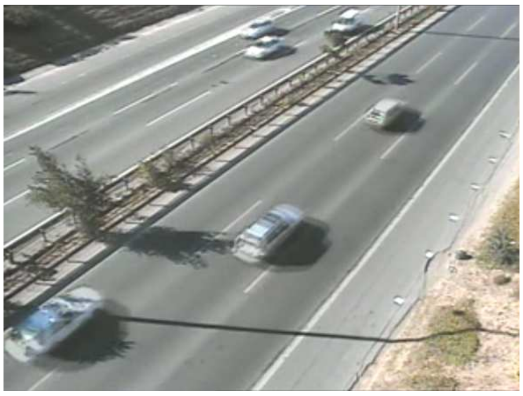


Figure 4.2. Video-2

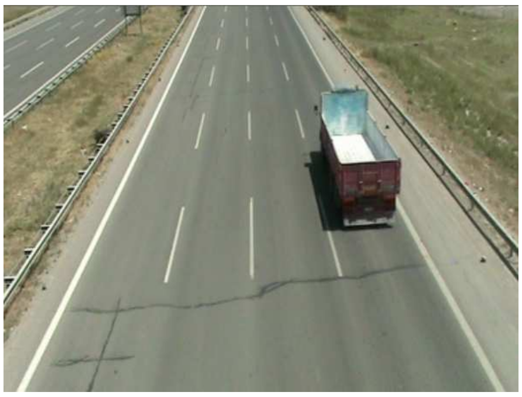


Figure 4.3. Video-3

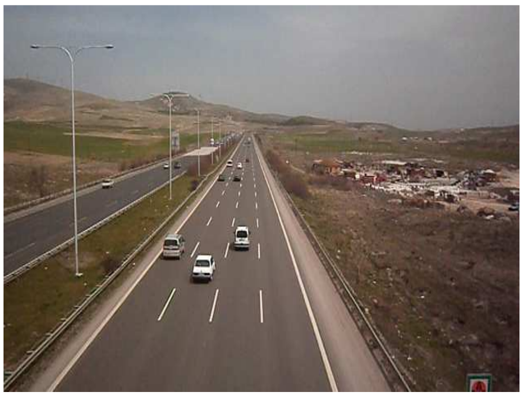


Figure 4.4. Video-4

Video-1 and Video-2 are recorded on a bridge looking at the same highway. However, they are not captured from exactly the same angular direction. In addition, they are recorded at different times of the day. Video-1 is captured in the morning time while Video-2 is captured at late-afternoon, so the lighting and shadows are different. The resolution of Video-1 and Video-2 is 320 x 240 pixels and they have a frame rate of 25 fps.

Video-3 and Video-4 are recorded on a bridge from different highways. As in the previous videos they are captured as their views of the vehicles are nearly same, but not exactly the same angular direction. They are also recorded at different times of the day as in Video-1 and Video-2 so that the lighting and shadows are different. Video-3 is captured in the morning time while Video-4 is captured in the afternoon time. Video-3 has a frame rate of 25 fps and has a resolution of 384 x 288 pixels. Video-4 has a frame rate of 30 fps and has a resolution of 1280 x 720 pixels. Their resolutions are also different.

It is an important point that there is no information about the highways and the cameras that capture these videos. We do not have any camera parameters like height of the camera, angular direction of the camera; and cameras are not calibrated.

Speed of the algorithm which runs on Core 2 Duo 2.2GHz PC and on MATLAB is calculated as around 3fps for a 320 x 240 pixels resolution video. Three experiments are conducted on the videos. In the first experiment in order to see the difference between using HOG and blob features together and alone, three types of classification are conducted on the videos by using:

- HOG and blob features together,
- Only HOG features,
- Only blob features.

In the second experiment proposed hierarchical classification versus multi-class SVM performance is observed. Finally, in the third experiment, proposed hierarchical classification versus a hierarchical classification which uses blob features in Classifier-1 and uses HOG features in Classifier-2 performance is observed.

4.2.1 Experiment-1

In the first experiment, effectiveness of the selected features is observed. For that purpose three classifications are done by using HOG and blob features together, using only HOG features and using only blob features in Classifier-1 and Classifier-2.

4.2.1.1 Results of Video-1 and Video-2

In this experiment, a training set including car, van and truck images is created by using a video which has exactly same characteristics as Video-1. In the training set each of the vehicle classes has 750 samples. As it is also mentioned in Section 3.3, in order to eliminate the dimension problem in HOG feature vectors each vehicle image is scaled to the maximum sized image in the training set. In this training set maximum image size is 170 x 227 pixels.

By using this training set, proposed algorithm is tested on Video-1 and Video-2 which have nearly same view of the highway, but not the same angular direction and are captured at different times of the day. Total length of these test videos on which the experiments are conducted is 15 minutes and there are 525 vehicles which include 416 cars, 57 vans and 52 trucks. Table 4.2 and Table 4.3 show the classification results when HOG and blob features are used together as proposed in our algorithm. Table 4.4 and Table 4.5 show the classification results when only HOG features are used while Table 4.6 and Table 4.7 show the classification results when only blob features are used.

Table 4.2. Results of Video-1 and Video-2 by using HOG and blob features together

Vehicle Class	**Precision**	**Recall**	**Total Accuracy**
car	%98.7	%94.2	%91.6
van	%71.4	%78.9	
truck	%67.6	%84.6	

Table 4.3. Confusion matrix of Video-1 and Video-2 by using HOG and blob features together

	Prediction			
Real		car	van	truck
	car	392	10	14
	van	5	45	7
	truck	0	8	44

Table 4.4. Results of Video-1 and Video-2 by using only HOG features

Vehicle Class	**Precision**	**Recall**	**Total Accuracy**
car	%98.1	%89.1	%84.9
van	%46.5	%71.9	
truck	%57.6	%65.3	

Table 4.5. Confusion matrix of Video-1 and Video-2 by using only HOG features

	Prediction			
Real		**car**	**van**	**truck**
	car	371	31	14
	van	5	41	11
	truck	2	16	34

Table 4.6. Results of Video-1 and Video-2 by using only blob features

Vehicle Class	**Precision**	**Recall**	**Total Accuracy**
car	%90.1	%90.1	%78.0
van	%46.0	%26.3	
truck	%60.6	%38.4	

Table 4.7. Confusion matrix of Video-1 and Video-2 by using only blob features

	Prediction			
Real		**car**	**van**	**truck**
	car	375	35	6
	van	35	15	7
	truck	6	26	20

As it is observed from above tables, using HOG and blob features together gives the best accuracy comparing to the others. Figure 4.5 and Figure 4.6 show the examples of correct classification of the vehicles with the proposed algorithm on Video-1 and Video-2, respectively.

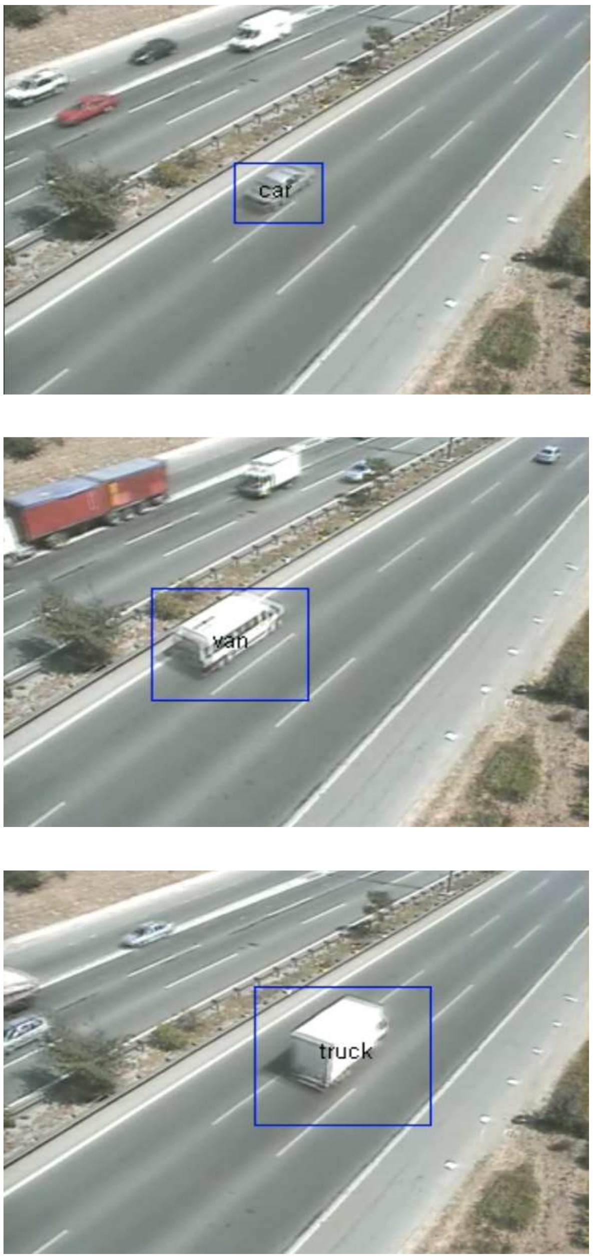
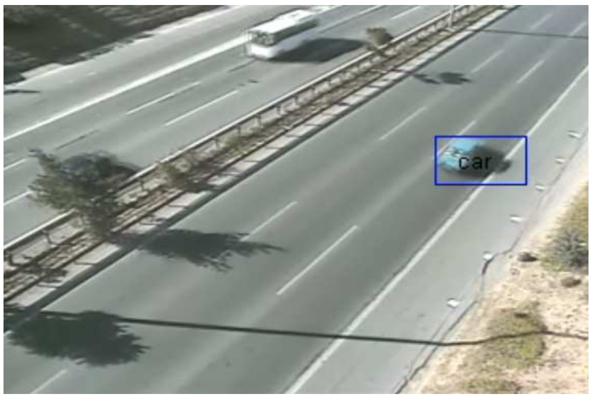
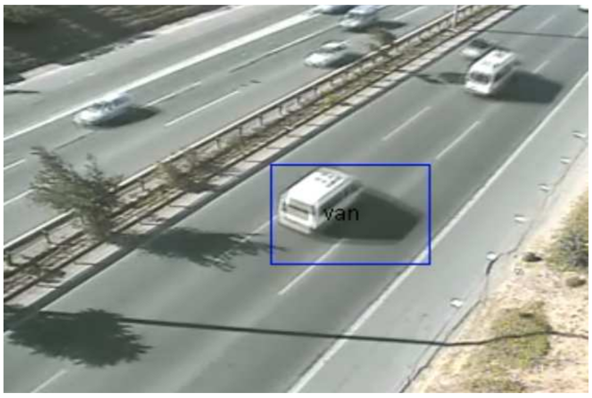


Figure 4.5. Examples of correct classification of the vehicles on Video-1





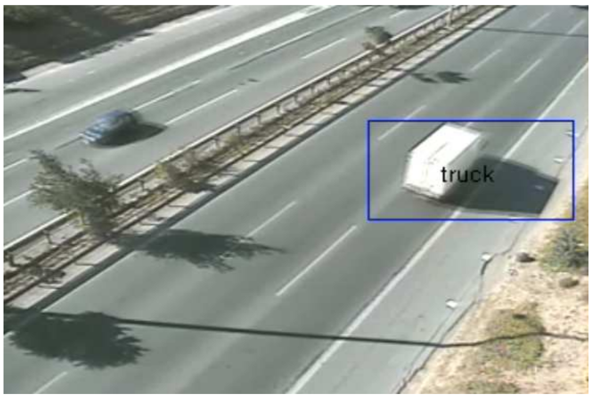


Figure 4.6. Examples of correct classification of the vehicles on Video-2

On Video-1 and Video-2 most of the misclassifications are due to the perspective occlusion and shadow. Due to the perspective of these videos, one vehicle occludes the other vehicle. This occlusion causes merging of the vehicle blobs as seen in Figure 4.7. These merged blobs are interpreted as one vehicle by the algorithm. That's why, misclassification occurs as in Figure 4.8. In the same way, shadow causes merging of the vehicle blobs as seen in Figure 4.9 and causes misclassification as seen in Figure 4.10. In these videos, %97 of the misclassifications is due to the shadow and perspective occlusion. The perspective occlusion and shadow problems especially affect van and truck classes because of their large sizes. So, classification performance decreases in van and truck classes compared to car class as it is seen in Table 4.2 and Table 4.3.

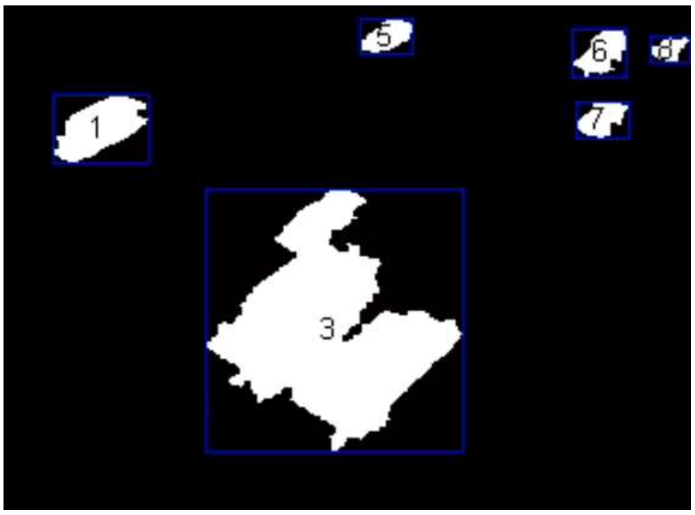


Figure 4.7. Merged blobs due to the perspective occlusion

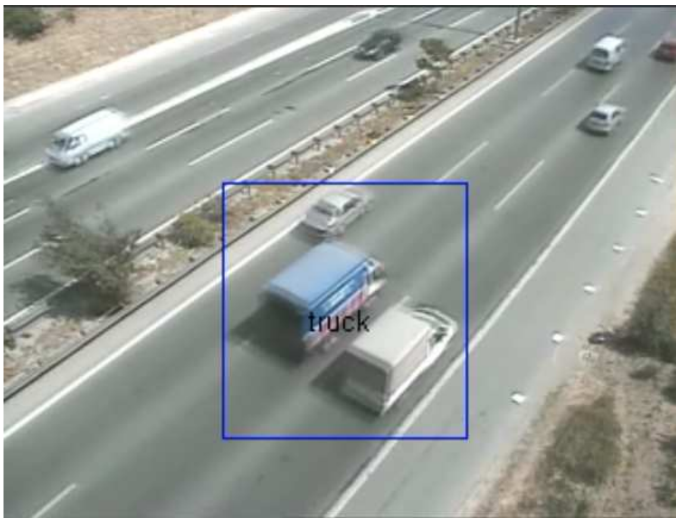


Figure 4.8. Misclassification due to the perspective occlusion

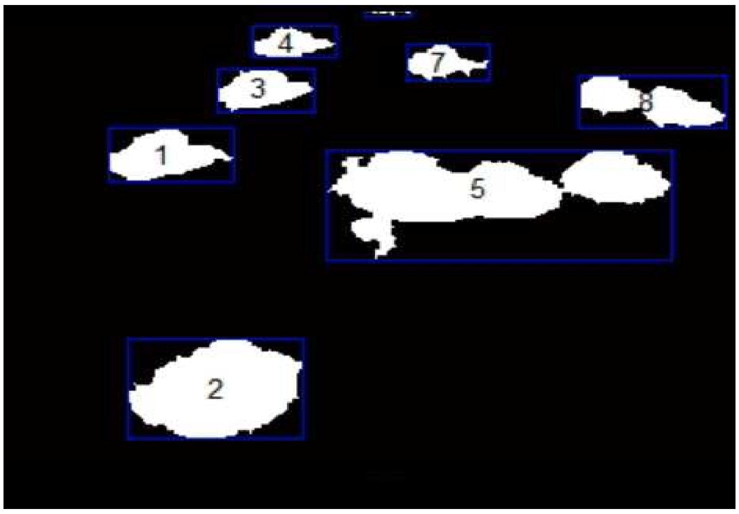


Figure 4.9. Merged blobs due to the shadow

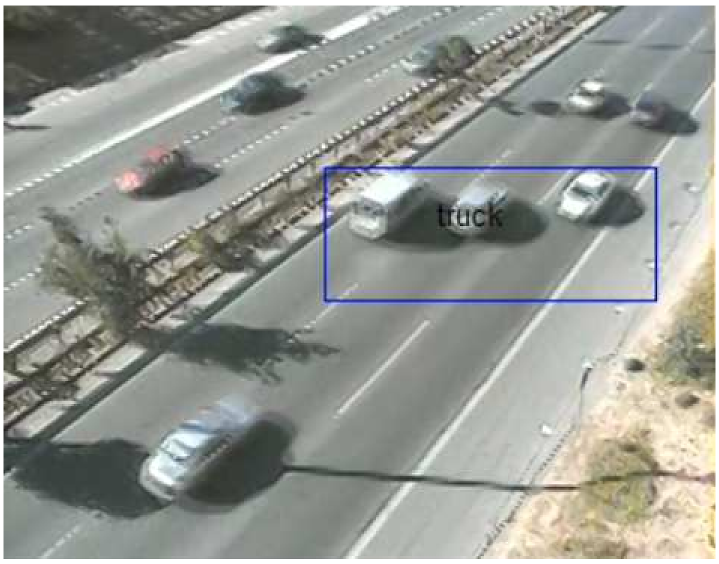


Figure 4.10. Misclassification due to the shadow

Using HOG and blob features together has slightly better performance than using only HOG features; because some vehicles are similar in terms of shape and appearance, although they are in different classes. Using only HOG features causes these vehicles with similar shape to be labelled incorrectly. By including blob features to the HOG features, these similar vehicles in terms of shape become distinguishable by their sizes. In addition, using only blob features gives the worse performance comparing to the others; because some vehicles have similar sizes although they are in different classes. Using only blob features causes such vehicles to be labelled incorrectly. In this experiment especially van and truck classes are confused with each other, since mostly their sizes are very close to each other. In Figure 4.11 height, width and extent from the blob features are plotted together, while in Figure 4.12 major axis length, minor axis length and ratio of the white pixels to the black pixels in the bounding box of the vehicle blob are plotted together. As it is seen, in the training set, blob features of all classes coincide, especially van and truck classes. This coincidence causes incorrect classification when only blob features are used. Figure 4.13 and Figure 4.14 show examples of incorrect classification when only blob features are used. Although vehicle is truck it is labelled as van in Figure 4.14 and although vehicle is van it is labelled as car in Figure 4.13.

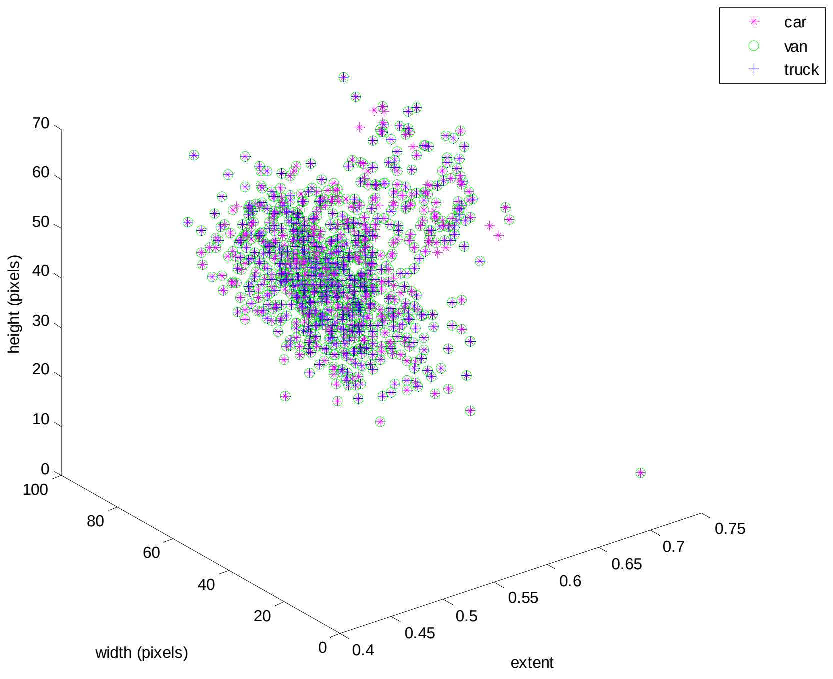


Figure 4.11. Coincidence of blob features visualization-1

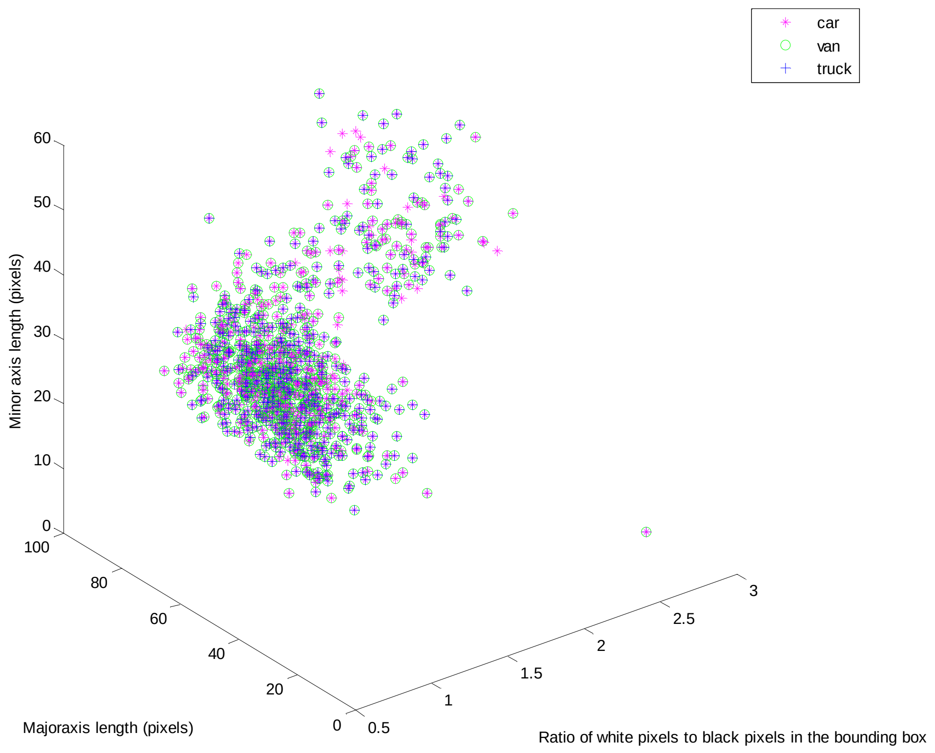


Figure 4.12. Coincidence of blob features visualization-2

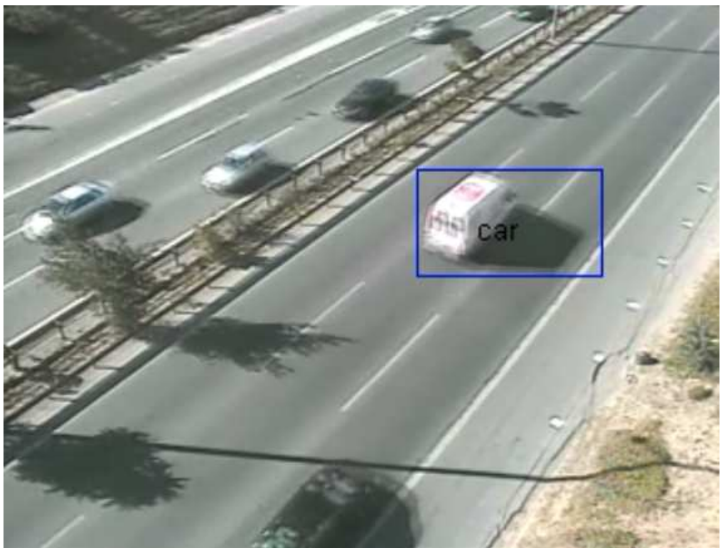


Figure 4.13. An example of incorrect classification when only blob feature used

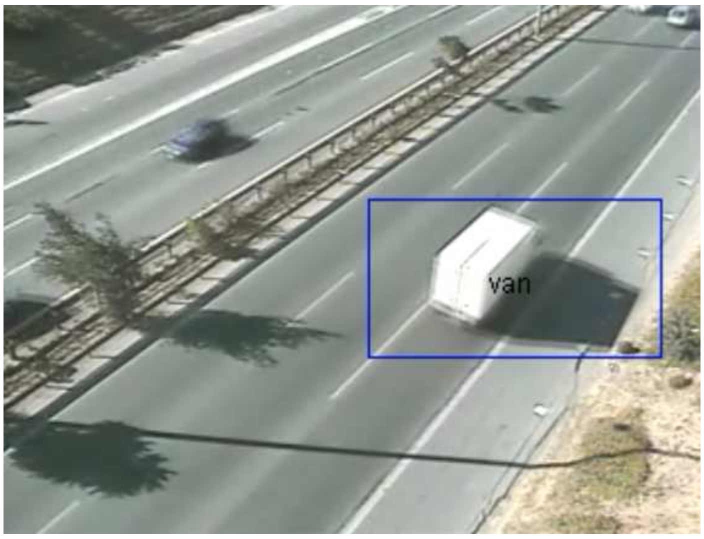


Figure 4.14. An example of incorrect classification when only blob feature used

4.2.1.2 Results of Video-3 and Video-4

In this experiment, a training set including car, van and truck images is created by using a video which has exactly same characteristics as Video-3. In the training set there are 450 samples for car, 130 samples for van and 322 samples for truck. It is intended to have same number of samples for each class in order to prevent the bias between them, but in the training video van and truck vehicles are rarely observed. In the training set the maximum image size which is needed for scaling of detected vehicle images before HOG feature extraction is 202 x 169 pixels.

By using this training set, proposed algorithm is tested on Video-3 and Video-4. These videos are captured from different highways at different times of the day. Their resolution is different. They do not have the same angular direction. Total length of these test videos on which the experiments are conducted is 15 minutes and there are 109 vehicles which include 65 cars, 16 vans and 28 trucks. Table 4.8 and Table 4.9 show the classification results when HOG and blob features are used together as proposed in our algorithm. Table 4.10 and Table 4.11 show the classification results when only HOG features are used while Table 4.12 and Table 4.13 show the classification results when only blob features are used.

Table 4.8. Results of Video-3 and Video-4 by using HOG and blob features together

Vehicle Class	**Precision**	**Recall**	**Total Accuracy**
car	%100	%98.5	%98.1
van	%88.8	%100	
truck	%100	%93.1	

Table 4.9. Confusion matrix of Video-3 and Video-4 by using HOG and blob features together

	Prediction			
Real		**car**	**van**	**truck**
	car	64	1	0
	van	0	16	0
	truck	0	1	27

Table 4.10. Results of Video-3 and Video-4 by using only HOG features

Vehicle Class	**Precision**	**Recall**	**Total Accuracy**
car	%98.5	%98.5	%95.4
van	%82.3	%87.5	
truck	%96.2	%92.8	

Table 4.11. Confusion matrix of Video-3 and Video-4 by using only HOG features

		Prediction		
Real		**car**	**van**	**truck**
	car	64	1	0
	van	1	14	1
	truck	0	2	26

Table 4.12. Results of Video-3 and Video-4 by using only blob features

Vehicle Class	**Precision**	**Recall**	**Total Accuracy**
car	%76.9	%76.9	%66.0
van	%15.3	%12.5	
truck	%62.5	%71.4	

Table 4.13. Confusion matrix of Video-3 and Video-4 by using only blob features

		Prediction		
Real		car	van	truck
	car	50	4	11
	van	13	2	1
	truck	1	7	20

As it is observed from above tables, using HOG and blob features together gives the best accuracy comparing to the others as in the case of Video-1 and Video-2. Figure 4.15 and Figure 4.16 show the examples of correct classification of the vehicles with the proposed algorithm on Video-3 and Video-4, respectively.

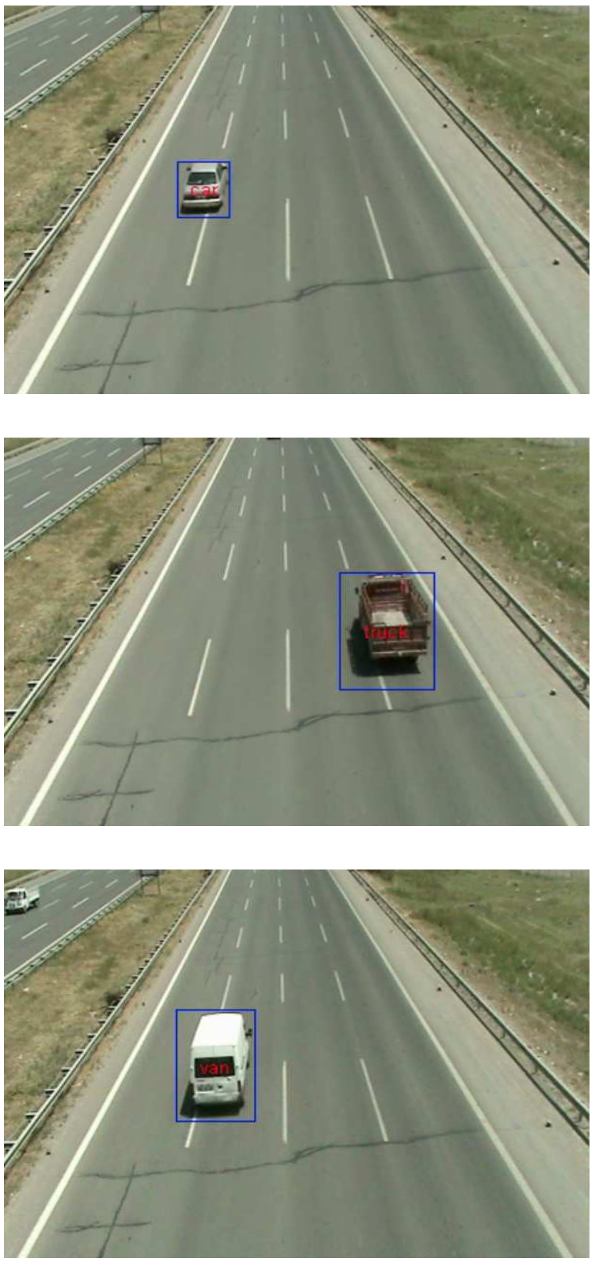


Figure 4.15. Examples of correct classification of the vehicles on Video-3

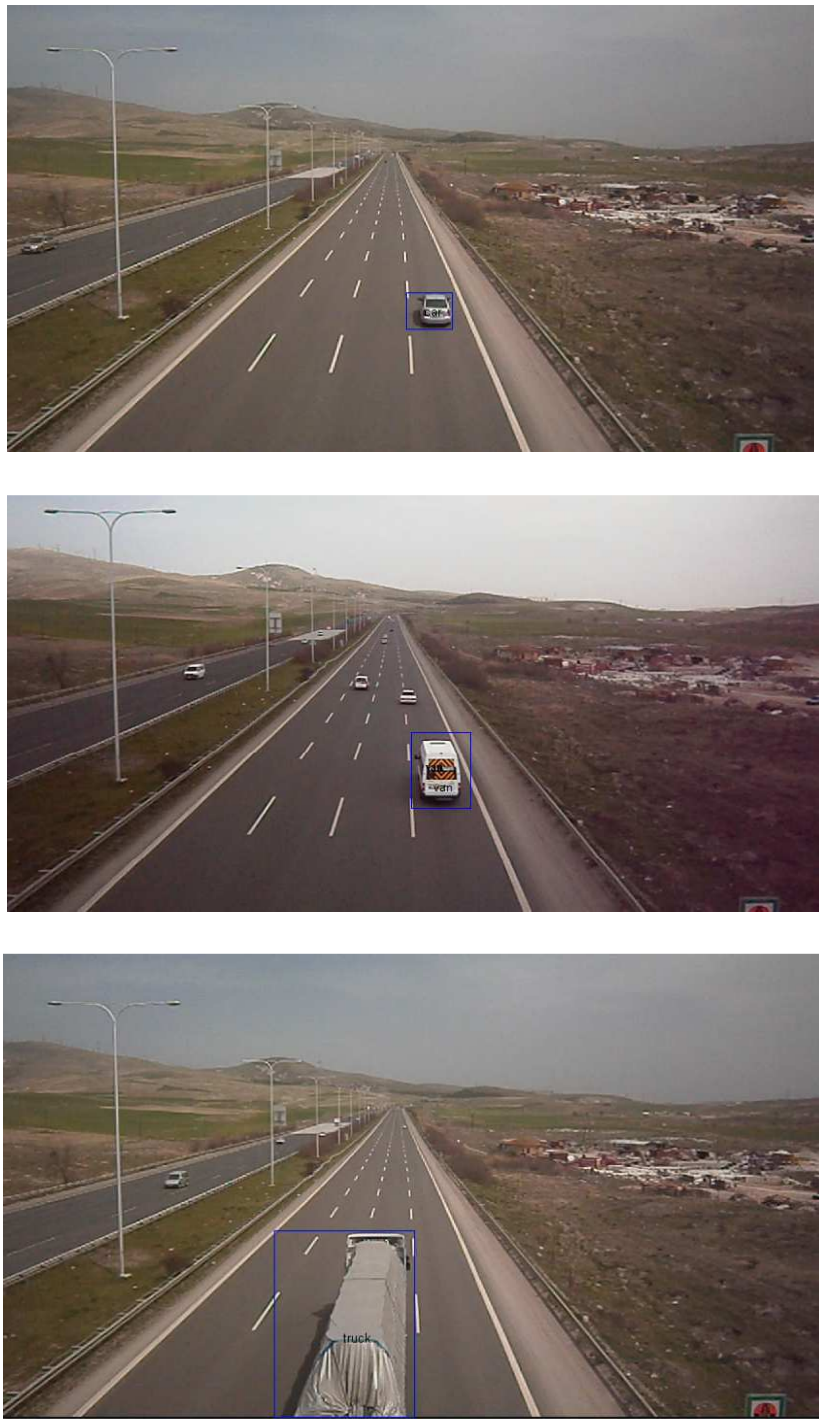


Figure 4.16. Examples of correct classification of the vehicles on Video-4

The results of Video-3 and Video-4 are much better than Video-1 and Video-2. The reason is that misclassifications due to perspective occlusion do not exist. %95 of the misclassifications is due to the shadow. Shadow problem especially affect van and truck classes because of their large sizes. So, classification performance decreases in van and truck classes compared to car class as it is seen in Table 4.8 Table 4.2and Table 4.9. Figure 4.17 and Figure 4.18 show an example misclassification due to the shadow.

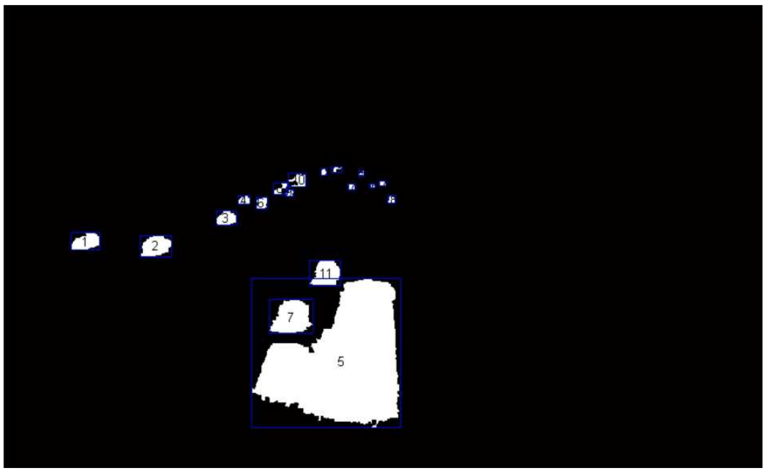


Figure 4.17. Merged blobs due to the shadow

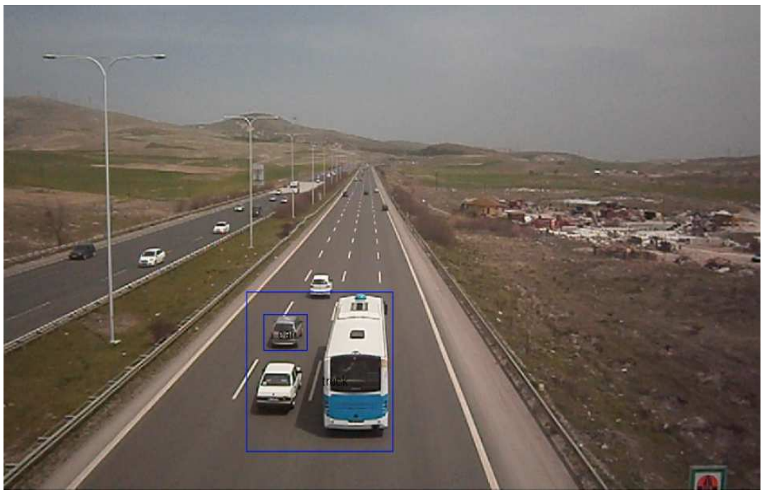


Figure 4.18. Misclassification due to the shadow

In this experiment, similar to previous experiment it is seen that using HOG and blob features together has slightly better performance than using only HOG features and using only blob features gives the worse performance comparing to the others because of the reasons mentioned in the first experiment. In Figure 4.19 height, width and extent from the blob features are plotted together, while in Figure 4.20 major axis length, minor axis length and ratio of the white pixels to the black pixels in the bounding box of the vehicle blob are plotted together. In this training set, it is again observed that blob features of all classes coincide. This coincidence causes incorrect classification when only blob features are used. Figure 4.21 and Figure 4.22 show examples of incorrect classification when only blob features are used. Although vehicle is truck it is labelled as van in Figure 4.21 and although vehicle is van it is labelled as car in Figure 4.22.

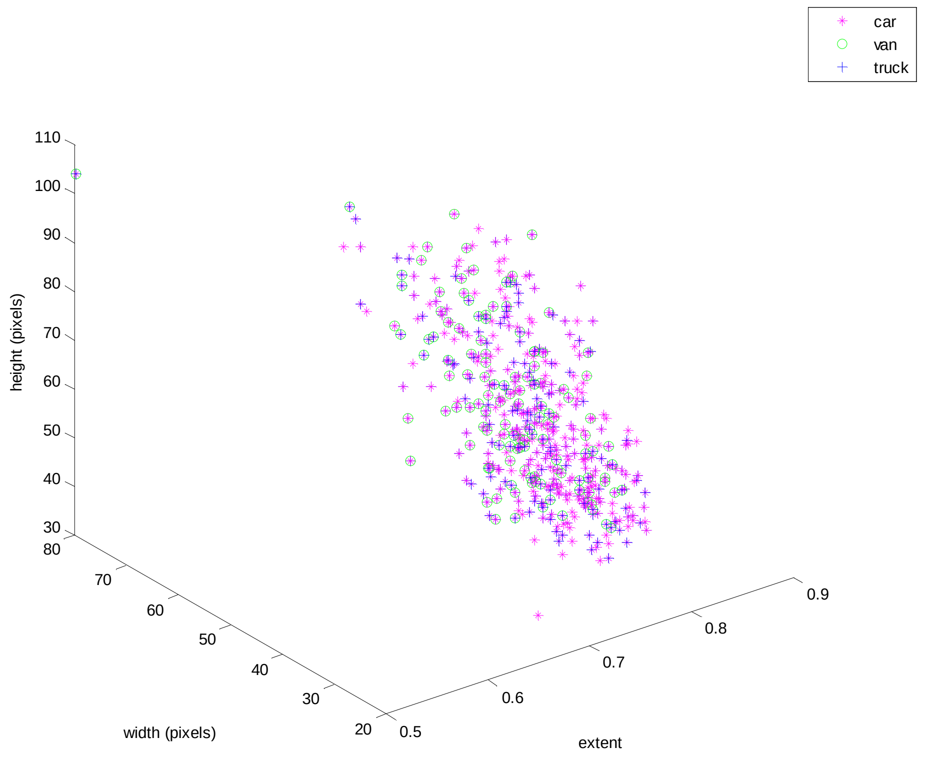


Figure 4.19. Coincidence of blob features visualization-1

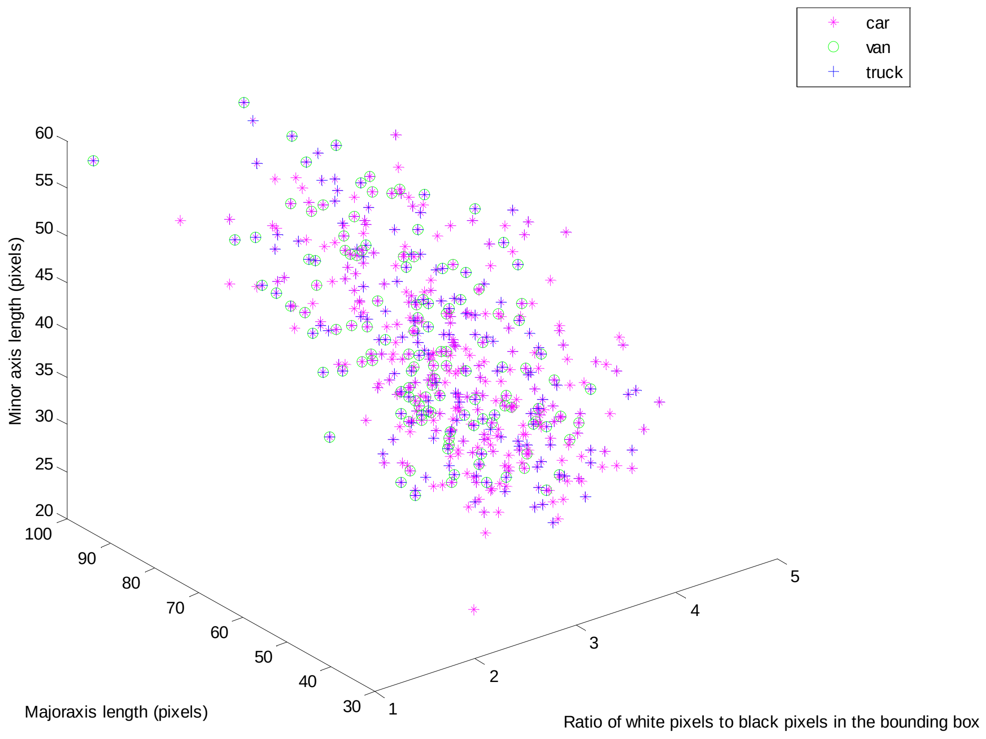


Figure 4.20. Coincidence of blob features visualization-2

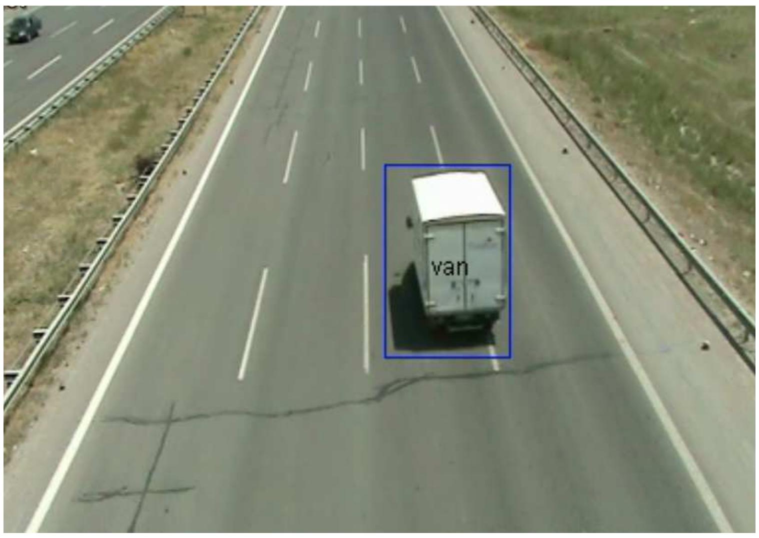


Figure 4.21. An example of incorrect classification when only blob feature used

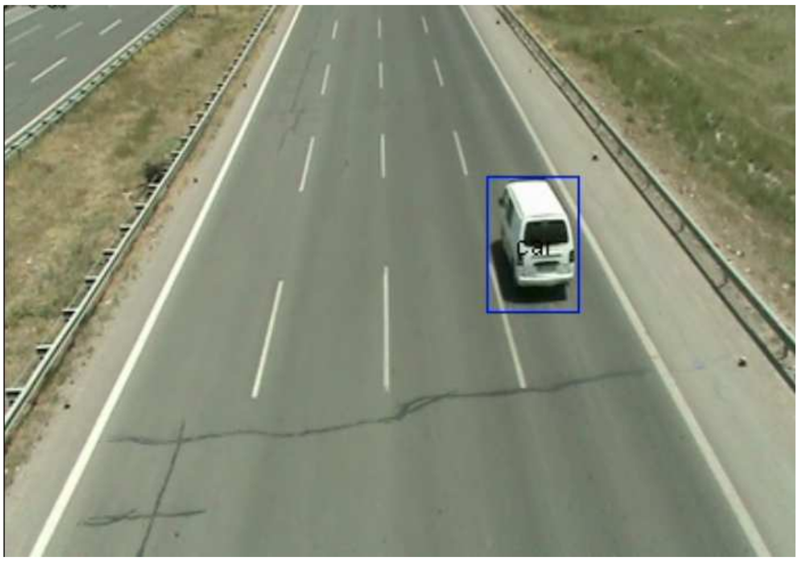


Figure 4.22. An example of incorrect classification when only blob feature used

4.2.2 Experiment-2

In the second experiment, proposed hierarchical classification versus multi-class SVM performance is observed. Combination of the HOG and blob features which give the best accuracy result in the previous experiment is used for training the classifiers. Liblinear library [55] is used for multi-class SVM with same C parameter (0.01) used in the first experiment.

4.2.2.1 Results of Video-1 and Video-2

In this experiment, same training sets in order to train the multi-class SVM and same test videos are used as in the first experiment for Video-1 and Video-2. Table 4.14 shows the classification results of multi-class SVM.

Table 4.14. Confusion matrix of Video-1 and Video-2 for multi-class SVM

	Prediction			
Real		car	van	truck
	car	393	8	15
	van	7	43	7
	truck	0	10	42

When we look at the results of hierarchical classification which is given in Table 4.3 and results of multi-class SVM, it is observed that they have almost same classification performance. Most of the misclassifications for multi-class SVM are due to the perspective occlusion and shadow as in the first experiment.

4.2.2.2 Results of Video-3 and Video-4

In this experiment, same training sets in order to train the multi-class SVM and same test videos are used as in the first experiment for Video-3 and Video-4. Table 4.15 shows the classification results of multi-class SVM.

Table 4.15. Confusion matrix of Video-3 and Video-4 for multi-class SVM

	Prediction			
Real		**car**	**van**	**truck**
	car	64	1	0
	van	0	16	0
	truck	0	3	25

When we look at the results of hierarchical classification which is given in Table 4.9 and results of multi-class SVM, it is observed that they have almost same classification performance. Most of the misclassifications for multi-class SVM are due to the shadow as in the first experiment.

4.2.3 Experiment-3

In the third experiment, performance of the proposed hierarchical classification versus performance of a hierarchical classification which uses only blob features in Classifier-1 and uses only HOG features in Classifier-2 is observed. As in the proposed classification, Classifier-1 and Classifier-2 are one class linear SVM with C equal to 0.01.

4.2.3.1 Results of Video-1 and Video-2

In this experiment, in order to train Classifier-1 and Classifier-2 same training sets are used as in the first experiment. Tests are done on the same test videos. Table 4.16 shows the classification results of the hierarchical classification by using only blob features in Classifier-1 and using only HOG features in Classifier-2.

Table 4.16. Confusion matrix of Video-1 and Video-2 by using only blob features in Classifier-1 and using only HOG features in Classifier-2

		Prediction		
Real		**car**	**van**	**truck**
	car	375	35	6
	van	37	13	7
	truck	0	27	25

When we look at the results of proposed hierarchical classification which is given in Table 4.3 and the above results, it is observed that the hierarchical classification by using only blob features in Classifier-1 and using only HOG features in Classifier-2 has worse performance than the proposed classification which uses HOG and blob features in Classifer-1 and Classifier-2. Using only blob features in Classifier-1 causes the vehicles which have same size, but are in different classes to be labeled incorrectly. In addition, using only HOG features in Classifier-2 causes the vehicles which have similar shape, but are in different classes to be labeled incorrectly. So, performance of this type of classification degrades.

4.2.3.2 Results of Video-3 and Video-4

In this experiment, in order to train Classifier-1 and Classifier-2 same training sets are used as in the first experiment. Tests are done on the same test videos. Table 4.17 shows the classification results of the hierarchical classification by using only blob features in Classifier-1 and using only HOG features in Classifier-2.

Table 4.17. Confusion matrix of Video-3 and Video-4 by using only blob features in Classifier-1 and using only HOG features in Classifier-2

	Prediction			
Real		car	van	truck
	car	51	11	3
	van	11	2	3
	truck	0	9	18

When we look at the results of proposed hierarchical classification which is given in Table 4.9 and the above results, as in the previous experiment it is observed that the hierarchical classification by using only blob features in Classifier-1 and using only HOG features in Classifier-2 has worse performance than the proposed classification which uses HOG and blob features in Classifer-1 and Classifier-2. Performance of this type of classification degrades due to the same reasons as in the previous experiment.

4.2.4 Comparison with the State of the Art Literature

Direct comparison of quantitative results with the literature is difficult due to the lack of a common data set for vehicle classification. In Section 1.1 detailed information about the state of the art in vehicle classification has been given. [21], [22] and [23] have high accuracy results about 90%. However, these algorithms rely on 3D modeling of the vehicles which is computationally costly and needs camera parameters like height of the camera and the angular direction of the camera. Our approach to the vehicle classification is in 2D approaches, that is, it works in the camera coordinate domain. 2D approach algorithms mostly want camera parameters (height, angle of vision, calibration, etc.) and parameters about the highways (direction of the vehicles, place of the lane division lines, viewing side of the highway etc.). From those algorithms ones which have classification performance nearly close to our approach are like that: [15] has a recall 65% and a precision 75% for classifying into 3 classes. However, in this algorithm there exists a normalization step of vehicle blobs where camera parameters are needed. [3] has a classification accuracy 70%. However, classification is done only as car and non-car and uses vehicle parameters such as length and height which are recovered from the 2D projections of the vehicles; so, camera parameters are needed.. [9],[10] and [7] have classification accuracies 82,9% classifying into 3 classes,74,4% classifying into 7 classes, and 91% classifying into 7 classes, respectively. However, these works use only blob features which are not reliable as seen in our experiments and their algorithms work only on the videos which are taken from side view of the highway. [11] gives a classification accuracy 88% for 4 classes, but it requires that the camera is in axis with the highway to see the lane division lines. [16] uses HOG features to classify vehicles in 3 classes. However, it is only proposed for in vehicle usage and also has a low classification accuracy as 64,3%.

The proposed algorithm in this thesis study is not computationally costly as in the case of 3D approaches. It does not need camera calibration and specifically any precise knowledge of camera parameters like height of the camera, angular direction of the camera. In addition, it does not depend on the parameters of the highways like the knowledge about the place of the lane division lines, viewing side of the highway. This approach only needs a detection region given by the user in which the camera will see a vehicle in full view and a training set created from the nearly same view of the highway on which classification will be performed, but it is not needed to be exactly in the same height and the same angular direction.

CHAPTER 5

CONCLUSIONS AND FUTURE WORKS

5.1 Summary and Conclusions

In this study, a new algorithm to classify moving vehicles in highways has been presented. The aim is to provide an automated system that classifies the vehicles and by this way provide more information to traffic system managers in order to improve the travel experience.

In the algorithm moving vehicle detection is performed by using an adaptive mixture of Gaussians that selects number of Gaussian components adaptively on run-time. Mixture of Gaussians is selected because it can cope with complexities in outdoor environments like lighting changes and repetitive clutters. During extraction of the foreground regions, some noises are also detected as foreground. In order to minimize the effects of these unwanted components, morphological operations are performed and by that way system performance is improved.

After detecting the vehicles, feature extraction step comes. In the algorithm two features are selected in order to make discrimination between 3 vehicle classes which are car, van and truck. The main difference between these vehicle types is their sizes, so 7 blob features of each vehicle blob is obtained. Another discriminative feature of these vehicle types is their shapes and appearances, so HOG features are extracted as a shape descriptor. By using these two features a global feature vector is created for each vehicle blob in order to use in classification step.

In the classification step, a robust classifier SVM is chosen in order to label the vehicles as car, van or truck. Each feature vector coming from each vehicle blob which includes HOG and blob features is classified as car, van or truck in a hierarchical manner. Firstly, the feature vector coming from a test data is classified as car or other (van + truck) with the Classifier-1 which is trained with a training set created before. Then, if the feature vector is labeled as car, classification step ends. Otherwise, if the feature vector is labeled as other, this feature vector is again classified with the Classifier-2 which is trained with a training set created before and discriminates between van and truck. At the end, the feature vector is labeled as van or truck.

Proposed algorithm is tested on 4 different videos. First two sets of videos are recorded from the same highway at different times of the day as their views of the vehicles are nearly same, but not exactly the same angular direction and their resolutions are same. With a training set created as the view of the vehicles nearly same as these two videos, the algorithm gives a satisfying result. Second two sets of videos are recorded from the different highways at different times of the day as their views of the vehicles are nearly same, but again not exactly the same angular direction and their resolutions are also different. With a training set created as the view of the vehicles nearly same as these two videos, the algorithm gives a satisfying result on these videos, too.

By looking at the results of experiments HOG and blob features together with the SVM classifier give a successful performance to classify vehicles into car, van or truck. It can be concluded that local object appearance and shape within an image can be described by the distributions of intensity gradients and edge directions, which refers to HOG features. HOG features are robust feature set which is invariant to local geometric and photometric transformations. Within cell rotations and translations do not affect the HOG values and illumination invariance achieved through normalization. Also, selection of SVM as a classifier provides robust classification. SVM is less overfitting and robust to noise.

As a conclusion, the algorithm classifies the vehicles into three main classes as car, van or truck by providing independence of knowing specifically the camera parameters like height of the camera, angular direction of the camera, camera calibration etc. and any information about the highway like the place of the lane division lines. Algorithm assumes a stationary camera and only needs a user defined classification region in order to see the vehicles as full view and trained classifiers with a training set created with the vehicle images nearly the same view of the highway on which the classification will be performed, but it is not necessary to be exactly the same view (same camera, same height of the camera, same angular direction of the camera).

5.2 Future Works

In this thesis, moving vehicle classification into three classes as car, van or truck is performed. One of the improvements which can be suggested is to combine tracking with the classification. Tracking can help to improve the results by eliminating the occlusion problem. Another improvement can be working on the shadow elimination. In addition, algorithm can be extended to the moving cameras by incorporating a different algorithm in the detection stage of the vehicles and vehicle types can be increased.

REFERENCES

[1] Lipton, Alan J., Fujiyoshi, H. and Patil, Raju S., *Moving Target Classification and Tracking From Real-time Video*, Proceedings of 4th IEEE Workshop on Applications of Computer Vision, pp. 8-14, 1998.

[2] Gupte, S., Masoud, O. and Papanikolopoulos, N., *Vision-based Vehicle Classification*, Intelligent Transportation Systems Conference IEEE, pp. 46-51, 2000.

[3] Gupte, S., Masoud, O., Martin, R. and Papanikolopoulos, N., *Detection and Classification of Vehicles*, Intelligent Transportation Systems Conference IEEE, vol. 3(1):37–47, 2002.

[4] Lai, Andrew H. S., Fung, George S. K., Yung, N.H.C., *Vehicle Type Classification From Visual-based Dimension Estimation*, Intelligent Transportation Systems Conference IEEE, pp. 201-206, 2001.

[5] Veeraraghavan, H., Masoud, O. and Papanikolopoulos, N., *Vision-based Monitoring of Intersections*, IEEE 5th International Conference on Intelligent Transportation Systems, pp. 7–12, 2002.

[6] Avery, R.P., Wang, Y., Rutherford, G. Scott, *Length-Based Vehicle Classification Using Images from Uncalibrated Video Cameras*, IEEE 7th International Conference on Intelligent Transportation Systems, pp. 737- 742, 2004.

[7] Huang, C. L. and Liao, W. C., *A Vision-based Vehicle Identification System*, Proceedings Of The 17th International Conference On Pattern Recognition, vol. 4, pp. 364–367, 2004.

[8] Rad, R. and Jamzad, M., *Real Time Classification and Tracking of Multiple Vehicles In Highways*, Pattern Recognition Letters, vol. 26(10), pp. 1597–1607, 2005.

[9] Morris, B. and Trivedi, M., *Robust Classification and Tracking of Vehicles In Traffic Video Streams*, Intelligent Transportation Systems Conference IEEE, pp. 1078–1083, 2006.

[10] Morris, B. and Trivedi, M., *Improved Vehicle Classification In Long Traffic Video by Cooperating Tracker and Classifier Modules*, Proceedings of the IEEE International Conference on Video and Signal Based Surveillance, pp. 9, 2006a.

[11] Hsieh, J. W., Yu, S. H., Chen, Y. S. and Hu, W. F., *Automatic Traffic Surveillance System For Vehicle Tracking and Classification*, IEEE Transactions On Intelligent Transportation Systems, vol. 7(2), pp.175–187, 2006.

[12] Zhang, C., Chen, X., Chen, W., *A PCA-based Vehicle Classification Framework*, Proceedings.of IEEE International Workshop on Multimedia Databases and Data Management, in conjunction with IEEE International Conference on Data Engineering, pp.17, 2006.

[13] Arrospide, J. and Salgado, L., *Region-dependent Vehicle Classification using PCA Features*, IEEE International Conference on Image Processing, pp. 453–456, 2012.

[14] Arrospide, J., Salgado, L. and Marinas, J., *HOG-like Gradient-based Descriptor for Visual Vehicle Detection*, IEEE Intelligent Vehicles Symposium, pp. 223-228, 2012.

[15] Chen, X. and Zhang, C. C., *Vehicle Classification From Traffic Surveillance Videos at a Finer Granularity*, Advances In Multimedia Modeling, Lecture Notes in Computer Science, vol. 4351, pp. 772–781, 2007.

[16] Gandhi, T. and Trivedi, M. M., *Video Based Surround Vehicle Detection, Classification and Logging From Moving Platforms: Issues and Approaches*, Intelligent Vehicles Symposium IEEE, pp. 1067–1071, 2007.

[17] Alonso, D., Salgado, L., and Nieto, M., *Robust Vehicle Detection Through Multidimensional Classification For on Board Video Based Systems*, IEEE International Conference on Image Processing, vol. 4, pp. 321–324, 2007.

[18] Zhang, G., Avery, R. P., and Wang, Y., *Video-Based Vehicle Detection and Classification System For Real-time Traffic Data Collection Using Uncalibrated Video Cameras*, Transportation Research Record, pp. 138–147, 2007a.

[19] Thi, T. H., Robert, K., Lu, S., and Zhang, J., *Vehicle Classification at Nighttime Using Eigenspaces and Support Vector Machine*, Image and Signal Processing Congress, vol. 2, pp. 422–426, 2008.

[20] Robert, K., *Night-time Traffic Surveillance: A Robust Framework for Multi-vehicle Detection, Classification and Tracking*, IEEE Conference on Advanced Video and Signal Based Surveillance, pp 1–6, 2009a.

[21] Sullivan, G. D., Baker, K. D., Worrall, A. D., Attwood, C. I. and Remagnino, P. R., *Model-based Vehicle Detection and Classification Using Orthographic Approximations*, Proceedings of 7th British Machine Vision Conference, vol. 2, pp. 695–704, 1996.

[22] Messelodi, S., Modena, C. M., and Zanin, M., *A Computer Vision System for The Detection and Classification of Vehicles at Urban Road Intersections*, Pattern Analysis & Applications, vol. 8(1-2), pp. 17–31, 2005b.

[23] Buch, N., Orwell, J., Velastin, S.A., *Detection and Classification of Vehicles for Urban Traffic Scenes*, 5th International Conference on Visual Information Engineering, pp.182-187, 2008.

[24] W.R. Christopher, A. Azarbayejani, T. Darrell and A. Pentland, *Pfinder: Real-Time Tracking of the Human Body*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, pp. 780-785, 1997.

[25] C. Stauffer and W. E. L. Grimson, *Learning Patterns of Activity Using Real-time Tracking*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pp. 747–757, 2000.

[26] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, New York, 1995.

[27] Zivkovic Z., *Improved Adaptive Gaussian Mixture Model for Background Subtraction*, ICPR 2004, pp.2: 28-31, 2004.

[28] Z.Zivkovic and F.van der Heijden, *Recursive Unsupervised Learning of Finite Mixture Models*, IEEE Trans. on PAMI, vol.26., no.5, 2004.

[29] Jain, R., Kasturi, R., Schunk, B.G., *Machine Vision*, McGraw-Hill Inc., pp. 63- 69, 1995.

[30] R.M. Haralick, and L.G. Shapiro, *Computer and Robot Vision*, Volume I, Addison-Wesley, pp. 28-48, 1992.

[31] N. Dalal , B. Triggs, *Histograms of Oriented Gradients for Human Detection*, in Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, pp.886-893, June 20-26, 2005

[32] MathWorks, Support Vector Machines, http://www.mathworks.com/help/stats/support-vector-machines.html

[33] Manjunath, B. S., Ohm, R., Vasudevan, V. V. and Yamada A., *MPEG-7 Color and Texture Descriptors*, IEEE Trans. Circuits Syst. Video Technol., vol. 11, pp. 703–715, 2001.

[34] Lowe, D. G., *Object Recognition From Local Scale-Invariant Features*, In International Conference on Computer Vision, vol. 2, pp. 1150–1157, 1999.

[35] Ma, X. and Grimson, W., *Edge-based Rich Representation for Vehicle Classification*, ICCV, Tenth IEEE International Conference, vol. 2, pp. 1185–1192, 2005.

[36] Bay, H., Tuytelaars, T., and Gool, L. V., *SURF: Speeded Up Robust Features*, In European Conference on Computer Vision (ECCV), vol. 3951 of *LNCS*, pp. 404–17, 2006

[37] Zhang, D., Qu, S., and Liu, Z., *Robust Classification of Vehicle Based On Fusion of Tsrp and Wavelet Fractal Signature*, In Networking, Sensing and Control (ICNSC) IEEE International Conference on, pp. 1788–1793, 2008a.

[38] Harris, C. and Stephens, M., *A Combined Corner and Edge Detector*, Proceedings of the 4th Alvey Vision Conference, pp.147—151, 1988.

[39] Beaudet, P.R., *Rotationally Invariant Image Operators*, In Proc. Of the 4th International Joint Conference on Pattern Recognition, pp. 579–583, 1978.

[40] Lindeberg, T., *Feature Detection With Automatic Scale Selection*, International Journal of Computer Vision, vol. 30 (2)., pp. 77–116, 1998.

[41] Lowe, D. G., *Distinctive Image Features from Scale-Invariant Keypoints*, International Journal of Computer Vision, vol. 60 (2), pp. 91, 2004.

[42] Canny, J., *A Computational Approach To Edge Detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, vol 8 (6), pp. 679–698, 1986.

[43] Gonzalez, R. and Woods, R., *Digital Image Processing*, Addison Wesley, pp 414 – 428, 1992.

[44] Prewitt, J.M.S., *Object Enhancement and Extraction*, In Picture processing and Psychopictorics, Academic Press, 1970.

[45] Shi, J. and Tomasi, C., *Good Features to Track*, 9th IEEE Conference on Computer Vision and Pattern Recognition, Springer, 1994.
[46] Rosten, E. and Drummond, T., *Machine Learning for High Speed Corner Detection*, In 9th European Conference on Computer Vision, vol. 1, pp. 430–443, 2006.

[47] Matas, J., Chum, O., Urban, M., and Pajdla, T., *Robust Wide Baseline Stereo from Maximally Stable Extremal Regions*, Proc. of British Machine Vision Conference, pp. 384-396, 2002.

[48] Haar, A., *On the Theory of Orthogonal Function Systems*, Mathematische Annalen, vol. 69 (3): 331–371, 1910.

[49] Duda, R. O. and P. E. Hart, *Use of the Hough Transformation to Detect Lines and Curves in Pictures*, Comm. ACM, Vol. 15, pp. 11–15, 1972.

[50] Rublee, E. at al., *ORB: An Efficient Alternative to SIFT or SURF*, In Computer Vision IEEE International Conference on, pp. 2564-2571, 2011.

[51] Mikolajczyk, K. and Schmid, C., *A Performance Evaluation of Local Descriptor*, IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1615--1630, 2005.

[52] Zheng, D., Zhao, Y. and Wang, J., Features Extraction Using a Gabor Filter Family, Proc. of the 6th IASTED International Conference, pp. 139-144, 2004.

[53] Kastrinaki, V., Zervakis, M., and Kalaitzakis, K., *A Survey of Video Processing Techniques for Traffic Applications*, Image and Vision Computing, vol. 21(4), pp. 359 – 381, 2003.

[54] Kanhere, N. and Birchfield, S., Real-time Incremental Segmentation and Tracking of Vehicles at Low Camera Angles Using Stable Features, Intelligent Transportation Systems, IEEE Transactions on, vol. 9(1):148–160, 2008.

[55]Liblinear SVM Library, http://www.csie.ntu.edu.tw/~cjlin/liblinear/