İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

AN AUTOMATED ROBUST VEHICLE DETECTION AND TRACKING SYSTEM FOR LOW RESOLUTION TRAFFIC VIDEO SEQUENCES

M.Sc. Thesis by
Mehmet KAPLAN

Department : Computer Engineering

Programme : Computer Engineering

JANUARY 2010

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

AN AUTOMATED ROBUST VEHICLE DETECTION AND TRACKING SYSTEM FOR LOW RESOLUTION TRAFFIC VIDEO SEQUENCES

M.Sc. Thesis by
Mehmet KAPLAN
(504071523)

Date of submission : 24 December 2009
Date of defence examination: 29 January 2010

Supervisor (Chairman) : Prof. Dr. Muhittin GÖKMEN (ITU)
Members of the Examining Committee : Assoc. Prof. Dr. Zehra ÇATALTEPE (ITU)
Prof. Dr. Coşkun SÖNMEZ (YTU)

JANUARY 2010

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

DÜŞÜK ÇÖZÜNÜRLÜKLÜ TRAFİK GÖRÜNTÜ DİZİLERİ İÇİN OTOMATİK GÜRBÜZ ARAÇ TANIMA VE İZLEME SİSTEMİ

YÜKSEK LİSANS TEZİ
Mehmet KAPLAN
(504071523)

Tezin Enstitüye Verildiği Tarih : 24 Aralık 2009
Tezin Savunulduğu Tarih : 29 Ocak 2010

Tez Danışmanı : Prof. Dr. Muhittin GÖKMEN (İTÜ)
Diğer Jüri Üyeleri : Doç. Dr. Zehra ÇATALTEPE (İTÜ)
Prof. Dr. Coşkun SÖNMEZ (YTÜ)

OCAK 2010

FOREWORD

I would like to express my deep appreciation to my family for their continuous support during my educational life and social life.

I also want to thank my advisor Prof. Dr. Muhittin GÖKMEN for his patient and endless guidance during my education and this research.

I would also like to thank to TÜBİTAK (The Scientific and Technological Research Council of Turkey) for supporting me during my Master study under the grant "National Scholarship Programme for Master Science Students".

December 2009

Mehmet Kaplan

Computer Engineer

TABLE OF CONTENTS

Page

ABBREVIATIONS

ROI **:** Region of Interest
HSV **:** Hue-Saturation-Value color space
RGB **:** Red Green Blue color space
YCbCr **:** Luma, blue difference and red difference chromaticity color space
VSAM **:** Visual Surveillance and Monitoring System
EU **:** European Union
SIFT **:** Scale-invariant Feature Transform
SVM **:** Support Vector Machine
DCT **:** Discrete Cosine Transform
MPEG **:** Moving Picture Experts Group
WSMM **:** Windowed Second Moment Matrix

LIST OF TABLES

Page

LIST OF FIGURES

Page

AN AUTOMATED ROBUST VEHICLE DETECTION AND TRACKING SYSTEM FOR LOW RESOLUTION TRAFFIC VIDEO SEQUENCES

SUMMARY

Traffic surveillance systems are widely used in numerous municipalities for controlling urban and highway traffic. While some of them are used for only monitoring traffic conditions, cameras are generally installed for extracting traffic parameters such as number of vehicles, traffic flow density, mean vehicle speed and individual vehicles speeds. For instance, in Istanbul there are nearly 175 traffic cameras for monitoring urban traffic. Although these cameras are utilized for traffic analysis, traffic sensors perform most of the analysis work. However, traffic cameras can obtain all parameters with the help of video processing algorithms, without requiring another hardware equipment such as traffic sensors.

In order to obtain corresponding parameters, a traffic surveillance system is developed in this thesis. System is designed with following steps: background subtraction and moving object detection, occlusion handling and tracking. Firstly, moving object detection is realized with an efficient and simple background subtraction algorithm. Moreover, utilized main algorithm is improved with some contributions such as proposed edge adaptive thresholding approach and 3-d connected component analysis. Edge adaptive thresholding provides an improvement in detecting all vehicles (especially small ones) and obtaining correct shapes for vehicles. Additionally, 3-d connected component analysis is used to eliminate irregularities in foreground regions. Together with these contributions, foreground masks and moving objects are determined accurately. Accuracy of the approach is also proved by numerical results while comparing the system with another succeeding algorithm.

After moving object detection, an existing occlusion handling system is implemented to obtain single vehicles from occluded blobs, which are determined by an efficient occlusion detection algorithm. This approach is a classification-based algorithm and is fully automated by obtaining training examples from the video sequence automatically. In order to train a model, these examples are used; subsequently, vehicles in occluded blobs are located by binary classification. The advantage of the system is that system adapts to different video sequences by acquiring train examples from the video sequence itself. Instead of a general vehicle model, video specific model is more sufficient in this manner. Furthermore, an automatic ROI detection algorithm is proposed in addition to corresponding approach to make the system fully automated, while reducing the possible errors from the user selections. Moreover, feature extraction method in existing occlusion handling system is improved with adding new features to the algorithm. The improvement in accuracy is also indicated with visual and numerical test results.

Finally, a simple and efficient tracking method is presented to obtain mean and individual vehicle speeds.

DÜŞÜK ÇÖZÜNÜRLÜKLÜ TRAFİK GÖRÜNTÜ DİZİLERİ İÇİN OTOMATİK GÜRBÜZ ARAÇ TANIMA VE İZLEME SİSTEMİ

ÖZET

Trafik denetim sistemleri şehir içi ve şehirlerarası trafiği kontrol etmek için belediyeler tarafından sıklıkla kullanılmaktadır. Bazıları sadece trafik koşullarını gözetlemek için kullanılırken; kameralar genellikle araç sayısı, trafik akış yoğunluğu, ortalama araç hızı ve tek tek araç hızları gibi trafik parametrelerin elde edilmesi için kurulmaktadır. Örneğin, İstanbul'da şehir içi trafiği gözetlemek amacıyla yaklaşık 175 adet kamera yer almaktadır. Bu kameralardan trafik analizi için yararlanılmasına rağmen, trafik duyargaları analiz görevinin büyük bir kısmını yerine getirmektedir. Halbuki, trafik kameraları görüntü işleme algoritmaları yardımıyla, trafik duyargaları gibi başka bir donanım malzemesine gerek duymadan tüm parametreleri elde edebilir.

İlgili parametreleri elde etmek amacıyla, bu tezde bir trafik denetleme sistemi geliştirilmiştir. Sistem şu adımlarla tasarlanmıştır: arka plan ayrıştırma ve hareketli nesne tespiti, örtüşme giderilmesi ve izleme. Öncelikle, hareketli nesnelerin belirlenmesi verimli ve basit bir arka plan ayrıştırma algoritması ile gerçeklenmiştir. Ayrıca, yararlanılan ana algoritma önerilen ayrıt uyarlanabilir eşikleme yaklaşımı ve 3 boyutlu bağlı bileşen analizi gibi bazı katkılarla iyileştirilmiştir. Ayrıt uyarlanabilir ayrıştırma tüm araçların belirlenmesinde (özellikle küçük olanların) ve araçlar için düzgün şekiller elde edilmesinde iyileştirme sağlamaktadır. Ek olarak, ön plan alanlarındaki tutarsızlıkları ortadan kaldırmak için 3 boyutlu bağlı bileşen analizi kullanılmaktadır. Bu katkılarla birlikte, ön plan maskeleri ve hareketli nesneler doğru olarak tespit edilmektedir. Yaklaşımın doğruluğu sistemi bir başka başarılı algoritma ile karşılaştırarak, sayısal sonuçlar ile kanıtlanmıştır.

Hareketli nesnelerin tespitinden sonra, başarılı bir örtüşme tespiti algoritması tarafından elde edilen örtüşme olan bölgelerden tekil araçların elde edilmesi için var olan bir örtüşme giderilme sistemi gerçeklenmiştir. Bu yaklaşım sınıflandırma tabanlı bir algoritmadır ve öğrenme amaçlı örnekleri görüntü dizisinden otomatik olarak elde ederek tamamıyla otomatikleştirilmiştir. Bir model öğrenilmesi amacıyla bu örnekler kullanılır; akabinde, ikili sınıflandırma ile örtüşme olan alanlardaki araçların yeri belirlenir. Sistemin avantajı sistemin öğrenme örneklerini görüntü dizisinin kendisinden elde ederek farklı görüntü dizilerine uyum sağlamasıdır. Bu anlamda, genel bir araç modeli yerine görüntüye özgü bir model daha uygun olmaktadır. Ayrıca, kullanıcı seçimlerinden kaynaklanan olası hataları azaltırken sistemi tamamıyla otomatik yapmak amacıyla, ilgili yaklaşıma ek olarak bir otomatik ilgi alanı tespiti algoritması da tasarlanmıştır. Diğer taraftan, sisteme yeni öznitelikler eklenerek var olan sistemdeki öznitelik çıkarılması aşaması geliştirilmiştir. Başarımdaki ilerleme de ayrıca sayısal ve görsel test sonuçları ile kanıtlanmıştır.

Son olarak, ortalama hızı ve tek tek araç hızlarını tespit etmek için basit ve verimli bir izleme algoritması sunulmuştur.

1. INTRODUCTION

Visual Surveillance Systems play an important role in daily life. Nearly every place used in social and business life is controlled by visual surveillance systems. Also for security and military purposes, these systems are very important. General aims in these systems are entrance surveillance in important points, human detection and recognition, density estimation of people and vehicles for congestion analysis, behavior analysis and detecting abnormal behaviors [1]. For these purposes in a wide variety of applications, such as controlling public areas like airports, maritime, railway stations, metro stations, banks, shopping malls, parking areas; detecting human behaviors in entrance in sport activities etc.; surveillance in military and forensic applications; surveillance in highway and urban traffic surveillance, cameras gives an important information [2].

For the huge demand in security, visual surveillance systems became indispensable. Governments make big investments in visual surveillances systems. For instance, VSAM (Visual Surveillance and Monitoring System) [3] of DARPA's Image Understanding for Battlefield Awareness (IUBA) program, Cooperative Distribution Vision (CDV) Program of Japan, EU Chromatica and Prismatica program are these kind of applications [4]. In addition, ObjectVideo's Video Early Warning (VEW) product is a very competent application in this area. This product has very evolutionary abilities, which can handle various necessities of visual surveillance systems. Some of these abilities are given in Figure 1.1 [4]. In Figure 1.1, indicated abilities are imaginary control line determination, ROI control, left object detection, congestion detection, detecting movement in restricted way and object count, respectively.

As already stated, visual surveillance systems have lots of varieties. In this research, the main topic is traffic surveillance systems. In recent years, traffic surveillance systems are widely used for analyzing and learning the structure of urban and highway traffic. After analyzing the density of traffic flow in critical directions, traffic control centers are informed about the situation of the traffic flow; hence,

drivers can be canalized into available, low crowded roads. Moreover, vehicle velocity (individual or mean velocity) and number of vehicles that pass from imaginary lines are other significant features for traffic surveillance. Traffic cameras are also used in detection of accidents, detection of stopping vehicles and simulation of traffic flow in road junctions in recent projects. As Machy et al. mentioned in their study [5], traffic surveillance systems are also used in traffic sign detection for driver guidance and driver fatigue detection.

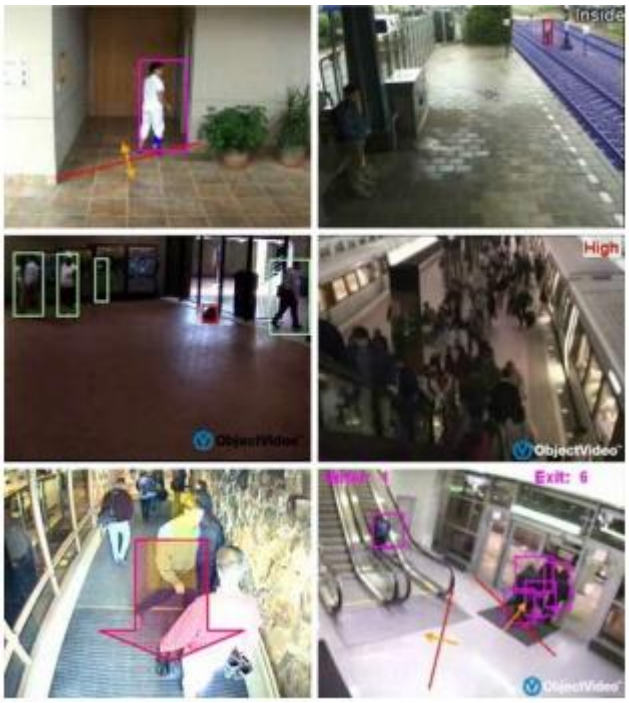

Figure 1.1 : Some abilities of Object Video VEW product

In the traffic surveillance area, lots of researches were done in past years. Some of these works were canalized into develop complete traffic surveillance systems, however, some of the works were focused on providing improvement on a step of traffic surveillance systems. Although some researches have specific approaches; as illustrated in Figure 1.2 [1], video surveillance systems (also traffic surveillance systems) have general steps like environment modeling, object detection, object tracking and getting further information such as mean vehicle speed and behavior analysis. Now, some of these steps and previous work in these steps will be explained briefly. Further information in these steps will be given in other chapters of the thesis when necessary.

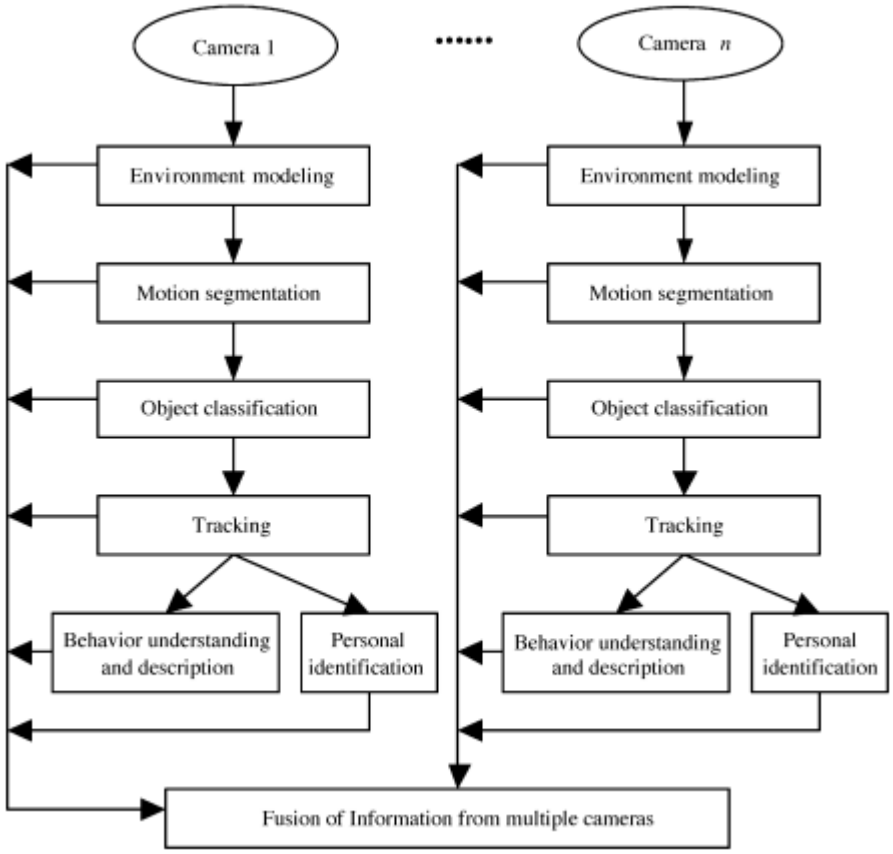

Figure 1.2 : General steps of visual surveillance systems

1.1 Environment Modeling and Motion Segmentation

The main purpose of environment modeling and motion segmentation is finding moving or foreground objects in a video sequence. Environment modeling is finding stationary scene in a sequence. For example, in traffic surveillance systems, modeling background image is necessary to find moving vehicles. Motion segmentation and object detection are based on environment modeling.

1.1.1 Temporal Differencing

Temporal differencing is a simple and direct way of motion segmentation. In this approach, two or more consecutive frames are examined for detecting remarkable change in intensity values. If an intensity change is more than a threshold, this pixel is assigned as a foreground pixel. Lipton et al. proposed using two frame temporal differencing and clustering of foreground pixels (connected component analysis) in order to detect moving regions [6].

1.1.2 Background Subtraction

Background subtraction is the widely used solution for moving object detection. General steps of background subtraction are summarized in Figure 1.3 [7].

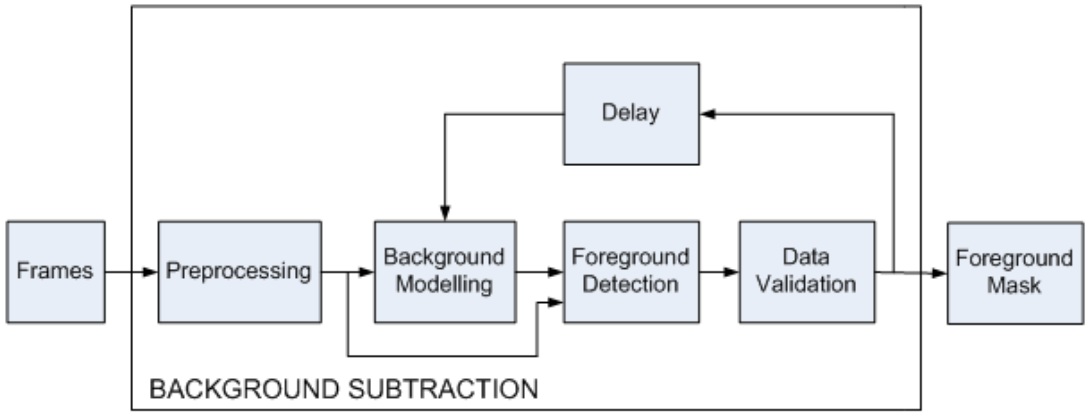

Figure 1.3 : General steps of background subtraction

Primary step in background subtraction is modeling background image. With the aim of finding foreground objects, background image is differentiated from the original scene (foreground detection). After differentiation, differences are compared with a threshold to find foreground pixels. If a pixel in current frame is denoted by $I(x, y)$ and corresponding intensity value in background image is denoted by $B(x, y)$, a point is a foreground pixel if

$$\left|I(x, y) - B(x, y)\right| > Threshold. \qquad \textbf{(1.1)}$$

Another approach is using normalized statistics. In this approach, a pixel is defined as foreground if

$$\frac{\left|I(x, y) - B(x, y) - mean\right|}{std} > Threshold. \qquad \textbf{(1.2)}$$

Mean and std terms in equation 1.2 are mean value and standard deviation of the difference $I(x, y) - B(x, y)$.

In addition, Fuentes and Velastin [8] determined foreground pixels by relative difference as:

$$\frac{|I(x,y) - B(x,y)|}{B(x,y)} > Threshold. \quad \textbf{(1.3)}$$

Threshold values can be found after experiments or adaptively in real time.

Background modeling is a challenging problem because of the dynamic nature of environment. There are lots of possible problems in video sequences, hence it is very difficult to develop a robust background model that can cover all of the situations. Toyama et al. [9] figured out some examples of these problems as follows:

- Movement of stationary background objects
- Variation of illumination during the day and change according to weather etc.
- Rapid variation in illumination
- Fluctuation of some objects like trees
- Occlusion of a foreground object by background objects
- Uniform structure of an object, which can cause misdetection of interior pixels belonging to that object
- Stopping situation of a moving object: If a foreground object stops, it becomes a background object.
- Movement of background object: If a background object starts to move, in that position of the background model, anomalies can appear.
- Shadows

A robust algorithm should work efficiently in these situations.

Many background subtraction algorithms were developed so far. Some algorithms aimed to obtain a background model; however, some algorithms intended to find foreground pixels directly. The basic approach can be finding mean or median of the frames in a time period. This mean or median gives background model in the sequence. Matsuyama et al. [10] described a normalized block correlation algorithm, which compares images with median images to detect foreground pixels. Comparison is done in block level. Oliver et al. [11] suggested using Principal Component Analysis (Eigenbackground algorithm). According to this suggestion, some stationary scenes are collected. After that, all new frames are transformed into

PCA space. If the difference between the original and transformed image is larger than a threshold at a pixel, this pixel is defined as foreground. Another approach is using Kalman filter approach. Karmann and Brandt [12] modeled a system using background model B_t and its temporal derivative B_t'

$$\begin{bmatrix} B_t \\ B_t' \end{bmatrix} = \mathbf{A}.\begin{bmatrix} B_{t-1} \\ B_{t-1}' \end{bmatrix} + \mathbf{K_t}\left(I_t - \mathbf{H}.\mathbf{A}.\begin{bmatrix} B_{t-1} \\ B_{t-1}' \end{bmatrix}\right), \quad \mathbf{A} = \begin{bmatrix} 1 & 0.7 \\ 0 & 0.7 \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} 1 & 0 \end{bmatrix} \qquad \textbf{(1.4)}$$

where**,** $\mathbf{K_t}$ is the adaptation rate that changes according to pixel in the previous frame. If it was a foreground pixel $\mathbf{K_t}$ is $[\alpha 1 \quad \alpha 1]^T$, otherwise $\mathbf{K_t}$ is $[\alpha 2 \quad \alpha 2]^T$ $(\alpha 2 > \alpha 1)$.

Toyama et al. [9] developed a three level algorithm. First level (pixel level) uses Wiener filtering to estimate background statistics. The purpose of the second level (region level) is filling foreground blobs. Last level (frame level) deals with sudden and global changes.

Stauffer and Grimson [13] modeled pixel values with a mixture of Gaussian random variables. Gaussian random distribution is a good way to define behavior of a variable. Pixel values in an image can also be defined by a Gaussian random variable. Every pixel in an image is a combination value of some effects (lighting change, changing objects etc.). As a result, a mixture of Gaussian random distributions rather than only one Gaussian distribution can be a better way to characterize a pixel. These distributions are updated according to the illumination change in pixel values with an expectation maximization approach.

The main problem of the mixture of Gaussian algorithm of Stauffer and Grimson [13] is the learning rate. KaewTraKulPong and Bowden [14] stated the following situation. Think that %60 of the time, background is present and α is 0.002 (500 recent frames), it will take 255 frames to include the new pixel value to the model and 346 frames to become dominant of new pixel in the model. As a solution to this problem, KaewTraKulPong and Bowden [14] used update equations of the algorithm of Stauffer and Grimson [13] only in first L frames. After L frames, update operation was done by exploring L recent frames.

In an advanced version of algorithm of Stauffer and Grimson [13], Jun et al. [15] suggested using 3-d connected component analysis in order to provide enhanced foreground blobs. After making the foreground and background classification in the

current frame with Mixture of Gaussian approach [13], the holes belonging to foreground blobs in the current frame are filled through foreground masks of previous and next K frames. As a conclusion, more accurate foreground blobs are obtained for future work like occlusion detection etc.

Collins et al. [3] presented a simple and efficient moving object detection and background-modeling algorithm in the VSAM (Visual Surveillance and Monitoring System) project. This algorithm is a mixture of frame differencing and background subtraction. Moving objects are detected by three frame temporal differencing approach. Afterwards, background model is updated according to this information. Pixels not containing moving object is considered as a background pixel and the intensity value in the background model is updated with the current pixel value by a learning rate.

Horprasert et al. [16] used a color model that separates brightness from chromaticity component. A pixel was modeled by four components, which are expected color values, standard deviation of color values, brightness distortion and chromaticity distortion. In consequence of comparison operations in these four components, a pixel was classified into background, foreground, shadow or highlighted background.

Kim et al. [17] modeled pixel values with a 6-tuple codeword. Codeword contains minimum and maximum values of pixel values, frequency of the codeword, maximum negative run length of the codeword, first and last access times that the codeword has occurred. This approach represents a compressed model of long video sequences and can handle moving backgrounds and illumination changes.

Javed et al. [18] presented a background subtraction algorithm utilizing color and edge information. In the pixel level, color information was used for background subtraction. Clustered foreground blobs were detected in the region level and foreground regions were validated according to gradient information. Yao and Odobez [19] benefited from color and texture information for robust background subtraction. This approach takes advantage of the texture information (represented by local binary patterns) in rich texture regions, contributing the stable results of color information in uniform regions.

Vargas et al. [20] introduced improved sigma delta background estimation method for a robust estimation in urban traffic video. Slow vehicles or stopped vehicles can

corrupt the background model in corresponding areas. A confidence measurement was provided to make decision for updating the background model. This validation uses not only the intensity change in a pixel, but also the estimated motion flow in the corresponding pixel.

1.1.3 Optical Flow

In addition to temporal frame differencing and background subtraction, optical flow information is another alternative for the purpose of moving object detection. Optical flow estimates pixel based motion information. Consequently, this information is very significant to detect moving objects. Meyer et al. [21] took advantage of optical flow information for initializing object segmentation. Subsequently, segmented body parts were tracked with a contour based tracking algorithm.

1.2 Object Tracking

Object tracking is the final step for visual surveillance systems. After tracking step, object behaviors, object trajectories are obtained. There are several tracking approaches such as region based tracking, active contour based tracking, feature based tracking and model based tracking. Some advantages and disadvantages of these approaches can be summarized as follows [1]:

- Region based algorithms can handle scenes where there are few objects; however, occlusion handling is very poor. Obtaining 3-d pose in region-based approach is very difficult and necessity of tracking multiple objects with occlusion cannot be handled.
- On the contrary, active contour based tracking algorithms simply track objects while consuming little resource, because only contour information is used. In addition to that, occlusion handling can be done partially. The main problem is the initialization; since, it is very difficult to start tracking automatically with active contour based algorithms.
- Although there are complicated algorithms like dependency graph based algorithms, feature based tracking is generally adaptable to real-time tracking. Moreover, feature based algorithms are suitable for congested and partially occluded scenes. Despite all these advantages, acquiring 3-d pose and object recognition are very difficult in feature based tracking algorithms.

- Model based algorithms provide robust object tracking, even under occlusion. By using projection between 2-d image plane and 3-d world plane, 3-d pose of objects can be acquired efficiently. Furthermore, model based approach gives more robust result despite orientation variation according to motion. The main challenges of model based algorithms are their complex structure and computational cost.

As stated above, all algorithms have distinctive advantages and disadvantages. Consequently, a suitable algorithm can be chosen according to the demand of the application. Main trend in vehicle tracking is using feature based tracking approaches and Kalman based models. However, some other approaches are also used in traffic surveillance systems. Koller et al. [22] presented a contour based motion tracker with an affine model based Kalman filter model to track vehicles in traffic scenes. In model based tracking algorithms, the main approach is using 3-d wire-frame vehicle model. Karlsruhe group (Koller et al.) [23] applied this model by using edge features. A vehicle was modeled according to vehicle pose parameters. The algorithm provided robust results with the aim of modeling smooth trajectories of vehicles in complicated illumination environments and cluttered traffic scenes. Haag and Nagel [24] combined image gradients and optical flow in parameter extraction. In this approach, image gradients give orientation and position parameters accurately; additionally, optical flow provides orientation, speed and angular speed parameters. Gradient information is local information and affected by model parameters. From the other point of view, optical flow information can be inaccurate when vehicle moves slowly or stops. In conclusion, global approach of optical flow and local approach of image gradients are combined in this algorithm to obtain more robust tracking results.

On the other hand, feature based algorithms are widely used in vehicle tracking purposes. Tomasi and Kanade [25] stated a way to choose best features for tracking and described how to track those features. A 2x2 matrix was created from the weighted averages of vertical and horizontal derivates in a window around a pixel. If the eigenvalues of matrix A are large enough, this point was classified as a good feature that will be tracked. Afterwards, point was tracked according to the mean squared error between two windows in different frames.

Beymer et al. [26] suggested to track sub-features of a vehicle instead of the vehicle itself, in order to provide occlusion handling. In their traffic surveillance system, they used corner features for tracking. After tracking corner features, features were clustered according to their motion information to compose vehicle blobs. She et al. [27] utilized color and shape features together for vehicle tracking. They used HSV color space, and vertical, horizontal, diagonal edge spaces as an input for a mean shift estimator. Target positions provided from mean shift estimator for different feature spaces were combined to track a vehicle.

Most general approach for vehicle tracking is Kalman based filters. Kalman filters are state based filters that estimate the position of a vehicle from noisy measurements. Before processing Kalman filter, the position of a vehicle must be obtained in different frames. In a vehicle tracking problem, a Kalman filter model can be determined according to Newton's law of motion with a static acceleration constraint [28]. As a result of measuring the positions of the vehicle in consecutive frames, more accurate estimated positions can be found according to Kalman filter approach.

Additionally, particle filter is also another approach for getting results that are more accurate than measured data. In particle filtering, firstly, particles are generated from the measured data. Then, probabilities of the particles are calculated according to the confidence of those particles. Finally, a weighted sum of generated particles is assigned as the estimated result. In this summation, a weight is the probability of corresponding particle. Yang et al. [29] developed an object-tracking algorithm based on the particle filter approach. They calculated color and edge histogram features to define objects that will be tracked. Moreover, they suggested sampling particles with Quasi-random distribution in order to improve the probability of convergence.

Research of Grammatikopoulos et al. [30] can be presented as a simple and efficient approach in vehicle tracking. In this work, all frames are rectified with affine transformation. Corresponding affine transformation parameters are obtained from the vanishing point calculated automatically through road borders. After rectification, vehicles remain in the same size throughout the consecutive frames. The algorithm obtains a window from bottom part of a vehicle and calculates cross correlation of this window and windows of all vehicles in the following frame. Matched vehicles

are determined according to the cross correlation information obtained from pixel values in corresponding windows.

1.3 Other Required Steps

Several steps in visual surveillance systems are stated above. Additionally, some other improvements can be done for more robust and accurate results. Two examples of additional steps, shadow removal and occlusion handling are discussed in following sections.

1.3.1 Shadow Removal

Shadow affects accuracy of visual surveillance systems in daylight. According to shadows, shape and orientation of vehicles can be irregular. Additionally, occlusion rate increases, because shadows connect individual vehicles as a single object blob. As a solution to this problem, recent visual surveillance systems involves shadow removal step.

In their work Cucchiara et al. [31] classified image pixels as background, foreground and shadow. They used Hue-Saturation-Value (HSV) color space for detecting shadow points. Cucchiara et al. stated that if a shadow is apparent on the background, hue and saturation values change in a certain limit. As a result, they controlled hue and saturation changes with upper and lower bound thresholds to determine shadow regions on the scene. Bo and Qi-mei [32] proved that the constancy of hue value in shadow detection can fail in traffic surveillance systems. Instead of the algorithm stated by Cucchiara et al. [31], they proposed using ratio of the reflection rate for shadow detection in their framework. They calculated ratio of reflection rates between red, green components and green, blue components; afterwards, they limited these rates with upper and lower bounds for shadow detection and removal.

In addition to HSV and RGB color spaces Kristensen et al. [33] showed that YCbCr color space can be another alternative for shadow detection. They observed that shadow points have similar chrominance and lower luminance as compared with background points. According to their approach, a shadow point is determined by a limiting ratio between luminance components and difference between chrominance components of background and shadow points.

Mikic et al. [34] described a statistical approach for shadow removal. In this research, they calculated posterior probability of being a background, foreground and shadow pixel for every point. They projected the value of the pixel (assuming that the pixel is not a shadow) by a diagonal matrix for estimating the shadowed value of that pixel and used this value for calculating the priori probability of being a shadow pixel.

Horprasert et al. [16] suggested a color model that separates brightness from chromaticity component. They calculated brightness distortion and chromaticity distortion with the difference between the value of the pixel and its estimated value. By using these distortions, pixels were classified into foreground, background, shadow and highlighted background.

Liu et al. benefited from gradient features for moving shadow elimination in their work [35]. Their assumption is gradient information of shadows being similar with the gradient information of the background model. On the contrary, gradient features of moving objects are different from the gradient features of background. This approach is very simple and robust to illumination changes.

1.3.2 Occlusion Handling

Occlusion handling is the most challenging problem in visual surveillance systems. Most of the approaches develop tracking algorithms that can work robustly in occlusion situations. For example, Beymer et al. [26] used sub-features (corner features) for vehicle tracking. Because sub-features are independent from occluded blobs, tracking algorithm can work robustly despite the occlusion problem. Jung and Ho [36] suggested a tracking based occlusion handling method for providing continuous trajectories of vehicles even though vehicles occlude in some frames. They presented two types of occlusions: implicit and explicit occlusion. In implicit occlusion, initially, two vehicles occlude each other. After merged object A separates into two vehicles B and C, trajectory of A is continued by trajectories of B and C. In explicit occlusion, initially, A and B objects are tracked individually. After A and B merges, trajectory of merged object C is connected with trajectory of A and B, until the end of occlusion. As soon as the occlusion ended, trajectories of A and B before occlusion is merged with the trajectories after the end of occlusion. This merging operation is done according to the feature matching.

Senior et al. [37] defined appearance models for occlusion handling. These models were presented for solving partial and complete occlusion and acquiring depth information of occluded objects. For each vehicle, an appearance model is generated showing the appearance of the object throughout the video frames. Once the object is segmented in following frames, appearance model is updated with new information by a small learning rate. As a result, model changes slowly for remembering the old information about the object. Although new information is added slowly, scale and orientation changes can be handled. When objects form an occluded single object region, appearance models produce a solution for solving occlusion and gathering depth information.

Pang et al. [38] described individual vehicles by 3-d cubic models. Occluded blobs were also detected by object dimensions; additionally, type of the occlusion (side by side or front and back) was determined from blob dimension. Afterwards, occlusion was solved by fitting curvature based 3-d models to the vehicles causing occlusion.

Jun et al. [15] introduced feature based occlusion handling algorithm. First of all, they detected occluded blobs by controlling solidity and orientation of all blobs. They stated that vehicles are almost a convex object. When objects occlude each other, connected region has a small solidity and orientation of the blob is relatively different from the orientation of the road. Subsequently, they calculated SIFT features in this irregular blob and found two cluster of motion vectors (two clusters for two vehicles) from SIFT features. The next step in their algorithm was forming over-segmented patches from the irregular blob and assigning the corresponding motion vector to each patch. For this purpose in each patch, average intensity error between the current patch and patch in the motion compensated frame was calculated for two clustered motion vectors. Consequently, each patch was assigned to a vehicle by finding the suitable motion vector for that patch.

Tamersoy and Aggarwal [39] recommended using unsupervised learning for occlusion handling. They benefited from the occlusion detection algorithm of Jun et al. [15] for finding irregular blobs. Initially, among particular number of frames, they found positive and negative examples of vehicles. Positive examples were vehicles which are individually found by occlusion detection algorithm. Additionally, negative examples were created from these positive examples. Afterwards, median width and height of vehicles was automatically determined from these examples.

Later, they trained a SVM classifier with histogram of gradient features calculated from positive and negative examples. After the training period, occluded blobs in following frames were segmented according to SVM classifier. For every suspicious blob in the corresponding frame, a sliding window, which has a size of the median width and height of positive examples, was used to detect whether there is a vehicle in the center of the window or not. Consequently, a binary image, which determines the points that can be a center of a vehicle, was calculated for every blob. As a result, vehicle centers and individual vehicles were obtained from this binary image.

1.4 Examples of Some Complete Systems for Traffic Surveillance

General steps of visual surveillance systems were explained in details. Most of the general surveillance systems consist of these processes. For example in their work, Beymer et al. [26] implemented vehicle tracking according to tracked corner features to prevent vehicle occlusion. Afterwards, they grouped features of same vehicle with the assistance of motion information. Using the general steps of visual surveillance systems, Gupte et al. [40] developed a system with six stages:

- Segmentation provided by background subtraction
- Region tracking according to spatial correlation
- Gathering vehicle parameters such as width, height, length from camera calibration parameters
- Forming validated vehicles from tracked regions
- Vehicle tracking at two stages: region and vehicle level
- Vehicle classification

Ozkurt and Camci [41] presented another complete system for video sequences from Istanbul. They detected moving objects with a background subtraction approach. After that, they used neural networks for classifying detected vehicles. Their neural network model consists of 14 input parameters and 4 output classes. Input parameters are vehicle parameters like orientation, bounding box coordinates, centroid coordinates and diameter. Output classes are small (cars), medium (van), big (bus) vehicles and erroneously detected vehicles. They obtained accurate results for vehicle detection and classification. The main problem in this system is that vehicles

were not separated under occlusion situation; as a result, number of vehicles cannot be acquired efficiently.

In addition to general traffic surveillance systems, some goal-oriented approaches also exist. In their work, Porikli and Li [42] developed a congestion analysis algorithm using Gaussian Mixture Hidden Markov Models (GM-HMM) that performs on MPEG video data. They trained HMM chains according to DCT coefficients and motion vectors. As a result, they classified traffic density into five congestion levels from empty traffic to stopped traffic. However, in this approach, finding number of vehicles and speeds of each vehicle is impossible. The other interesting approach is the method developed by Balcilar and Sonmez [43], which aims calculating mean vehicle speed accurately and efficiently. For this purpose, they benefited from the specific characteristic of MPEG video format. They filtered the motion vectors obtained from MPEG, to reduce noise in vectors. Afterwards, they projected motion vectors into world plane in order to gather speed parameters. They obtained robust and accurate results for mean vehicle speeds in different video sequences. Nevertheless, number of the vehicles and individual vehicle speeds cannot be acquired in this approach.

1.5 Organization of the Thesis

In this research, the main aim is developing a visual surveillance system that can robustly extract traffic parameters like number of vehicles, traffic density, individual speed of vehicles and mean vehicle speed. Although other systems focus on specific parameters such as mean vehicle speed or congestion detection, in this system all parameters are extracted. In addition, these parameters are acquired accurately from different video sequences which have different characteristics (low crowded, reasonably crowded and high crowded sequences). For these purposes, steps in Figure 1.4 were implemented; moreover, some improvements were done in these steps for obtaining a robust and more accurate traffic surveillance system.

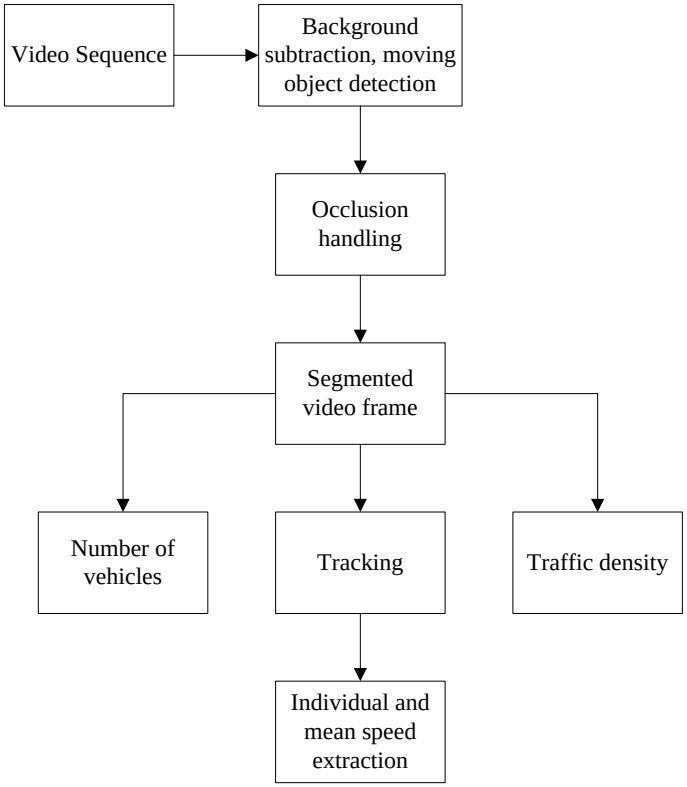

Figure 1.4 : Fundamental steps of the developed system

In the following chapters, these steps are discussed. In chapter 2, background subtraction and moving object detection step is covered. Furthermore, improvements in this step and accurate results are presented in chapter 2. After second chapter, occlusion-handling approach used in this work and accuracy of the method are explained in chapter 3. Subsequently, tracking step and gathering speed information are proposed in chapter 4. Finally, conclusions and future work are given in chapter 5.

2. BACKGROUND SUBTRACTION AND MOVING OBJECT DETECTION

As stated in first chapter, background subtraction is the general approach for moving object detection. In this research, an improved background subtraction algorithm was used to detect vehicles accurately. The accuracy in this step can be defined as reducing false positives and false negatives in vehicle detection. Additionally, the algorithm should provide efficient results as an initialization to occlusion detection and handling steps. For instance, if shapes of vehicles are estimated faulty, individual vehicles can be considered as occluded blobs in shape based occlusion detection algorithms.

For obtaining appropriate results, many experiments were done for implementing background subtraction. Likewise, various preprocessing and post processing steps were examined for increasing accuracy. As a result, object detection was done in this manner:

- Background modeling
- Edge adaptive threshold mechanism for object detection
- 3-d connected component analysis

In addition to object detection, occlusion detection step is also examined in this chapter. Finally, some numerical results are given at the end of the chapter in order to indicate the succeeding performance of the proposed approach.

2.1 Related Work

Most of visual surveillance systems implement background subtraction algorithm (Mixture of Gaussian) developed by Stauffer and Grimson [13]. As a result, object detection step in this thesis will be compared with Mixture of Gaussian approach and its extended version presented by Jun et al. [15]. Hereby, background subtraction algorithm using Mixture of Gaussians [13] will be summarized in this section.

Gaussian random distribution is a good way to define behavior of a variable. Intensity of a pixel in a video scene is affected by many external conditions such as lighting change, changing objects etc. As a result, a mixture of Gaussian random variables can be a useful approach to model variation of pixel values. Throughout a video sequence, a point with (x,y) coordinate is defined as $\{X_1, ...,X_t\} = \{I(x, y, i) : 1 \leq i \leq t\}$. In Mixture of Gaussian approach, the history of the pixel $\{X_1, ...,X_t\}$ is modeled by a mixture of K Gaussians. The probability of obtaining pixel value X_t is

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \qquad \textbf{(2.1)}$$

where, $\omega_{i,t}$ is the weight of the i^{th} Gaussian distribution in the mixture, $\mu_{i,t}$ is the mean value and $\Sigma_{i,t}$ is the covariance matrix of the i^{th} Gaussian distribution at frame t. $\eta(X, \mu, \Sigma)$ is a Gaussian random variable defined as:

$$\eta(X, \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(X-\mu)^T \Sigma^{-1}(X-\mu)}. \qquad \textbf{(2.2)}$$

According to memory and processing constraints, K can be between 3 and 5; moreover, $\Sigma_{i,t}$ can be modeled as $\Sigma_{i,t} = \sigma_i^2 \mathbf{I}$. As a result, it is considered that red, green and blue components of a pixel are independent and have common variance.

Every new pixel in new frames is compared with existing K Gaussian distributions. The first distribution with a smaller difference than 2.5 standard deviation is defined as the distribution of the new pixel. On the other hand, it is possible not to find even one matching distribution. In this situation, a new distribution takes place of the distribution that has minimum probability. Mean value of this new distribution is the pixel value of the recent frame, a high value is chosen as variance of distribution and a low weight is assigned to this distribution. Weight of i^{th} Gaussian distribution is updated at frame t as

$$\omega_{i,t} = (1-\alpha)\omega_{i,t-1} + \alpha M_{i,t} \qquad \textbf{(2.3)}$$

where, $M_{i,t}$ is 1 for the matched distribution and 0 for other distributions. In addition, learning rate α has significant effect on the performance of algorithm.

Now, the next step is updating matching distribution with new information, while other distributions preserve their parameters. Mean value and variance of matched distribution are updated as:

$$\begin{aligned}\mu_t &= (1-\rho)\mu_{t-1} + \rho X_t \\ \sigma_t^2 &= (1-\rho)\sigma_{t-1}^2 + \rho(X_t - \mu_t)^T (X_t - \mu_t) \\ \rho &= \alpha\eta(X_t | \mu_k, \sigma_k).\end{aligned} \quad \textbf{(2.4)}$$

After update operation, the next step is modeling background. Mixture of distributions that have bigger evidence and less variance can be chosen as the background model. For this purpose, all distributions are sorted according to ratio ω/σ. First B distributions are chosen as the background model where,

$$B = \arg\min_b \left(\sum_{k=1}^{b} \omega_k > T\right). \quad \textbf{(2.5)}$$

T is the minimum portion of data that should be consisted in the model.

2.2 Background Model

With the aim of background modeling, algorithm used in Video Surveillance and Monitoring (VSAM) project developed in the Robotics Institute at Carnegie Mellon University (presented by Collins et al. [3]) was implemented with some extensions. Proposed algorithm, which is a simple but a fast and surprisingly efficient algorithm, is a combination of adaptive background subtraction and frame differencing. Mentioned algorithm finds moving and non-moving parts of every frame using frame differencing and updates background model according to this information.

Let $I_n(x,y)$ is the intensity value at the position (x,y) at time t=n. First of all, motion information of pixel at position (x,y) is found. If a pixel $I_n(x,y)$ is moving, it satisfies,

$$|I_n(x,y) - I_{n-1}(x,y)| > T(x,y) \quad and \quad |I_n(x,y) - I_{n-2}(x,y)| > T(x,y) \quad \textbf{(2.6)}$$

where, T(x,y) is an automated threshold value, which is determined by the variation in intensity of pixel (x,y) (update equations of T will be described below).

The main purpose of this step is finding a background model $B_n(x,y)$. Values $B_n(x,y)$ and $T_n(x,y)$ are updated in an adaptive manner. B(x,y) and T(x,y) is initially set to the

first frame ($B_0(x,y) = I_0(x,y)$) and a value greater than zero ($T_0(x,y) = k$), respectively. Afterwards, $B_n(x,y)$ and $T_n(x,y)$ are updated over time as:

$$B_{n+1}(x,y) = \begin{cases} \alpha B_n(x,y) + (1-\alpha) I_n(x,y), & (x,y) \quad is \quad non-moving \\ B_n(x,y), & (x,y) \quad is \quad moving \end{cases} \qquad \textbf{(2.7)}$$

$$T_{n+1}(x,y) = \begin{cases} T_n(x,y), & (x,y) \quad is \quad moving \\ \alpha T_n(x,y) + (1-\alpha)(c)(|I_n(x,y) - B_n(x,y)|), & otherwise \end{cases} \qquad \textbf{(2.8)}$$

where, α is the learning rate which describes in what ratio of the new information is added to the old information ($0 < \alpha < 1$). Background model is updated with new intensity value, where the pixel is considered as non-moving. Furthermore, the local standard deviation at a pixel is added to $T_{n+1}(x,y)$ proportionally to the c value. If a pixel is defined as moving, B(x,y) and T(x,y) values remain same at this position. α and c values, which affect the performance of the presented algorithm significantly, were found in an empirical manner, and chosen 0.95 and 5, respectively.

In addition to original algorithm, some extensions were done for the purpose of increasing accuracy of the algorithm. First of all, HSV color space was utilized instead of RGB color space. As illustrated in the study of Molinier et al. [44], value (V) component was used to define the intensity of a pixel.

As stated in equation in 2.7 and 2.8, update condition is being a moving pixel or not. However, foreground mask is a better information, in this manner. After modeling background, through a thresholding operation, foreground mask is acquired. According to this result, B(x,y) and T(x,y) values remain same in foreground pixels, only background pixels are updated with new information as proposed by Gupte et al. [40]. Main challenge in this approach is obtaining insufficient foreground masks in early frames. Consequently, in L frames (chosen 50 in this research) original update equations (equation 2.7 and 2.8) were used. After L frames, update equations benefit from foreground mask to make update decision. If value of a pixel in foreground mask is denoted by FM(x,y), update equations will become as:

$$B_{n+1}(x,y) = \begin{cases} \alpha B_n(x,y) + (1-\alpha) I_n(x,y), & FM(x,y) = 0 \\ B_n(x,y), & FM(x,y) = 1 \end{cases} \qquad \textbf{(2.9)}$$

$$T_{n+1}(x,y)=\begin{cases} T_n(x,y),\ FM(x,y)=1 \\ \alpha T_n(x,y)+(1-\alpha)(c)(|I_n(x,y)-B_n(x,y)|),\ FM(x,y)=0. \end{cases} \quad \textbf{(2.10)}$$

In this research, experiments were done in three video sequences from different places of Istanbul. These sequences are Mecidiyekoy, Halic and Elmali videos, which have different levels of congestion from low to high. In following sections, all results will be given on these sequences. Moreover, in background modeling problem, estimated background images in 55th and 200th frames of Halic and Elmali video sequences are given in Figure 2.1.

Figure 2.1 : Background images in 55th and 200th frames of Halic and Elmali

2.3 Edge Adaptive Thresholding

Next step after background modeling is thresholding operation. In order to obtain foreground mask, background image is differentiated from original frame. If this difference is greater than a threshold, corresponding pixel is assigned as a foreground pixel. Collins et al. [3] proposed using T(x,y) as the threshold value. According to their algorithm, a foreground pixel is the pixel satisfying

$$|I_n(x,y)-B_n(x,y)|>T_n(x,y). \quad \textbf{(2.11)}$$

In this research, a new thresholding scheme is presented, which can be defined as edge adaptive thresholding. If the color of a vehicle is similar to background color,

original thresholding mechanism will not determine that vehicle. Moreover, if vehicles move slowly or stop, their information will be added to background model. In this situation, big threshold value of original algorithm causes to miss corresponding vehicles. As a result, edge information must be utilized in addition to color information to obtain accurate results. According to this approach, a smaller threshold is used in edge regions. Initially, edge points are determined according to WSMM (Windowed Second Moment Matrix) [45-47]. General edge detection algorithms work pixel based, however, WSMM works region based. As a result, edges of objects are determined more accurately.

In this approach, firstly, vertical and horizontal gradients are calculated. If a pixel value is denoted by I(x,y); image gradients $G_x(x,y)$ and $G_y(x,y)$ are determined as:

$$G_x(x,y) = \frac{I(x+1,y) - I(x-1,y)}{2}$$
$$G_y(x,y) = \frac{I(x,y+1) - I(x,y-1)}{2}. \qquad \textbf{(2.12)}$$

Afterwards, WSMM (Windowed Second Moment Matrix) of a pixel is obtained in an MxM window

$$W = \begin{bmatrix} \sum_{x,y \in MxM} w_b(x,y) G_x^2(x,y) & \sum_{x,y \in MxM} w_b(x,y) G_x(x,y) G_y(x,y) \\ \sum_{x,y \in MxM} w_b(x,y) G_x(x,y) G_y(x,y) & \sum_{x,y \in MxM} w_b(x,y) G_y^2(x,y) \end{bmatrix} \qquad \textbf{(2.13)}$$

where, $w_b(x,y)$ is a Gaussian smoothing function in the form

$$w_b(x,y) = e^{-(x^2+y^2)/2\tau^2}. \qquad \textbf{(2.14)}$$

Subsequently, if elements of W matrix are denoted as A=W(1,1) and B=W(2,2), a pixel is defined as an edge pixel where either A or B is larger than a predefined threshold.

In corresponding problem, main aim is finding edge pixels according to moving objects to use small thresholds at those points. Consequently, edge points coming from the background image must be eliminated. Utilizing from the calculated background image, two edge maps can be determined for the original frame and background image. In order to find only edge points according to interested objects,

these two edge maps are differentiated. For instance, edge maps of original frame and background image, original image and differentiated edge map of 100th frame of Mecidiyekoy video sequence are given in Figure 2.2. The first image (top left) in Figure 2.2 is the edge map of original frame. Image in top right is the edge map of background image. Moreover, edge map according to moving object is given in second row. This edge map is the difference of previous two images; also, difference is filtered with median filter to reduce salt and pepper noise. As given in Figure 2.2, edge map of moving objects is calculated accurately.

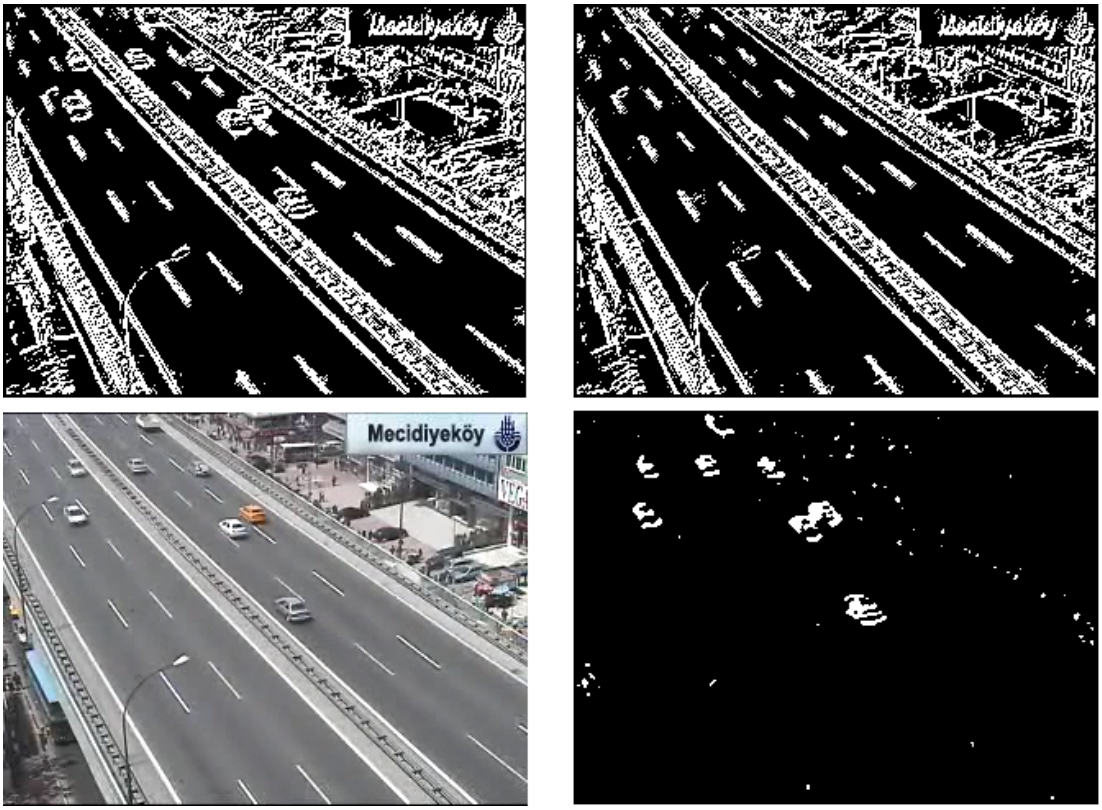

Figure 2.2 : Edge maps in 100th frame of Mecidiyekoy video

After finding the edge map of only moving objects, edge adaptive threshold mechanism is performed to find foreground objects. If the edge map of only interested objects is denoted by E and a point in this map is denoted by E(x,y); a foreground pixel is determined as:

$$\begin{aligned} &\left|I_n(x,y) - B_n(x,y)\right| > T_n(x,y), \; if \;\; E(x,y) = 0 \\ &\left|I_n(x,y) - B_n(x,y)\right| > \frac{T_n(x,y)}{k}, \; if \;\; E(x,y) = 1. \end{aligned} \qquad \textbf{(2.15)}$$

In this thresholding scheme, k was chosen 5, empirically. Moreover, while acquiring the edge map, window size (M) was assigned 5 and the standard deviation of Gaussian smoothing function was chosen 0.2. Otherwise, for large window sizes and standard deviation values edge adaptive thresholding can increase occlusion in close vehicles. Additionally, foreground masks were post processed with some morphological operations. Opening was used to eliminate noisy small objects and closing was used to increase regularity of vehicle shapes. Finally, success of this approach is shown in Figure 2.3. In Figure 2.3, the first image is original image. In second row, foreground masks determined from original threshold mechanism and edge adaptive threshold approach are given, respectively. As given in Figure 2.3 edge adaptive thresholding approach gives more accurate results, especially in regions, which are far away from traffic camera.

Figure 2.3 : Enhanced foreground objects by edge adaptive thresholding

2.4 3-D Connected Component Analysis

Another extension on the algorithm of Collins et al. [3] is post processing foreground mask with the 3-d connected component analysis, which is presented by Jun et al. [15]. The main aim of this processing is filling holes in objects and obtaining regular

shapes for objects. In binary foreground masks, there can be divided vehicles and vehicles can have missing parts because of noise in the background model. This approach tends to complete these parts from other frames of video sequence. 3-d component analysis is processed with these steps:

- In every frame, algorithm benefits from original binary masks of K previous and K following frames.
- First of all, objects are labeled (with values 1, 2, 3…) in each frame (assigning same label to neighboring pixels).
- Objects are matched in different frames by 3-d connected component analysis. For this analysis, neighbor of a pixel in consecutive frames are examined. Let, $FM_t(x,y)$ be the value of foreground mask at time t; $FM_t(x,y) = 1$ and the label of (x,y) point is L. In order to find the match of the current object (object with label L) in frame t+1, $FM_{t+1}(x,y)$ and its neighbors in frame t+1 are analyzed. Object in $FM_{t+1}(x,y)$ or in its neighbors is matched with the L^{th} object in current frame (frame t). As a result of matching objects in consecutive frames, all matches of an object in 2*K frames can be determined.
- For every object in current frame, foreground masks of matched objects in neighboring frames (totally 2*K neighbor frames) is added to foreground mask of corresponding object.
- Consequently, incomplete parts of all objects are filled with information from neighboring frames.

Accuracy of this approach is presented in Figure 2.4. Foreground mask on the right is enhanced version of original foreground mask, which is given on the left image, by 3-d connected component analysis.

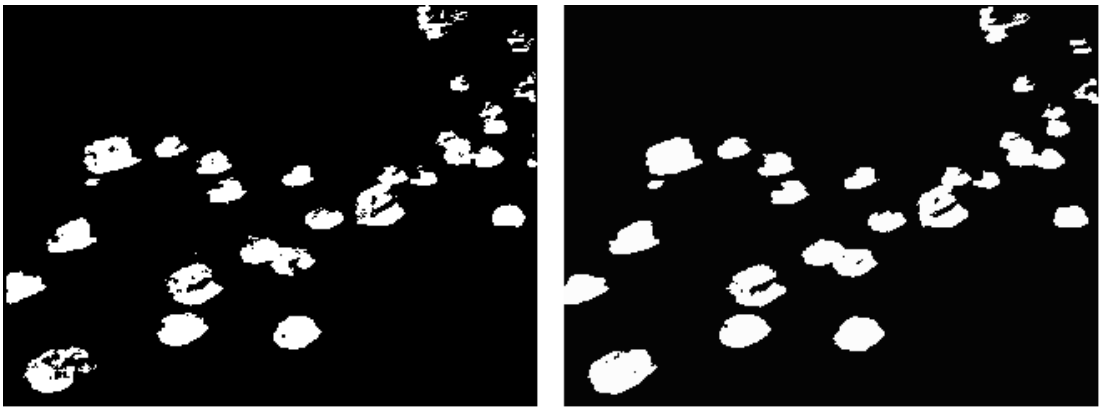

Figure 2.4 : Enhanced foreground mask by 3-d connected component analysis

2.5 Occlusion Detection

In previous sections, background subtraction and object detection algorithms were presented. According to these steps, foreground mask, which shows whether there is a vehicle on that pixel or not, was obtained. Afterwards, moving objects can be determined as connected pixel regions according to this information. The main challenge in this stage is some regions can have more than one vehicle, which is called occlusion. Occlusion happens because of some problems such as shadows, camera angle etc. The aim is to detect these occluded regions.

Occlusion detection approach of Jun et al. [15] is a significant solution for this problem. Jun et al. stated that a vehicle is nearly a convex object. However, if there are two or more vehicles in a blob, blob is less convex. As a result, for individual regions, shape of the blob and its convex hull are nearly the same. Solidity information is a good way to express similarity of original blob and its convex hull. Solidity is defined as the area of the blob divided by the area of the convex hull of the blob,

$$S_i = \frac{Area(B_i)}{Area(C_i)} \qquad \textbf{(2.16)}$$

where, B_i denotes the i^{th} blob, C_i denotes the convex hull of the blob and S_i is the solidity of corresponding blob.

In addition to solidity, the eccentricity and orientation of the blob are examined. Eccentricity of an object can be imagined as a measure of difference from a circular shape. Orientation of an object is usable only if the object is highly eccentric,

because orientation of an object with a circular shape is not realistic. After ensuring this condition, orientation of an object is compared with the orientation of the road. For single objects, orientation is similar with orientation of the road, which is determined by line detection with Hough transform (examined in section 3.3). If eccentricity of the object is E_i (E_i is between [0, 1]); orientation of the blob is θ_i (θ_i is between $[0, \pi]$) and orientation of the road is θ_L, a blob is considered as an occluded blob if

$$(S_i < T_S) \vee \{(E_i > T_E) \wedge (|\theta_i - \theta_L| > T_\theta)\} \quad \textbf{(2.17)}$$

where, T_S, T_E and T_θ are thresholds for solidity, eccentricity and orientation, respectively. These threshold values are obtained in an empirical manner.

In Figure 2.5, occlusion detection results are illustrated. In these frame instances, individual objects are represented with green rectangles. Moreover, estimated irregular blobs are highlighted with red boundaries. As indicated in Figure 2.5 occluded vehicles are detected accurately; also, there are small amount of false positives (a single object determined as occluded vehicle).

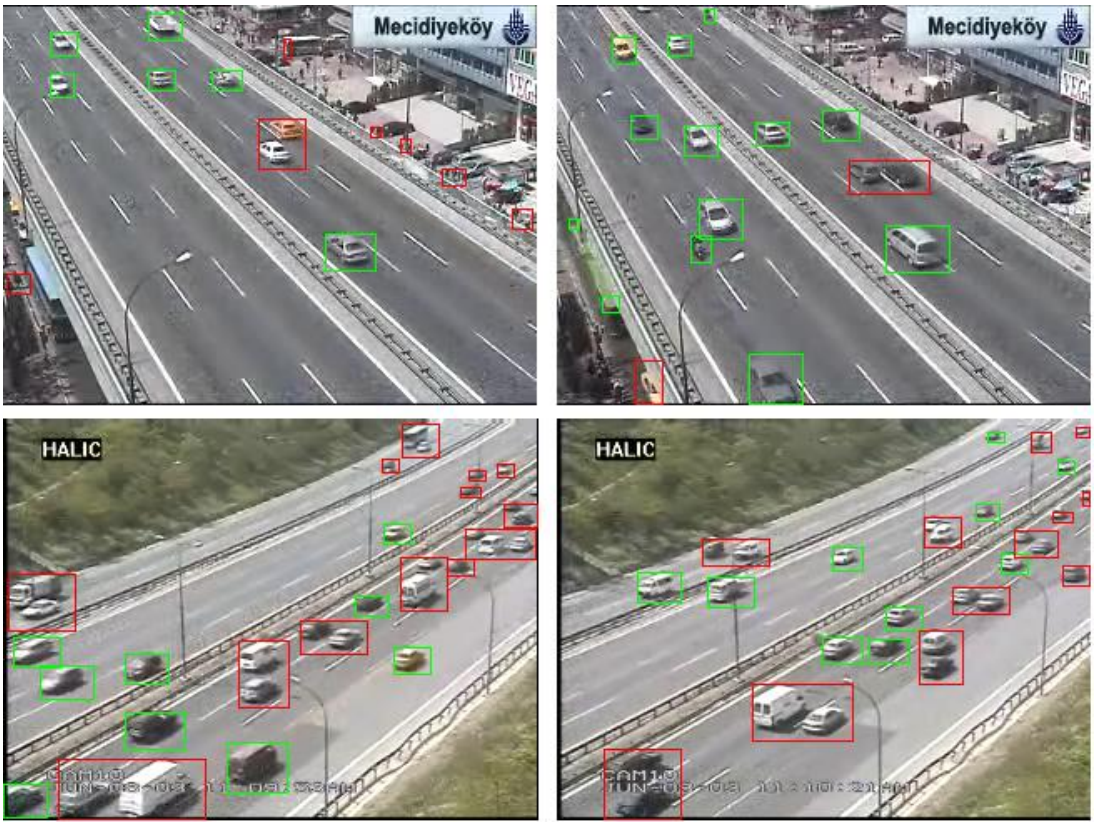

Figure 2.5 : Occlusion detection results

2.6 Results

In this chapter, a modified and accurately improved background subtraction and object detection algorithm was presented. Now, results of the algorithm will be compared with the approach proposed by Jun et al. [15], which is the extended version of Mixture of Gaussian algorithm of Stauffer and Grimson [13]. In the study of Jun et al. [15], Mixture of Gaussian approach, which was introduced by Stauffer and Grimson [13], was utilized to model background. Afterwards, 3-d connected component analysis was done to enhance foreground masks.

Proposed approach in this thesis performs background subtraction and moving object detection step as summarized below:

- Background modeling based on algorithm of Collins et al. [3], however update operations are done according to foreground mask
- Edge adaptive thresholding to obtain foreground mask
- 3-d connected component analysis (as presented by Jun et al. [15]) for enhancement of foreground mask

In order to acquire numerical results in comparison of the approach presented by Jun et al. [15] and the approach in this thesis, occlusion detection results were utilized. After occlusion detection, all results were classified into four classes: true vehicle, false vehicle, true occlusion and false occlusion. Here, true vehicle shows a correctly detected single vehicle and false vehicle is an incorrectly detected single blob, where there is not any vehicle. True occlusion represents a vehicle that is occluded with another vehicle and belongs to a detected irregular blob, which is obtained by occlusion detection. Moreover, false occlusion refers to a single vehicle, which is erroneously detected as an occluded blob. Assumption in this comparison is that an accurate algorithm provides more true vehicles and true occlusion, while reducing the number of false vehicles and false occlusions in occlusion detection. According to this assumption, some numerical results are given in Table 2.1, obtained from Mecidiyekoy, Halic and Elmali video sequences. For reducing temporal dependency, results were acquired in a 10 frame period (65, 75, 85^{th} frame etc.) from total 100 frames (65^{th} to 1055^{th}). In this table, results with no label are the results of proposed approach in this thesis. Results marked with [15] show the occlusion detection results provided from the algorithm of Jun et al. [15]. Moreover true vehicle, false vehicle,

true occlusion and false occlusion are abbreviated by TV, FV, TO and FO, respectively. Finally, the accuracy is calculated by the sum of true vehicles and true occlusions divided by total vehicles.

Table 2.1: Numerical results for occlusion detection

	Mecidiyekoy	Halic	Elmali
TV	648	541	199
TV [15]	534	441	163
FV	8	3	9
FV[15]	18	1	30
TO	72	639	574
TO [15]	69	638	514
FO	132	220	358
FO [15]	249	321	454
Total Vehicles	852	1400	1131
Total Vehicles [15]	852	1400	1131
Accuracy	0.8372	0.8411	0.6780
Accuracy [15]	0.6931	0.7702	0.5831

As stated in Table 2.1, detection results of the system presented in this thesis is more accurate than the results of proposed approach of Jun et al. [15]. Most effective improvement providing this accuracy is getting more true vehicles and less false occlusions, because of detecting foreground blobs closer to original vehicle shapes. In Figure 2.6, foreground masks and corresponding occlusion detection results are given (green rectangles for individual vehicles, red rectangles for occluded blobs). In first column, bounded vehicle (highlighted in second row with a rectangle) in foreground mask was detected with an appropriate shape with the system proposed in this thesis; as a result, this vehicle is assigned as an individual vehicle. However, in second column, the approach of Jun et al. [15] obtained the shape of the vehicle irregularly and considered this vehicle as an occlusion of more than one vehicle.

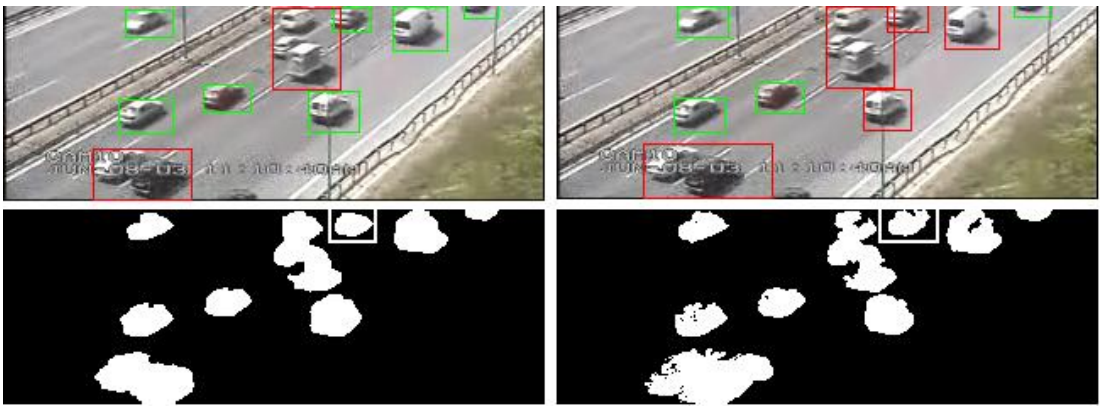

Figure 2.6 : Occlusion detection and foreground mask

Additionally, getting shapes that are more regular prevent occlusion of more than one vehicle in some situations. For instance, in third and fourth row of Figure 2.7, proposed system (first column) successfully separated close objects; nevertheless, the algorithm of Jun et al. [15] (second column) connected individual blobs of close vehicles which results occlusion. Consequently, proposed system in this thesis increases true vehicles, while reducing false occlusions in this situation.

As stated before, in Elmali video sequence, there is a high crowded traffic scene; moreover, quality of this sequence is lower. As a result, true occlusion (TO) number in this sequence is much more than true vehicle (TV) number. The road orientation in this video is determined from middle of the road, because there are significant lines only in this part of the road. Furthermore, traffic camera in Elmali video sequence is far away from the road and has a wide angle. As a result, significant perspective effect in this sequence also reduces true vehicle number, because of assigning occluded blob for individual vehicles, which are distant from middle of the road. Besides, a small Region of Interest was used to count number of vehicles; therefore, number of total vehicles is low for such a high crowded video sequence. On the other hand, in Elmali video sequence, approach of Jun et al. [15] could not detect some vehicles, especially which are far away from the camera. However, the system in this thesis gave accurate results even in this area. These vehicles are also counted as false occlusions to introduce the effect of false negatives in comparing accuracies of two approaches. A false negative class was not added to Table 2.1, because it is nearly zero in Halic and Mecidiyekoy video sequences. As a result, the accuracy of the system presented in this thesis provides increment in true occlusion, while decreasing false occlusions. As Table 2.1 is examined, it is clear that true occlusion numbers in Halic and Mecidiyekoy are nearly the same; however, in Elmali sequence true occlusion number of the proposed system is more than the approach of Jun et al. [15]. In last two rows of Figure 2.7, detection results from Elmali video sequence are given. In first column of these instances, proposed system accurately detected all vehicles, even though they are distant from the traffic camera. However, in the second column, some of these vehicles were missed by the approach of Jun et al. [15].

Finally, some visual results are given in Figure 2.7. In the figure, red boundaries represent occlusion, while green ones define single vehicles. Results from Halic,

Mecidiyekoy and Elmali video sequences are given by two rows each, respectively. The first column represents results of proposed system, while the second column belongs to the results of Jun et al. [15].

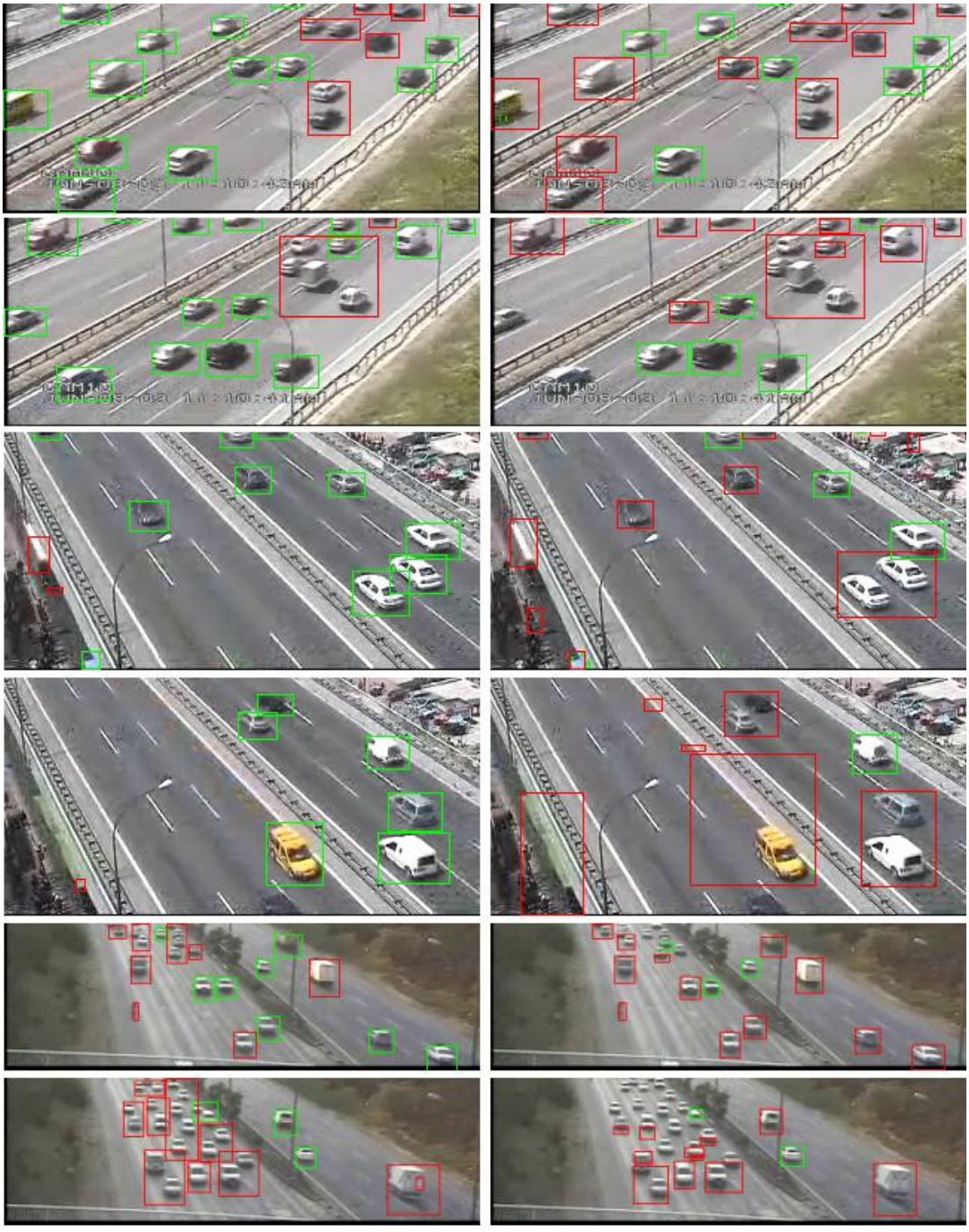

Figure 2.7 : Occlusion detection results in different video sequences

In the following chapter, a method will be presented to solve occlusion and obtain individual vehicles from occluded blob. Accuracy of that scheme is also dependent with the accuracy of this stage. For instance, occlusion-solving stage will try to

separate false occlusion results, as considering that there is more than one vehicle in corresponding blob; however, that blob contains only one vehicle actually. Consequently, Table 2.1 and Figure 2.7 prove that the system presented in this thesis successfully determines moving objects. Furthermore, proposed system is a sufficient initialization step for occlusion handling stage.

3. OCCLUSION HANDLING

In previous chapter, background modeling and moving object detection stages of the system were presented. After moving object detection, single vehicles and occluded blobs were segmented from the original frames. Next step in this traffic surveillance system is occlusion handling. The aim of occlusion handling is separating single vehicles from occluded blobs. As a result of this step, all vehicles will be obtained individually for further processes such as tracking.

3.1 Related Work

Occlusion handling process in this proposed system is based on the study of Tamersoy and Aggarwal [39]. In this study, Tamersoy and Aggarwal suggested an occlusion-handling algorithm, which is based on classification. According to their approach, a vehicle is modeled with edge orientation features of positive examples, which are obtained after background subtraction step with occlusion detection. A SVM classifier is trained with these features and further occluded blobs are separated according to SVM classifier by determining the position of individual vehicles. Occlusion handling in this thesis was implemented according to this presented framework [39]; moreover, new features were added to the system in order to improve the performance. In this chapter, the occlusion-handling system will be presented and accurate results will be given at the end of the chapter.

3.2 Support Vector Machines (SVM)

As stated before, occlusion handling in this system is based on learning and classification. In the learning step, all vehicles are projected into the vector space with obtained features. Afterwards, a classification rule is learned from training examples and further examples in occluded blobs are classified according to trained model to find vehicles. Support Vector Machines can be the solution for this problem.

SVM classifier is a succeeding method, which is widely used in classification problems. Moreover, SVM provides performance increase in time consumption. The learning phase is almost same in all classification methods by the means of time consumption. Additionally, in spite of the methods such as k-nearest neighbors algorithm (k-nn), a decision boundary is learned for classifying further examples in SVM. This approach provides performance increase in classification phase, because classification is done with a limited number of multiplications and summations. As a result, Support Vector Machines (SVM) were utilized for training and classification. In this section, SVM will be explained briefly. This summary is organized according to information from Alpaydin [48] and Yang [49].

Let us define the problem as a binary classification problem with classes -1 and +1. In corresponding classification problem, a mapping function is found between feature examples and classes as:

$$\begin{aligned} &y = f(x, w) \\ &\vec{w}^{T}\vec{x} > +1, \ for\ class +1 \\ &\vec{w}^{T}\vec{x} < -1, \ for\ class -1. \end{aligned} \qquad \textbf{(3.1)}$$

Here, **w** is the parameter vector for mapping and **x** is a point on feature space. General solution in classification is to find the optimum **w** to minimize the error between **x** points and classes. On the other hand, in SVM, a hyper-plane that separates two classes is found by maximizing the margin between two classes with minimizing ||w||, such as in Figure 3.1 [48].

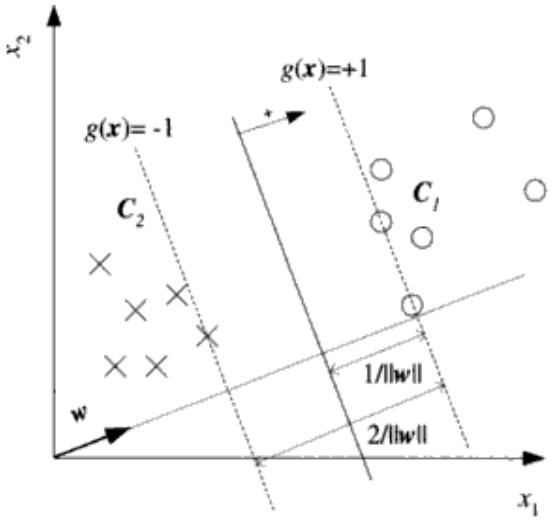

Figure 3.1 : SVM classification example

In SVM, support vectors are defined as the most difficult points for classification, which are on the ends of the margin of two classes C_1 and C_2 as given in Figure 3.1 [48]. While classifying further examples, decision is made according to the similarity between the corresponding vector and support vectors. As a result, future vectors are classified as

$$y = sign\left\{\sum \alpha_i t_i \psi(\vec{x}, \vec{x}_i)\right\} \quad \textbf{(3.2)}$$

where, α_i is the positive parameter of corresponding support vector x_i; t_i is the label of the class consisting i^{th} support vector and sign is the operator giving the sign of the calculation as +1 or -1. Additionally, $\psi(\vec{x}, \vec{x}_i)$ is the kernel function used in SVM classifier where $\psi(\vec{x}, \vec{x}_i) = \psi(\vec{x})\psi(\vec{x}_i)$ and $\psi(\vec{x})$ is a feature vector. Main challenge in SVM is designing an appropriate kernel function. Generally, linear functions, polynomial functions, radial basis functions and sigmoid hyperbolic tangent function are used as kernel functions.

In Figure 3.2 [49], SVM and conventional algorithms, which find decision boundary by minimizing error, are compared. In first image, two classes are separated accurately with conventional algorithms. However, when new examples are added to problem such as in second and third image, classes became unbalanced and examples shown with triangles are misclassified. On the other hand, SVM successfully classifies these examples as given in last column of Figure 3.2. Consequently, SVM is the most robust method for classifying future examples [49].

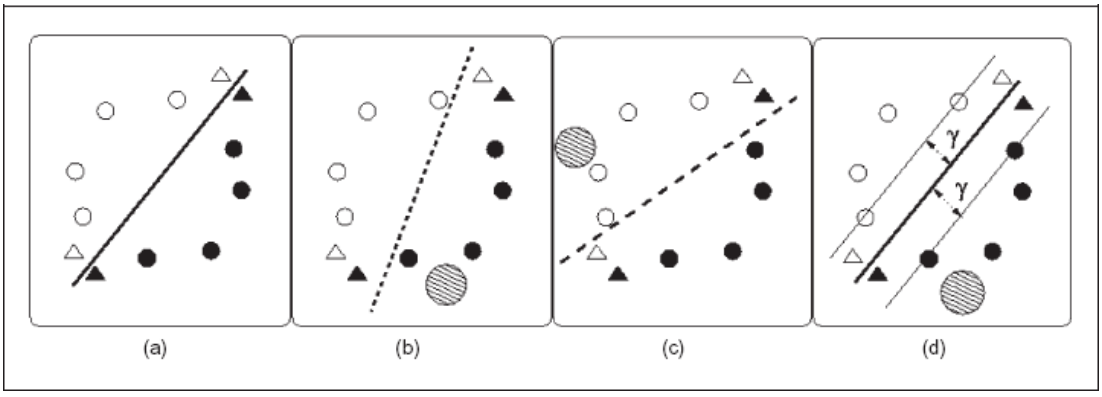

Figure 3.2 : SVM and conventional algorithms

In this thesis, implementation of SVM was realized with the help of SVM^{light} framework developed by Joachims [50].

3.3 Automatic Region of Interest (ROI) Detection

In traffic surveillance systems, image quality changes according to camera location and camera angle. In order to process high quality parts of the image, reducing source (time, memory etc.) consumption and developing real time applications, a region of interest is determined manually as an initialization step. All processes are done only in this region. Manual detection of ROI is a simple approach; however, it can bring some difficulties. For instance, in Istanbul, there are many cameras in different locations, as a result, obtaining ROI for all cameras is a time consuming operation. On the other hand, recent cameras have the ability of remote control and control centers may change the angle of camera when necessary, so every time a new ROI must be assigned before further processing. Consequently, automatic ROI detection can be necessary for robust and automated traffic surveillance. In this section, an automatic ROI detection algorithm will be presented. In this approach, the main aim is to determine a ROI, which is close to the camera and perpendicular to road orientation that provides appropriate solution for further processing such as rectifying image sequences in the study of Grammatikopoulos et al. [30].

First of all, an activity map is obtained from video sequence, as illustrated by Stewart et al. [51]. Activity map defines whether a vehicle pass through the pixel position or not. For this purpose, an accumulator is used to determine how many times a pixel is assigned as a foreground pixel. If the accumulator matrix is denoted by ACC and foreground mask is denoted by FM; ACC matrix is updated in each video frame as:

$$ACC(x,y) = \begin{cases} ACC(x,y)+1, if\ FM(x,y)=1 \\ ACC(x,y), if\ FM(x,y)=0. \end{cases} \quad \textbf{(3.3)}$$

After finding accumulator matrix in a predefined number of frames, activity map A is obtained by

$$A(x,y) = \begin{cases} 1, if\ ACC(x,y)>0 \\ 0, if\ ACC(x,y)=0. \end{cases} \quad \textbf{(3.4)}$$

For three video sequences (Mecidiyekoy, Halic and Elmali), acquired activity maps after 250 frames (nearly 10 seconds) are given in Figure 3.3.

Figure 3.3 : Activity maps obtained from 250 frames

Afterwards, road lines are extracted from Hough transform by using obtained activity maps. Hough transform mainly transforms image plane into parameter space and is widely used for detecting lines in an image. In this approach, lines are defined with their polar representation [52]

$$\rho = x\cos\theta + y\sin\theta. \qquad \textbf{(3.5)}$$

According to the given polar representation, following algorithm is implemented for line extraction [52]. Here, input image E is an MxN edge map, where an edge pixel is defined as E(i,j) is 1. Let ρ_d , θ_d be the arrays containing finite elements of probable

ρ, θ values, respectively and R, T are the number of elements in these arrays. Following steps are implemented for line detection in Hough Transform.

1) Initialize accumulator array A(R, T) with zero values.
2) For each points satisfying E(i, j) = 1 and for h=1,….,T
   a) Find $\rho = i\cos\theta_d(h) + j\sin\theta_d(h)$
   b) Obtain k, which is the index of closest element in ρ_d to ρ
   c) Increment value of A(k, h) by 1
3) Parameters of lines are local maxima found in A matrix such that A(k, h) > T, where T is a threshold value.

As stated in algorithm described above, polar parameters of all lines are extracted from Hough Transform. Figure 3.4 shows detected lines with Hough Transform according to the activity maps given in Figure 3.3.

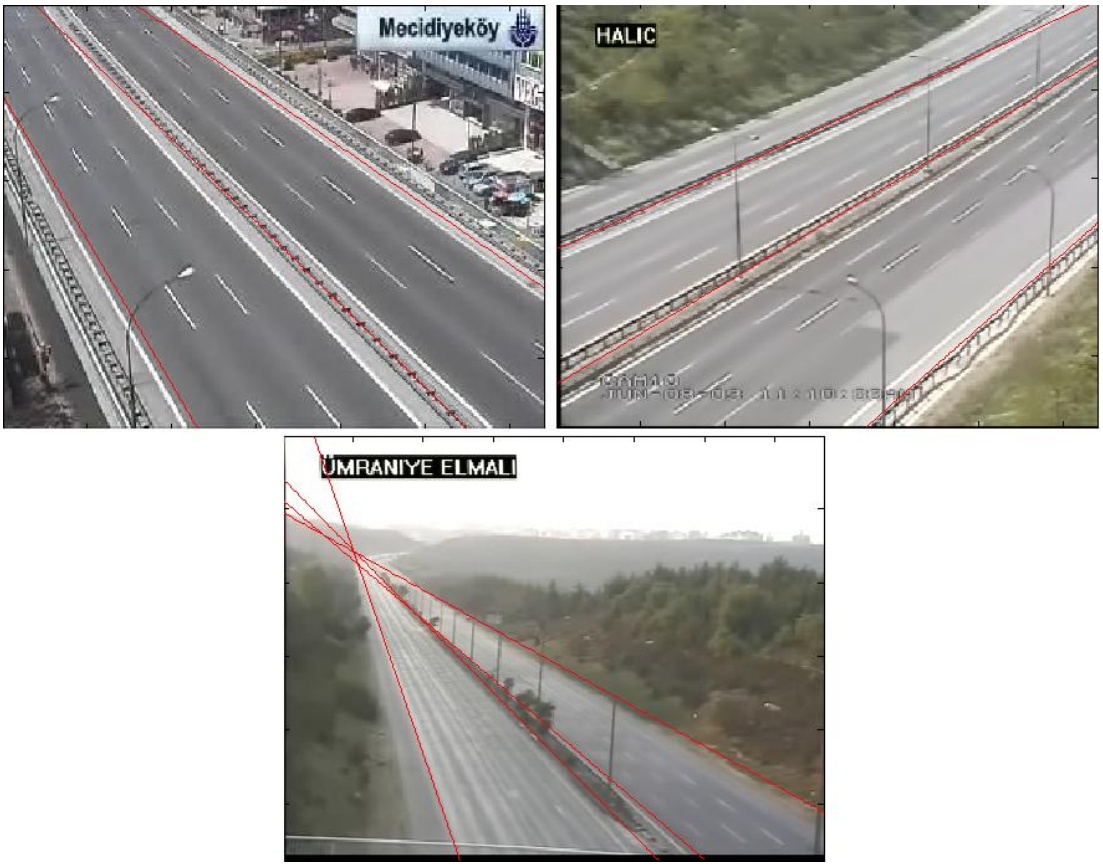

Figure 3.4 : Detected lines in different video sequences

After extracting all lines for video sequences, two boundary lines are used to find a middle line. In three video sequences shown in Figure 3.4, middle lines are detected, but it can be impossible for some other sequences. To develop a robust algorithm,

central line is determined from two boundary lines. Middle line has ρ and θ values, which are calculated as the mean of corresponding ρ and θ values of two boundary lines. The assumption in obtaining ROI is that road orientation is parallel to middle line in the image. Therefore, the lope of middle line (m) is calculated according to the arithmetic geometry equation of a line,

$$m = \frac{y_n - y_1}{x_n - x_1} \qquad \textbf{(3.6)}$$

where, (x_1,y_1) and (x_n,y_n) are the first and the last point on middle line, respectively. In this case, boundary of the ROI is a line that is perpendicular to the middle line and intersects with two boundary lines extracted in Figure 3.4. This approach is illustrated in Figure 3.5.

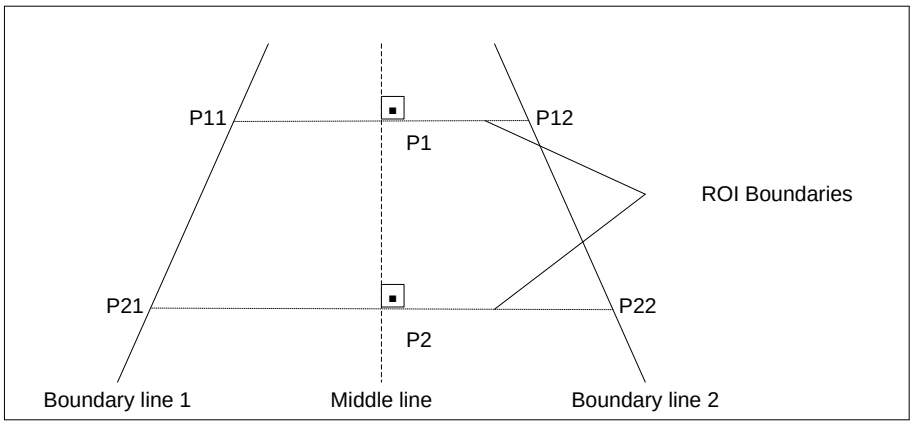

Figure 3.5 : ROI detection approach

As explained in Figure 3.5, two points (P1, P2) are selected on middle line and these points are connected with boundary lines by two perpendicular lines to middle line. These perpendicular lines are found according to the rule that multiplication of slopes of two perpendicular lines is -1. Afterwards, intersection of perpendicular lines and boundary lines provides four points (P11, P12, P21, P22), which are corners of obtained ROI. The challenge in this scheme is to choose P1 and P2 points on middle line. Still, these points can be extracted automatically, by proportionally to the length of the middle line. However, it must be controlled that perpendicular lines to these points (P1 and P2) must intersect with the boundary lines in image border. Automatically detected Region of Interest for different video sequences are given in Figure 3.6.

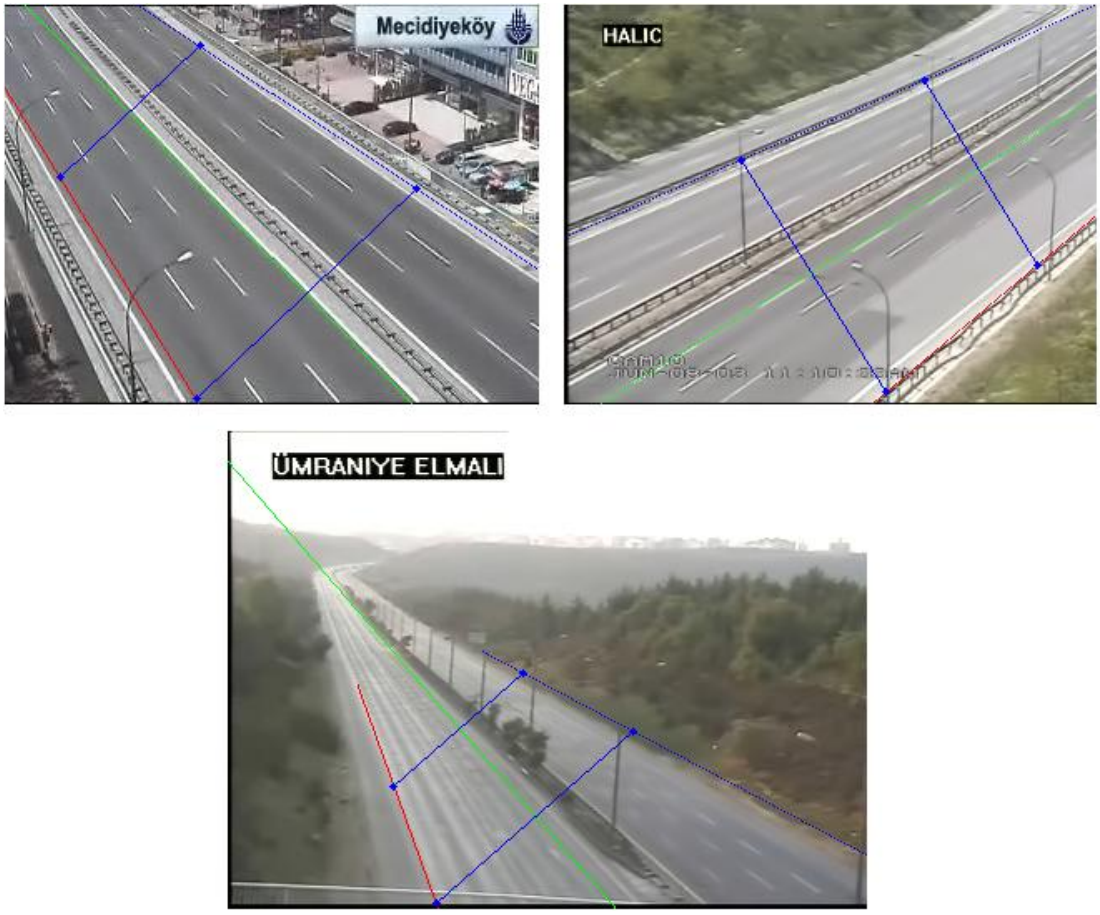

Figure 3.6 : ROI detection results

In Figure 3.6, boundary lines, middle line and ROI boundaries are shown. As figured out, proposed automatic ROI detection approach accurately and automatically obtains appropriate ROI in video sequences.

Although an accurate ROI detection algorithm is presented in this thesis; in further processes, detected regions in the study of Tamersoy and Aggarwal [39] will be used for efficient comparison with their approach, because they used same video sequences in their study. In their study, they determined one ROI each for Elmali and Halic video sequences. Additionally, they separated Mecidiyekoy video sequence into two parts (ROI) such as bottom and top. These four regions for three video sequences will be also used as detected ROI in this thesis.

3.4 Training Stage

As stated before, occlusion-handling step is based on training and classification. The training phase of this process will be explained in this section.

3.4.1 Positive and Negative Examples in Training

In all classification problems, a model must be trained with training examples. These examples are manually chosen in several systems. However, in traffic surveillance systems there is the ability of acquiring training examples automatically. As stated in the previous chapter, after background modeling and moving object segmentation, individual vehicles can be acquired by means of occlusion detection approach. A period of frames is assigned as training phase and in this period, single blobs, which are obtained from occlusion detection, are acquired as positive examples of vehicles. In this study, 600 frames were used for training period. In addition to positive examples, classifier also needs negative examples to train a sufficient model. Negative examples are derived from positive examples. After a positive example is obtained with its bounding box, this box is shifted in image space by a half of width and height of bounding box in eight neighboring directions. This process provides eight negative examples according to a positive example. Due to eight negative and one positive example, classifier learns to detect center of a vehicle accurately, which will help to separate vehicles in occluded blobs as described in following sections. Instances of a positive example from Mecidiyekoy video sequence and corresponding negative examples in eight directions are shown in Figure 3.7.

Figure 3.7 : Positive and corresponding negative examples

The advantage of this automated training approach is that obtained examples are robust to different video sequences and different illumination conditions, because, instead of using a general vehicle model, a model is determined from the video sequence automatically. Parameters such as orientation of the road, camera angle and camera distance affect the shape, orientation and size of the vehicles. Presented training approach handles these variations and provides a vehicle model appropriate to conditions of corresponding video sequence. In Figure 3.8, some instances of positive and negative examples for Mecidiyekoy, Halic and Elmali video sequences are exemplified.

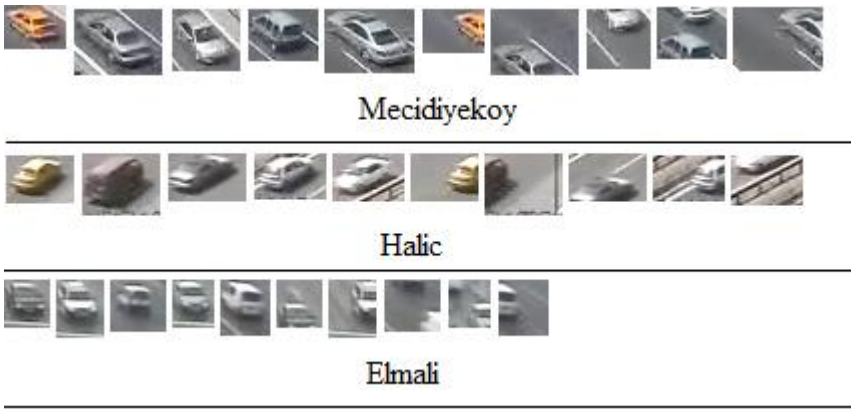

Figure 3.8 : Positive and negative examples from different video sequences

As illustrated in Figure 3.8, for different video sequences vehicles have distinctive properties in size and orientation. As a result, classification method should learn characteristic features of vehicles from positive and negative examples, which are collected from corresponding sequence, automatically. In next section, appropriate features for this purpose and feature extraction method will be presented.

3.4.2 Feature Extraction

Feature extraction is the method of projecting vehicle examples into vector space or deciding how to define a vehicle by features. After extracting positive and negative examples, the next step is finding feature vectors of these examples and determining the classifier model with these features using SVM based training. With the assistance of SVM training, a decision boundary is found to classify future examples. For every video sequence, a different train model is obtained to handle characteristics of corresponding video sequence.

As stated in previous section, the vehicle orientation is a distinctive parameter in different video sequences. Therefore, edge orientation of vehicles is a reasonable characteristic to model vehicles. Tamersoy and Aggarwal [39] used edge orientation to define a vehicle. In this approach, a vehicle is projected into vector space by using Histogram of Oriented Gradients (HoG), in a similar way with the usage of this method by Dalal and Triggs [53]. According to this approach, features are extracted with following steps:

1. Divide image into 2x2 cells, which results 4 sub-images.
2. For every sub-image, calculate edge gradients and edge orientations by Sobel edge operator.

3. Find 8 bin histogram of edge orientations in each sub-image.
4. Normalize histograms according to size of the sub-image
5. Combine normalized histograms of sub-images
6. As a result, a feature vector with 32 elements is obtained for corresponding image.

An advantage of this implementation is that size of the feature vector is constant, independently from image size. In this thesis, this approach was implemented with some differences. Instead of only using edge orientation features, some other features were added to the system. These features are locally normalized gradients, direct normalized gradients and square mapped gradients, which were described by Petrovic and Cootes in their study of vehicle type recognition [54].

Let S_x and Sy be the edge gradients obtained by Sobel edge operator, as a result, edge orientation and other features are calculated according to Table 3.1 [54].

Table 3.1: Different features used in feature extraction

Feature	Equation
Edge Orientation	$S_\alpha = \arctan(S_x / S_y)$
Locally Normalized Gradients	$(G_x^{LN}, G_y^{LN}) = \left(\frac{S_x}{\frac{1}{L^2}\sum_{LxL}\sqrt{S_x^2 + S_y^2}}, \frac{S_y}{\frac{1}{L^2}\sum_{LxL}\sqrt{S_x^2 + S_y^2}} \right)$
Direct Normalized Gradients	$(G_x^{DN}, G_y^{DN}) = \left(\frac{S_x}{\sqrt{S_x^2 + S_y^2}}, \frac{S_y}{\sqrt{S_x^2 + S_y^2}} \right)$
Square Mapped Gradients	$(G_x^{SM}, G_y^{SM}) = \left(\frac{S_x^2 - S_y^2}{S_x^2 + S_y^2}, \frac{2S_xS_y}{S_x^2 + S_y^2} \right)$

As stated before, edge orientation is a prominent feature to define a vehicle in different video sequences. Additionally, edge orientation is a robust feature because of its independence from lighting conditions. Other features in Table 3.1 are derived from edge gradients. Locally normalized gradients are calculated by dividing edge gradients by the mean edge strength calculated in LxL neighborhood of corresponding point. Therefore, significant edges have maximum value in this manner. Square mapped gradients define parallel and diagonal edges to the axis. Moreover, this feature is robust to variation in noise and edge orientation.

Consequently, it is an appropriate solution for defining local structure [54]. On the other hand, square mapped gradients and direct normalized gradients lie on unit circle. In addition to edge orientation, these gradient-based features add new information to vehicle model and vehicles are represented more accurately than the study of Tamersoy and Aggarwal [39], which proposes to use only edge orientation.

Performances of these features were also examined. For this purpose, in training period, feature vectors were extracted by the means of Table 3.1 and train models were extracted according to these features, individually. On the other hand, a train model was created by combining all these four features. Afterwards, test examples were generated in an automated manner from following frames. Performances of classifiers were evaluated according to the accuracy of these feature sets in test examples from different video sequences. In this process, all operations were done in interest regions (ROI), determined in the study of Tamersoy and Aggarwal [39]. This performance comparison is given in Table 3.2.

Table 3.2: Performance comparison of feature sets in different video sequences

ROI	Pos. Train	Neg. Train	Pos. Test	Neg. Test	EO [39]	DN	LN	SM	Combined features
Mecid. Bottom	1443	11323	1461	1426	0.867	0.811	0.748	0.886	0.944
Halic	1116	8999	1545	1506	0.916	0.967	0.931	0.933	0.98
Mecid. Top	1815	14388	1693	1673	0.841	0.823	0.791	0.824	0.933
Elmali	1186	8867	1692	1525	0.883	0.922	0.915	0.875	0.94

In Table 3.2, there are the names of four ROI for three video sequences (Mecidiyekoy, Halic and Elmali) in first column. Additionally, 2^{nd} to 5^{th} columns (Pos. Train to Neg. Test) show the numbers of positive train examples, negative train examples, positive test examples and negative test examples in different video sequences, respectively. Last five columns represent the accuracy of different features in test examples. These features are edge orientation, direct normalized gradients, locally normalized gradients, square mapped gradients and a combination of all four features, respectively.

As stated before, negative examples are obtained from eight corners of positive examples in training stage. As a result, number of negative train examples is nearly eight times of number of positive train examples. However, test examples were

acquired in approximately same number in order to calculate the performance of features more efficiently.

Table 3.2 proves that combination of features generates the most accurate feature set in all video sequences and it is more appropriate than only using edge orientation such as in the study of Tamersoy and Aggarwal [39]. As a result, this concatenated feature set was used to train a SVM classifier in this thesis.

Finally, Receiver Operating Characteristic (ROC) curves of combined feature set in different video sequences are given in Figure 3.9.

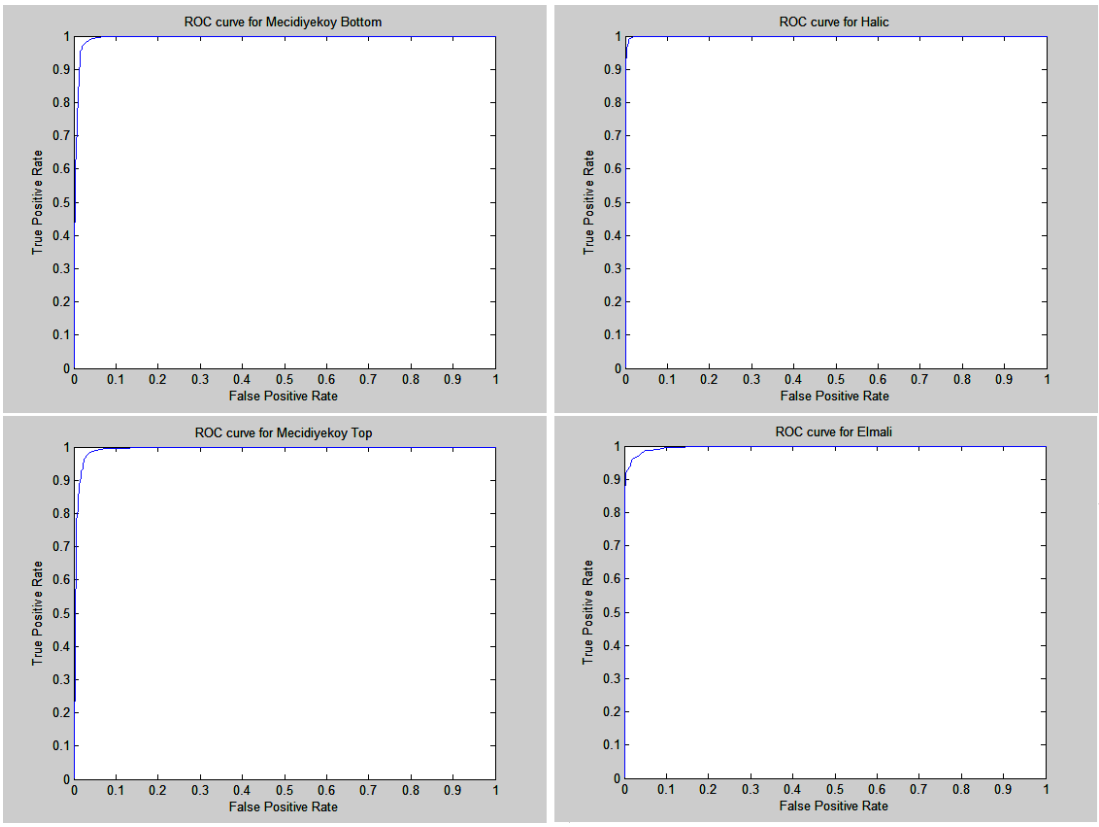

Figure 3.9 : ROC curves of combined feature set

3.5 Occlusion Handling in Irregular Blobs

As stated before, occlusion detection, which is done after background subtraction and moving object segmentation, produces two results: individual vehicles and occluded blobs. In occlusion handling, the main aim is to determine single vehicles from occluded regions. For this purpose, a classification-based approach, which is presented by Tamersoy and Aggarwal [39], was implemented. After training phase that is explained in section 3.4 a classifier model was extracted. With the help of this

model, an image patch can be examined to decide whether there is a vehicle in it or not. Consequently, all vehicles in an occluded blob can be determined according to the binary classifier obtained by training.

An occluded blob is a large image patch, containing more than one vehicle in it. Occlusion handling process should determine the positions of single vehicles in occluded region. In order to detect these positions, a sliding window is utilized for recognition of a vehicle in occluded image patch. In each position of this sliding window, the feature vector representing corresponding window is calculated according to the feature extraction method that is described in section 3.4.2. Afterwards, this feature vector is classified with trained classifier model. Classification of the window produces the decision whether this window represents a vehicle or not.

The main challenge in this approach is to decide the size of sliding window. This decision can affect the accuracy of the system dramatically. In this study, size of the window was determined from the positive training examples. Because of the camera angle and distance of camera, sizes of vehicles change in different video sequences. Therefore, for each video sequence, median of the width and height values of positive training examples was assigned as the width and height of the sliding window. As stated before, positive examples are obtained automatically from individual vehicles; as a result, size of the sliding window is also calculated automatically. In Figure 3.10, sorted width, height values and median values (with circles on lines) are given for different video sequences.

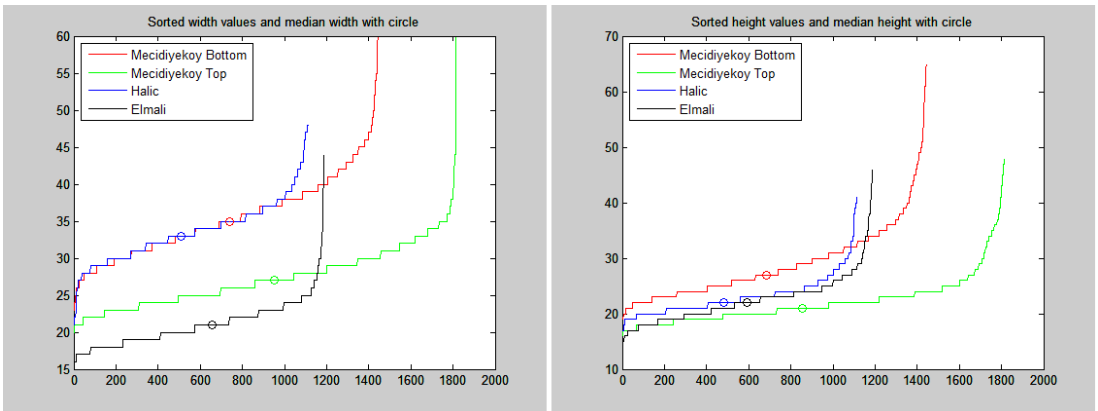

Figure 3.10 : Obtaining width and height of sliding window

As previously stated, using the obtained training model and sliding window for each video sequence, occluded blobs are segmented in corresponding sequences. After extracting the feature vector of sliding window in a point in occluded blob, this window is classified with training model. If a vehicle exists in this window and the vehicle is in the center of the window, the classifier produces a positive result. After classifying all image patches of occluded blob with a sliding window manner, a binary image is obtained for the corresponding occluded blob, which shows the vehicle centers. This process is illustrated in Figure 3.11.

Figure 3.11 : Sliding window approach

The first image in Figure 3.11 is the foreground mask of an occluded blob. Afterwards, corresponding image patch is obtained from the original frame and this patch is padded for the sliding window process. This padded image is the second one in Figure 3.11. Classification results of the sliding window for every point in image patch provides the third image in Figure 3.11. As indicated in Figure 3.11, binary image also consists of some connected regions. These regions belong to individual vehicles. However, this binary image includes some noise, which can produce false positive vehicles around significant connected regions. Therefore, these noisy small regions must be eliminated. The assumption in this control is that two vehicles cannot be closer than a distance threshold (it will be explained in details). After elimination of noisy binary regions, detected vehicles are found as given in the last image of Figure 3.11. All vehicles in this image are defined by a rectangle with a size of the sliding window. Centers of individual vehicles are centers of significant binary regions, which are illustrated in the third image of Figure 3.11.

In order to eliminate noisy regions, following steps are implemented in binary classification results of occluded blobs.

1. Sort all connecting regions according to their areas.
2. Start from the largest region and process all regions in order.

3. If the distance between center of any larger region and center of corresponding region is smaller than a threshold, processed region is eliminated as a noisy region.
4. Otherwise, mark the center of the region as a center of individual vehicle.

On the other hand, distance threshold is also determined automatically according to the size of the sliding window (median size of positive examples).

3.6 Results

Occlusion handling system consists of training and classification steps, as explained in previous sections. In all different video sequences, training phase learns a classification rule, which is a specific model for corresponding sequence. After training period, all occluded blobs are segmented according to the approach described in section 3.5.

An accurate vehicle segmentation system should separate all individual vehicles in an occluded blob; therefore, the system should produce as many as true positives and less false negatives. Additionally, all positive results should be an exact vehicle, which stands for reducing false positive number. In order to measure the accuracy of proposed occlusion handling system, these metrics were counted for 100 frames of all video sequences. To reduce temporal dependency, these instance frames were chosen in a 10-frame period. Additionally, accuracy results of the system were compared with the results of the similar study, which was introduced by Tamersoy and Aggarwal [39]. Results were taken from their paper that implements occlusion-handling on same video sequences. These results are presented in Table 3.3.

Table 3.3: Performance of occlusion handling approaches in different video sequences

Sequence	Total Vehicles	Detected Vehicles	False Pos.	False Neg.	Accuracy	Accuracy of [39]
Mecid. Bottom	307	299	3	8	0.9739	0.9343
Halic	536	508	9	28	0.9478	0.9075
Mecid. Top	710	696	13	14	0.9803	0.9605
Elmali	1129	1046	83	83	0.9265	0.9239

In Table 3.3, total vehicles represent the actual number of total vehicles in 100 frames of video sequence. Detected vehicles number is the amount of correctly segmented vehicles. False positive and false negative numbers are also given in the

fourth and fifth columns of the table, respectively. Finally, in last two columns of Table 3.3, accuracy of the proposed system is compared with the accuracy results of Tamersoy and Aggarwal [39]. As proved in the Table 3.3, the occlusion-handling system presented in this thesis produces more accurate results when compared with this similar study. The reason for this accuracy increase is the improvement in feature extraction method. Additionally, developed background subtraction and moving object detection approach in chapter 2 affects these results positively.

In addition to the numerical results, some visual results are illustrated in Figure 3.12.

Figure 3.12 : Occlusion handling results

In Figure 3.12, results from Mecidiyekoy Bottom, Mecidiyekoy Top, Halic and Elmali video sequences are given in two rows each, respectively. Green vehicles represent individually detected vehicles, red bounded vehicles show separated single vehicles from occluded blobs. As indicated in Figure 3.12, occlusion-handling

approach can separate occluded blobs to individual vehicles, even in a high crowded video sequence such as Elmali.

As stated before, because of the detected shape of single vehicles, a single vehicle can wrongly detected as an occluded blob. However, this occlusion handling approach also determines single vehicles, whether they are assigned as an irregular blob. These kinds of instances, which are single vehicles bounded with red rectangle, are given in the first and fourth row of Figure 3.12. Additionally, as illustrated in Figure 3.12, a region in size of the sliding window is assigned to separated vehicles from occluded blobs. However, regions of individual vehicles are the results obtained in background subtraction.

As presented in second and third chapter, moving object detection and occlusion handling steps were implemented accurately. In order to show this accuracy, some numerical and visual results were given in corresponding chapters. As a result of these two chapters, vehicles can be counted to find the number of individual vehicles and detecting traffic density, in a traffic scene or for a period of time. Additionally, all vehicles can be located, which is significant for further processes like tracking.

4. TRACKING

Until now, all vehicles were segmented through moving object detection and occlusion handling steps. As a result, traffic parameters such as vehicle number and traffic density were extracted. In addition to these parameters, another aim is to obtain individual vehicle speeds and mean vehicle speed in a traffic scene. In order to obtain speed information, all segmented vehicles must be tracked in a period of time and trajectory of a vehicle must be determined. In this chapter, tracking step of the system will be explained.

4.1 Tracking Method

In first chapter, related work on tracking approach was explained in detail. There are numerous methods in literature for this purpose. Main approaches can be classified into four groups: model based tracking, active contour based tracking, region based tracking and feature based tracking. These approaches have specific advantages. For instance, feature based methods track deterministic features rather than vehicles, which provide occlusion independent structure. However, in this study, all vehicles were segmented for tracking approach. As a result, there is significant information for tracking. As a result, a simple approach such as region matching can be sufficient to track vehicles. In this system, the fundamental aim is to match two vehicles in consecutive frames, to find a trajectory and speed of a vehicle.

As stated before, segmented vehicles, their position and region were obtained before tracking step. For individual vehicles, regions of objects were determined in the background subtraction step. However, regions of the separated vehicles from irregular blobs were assigned as the sliding window in occlusion handling approach. In order to match two vehicles in consecutive frames, these regions are used. Let denote the number of vehicles in following two frames be N1 and N2. The problem can be thought as connecting individual vehicles. There are N2 possibilities for matching a vehicle from frame t with another vehicle from frame t+1 as illustrated in Figure 4.1.

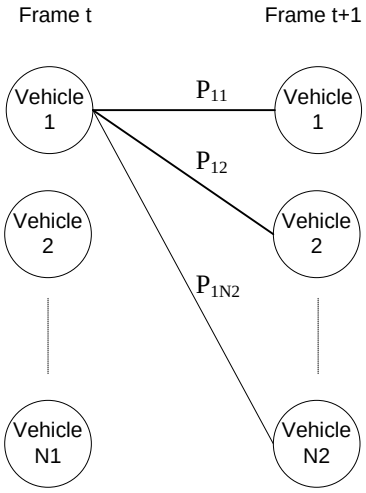

Figure 4.1 : Matching approach

In this example, matching is done between two frames t and t+1. As indicated in Figure 4.1, first vehicle can be matched with N2 vehicles in the frame t+1. This is similar for all vehicles in frame t. Moreover, all connections (matching) have a cost. Matching operation aims to find the appropriate matching combination, which has the minimum cost or has the maximum probability. In this manner, the cost is inverse proportional with the similarity of two vehicles. In other words, similarity of two vehicles defines the probability (for instance, P_{11}, P_{12}, and P_{1N2} for first vehicle) of being same vehicles in different frames.

As previously stated, regions of all individual vehicles were obtained in previous steps. Therefore, intensity correlation of two regions in consecutive frames can determine the similarity of two vehicles or the probability of being same vehicle. However, finding correlation of two regions is impossible, because regions do not have the same size in different frames. Utilizing correlation of intensity histograms can be a reasonable and more efficient approach, in this manner. To find the similarity of two vehicles, firstly, 255 bin intensity histograms of two vehicles are calculated. After that, these histograms are normalized with the number of pixels in corresponding regions. Finally, the similarity of two regions is assigned as the correlation value of two histograms. Three vehicles, their histograms and correlation value of histograms are given in Figure 4.2. In Figure 4.2, first and second vehicles are detected regions of same vehicle in different frames. Additionally, second and third regions belong to different vehicles in consecutive frames. As illustrated in Figure 4.2, same vehicles produce a high correlation, although they are in different

frames. However, histograms of different vehicles have substantially lower correlation.

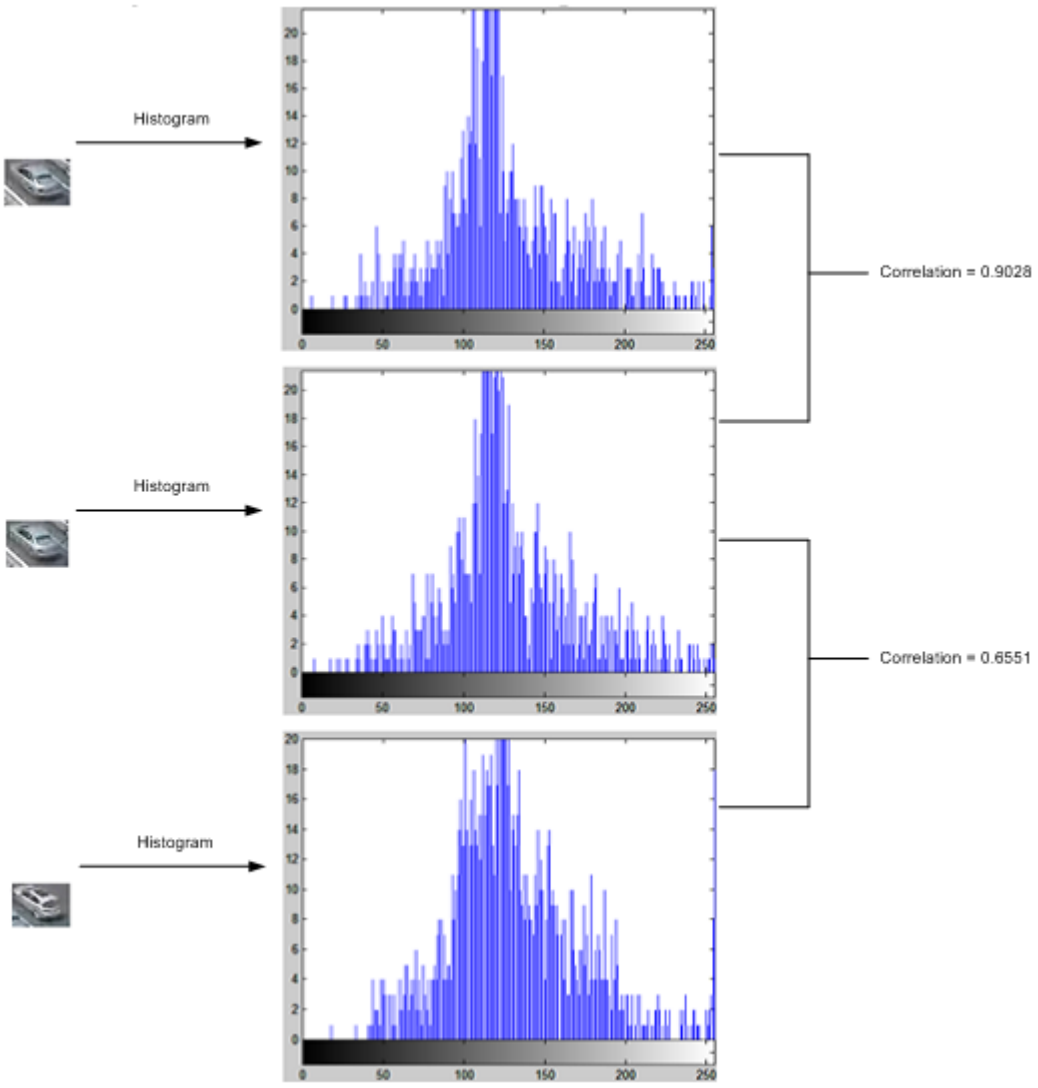

Figure 4.2 : Intensity histograms of different regions

After finding the similarity of all vehicles between consecutive frames, next step is to produce a matching probability matrix. In this matrix, a cell defines the probability or correlation between two regions. This matrix (PM) is in the form

$$PM = \begin{bmatrix} P_{11} & P_{12} & . & . & . & P_{1N2} \\ P_{21} & P_{22} & . & . & . & P_{2N2} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ P_{N11} & P_{N12} & . & . & . & P_{N1N2} \end{bmatrix}. \quad \textbf{(4.1)}$$

In this matrix, rows represent the vehicles in frame t; columns show the vehicles in frame t+1. On the other hand, number of vehicles can be different in consecutive frames, because of the entering vehicles into the ROI and exiting vehicles from the ROI. Moreover, false positive and false negative vehicles, which are obtained from faulty occlusion handling, can also affect the inequality. There exists three possibilities in this manner: N1 = N2, N1 > N2, N1 < N2. The aim is to find exact matches from the probability matrix, while maximizing the total probability and eliminating outlier regions (vehicles). In order to find optimum combination, the rule is that i^{th} vehicle from the frame t and j^{th} vehicle from the frame t+1 match only if P_{ij} is the maximum value in i^{th} row and j^{th} column of the matrix PM.

The rule above provides to find an appropriate match in consecutive frames. After obtaining this matching, position change of a vehicle can be determined to find the speed of the vehicle. However, finding the speed from only one frame is very noisy. Therefore, speed of vehicle is found from the frame vehicle entered the ROI until the current frame. Afterwards, result speed is assigned as the mean value of all speed values in corresponding frames. For this purpose, moving average is calculated for every vehicle. Two values are stored for each vehicle: mean speed until now and number of frames in which vehicle was tracked. These values are updated in each frame and are transferred between matched vehicles. If i^{th} vehicle in previous frame and j^{th} vehicle in current frame are matched, updating is performed as:

$$ms(j) = \frac{ms(i) * c(i) + s(j)}{c(i) + 1}$$
$$c(j) = c(i) + 1 \qquad \textbf{(4.2)}$$

where, ms, c and s indicate mean speed, frame counter and current speed found according to the position of i^{th} vehicle in previous frame and j^{th} vehicle in current frame, respectively. According to the approach above, an average speed is assigned to all vehicles in a frame.

4.2 Results

The approach explained above was implemented in four video sequences to find mean and individual vehicle speeds. The main challenge in this calculation is to transform pixel based distances to world plane distances. In other words, equivalent

world space length of a pixel must be determined. Generally, calibration is used to produce such a transformation. However, calibration needs some parameters such as camera angle and camera distance, which are not available in this thesis. In order to find a transformation, size of the lines that separates lanes and car sizes were used. According to highway rules [55], length of these lines is 3 meters in urban traffic roads, such as the utilized sequences. Additionally, some recognized vehicles were used to obtain the length of a pixel. Some visual tracking results in four video sequences are illustrated in Figure 4.3 and Figure 4.4.

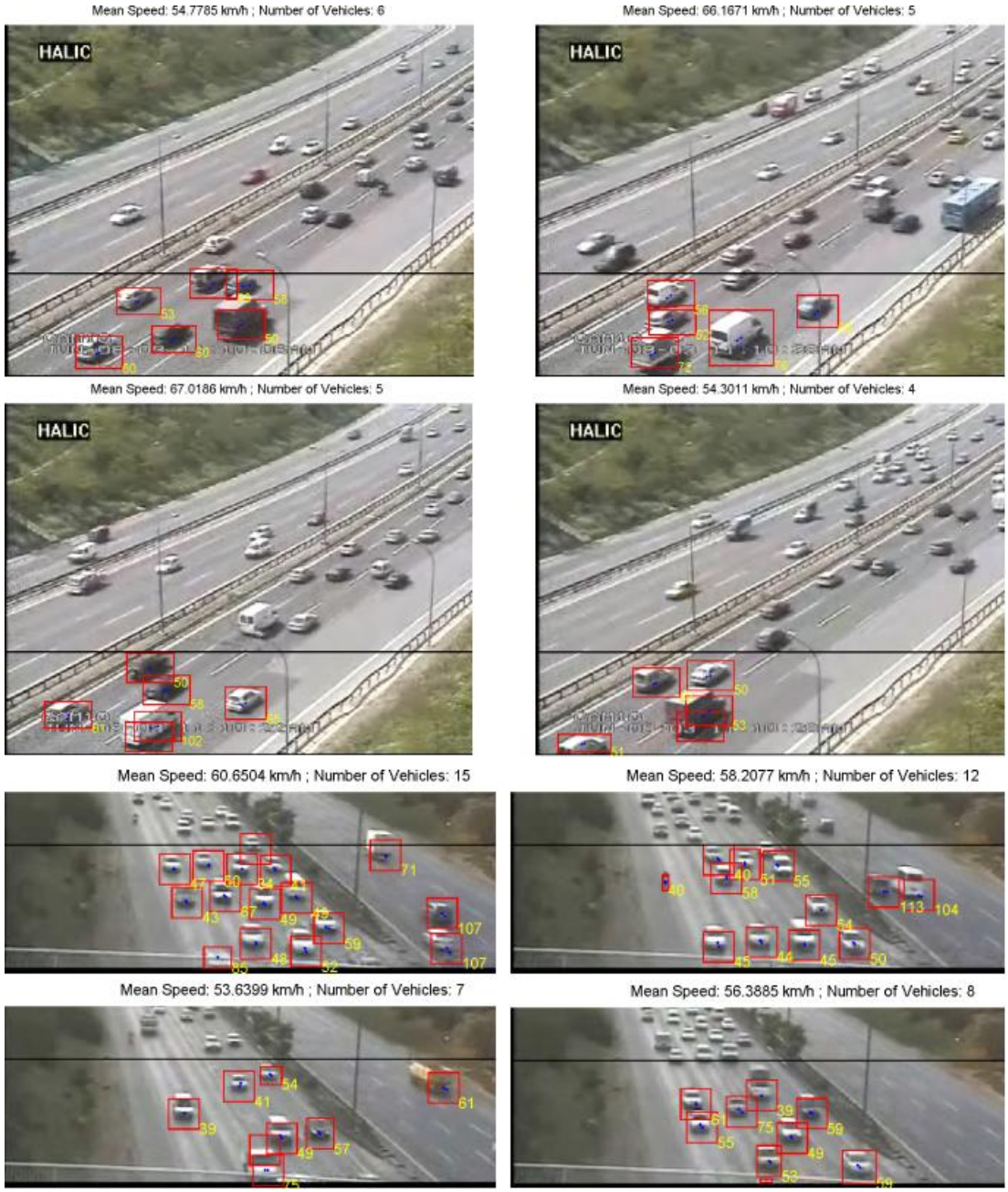

Figure 4.3 : Tracking results in Halic and Elmali sequences

Figure 4.4 : Tracking results in Mecidiyekoy sequences (top and bottom)

In these results, horizontal lines represent the selected ROI in corresponding video sequences. All vehicles are tracked only if their center is in the ROI. As indicated in Figure 4.3 and Figure 4.4, individual speed of each vehicle is presented in the right bottom corner of the vehicle. Additionally, mean vehicle speed and number of vehicles in ROI are given in the top of all images. In Mecidiyekoy video sequences, there exists some vehicles, which affect the number of vehicles and the mean speed, outside the road. Therefore, these vehicles can be eliminated with selecting an appropriate ROI in order to improve accuracy. Finally, a blue line over each vehicle shows location change of a vehicle in consecutive frames.

As stated in these figures, vehicles were tracked accurately, even though there is a high crowded traffic condition. On the other hand, tracking also contributes to eliminate some false positive vehicle detections. In this case, it is assumed that an obtained region is false positive if it is not matched with another region in previous frame. As a result, these regions are not counted as a vehicle for improving accuracy of the system. Such situations are illustrated in second and fourth rows of each figure (Figure 4.3 and Figure 4.4).

5. CONCLUSION

In this thesis, an accurate traffic surveillance system was developed. The aim of the system is to segment all vehicles, in order to find number of total vehicles, obtain the density in a traffic scene and find mean and individual vehicle speeds. To extract these traffic parameters, background subtraction, occlusion handling and tracking steps were implemented.

In first step, an improved background subtraction approach was presented, which is based on a simple and efficient method. The first contribution was done on update situations of the background model by using foreground masks to decide where the update will be applied. Afterwards, an edge adaptive thresholding approach was proposed to determine shape of vehicles more accurately. Additionally, 3-d connected component analysis was performed to eliminate irregularities in foreground regions. Furthermore, an efficient occlusion detection algorithm was implemented to separate individual vehicles and occluded blobs in video sequences. Finally, the accuracy of the system was proved with numerical results, according to the success in correctly obtaining the single vehicles and occlusion detections.

After finding irregular blobs, next step was to locate individual vehicles in these blobs. For this purpose, a robust occlusion handing approach was implemented. This system was a classification-based method. In this method, some contributions were done, especially in feature extraction phase. As a result, vehicles were segmented accurately and this accuracy was presented numerically. The other advantage of this method was being fully automatic, because training examples were obtained from the video sequence automatically. On the other hand, with this approach robust training sets, which are compatible with the conditions of corresponding video sequence, were determined by obtaining the sets adaptively for each sequence. Additionally, an automatic ROI detection algorithm was presented to make the system fully automated. With the help of remotely controllable camera technologies, camera angles can be changed by control centers easily. However, this change brings out the necessity of determining a new ROI in sequence. Therefore, automatic ROI detection

solves this problem by simplifying the work of traffic surveillance system controllers, while also reducing the possible user errors. In consequence of occlusion handling step, all vehicles were segmented in order to find number of vehicles and estimating traffic density.

Finally, a simple but efficient tracking approach was proposed to find individual vehicle speeds and mean vehicle speed in a video scene.

To sum up, an accurate traffic visual surveillance system was developed by improving the performance of individual steps of the work. Additionally, the system was designed as an automatic and adaptive system by an automatic ROI detection approach and fully automated and adaptive occlusion-handling algorithm.

5.1 Future Work

In background subtraction and moving object detection, some tests were done on different video sequences, which have various levels of traffic densities and accuracy of the system was obtained in these sequences. However, these sequences are similar by the means of lighting and weather conditions. Additionally, performance of the system can be tested in different conditions and system can be improved in order to acquire robust performance in different situations.

Occlusion handling approach used in this system tries to segment occluded blobs into individual vehicles with a sliding window approach. Size of the sliding window is automatically obtained from the median value of training examples. As a result, an appropriate window is found for vehicles. However, this window can be relatively small for big vehicles such as buses and trucks. As a result, some false positive detection can be obtained in such vehicles, by separating the big vehicles into interior parts. Although it is very difficult in implementation, different occlusion handling systems (different classifiers) can be developed for different classes of vehicles.

As stated before, tracking was implemented in a simple manner. In spite of the efficiency of the proposed tracking approach, some post processing can be done in order to obtain better results. Kalman filters or particle filters can be appropriate solutions for this purpose.

Finally, some specific analysis can be added to the system such as accident detection, detecting stopped vehicles or vehicles not staying in lane.

REFERENCES

[1] **Hu, W., Tan, T., Wang, L., and Maybank, S.,** 2004, A Survey on visual surveillance of object motion and behaviors, *Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, Vol. 34, no. 3, pp. 334-352.

[2] **Valera, M., and Velastin S. A.,** 2005, Intelligent distributed surveillance systems: a review, *IEE Proceedings: Vision, Image and Signal Processing*, Vol. 152, no. 2, pp. 192-204.

[3] **Collins, R. T., Lipton, A. J., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, N., Hasegawa, O., Burt, P., and Wixson, L.,** 2000, A System for Video Surveillance and Monitoring: VSAM Final Report, *Tech. Report CMU-RI-TR-00-12, Carnegie Mellon University*.

[4] **Haering, N., Venetianer, P. L., and Lipton, A.,** 2008, The evolution of video surveillance: an overview, *Machine Vision and Applications*, Vol. 19, no. 5-6, pp. 279-290.

[5] **Machy, C., Carincotte, C., and Desurmont, X.,** 2009, On the use of Video Content Analysis in ITS: a review from academic to commercial applications, *Int. Conf. on ITS Telecommunications*, October 2009.

[6] **Lipton, A. J., Fujiyoshi, H., and Patil, R. S.,** 1998, Moving target classification and tracking from real-time video, *in Proc. IEEE Workshop on Applications of Computer Vision*, pp. 8-14.

[7] **Cheung, S. C. S., and Kamath, C.,** 2004, Robust techniques for background subtraction in urban traffic video, *Proceedings of SPIE*, Vol. SPIE-5308, pp. 881-892.

[8] **Fuentes, L. M., and Velastin, S. A.,** 2003, From tracking to advanced surveillance, *Proceedings of IEEE International Conference on Image Processing (ICIP)*, Vol. 3, pp. III – 121-4.

[9] **Toyama, K., Krumm, J., Brumitt, B., and Meyers B.,** 1999, Wallflower: Principles and Practice of Background Maintenance, *Seventh International Conference on Computer Vision (ICCV)*, Vol. 1, pp. 255-261.

[10] **Matsuyama, T., Ohya, T., and Habe, H.,** 2000, Background Subtraction for Non-Stationary Scenes, *Proceedings of Asian Conference on Computer Vision*, pp. 622-667.

[11] **Oliver, N. M., Rosario, B., and Pentland, A.,** 2000, A Bayesian Computer Vision System for Modeling Human Interactions, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, no. 8, pp. 831-843.

[12] **Karmann, K. P., and Brandt, A.,** 1990, Moving object recognition using adaptive background memory, *Time-Varying Image Processing and Moving Object Recognition*, The Netherlands : Elsevier, Vol. 2.

[13] **Stauffer, C., and Grimson, W. E. L.,** 1999, Adaptive Background Mixture Models for Real-Time Tracking, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Vol. 2, pp. 246-252.

[14] **KaewTraKulPong, P., and Bowden, R.**, 2001, An improved adaptive background mixture model for real-time tracking with shadow detection, *Proceedings 2nd European Workshop on Advanced Video-Based Surveillance Systems*, September 2001.

[15] **Jun, G., Aggarwal, J. K., and Gokmen, M.,** 2008, Tracking and Segmentation of Highway Vehicles in Cluttered and Crowded Scenes, *WACV 2008 – IEEE Workshop on Applications of Computer Vision*, pp. 1-6.

[16] **Horprasert, T., Harwood, D., and Davis, L.,** 1999, A statistical approach for real-time robust background subtraction and shadow detection, *Proceedings of International Conference on Computer Vision (ICCV'99)*, 1999.

[17] **Kim, K., Chalidabhongse, T. H., Harwood, D., and Davis, L.,** 2004, Background modeling and subtraction by codebook construction, *International Conference on Image Processing (ICIP'04)*, Vol. 5, pp. 3061-3064.

[18] **Javed, O., Shafique, K., and Shah, M.,** 2002, A hierarchical approach to robust background subtraction using color and gradient information, *Workshop on Motion and Video Computing*, pp. 22-27.

[19] **Yao, J., and Odobez, J.-M.,** 2007, Multi-Layer Background Subtraction Based on Color and Texture, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07)*, pp. 1-8.

[20] **Vargas, M., Toral, S. L., Barrero, F., and Milla, J. M.,** 2008, An Enhanced Background Estimation Algorithm for Vehicle Detection in Urban Traffic Video, *IEEE 11th International Conference on Intelligent Transportation Systems (ITSC 2008)*, pp. 784-790.

[21] **Meyer, D., Denzler, J., and Niemann, H.,** 1997, Model based extraction of articulated objects in image sequences for gait analysis, *International Conference on Image Processing* , Vol. 3, pp. 78-81.

[22] **Koller, D., Weber, J., Huang T., Malik, J., Ogasawara, G., Rao, B., and Russell, S.,** 1994, Towards robust automatic traffic scene analysis in real-time, *Proceedings of the 12th International Conference on Pattern Recognition* , Vol. 1, pp. 126-131.

[23] **Koller, D., Daniilidis, K., and Nagel, H. H.,** 1993, Model-based object tracking in monocular image sequences of road traffic scenes, *International Journal of Computer Vision*, Vol. 10, no. 3, pp. 257-281.

[24] **Haag, M., and Nagel, H. H.,** 1999, Combination of edge element and optical flow estimates for 3D-model-based vehicle tracking in traffic image sequences, *International Journal of Computer Vision*, Vol. 35, no.3 pp. 295-319.

[25] **Tomasi, C., and Kanade, T.,** 1991, Detection and tracking of point features, *Technical Report CMU-CS-91-132 Carnegie Mellon University*.

[26] **Beymer, D., McLauchlan, P., Coifman, B., and Malik, J.,** 1997, A real-time computer vision system for measuring traffic parameters, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 495-501.

[27] **She, K., Bebis, G., Gu, H., and Miller, R.,** 2004, Vehicle tracking using on-line fusion of color and shape features, *IEEE International Conference on Intelligent Transportation Systems*, pp. 731-736.

[28] **Forsyth, D. A., and Ponce, J.,** 2003, Computer Vision: A Modern Approach, *Prentice Hall*, 2003.

[29] **Yang, C., Duraiswami, R., and Davis, L.,** 2005, Fast multiple object tracking via a hierarchical particle filter, *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 212-219.

[30] **Grammatikopoulos, L., Karras, G., and Petsa, E.,** 2005, Automatic estimation of vehicle speed from uncalibrated video sequences, *Proceedings of International Symposium on Modern Technologies, Education and professional Practice in Geodesy and Related Fields*, pp. 332-338.

[31] **Cucchiara, R., Grana, C., Piccardi, M., and Prati, A.,** 2003, Detecting moving objects, ghosts and shadows in video streams, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, pp. 1337-1342.

[32] **Bo, L., and Qi-mei, C.,** 2009, Framework for Freeway Auto-Surveillance from Traffic Video, *WRI World Congress on Computer Science and Information Engineering*, Vol. 6, pp. 360-365.

[33] **Kristensen, F., Nilsson, P., and Owall, V.,** 2006, Background segmentation beyond RGB, *Computer Vision - ACCV 2006*, pp. 602-612.

[34] **Mikic, I., Cosman, P. C., Kogut, G. T., and Trivedi, M. M.,** 2000, Moving shadow and object detection in traffic scenes, *15th International Conference on Pattern Recognition*, Vol. 1, pp. 321-324.

[35] **Liu, H., Li, J., Liu, Q., and Qian, Y.,** 2007, Shadow elimination in traffic video segmentation, *MVA 2007 IAPR Conference on Machine Vision Applications*, pp. 445-448.

[36] **Jung, Y. K., and Ho, Y. S.,** 1999, Traffic parameter extraction using vide-based vehicle tracking, *IEEE Proceedings International Conference on Intelligent Transportation Systems*, pp. 754-769.

[37] **Senior, A., Hampapur, A., Tian, Y., Brown, L., Pankanti, S., and Bolle, R.,** 2001, Appearance models for occlusion handling, *2nd International Workshop on Performance Evaluation of Tracking and Surveillance (PETS 2001)*, Kauai, Hawaii.

[38] **Pang, C. C. C., Lam, W. W. L., and Yung, N. H. C.,** 2004, A novel method for resolving vehicle occlusion in a monocular traffic-image sequence, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 5, pp. 129-141.

[39] **Tamersoy, B., and Aggarwal, J. K.,** 2009, Robust vehicle detection for tracking in highway surveillance videos using unsupervised learning, *Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, pp. 529-534.

[40] **Gupte, S., Masoud, O., Martin, R. F. K., and Papanikolopoulos, N. P.,** 2002, Detection and classification of vehicles, *IEEE Transactions on Intelligent Transportation Systems*, Vol. 3, pp. 37-47.

[41] **Ozkurt, C., and Camci, F.,** 2009, Automatic traffic density estimation and vehicle classification for traffic surveillance systems using neural networks, *Mathematical and Computational Applications*, Vol. 14, no. 3, pp. 187-196.

[42] **Porikli, F., and Li, X.,** 2004, Traffic congestion estimation using HMM models without vehicle tracking, *IEEE Intelligent Vehicles Symposium*, pp. 188-193.

[43] **Balcilar, M., and Sonmez, A. C.,** 2008, Trafik akış hızının hareket vektörleriyle belirlenmesi (Extracting traffic flow velocity with motion vector), *Signal Processing, Communication and Applications Conference, SIU 2008*, pp. 1-4.

[44] **Molinier, M., Hame, T., and Ahola, H.,** 2005, 3D-Connected Components Analysis for Traffic Monitoring in Image Sequences Acquired from a Helicopter, *Springer, Lecture Notes in Computer Science, Image Analysis (SCIA 2005)*, Vol. 3540/2005, pp. 141-150.

[45] **Lindeberg, T., and Garding, J.,** 1993, Shape from Texture from a Multi-Scale Perspective, *Proceedings of 4th International Conference of Computer Vision*, pp. 683-691.

[46] **Forstner, W., and Gulch, E.,** 1987, A Fast Operator for Detection and Precise Location of Distinct Points, Corners and Centres of Circular Features, *ISPRS Intercommission Workshop*, pp. 281-304.

[47] **Triantafyllidis, G. A., Varnuska, M., Sampson, D., Tzovaras, D., and Strintzis, M. G.,** 2003, An efficient algorithm for the enhancement of JPEG-coded images, *Elsevier Computers and Graphics*, Vol. 27, pp. 529-534.

[48] **Alpaydin, E.,** 2004, Introduction to Machine Learning, *MIT Press*, Cambridge, USA, pp. 218-235.

[49] **Yang, Z. R.,** 2004, Biological Applications of Support Vector Machines, *Briefings in Bioinformatics*, Vol. 5, pp. 328-338.

[50] **Joachims, T.,** 2008, SVM^{light} Support Vector Machine, Access date: 2009-12-05, http://www.cs.cornell.edu/People/tj/svm_light/.

[51] **Stewart, B. D., Reading, I., Thomson, M. S., Binnie, T. D., Dickinson, K. W., and Wan, C. L.,** 1994, Adaptive Lane Finding in Road Traffic Image Analysis, *Seventh International Conference on Road Traffic Monitoring and Control*, pp. 133-136.

[52] **Trucco, E., and Verri, A.,** 1998, Introductory Techniques for 3-D Computer Vision, *Prentice Hall*, New Jersey, USA, pp. 97-99.

[53] **Dalal, N., and Triggs, B.,** 2005, Histograms of Oriented Gradients for Human Detection, *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 2005)*, Vol. 1, pp. 886-893.

[54] **Petrovic, V. S., and Cootes, T. F.,** 2004, Analysis of Features for Rigid Structure Vehicle Type Recognition, *British Machine Vision Conference*, Vol. 2, pp. 587-596.

[55] **Karayolları Genel Müdürlüğü,** Trafik İşaretleri El Kitabı I, Access Date: 2009-12-14, http://www.kgm.gov.tr/trafikkitap/bolum1.pdf.

CURRICULUM VITA

Candidate's full name:	**Mehmet KAPLAN**
Place and date of birth:	**Gazipaşa / ANTALYA, April 23, 1985**
Permanent Address:	**Sanayi Mah. Batanay Sk. Hüseyin Bey Apt. No:31/17 Kağıthane / Istanbul**
Universities and Colleges attended:	**Istanbul Technical University [Undergraduate]**