

A VIDEO TRACKER SYSTEM FOR TRAFFIC MONITORING AND ANALYSIS

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

BY

MEHMET OCAKLI

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

AUGUST 2007

Approval of the thesis:

**A VIDEO TRACKER SYSTEM FOR TRAFFIC MONITORING AND ANALYSIS**

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

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

Prof. Dr. İsmet Erkmn \_\_\_\_\_  
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. Mübeccel Demirekler \_\_\_\_\_  
Supervisor, **Electrical and Electronics Engineering Dept., METU**

Examining Committee Members

Assoc. Prof. Dr. Aydın Alatan \_\_\_\_\_  
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Mübeccel Demirekler \_\_\_\_\_  
Electrical and Electronics Engineering Dept., METU

Assist. Prof. Dr. Afşar Saranlı \_\_\_\_\_  
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Aytül Erçil \_\_\_\_\_  
Electrical and Electronics Engineering Dept., Sabancı University

Dr. H. Burak Kaygısız \_\_\_\_\_  
Chief of Guidance and Control Division, TÜBİTAK-SAGE

**Date:** \_\_\_\_\_

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Mehmet OCAKLI

Signature :

# **ABSTRACT**

## **A VIDEO TRACKER SYSTEM FOR TRAFFIC MONITORING AND ANALYSIS**

OCAKLI, Mehmet

M.S., Department of Electrical and Electronics Engineering

Supervisor: Prof. Dr. Mübeccel Demirekler

AUGUST 2007, 126 Pages

In this study, a video tracker system for traffic monitoring and analysis is developed. This system is able to detect and track vehicles as they move through the camera's field of view. This provides to perform traffic analysis about the scene, which can be used to optimize traffic flows and identify potential accidents. The scene inspected in this study is assumed stationary to achieve high performance solution to the problem. This assumption provides to detect moving objects more accurately, as well as ability of collecting a-priori information about the scene.

A new algorithm is proposed to solve the multi-vehicle tracking problem that can deal with problems such as occlusion, short period object lost or inaccurate object detection. Two different tracking methods are used together in the developed tracking system, namely, the multi-model Kalman tracker and the Markov scene partition tracker. By the combination of these vehicle trackers with the developed occlusion reasoning approach, the continuity of the track is achieved for situations such as target loss and occlusion. The developed system is a system that collects a-priori information about the junction and then used it for scene modeling in order to increase the performance of the tracking system.

The proposed system is implemented on real-world image sequences. The simulation results demonstrates that, the proposed multi-vehicle tracking system is capable of tracking a target in a complex environment and able to overcome occlusion and inaccurate detection problems as well as abrupt changes in its trajectory.

Keywords: Video tracking, traffic monitoring and analysis, multi-model Kalman tracker, Markov scene partition tracker, scene modeling.

# ÖZ

## TRAFİK ANALİZİ VE GÖZLENMESİ AMAÇLI VİDEO İZLEYİCİ SİSTEMİ

OCAKLI, Mehmet

Yüksek Lisans., Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Mübeccel Demirekler

AĞUSTUS 2007, 126 Sayfa

Bu çalışmada, trafik analizi ve gözlenmesi amaçlı bir video izleyici sistemi geliştirmiştir. Bu sistem, taşıtlar kameranın görüş alanında hareket ettikleri sürece taşıtları tespit eder ve izler. Bu sayede incelenen sahne hakkında trafik analizleri gerçekleştirilebilir. Bu analizler trafik akışlarını eniyilemekte ve potansiyel kazaları tespit etmekte kullanılabilir. Tanımlanan problemin yüksek başarılı çözümü için bu çalışmada incelenen sahnenin sabit olduğu varsayılmıştır. Bu varsayım, hareketli nesnelerin sahnede daha doğru tespit edilmesini sağladığı gibi sahne hakkında ön-bilgi toplanabilmesini de sağlar.

Bu çalışmada, çoklu taşıt takibi probleminin çözümü için, örtüşme, kısa süreli hedef kayıpları veya hatalı nesne tespitleri gibi sorunlarla başa çıkabilecek bir algoritma önerilmiştir. Geliştirilen taşıt takip sistemi iki farklı takip algoritması birlikte kullanılmaktadır: Çok modelli Kalman takibi ve Markov sahne bölmesi takibi. Bu taşıt izleme algoritmalarının geliştirilen örtüşme akıl yürütücüsüyle birleşimi ile hedef kaybı ve örtüşme gibi durumlarda takibin devamlılığı sağlanmaktadır. Geliştirilen sistem kavşak hakkında ön-bilgi toplayan ve daha sonra, bu ön-bilgileri takip sisteminin başarımını arttırmak için sahne modellemesinde kullanan bir sistemdir.

Önerilen sistem gerçek görüntü dizileri üzerinde uygulanmıştır. Benzetim sonuçları, önerilen çoklu taşıt takibi sisteminin karmaşık ortam koşullarında taşıt takibi sürdürebildiğini ve örtüşme, hatalı taşıt tespiti ve ani yörünge değişikliği gibi sorunlarla başa çıkabildiğini kanıtlamıştır.

Anahtar Kelimeler: Video izleme, trafik gözlenme ve analiz, çok modelli Kalman takibi, Markov sahne bölmesi takibi, sahne modelleme.

*To my family,  
To my wife*

## ACKNOWLEDGMENTS

I am most thankful to my supervisor Prof. Dr. Mübeccel Demirekler for sharing his invaluable ideas and experiences on the subject of my thesis. Thanks to his advices and helpful criticisms, this thesis is completed.

I would like to extend my thanks to all lecturers at the Department of Electrical and Electronics Engineering, who greatly helped me to store the basic knowledge onto which I have built my thesis.

I am very grateful to TÜBİTAK-SAGE for providing tools and other facilities throughout the production of my thesis.

I would like to forward my appreciation to all my friends and colleagues who contributed to my thesis with their continuous encouragement.

I would like to express my deep gratitude to my family, who has always provided me with constant support and help. My dear mother, I have always felt your endless love and best wishes with me. My dear father, your kind guidance, cooperation, and constructive criticism have been always enlightened me in establishing my life and career.

Special thanks to my wife for all her help and showing great patience during the writing process of my thesis. I have always felt her endless support with me.

# TABLE OF CONTENTS

<b>ABSTRACT</b> .....	<b>IV</b>
<b>ÖZ</b> .....	<b>VI</b>
<b>ACKNOWLEDGMENTS</b> .....	<b>IX</b>
<b>TABLE OF CONTENTS</b> .....	<b>X</b>
<b>LIST OF TABLES</b> .....	<b>XIII</b>
<b>LIST OF FIGURES</b> .....	<b>XIV</b>
<b>LIST OF ABBREVIATIONS</b> .....	<b>XVI</b>
<b>CHAPTERS</b>	
<b>1 INTRODUCTION</b> .....	<b>1</b>
<b>1.1 Introduction to Video Tracking</b> .....	<b>1</b>
<b>1.2 Traffic Surveillance and Vehicle Tracking</b> .....	<b>3</b>
<b>1.3 Problem Definition and Motivation</b> .....	<b>5</b>
1.3.1 Motivation.....	5
1.3.2 Problem Definition.....	6
1.3.3 Proposed System.....	6
1.3.4 Contributions.....	8
<b>1.4 Outline of the Thesis</b> .....	<b>10</b>
<b>2 IMAGE PROCESSING</b> .....	<b>11</b>
<b>2.1 Thresholding</b> .....	<b>12</b>
<b>2.2 Morphological Operators</b> .....	<b>14</b>
2.2.1 Structuring Elements.....	14

2.2.2	Binary Dilation and Erosion.....	15
2.2.3	Binary Opening and Closing .....	16
<b>2.3</b>	<b>Edge Detection .....</b>	<b>17</b>
<b>2.4</b>	<b>Edge Labeling.....</b>	<b>20</b>
<b>3</b>	<b>OBJECT DETECTION .....</b>	<b>22</b>
<b>3.1</b>	<b>Background Subtraction .....</b>	<b>23</b>
<b>3.2</b>	<b>Frame Differentiation .....</b>	<b>27</b>
3.2.1	Object Detection Using Frame Differentiation.....	27
3.2.2	Object Detection Using Edge Map.....	30
<b>3.3</b>	<b>Proposed Vehicle Detection System.....</b>	<b>34</b>
3.3.1	Shadow Removal.....	35
<b>4</b>	<b>TRAINING: MODELING OF THE SCENE AND THE MOTION.....</b>	<b>38</b>
<b>4.1</b>	<b>Parameter Extraction .....</b>	<b>39</b>
4.1.1	Vehicle Parameters.....	39
4.1.2	Traffic Parameters .....	41
<b>4.2</b>	<b>A-priori Information Collection .....</b>	<b>42</b>
4.2.1	Scene Partitioning.....	43
4.2.2	Partition Modeling: Statistics of Partitions.....	45
4.2.3	Motion Region Detection.....	46
4.2.4	Path Detection .....	48
4.2.5	Motion Model Extraction Based on Partition Statistics.....	53
<b>5</b>	<b>VEHICLE TRACKING.....</b>	<b>59</b>
<b>5.1</b>	<b>Target Analysis .....</b>	<b>60</b>
5.1.1	Target Shape and Representation.....	60
5.1.2	Target Coordinate Systems .....	63
<b>5.2</b>	<b>Motion Models .....</b>	<b>65</b>
5.2.1	Basic Motion Model.....	66
5.2.2	Advanced Motion Models.....	67
5.2.3	Scene Specific Motion Models.....	71
5.2.4	Path Specific Motion Models.....	72
5.2.5	Modification of Noise Covariance Matrices.....	73

<b>5.3</b>	<b>Estimation</b> .....	<b>75</b>
5.3.1	Kalman Filtering .....	76
5.3.2	Basic Kalman Tracker .....	79
5.3.3	Multi Model Kalman Tracker .....	79
5.3.3.1	Motion Model Selection .....	80
5.3.4	Markov Scene Partition Tracker.....	81
5.3.4.1	Path Model Estimation .....	82
5.3.4.2	State Estimation.....	83
5.3.5	Hidden Vehicle Tracker.....	84
<b>5.4</b>	<b>Association Problem</b> .....	<b>86</b>
5.4.1	Gating and Validation.....	86
5.4.1.1	Rectangular Gating .....	86
5.4.1.2	Ellipsoidal Gating .....	87
5.4.2	Assignment Problem .....	88
5.4.2.1	Global Nearest Neighbor (NN) Algorithm.....	90
5.4.2.2	Auction Algorithm.....	91
5.4.2.3	Modified Auction Algorithm .....	94
5.4.3	Occlusion Problem .....	96
5.4.4	Assignment Post Processing .....	102
5.4.4.1	Object Loss Handling.....	103
5.4.4.2	New Object Handling .....	104
<b>6</b>	<b>IMPLEMENTATION AND SIMULATION</b> .....	<b>108</b>
6.1	Implementation Issues .....	108
6.2	Simulation Results.....	109
<b>7</b>	<b>CONCLUSIONS</b> .....	<b>119</b>
7.1	Contributions.....	121
7.2	Future Work .....	122
	<b>REFERENCES</b> .....	<b>124</b>

## LIST OF TABLES

Table 2-1: Canny Edge Detection Algorithm .....	19
Table 4-1: Image Parameters of a Vehicle .....	40
Table 4-2: Motion Parameters of a Vehicle .....	41
Table 4-3: Traffic Parameters .....	42
Table 4-4: Vehicle Features Used in the Partition Modeling .....	45
Table 4-5: Partition Feature Statistics Used in Partition Modeling.....	46
Table 4-6: Velocity Depended State Transition Probabilities .....	58
Table 5-1: Prediction Equations .....	77
Table 5-2: Measurement Update Equations .....	77
Table 5-3: Measurement Estimates .....	84
Table 5-4: Example of a General Assignment Matrix .....	92
Table 5-5: Assignment Matrix for the Previous Example.....	92
Table 5-6: Modified Assignment Matrix for Auction Algorithm .....	95
Table 5-7: Disappearance Conditions .....	103
Table 5-8: Appearance Conditions .....	104
Table 6-1: Results of the Vehicle Count .....	110
Table 6-2: The Development of the Proposed Method.....	111
Table 6-3: Occlusion Detection and Reasoning .....	112
Table 6-4: Explicit Occlusion Reasoning .....	113

## LIST OF FIGURES

Figure 1-1: Block Diagram of the Proposed Video Tracker System .....	7
Figure 2-1: RGB Image and Grayscale (Intensity) Image.....	12
Figure 2-2: Thresholding Results .....	13
Figure 2-3: Some Useful Structuring Elements [16] .....	14
Figure 2-4: Morphologic Operators – Dilation Process Example from [16].....	15
Figure 2-5: Morphologic Operators – Erosion Process Example from [16] .....	16
Figure 2-6: Morphologic Operators – Closing Process Example.....	16
Figure 2-7: Morphologic Operators – Opening Process Example .....	17
Figure 2-8: Canny Edge Detection Algorithm .....	20
Figure 2-9: Connected Components .....	20
Figure 2-10: Edge Labeling Example .....	21
Figure 3-1: Object Detection Using Background Subtraction .....	25
Figure 3-2: Object Detection Using Background Subtraction .....	26
Figure 3-3: Object Detection Using Frame Differentiation .....	28
Figure 3-4: Object Detection Using Frame Differentiation .....	29
Figure 3-5: Edge Map Extraction Method.....	31
Figure 3-6: An Example of the Proposed Edge Map Extraction Method .....	32
Figure 3-7: Object Detection Using Edge Map and Frame Differentiation.....	33
Figure 3-8: Shadow Removal .....	36
Figure 3-9: Vehicle Detection System .....	37
Figure 4-1: Vehicle Parameter Extraction Examples.....	39
Figure 4-2: Scene Partitioning Example .....	43
Figure 4-3: Partition Traffic Based Scene Partitioning.....	44
Figure 4-4: Motion Region Detector .....	46
Figure 4-5: Motion Detected Regions of the Scene.....	47
Figure 4-6: Motion Region Map of the Scene .....	48
Figure 4-7: Vehicle Motion Trajectories .....	49
Figure 4-8: Determination of the Entrance Gates of the Scene.....	50
Figure 4-9: Determination of the Exit Gates of the Scene .....	51

Figure 4-10: Entrance and Exit Regions (Gates) of the Scene.....	52
Figure 4-11: Path Candidates of the Scene .....	52
Figure 4-12: Paths Defined in the Scene.....	53
Figure 4-13 : Markov State Representation of the Scene Partitions.....	54
Figure 4-14 : State Transition Example .....	55
Figure 5-1: Block Diagram of the Proposed Vehicle Tracking System.....	60
Figure 5-2: Some Vehicles Concerned in this Work.....	61
Figure 5-3: Shape and Appearance Model Representations .....	62
Figure 5-4: Some Vehicle Representations used in this Thesis .....	63
Figure 5-5: Image Pixel Coordinate System .....	64
Figure 5-6: Body Fixed Coordinate Systems.....	65
Figure 5-7: Derivation Steps of the Motion Models .....	66
Figure 5-8: Orientation Angle ( $\theta$ ) and Object Area (A) .....	68
Figure 5-9: Path Models Extracted from the Scene.....	73
Figure 5-10: Transform of Noise Covariance Matrices .....	74
Figure 5-11: One Cycle of Kalman Filter Algorithm .....	78
Figure 5-12: Path Model Estimation Example .....	83
Figure 5-13: Example of an Association Conflict Situation .....	91
Figure 5-14: An Example of the Occlusion Problem.....	96
Figure 5-15: Explicit Occlusion Reasoning .....	99
Figure 5-16: Explicit Occlusion Example .....	100
Figure 5-17: Implicit Occlusion Reasoning .....	101
Figure 5-18: Implicit Occlusion Example .....	102
Figure 5-19: Disappearance Reasoning Algorithm Flowchart .....	106
Figure 5-20: Appearance Reasoning Algorithm Flowchart .....	107
Figure 6-1: Scene Inspected in this Study .....	109
Figure 6-2: Key Frames 1 – Occlusion Detection and Reasoning.....	115
Figure 6-3: Key Frames 2 – Occlusion Detection and Reasoning.....	116
Figure 6-4: Key Frames 3 – Occlusion Detection and Reasoning.....	117
Figure 7-1: Block Diagram of the Proposed Video Tracker System .....	120

## LIST OF ABBREVIATIONS

<b>AA</b>	Auction Algorithm
<b>GNN</b>	Global Nearest Neighbor
<b>GPB1</b>	First Order Generalized Pseudo Bayesian
<b>GPB2</b>	Second Order Generalized Pseudo Bayesian
<b>HVT</b>	Hidden Vehicle Tracker
<b>IMM</b>	Interacting Multiple Model
<b>ITS</b>	Intelligent Transportation Systems
<b>ITS</b>	Intelligent Traffic System
<b>KF</b>	Kalman Filter
<b>KT</b>	Kalman Tracker
<b>MHT</b>	Multi Hypothesis Tracking
<b>MM</b>	Motion Model
<b>MMKT</b>	Multi-Model Kalman Tracker
<b>MSPT</b>	Markov Scene Partition Tracker
<b>NN</b>	Nearest Neighbor
<b>OR</b>	Occlusion Reasoning
<b>RGB</b>	Red-Green-Blue
<b>STM</b>	State Transition Matrix

# CHAPTER 1

## INTRODUCTION

### 1.1 Introduction to Video Tracking

Object tracking can be defined as the problem of following image elements from frame to frame in a video sequence and estimating the trajectory of the object in the image plane as object moves around a scene. Object tracking is an important task within the field of computer vision and includes methods from signal processing, estimation and control disciplines [1]. Video tracking is an essential building block for vision systems addressing:

- military tasks like motion based target recognition, vehicle navigation and guidance,
- security and surveillance tasks like scene monitoring, fire control, traffic surveillance,
- industry tasks like, automation, robotics, human-computer interaction.

Tracking objects can be complex due to loss of information caused by projection of the 3D world on a 2D image, noise in images, complex object motion, complex or non-rigid object shape, partial and full object occlusions, scene illumination changes and real-time processing requirements [2].

*Comaniciu et al.* [3] define two major components distinguished in a typical visual tracker. The first component, *target representation and localization*, is mostly a bottom-up process, which has also to cope with the changes in appearance of the target. The second component, *filtering and data association* is mostly a top-down process dealing with the dynamics of the tracked object, learning of scene priors, and evaluation of different hypotheses.

*Trucco et al.* [4] defines video tracking problem as a problem of following moving targets automatically over a video sequence. The paper introduces video tracking problem in computer vision, including design requirements and a review of recent techniques. The survey claims that tracking systems must address two basic problems: *motion* and *matching*.

*Motion problem* is defined as predicting the location of an image element being tracked in the next frame, that is, a limited search region is identified in which the element is expected to be found with high probability. *Matching problem* is defined as identifying the image element in the next frame within the designated search region.

Numerous approaches for object tracking have been proposed in the literature [5, 6, 7, 8, 9, 10]. These approaches primarily differ from each other on the object representation selected for tracking and the image features that are used. *Yilmaz et al.* [2] categorize and describe the existing tracking methods and explain their strengths and weaknesses. They categorize the tracking methods by considering the object representations and introduce three main tracking categories: point, kernel and silhouette tracking. In *Point tracking*, objects detected in consecutive frames are represented by points, and the association of the points is based on the previous object state, which can include object position and motion. In *Kernel tracking*, kernel refers to the object shape and appearance. For example, the kernel can be a rectangular template or an elliptical shape with an associated histogram. Objects are tracked by computing the motion of the kernel in consecutive frames. In *Silhouette tracking*, tracking is performed by estimating the object region in each frame. Silhouette tracking methods use the information encoded inside the object region. This information can be in the form of appearance density and shape models, which are usually in the form of edge maps. Given the object models, silhouettes are tracked by either shape matching or contour evolution.

*Cavallaro et al.* [5] classify object tracking methods into five groups: *model-based*, *appearance-based*, *contour-based*, *mesh-based*, *feature-based*, and *hybrid methods*. They present an algorithm for tracking video objects, which is based on

a hybrid strategy. This strategy uses both object and region information to solve the correspondence problem. Low-level descriptors are exploited to track object's regions and to cope with track management issues. Appearance and disappearance of objects, splitting and partial occlusions are resolved through interactions between regions and objects.

## 1.2 Traffic Surveillance and Vehicle Tracking

Traffic surveillance becomes an important topic in the *intelligent traffic systems* (ITS). Intelligent traffic systems not only provide basic traffic parameters including the vehicle count and average speed, but also provide traffic flows such as fast or slow traveling, stopped vehicles, lane changes of vehicles and vehicles traveling on wrong direction. This traffic scene information can be used to optimize traffic flow during busy periods, identify stalled vehicles and accidents. Traffic scene analysis using vision-based surveillance systems has been previously investigated by several research groups [6, 7, 8, 9, 10, 11]. These works are summarized below.

*Jung and Ho* [6] use the recursive Kalman filtering algorithm that consists of three steps of operations: initialization, state prediction, and measurement update. The state variable is defined as a four-dimensional vector, which represents the positional change of the target object per unit time interval and the size change of the target object. An original occlusion reasoning approach is used for both implicit and explicit occlusion situations. They also estimate some basic traffic parameters such as vehicle count and the average speed.

*Gil et al.* [7] track vehicles in real-world highway scene by using Kalman filters on two state vectors, which represent each target's position and velocity. A vehicle's motion is represented by an affine model, taking into account translations and scale changes. Three types of features have been used for the vehicle's description state vectors. Two of them are contour-based: the bounding box and the centroid of the convex polygon approximating the vehicles contour. The third one is region-based and consists of the 2-D pattern of the vehicle in the image.

A feature based tracking approach is generated by *Beymer et al.* [8] which is used for traffic monitoring under congestions. Instead of tracking entire vehicles, vehicle sub-features are tracked to make the system robust to partial occlusion. The sub-features detected from same vehicle are grouped using common motion approach. Corner features are chosen as sub-features, since they can be reliably tracked. In this work, the road plane is determined by user. Finally, traffic parameters such as flow rate and average speed are computed from the vehicle tracks.

*Koller et al.* [9, 10, 11] have described a few different early approaches on vehicle tracking. In [9], parameterized 3D polyhedral vehicle models are matched to coherently moving image features. This algorithm uses an offline camera calibration step to aid in recovery of the 3D pose. An iterated extended Kalman Filter is used to update estimates of the model location and pose.

In [10] and [11], the authors use background subtraction method to segment moving blobs from background image and the background updated using an adaptive background model. The moving blobs are tracked using a Kalman filter based motion model. An explicit occlusion-reasoning step is used to maintain tracking after separation of overlapped vehicles. They also describe high-level events in the scene, such as lane changes and stalled vehicles. Computing this type of information from the individual vehicle tracks is a useful surveillance tool.

As a result, a series of problems has to be addressed in a vision based traffic surveillance and tracking system [12]:

- difficult illumination conditions,
- great variability of the possible scenes,
- great variability of the vehicle classes,
- occlusions of vehicles in heavy traffic conditions
- real-time processing requirements,
- cost aspects.

On the other hand, the envisioned application allows taking advantage of a couple of assumptions [12]:

- only objects, which are moving or have been moved in the past have to be registered,
- the movement of the vehicles can be modeled using simple physical laws,
- the shape of the vehicles can be modeled using simple geometric models,
- only moving part of the image (the traffic region) is of interested.

### **1.3 Problem Definition and Motivation**

#### **1.3.1 Motivation**

Analyzing the traffic sequences in order to get high-level semantic scene interpretation based on segmentation and tracking of moving vehicles becomes an important topic in the intelligent traffic systems (ITS). Traffic scene analysis can be used to optimize traffic flows, identify stalled vehicles, determine potential accidents and aid the decision making of an automated traffic management. These reasons make the multi-vehicle tracking problem an interesting problem.

One of the main problems to be addressed in the multi-vehicle tracking is occlusion problem. It is usually difficult to maintain a tracking before, during and after an occlusion of vehicles. Several approaches are proposed in order to solve occlusion problems. Most of these approaches are not sufficient to maintain track continuity for long-time occlusions at a cluttered background. Therefore, the most promising research approaches are focused on occlusion problems and detection system errors to maintain tracking during such cases.

The motion model determination is a very critical issue in vehicle tracking scenarios. To develop a robust and reliable tracking system the vehicle motions must be analyzed carefully and most suitable motion model must be defined for each vehicle maneuver. The motion model extraction becomes a difficult task especially for traffic sequences having several different trajectories with different maneuvers. This is due to the fact that many traditional multi-vehicle tracking

systems are employed only for highway traffic scenes where motion model can be determined relatively more easily. Therefore, there is an urgent need for such a tracking system that can deal with complicated vehicle maneuvers and different motion trajectories especially in traffic junctions with high traffic load.

### **1.3.2 Problem Definition**

The objective of this thesis is to develop a robust and reliable multi-vehicle tracking system, which is capable of tracking a target in a complex environment and able to overcome occlusion and inaccurate detection problems as well as abrupt changes in its trajectory.

The scene, which is inspected in this study, is an uncontrolled junction with high traffic density where the above-defined problems occur frequently. The scene is assumed stationary to achieve high performance solution to the problem. This provides to detect moving objects more accurately in the scene as well as ability of collecting a-priori information about the scene at both detection and tracking parts.

The core idea is to have stationary video acquisition camera placed on a pole or other tall structures like building, mound or bridge looking down to the scene. The video acquisition cameras are positioned and focused to have optimal view of the road and the passing vehicles.

### **1.3.3 Proposed System**

A new algorithm is proposed to solve the multi-vehicle tracking problem that can deal with problems such as occlusion, short period object lost or inaccurate object detection. The block diagram of the proposed system is given in Figure 1-1. The proposed system consists of three main parts.

- vehicle detection
- training: modeling of the scene and the motion
- vehicle tracking

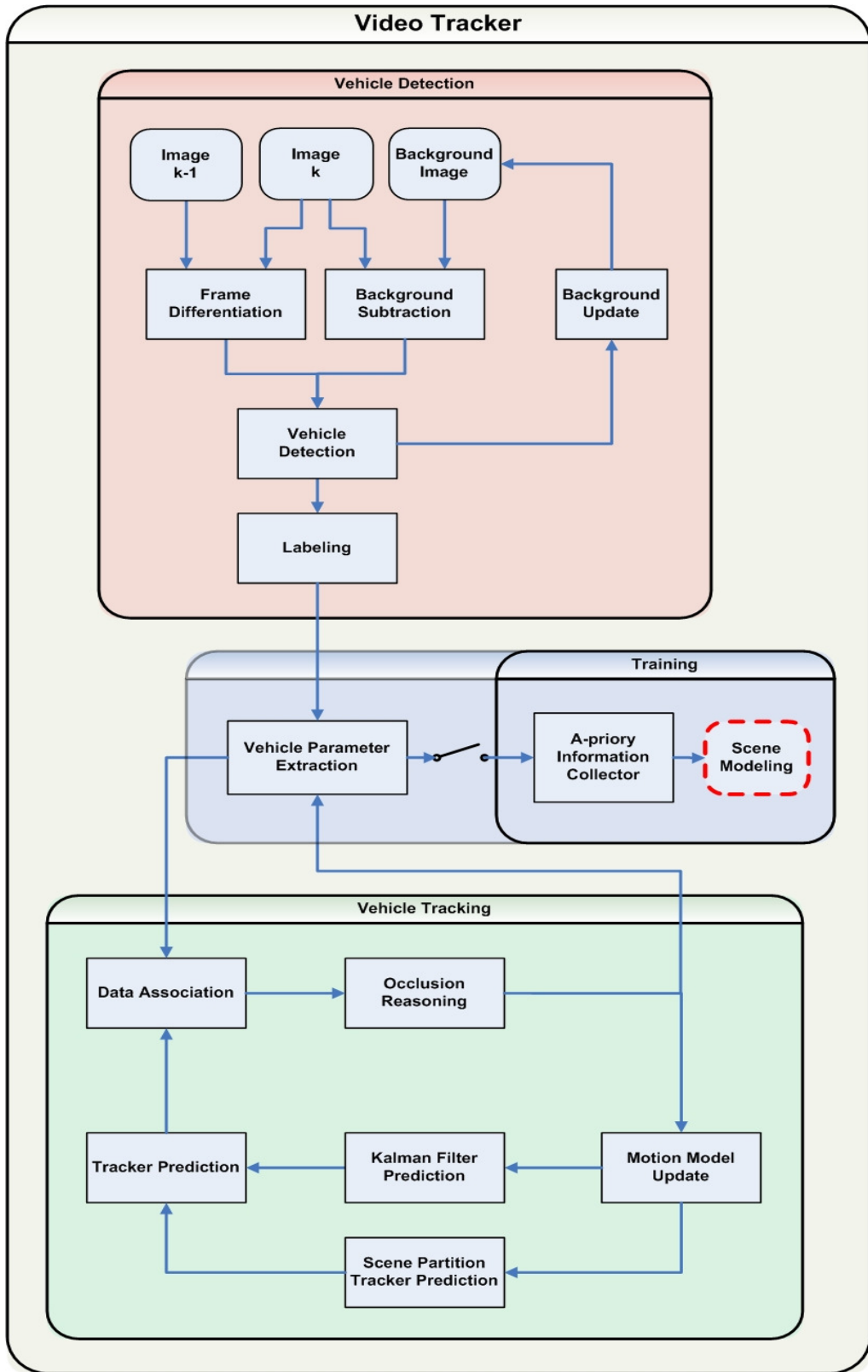


Figure 1-1: Block Diagram of the Proposed Video Tracker System

Training is done offline using the training data. The inputs of the training sub module are the output of the detection part as well as the tracks obtained by using a basic immature tracker. The output of this module consists of motion region of the scene, path models and the partition modeling of the scene. All these concepts are named as 'a priori information' and will be explained in detail later.

In this thesis, both *frame differentiation* and *background subtraction* approaches are used for vehicle detection problem. The result of background subtraction method is modified, whenever necessary by the output of the frame differentiation method to eliminate some disadvantages of it.

Tracking is done by means of a multi-model Kalman tracker, which is scene dependent. Such a tracker when performed alone is not very satisfactory especially during occlusions. The performance of the tracking process is increased by using a Markov chain based tracker together with the Kalman tracker. By combining these two vehicle trackers with the developed occlusion reasoning approach, the continuity of the track is well improved for situations such as target loss and occlusion.

#### **1.3.4 Contributions**

The major contributions of this thesis can be summarized as follows:

##### ***Vehicle Detection***

- Implementation of a system for moving object segmentation in Matlab by combining the advantages of both frame differentiation and background subtraction methods. The performance of the frame differentiation method is increased using the edge information of the vehicles for accurate vehicle localization and shape.

### ***Scene Modeling***

- Definition of partition modeling, which gives the a-priori information about the scene.
- Definition of Markov model of the scene, which gives the transition probabilities between scene partitions. By extracting the scene partition probabilities, we can estimate the motion of a vehicle between partitions.

### ***Estimation***

- Development of motion models for each path and each vehicle that will be used in multi-model Kalman tracker. This improves the estimation performance of the tracks.
- Definition of a Markov scene partition tracker, which uses scene partition transition probabilities to estimate the motion of a vehicle and checks the validation of Kalman tracker especially for cases like occlusion or detection system error.
- Definition of a hidden vehicle tracker, which is used to estimate the pseudo measurements of the Kalman tracker. The hidden vehicle tracker is used when the measurements cannot be obtained from the vehicle because of occlusion or detection system errors.

### ***Association***

- Adaptation of the auction algorithm according to hidden tracking and occlusion reasoning algorithm needs. The adapted association algorithm can cope with occlusion problems.
- Adaptation of the occlusion reasoning algorithm according to hidden vehicle tracking systems to cope with difficult occlusion problems.
- Development of the occlusion detection algorithm, which is based on searching the assignment results of the tracking system. The main idea behind the occlusion detection is to find disappeared-appeared vehicles, which are not assigned in the association algorithm.

## 1.4 Outline of the Thesis

The thesis is organized as follows:

Chapter 2 is an overview of fundamental image processing algorithms used in this thesis. The aim of this chapter is to give introductory information about thresholding, morphological operations and edge detection algorithms that will be used in vehicle detection part.

Chapter 3 describes moving object detection methods used in this thesis. These moving object detection methods are explained and a vehicle detection algorithm is proposed in this chapter.

Chapter 4 explains the training part of the proposed system. Firstly, a comprehensive parameter extraction system is developed to extract the vehicle parameters and traffic parameters. Then, a-priori information collector is designed to perform analysis that is necessary to obtain the scene model.

Chapter 5 introduces the main issues, which influence the design of a multi target tracking system. Firstly, an overview about object tracking algorithms is given. The target analysis such as target representation and motion modeling are introduced. Afterwards, the vehicle tracking system, which is developed in this work, is presented.

The simulation results and performance analysis of the developed system are introduced in Chapter 6. The developed system is tested on real-world image sequences for different scenarios to analyze reliability and robustness.

The summary of the overall study and conclusions about this thesis are given in Chapter 7. Some possible future work is also covered in this chapter.

## CHAPTER 2

### IMAGE PROCESSING

This chapter focuses on simple image processing operations and introduces some basic techniques and notations that will be used in the thesis. The image processing techniques like thresholding, morphological operations, edge detection and labeling will be briefly described. These image processing techniques will be used in the moving object detection part of the thesis, which is explained in Chapter 3.

In this thesis, the images obtained from the camera are in the RGB image format. An RGB (Red-Green-Blue) image is stored in Matlab as a  $m \times n \times 3$  data array where  $m$  and  $n$  are the height and weight of the image in pixels. The camera used in this work provides  $180 \times 240 \times 3$  RGB images. Although the color information is very important for object segmentation, we convert the true color image (RGB) to the grayscale intensity image since processing the images in RGB format increases the processing load and requires additional storage. An example of gray-scale transformation from RGB color space is given in the Figure 2-1. Given an image in RGB format, the intensity component of each RGB pixel is obtained using the equation

$$I = \frac{(R + G + B)}{3} \quad (2.1)$$



**Figure 2-1: RGB Image and Grayscale (Intensity) Image**

## 2.1 Thresholding

Thresholding is used in image processing to separate foreground pixels from the background pixels. Thresholding is one of the simplest and most commonly used image segmentation method for images having sufficient contrast between the foreground and background. In this thesis, the intensity images are converted to binary images using thresholding techniques. The threshold operation is defined as

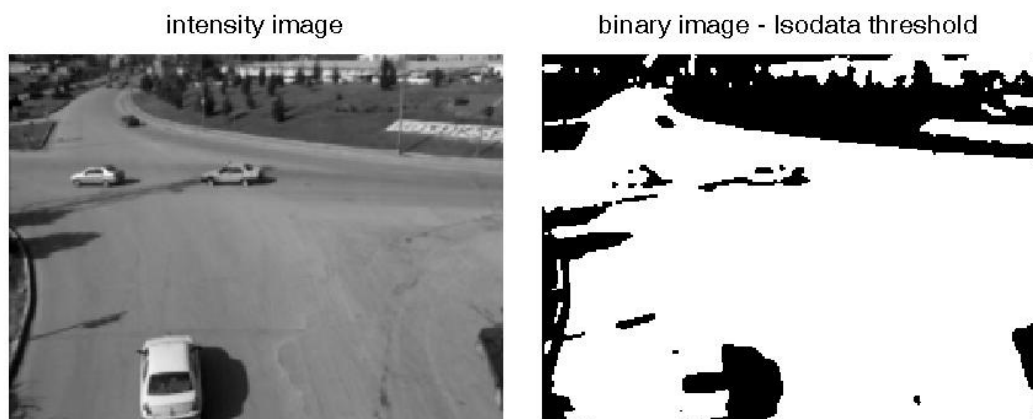
$$Bw(x, y) = \begin{cases} 1 & \text{if } I(x, y) > T \\ 0 & \text{if } I(x, y) \leq T \end{cases} \quad (2.2)$$

where  $I(x, y)$  is the original image, the  $Bw(x, y)$  is the binary image and  $T$  is the threshold value. The main problem in thresholding is to assign a threshold value according to intensity histogram of the gray-scale image. There are different methods for finding the threshold value  $T$  like *Isodata method*, *Triangle method* or *Otsu's method* [13]. In this work, we use *Isodata method* for thresholding. This iterative technique for choosing a threshold was developed by *Ridler and Calvard* [14].

The image intensity histogram is initially segmented into two parts using a starting threshold value ( $\theta_0$ ), which is the average intensity of the region. Then, the sample mean ( $m_{f,0}$ ) of the gray values associated with the foreground pixels and the sample mean ( $m_{b,0}$ ) of the gray values associated with the background pixels are computed. A new threshold value ( $\theta_1$ ) is now computed as the average of these two sample means. The process is repeated until the threshold value converges. The iterative thresholding can be formulated as

$$\theta_k = \frac{(m_{f,k-1} + m_{b,k-1})}{2} \quad \text{until } \theta_k \cong \theta_{k-1} \quad (2.3)$$

The results of Isodata thresholding for the road image are shown in Figure 2-2. The thresholding method is also used in shadow removing process, which will be explained in the next chapter.



**Figure 2-2: Thresholding Results**

## 2.2 Morphological Operators

Morphology is used as a tool for extracting image components that are useful in the representation and description of region shape. Morphological operators apply a structuring element to an input image, creating an output image of the same size. The most basic morphological operators are dilation and erosion.

A morphological operator maps the value of a pixel to a new value by comparing it with its neighbors. By choosing the size and shape of the neighborhood, one can construct a morphological operator that is sensitive to specific shapes in the input image.

### 2.2.1 Structuring Elements

A structuring element is a matrix consisting of only 0's and 1's that can have any arbitrary shape and size [15]. Some commonly used structuring elements are shown in Figure 2-3.

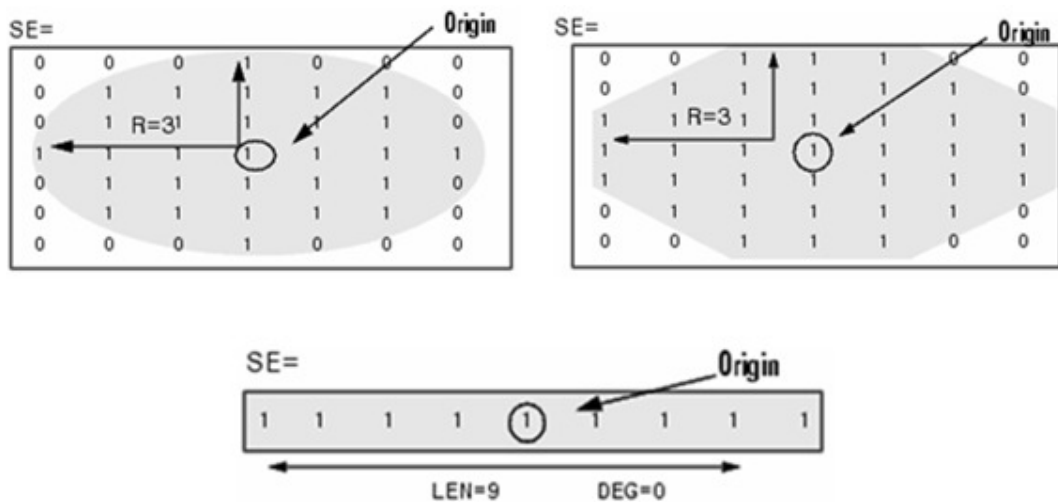


Figure 2-3: Some Useful Structuring Elements [16]

## 2.2.2 Binary Dilation and Erosion

Dilation is a morphological operator that enlarges the geometrical size of the objects within an image. The basic effect of the dilation operator on a binary image is to gradually enlarge the boundaries of regions of foreground pixels. The value of the output pixel is the maximum value of all the pixels in the input pixel's neighborhood. In a binary image, if any of the pixels is set to the value one, the output pixel is also set to one. Here, the neighborhood is determined by a structuring element. An example taken from [16] is shown in Figure 2-4.

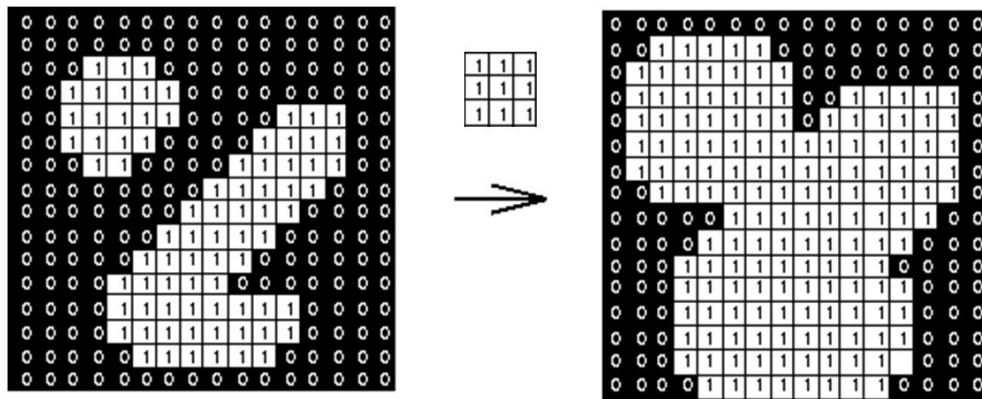


Figure 2-4: Morphologic Operators – Dilation Process Example from [16]

Erosion is a morphological operator that reduces the geometrical size of the objects within an image. The basic effect of the erosion operator on a binary image is to gradually erode the boundaries of regions of foreground pixels. The value of the output pixel is the minimum value of all the pixels in the input pixel's neighborhood. In a binary image, if any of the pixels is set to zero, the output pixel is set to zero. Here, the neighborhood is determined by a structuring element. Figure 2-5, which is taken from [16], is an example of erosion process.

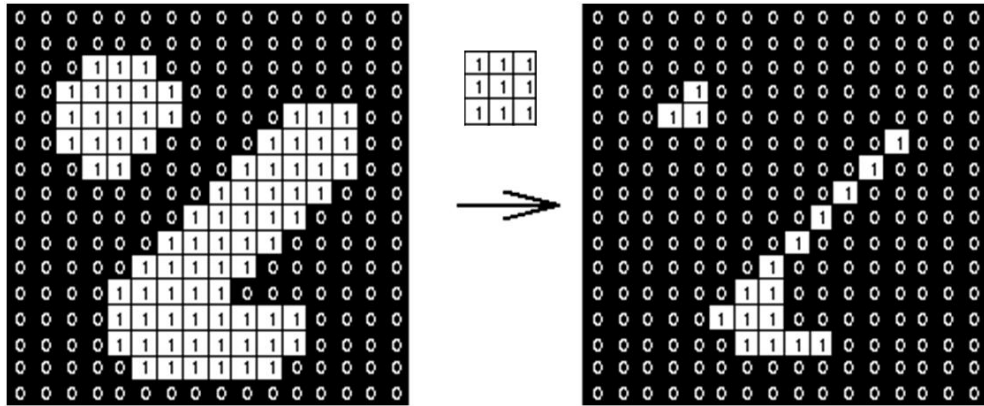


Figure 2-5: Morphologic Operators – Erosion Process Example from [16]

### 2.2.3 Binary Opening and Closing

Closing is a morphological operator that smooths the geometrical contour of the objects within an image. This operator is composed of a morphological dilation operation followed by a morphological erosion operation. The basic effect of the closing operator on a binary image is to fill or connect the boundaries of regions of foreground pixels. This operator firstly dilates an image and then erodes the dilated image using the same structuring element for both operations. An example of closing process is shown in Figure 2-6.

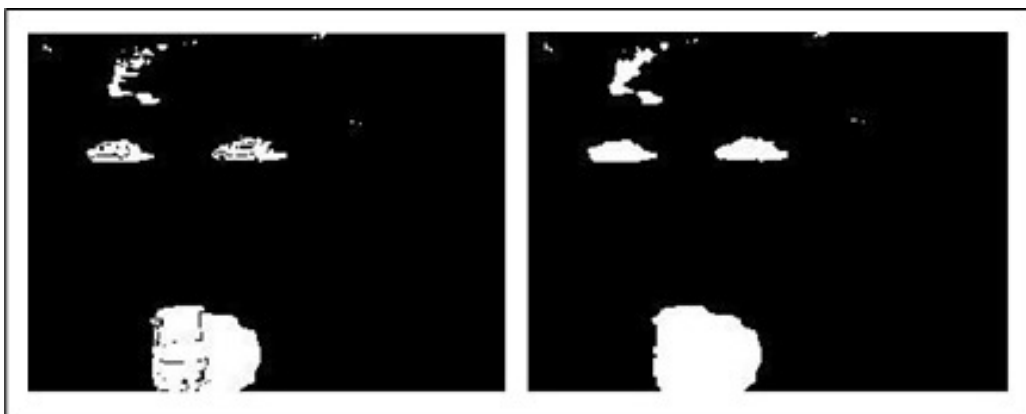


Figure 2-6: Morphologic Operators – Closing Process Example

Opening is a morphological operator that smoothes the geometrical shape of the objects within an image. This operator is composed of a morphological erosion operation followed by a morphological dilation operation. The basic effect of the closing operator on a binary image is to erode and smooth the boundaries of regions of foreground pixels. This operator firstly dilates an image and then erodes the dilated image using the same structuring element for both operations. An example of opening process is shown in Figure 2-7.



**Figure 2-7: Morphologic Operators – Opening Process Example**

### **2.3 Edge Detection**

Edge detection is a common image processing operation, which is used for detecting significant local intensity changes in the image. Edges are important features in an image since they provide clues to separate regions within an image. Edge detection is traditionally implemented by convolving the signal with a linear filter that usually approximates a first or second derivative operator.

Edge detection process has three main steps:

- Noise Smoothing: Suppress the image noise without destroying the edges.
- Edge Enhancement: Design a filter that gives large outputs at edge points and low elsewhere.
- Edge Localization: Decide which local maxima in the filter's output are edges and which are noise.

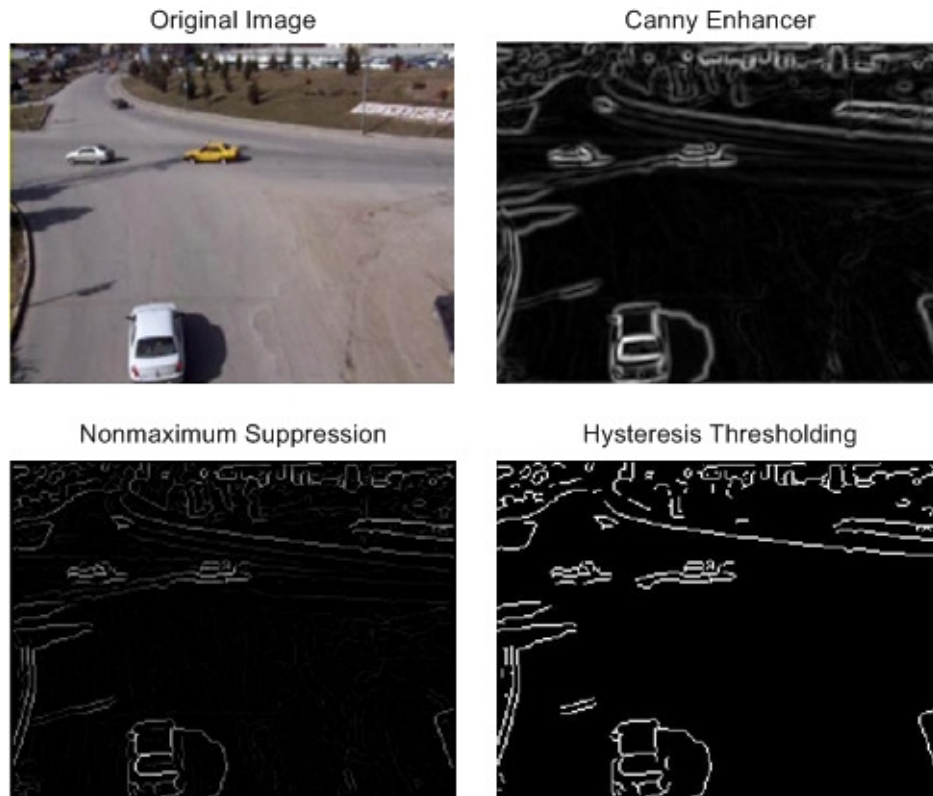
The most widely used edge detectors are Sobel, Prewitt, Canny and Marr-Hildreth [17, 18]. Canny Edge Detector [19] is used in this study due to its success in lots of applications. The objective function is designed to achieve the following optimization constraints [20]:

- Maximize the signal to noise ratio to give perfect detection. This favors the marking of true positives.
- Achieve perfect localization to accurately mark edges.
- Minimize the number of responses to a single edge. This favors the identification of true negatives, that is, non-edges are not marked.

The Canny edge detection algorithm is explained in Table 2-1. In the first step, a Gaussian filter is implied to eliminate the noise. The image gradient is found to highlight the regions with high spatial derivatives. Sobel filters are used to find the gradient of the image. Then, the algorithm tracks along these regions and suppresses any pixel that is not at the maximum. Finally, a hysteresis thresholding is used to track along the remaining pixels that have not been suppressed. In hysteresis thresholding, there are two threshold values. The pixels whose gradient values are larger than the first threshold are set to non-edge. On the other hand, if the gradient value of a pixel is smaller than the second threshold, it is detected as edge. The pixels whose gradient value is between two thresholds, it is set as an edge pixel if they are neighbors of an edge pixel. An example of Canny edge detection algorithm is given in Figure 2-8.

**Table 2-1: Canny Edge Detection Algorithm**

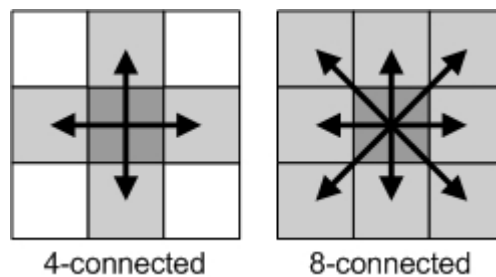
<b>CANNY EDGE DETECTION ALGORITHM</b>
<b>CANNY ENHANCER</b>
Apply Gaussian smoothing.
Compute the gradient components.
Find the edge magnitude and orientation.
<b>NONMAXIMUM SUPPRESSION</b>
Quantize edges for their orientations.
Suppress edge pixels smaller than neighbors along magnitude.
<b>HYSTERESIS THRESHOLDING</b>
Find heavy edge points, which have magnitude larger than high threshold.
Find weak edge points by searching the neighbors of the heavy edges that have larger magnitude than low threshold.



**Figure 2-8: Canny Edge Detection Algorithm**

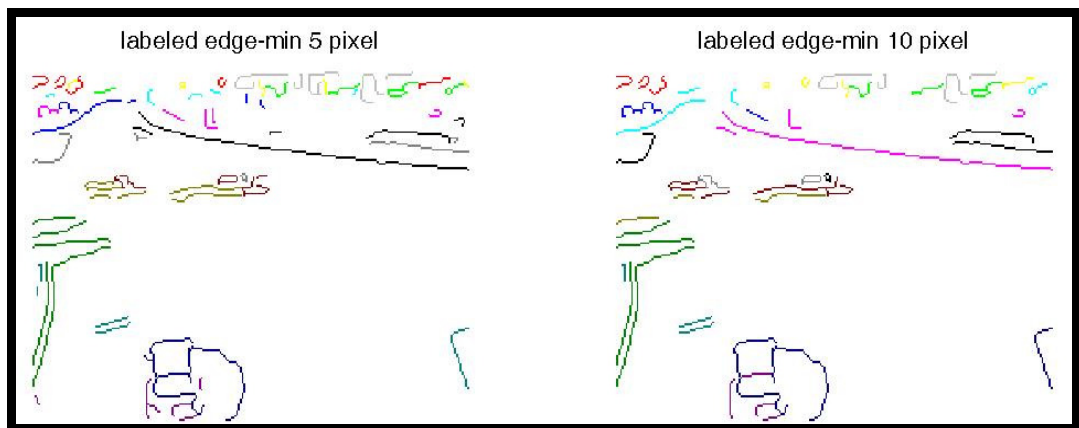
## 2.4 Edge Labeling

In this part, we label the connected components in a binary image. The two dimensional connectivity is determined by 4-connected neighborhood or 8-connected neighborhood, which are illustrated in Figure 2-9.



**Figure 2-9: Connected Components**

The connectedness of image pixels is determined by binary image searching using 4-connected neighborhood or 8-connected neighborhood [15]. Then, starting from one, a number is given to each unconnected edge pixel group. The labeling operation is also used for eliminating small edge groups in the image. If the size of a pixel group is smaller than a threshold value, then the pixels that generates that group are set to zero (decided as non-edge). An example of edge labeling operation is given below. The edge pixel groups, which have the same label, are shown with the same color. In the above example, the labeling process is done using minimum edge pixel sizes of five and ten pixels.



**Figure 2-10: Edge Labeling Example**

## CHAPTER 3

### OBJECT DETECTION

Tracking vehicles in a video sequence is composed of two main stages. The first stage is the segmentation of the vehicles from the background in each frame, and the second is association of the vehicles in successive frames in order to trace their trajectories. The vehicle segmentation is a major task in vehicle tracking applications. The main objective of the object detection is to divide the scene as background and foreground. Since we are dealing with tracking the moving objects for stationary camera, the differences between frames of a sequence gives information about moving objects in the scene.

There are many different methods for solving the moving object detection problems in the literature. These methods are examined in two classes: *frame differentiation* and *background subtraction*.

Frame differentiation consists of extracting the changes between consecutive image frames. The difference gives us information about the moving regions of the scene. However, it is difficult to extract the whole moving region from this difference mask. Edge detection methods can be used to find edges of the objects and this information when used together with moving object regions may determine the shapes of moving objects.

The second method is based on extracting a background model of the scene. If a background image of the scene is obtained, then the moving objects can be extracted from the difference between the current image and the background image. The results of the background subtraction method are usually superior to frame differentiation. The main issue in background subtraction method is the extraction and currency of the background image.

In this thesis, both frame differentiation and background subtraction approaches are examined for resolving the moving object detection problem. In the following section, firstly, the background subtraction technique is explained and main methods that are used in the literature for background model estimation are discussed. Then, the frame differentiation method is explained. The results of frame differentiation method are improved using some basic morphological operators. Afterwards, a moving object detector based on frame differentiation that contains an edge detection part is proposed to extract the vehicle location and shape more correctly. Finally, the moving object detector system developed in this thesis is explained. The developed system combines these two main methods to reach a vehicle detector that eliminates the disadvantages of both of the methods.

### 3.1 Background Subtraction

For stationary camera case, if the background image is known, then any change of it denotes moving objects in the image. This change can be calculated from the difference between the current and the background images. A significant change between the images can be registered by thresholding the difference image with a suitable threshold value. More precisely, the foreground and background regions of the image are separated using the formula given below.

$$Bw(x, y) = \begin{cases} 1 & \text{if } |I(x, y) - Bg(x, y)| > T \\ 0 & \text{if } |I(x, y) - Bg(x, y)| \leq T \end{cases} \quad (3.1)$$

In this formula, the term  $I(x, y)$  is the original image,  $Bg(x, y)$  is the background image and  $T$  is the threshold value. The term  $Bw(x, y)$  denotes the moving regions of the current image. The foreground pixels of  $Bw(x, y)$  are set to one and background pixels are set to zero.

The main issue in the background subtraction technique is to create a reference frame, often called the “*background model*” or “*background image*”. Several methods for performing background extraction have been proposed in the literature [10, 21, 22]. All of these methods try to estimate the background model effectively from a temporal sequence of the frames [21]. In this thesis, we use a

modified version of running average technique for background extraction because of its high speed and low memory requirements compared to more sophisticated approaches like mixture of Gaussians or kernel density estimation [21]. *Running average technique* for background extraction computes a long-term average of the image [22]. The computation of the background image is as follows.

$$Bg_k(x, y) = (1 - \alpha)Bg_{k-1}(x, y) + \alpha I(x, y) \quad (3.2)$$

where  $1/\alpha$  is the time constant of the forgetting process. For a complete description, we refer to [22].

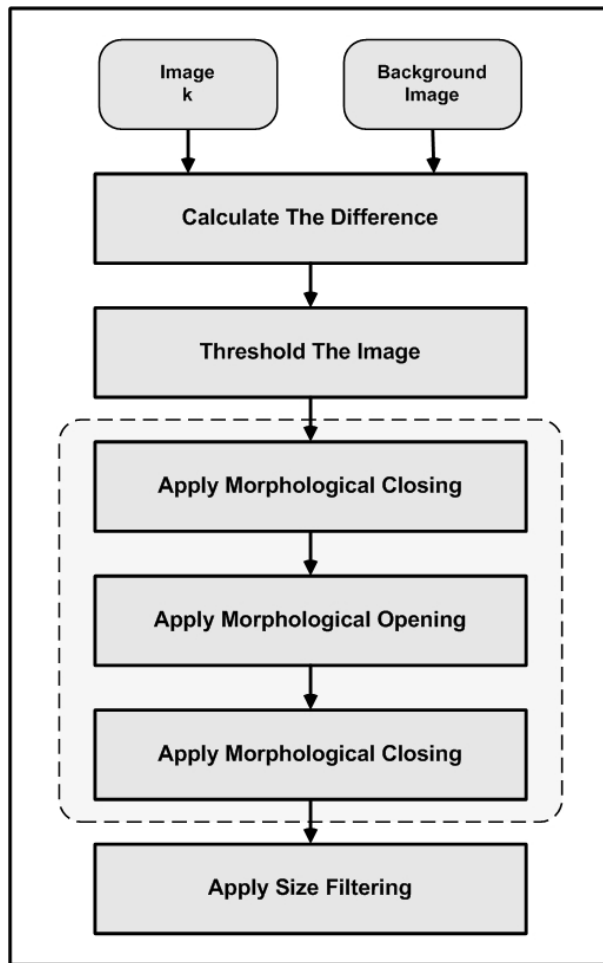
A modified version of the *running average technique* is suggested from *Koller et al.* [10, 11]. Their approach is known as *selective background update*. They modified the model update equation of the running average technique as:

$$Bg_k(x, y) = Bg_{k-1}(x, y) + (\alpha_1(1 - M_{k-1}(x, y)) + \alpha_2 M_{k-1}(x, y))D_{k-1}(x, y) \quad (3.3)$$

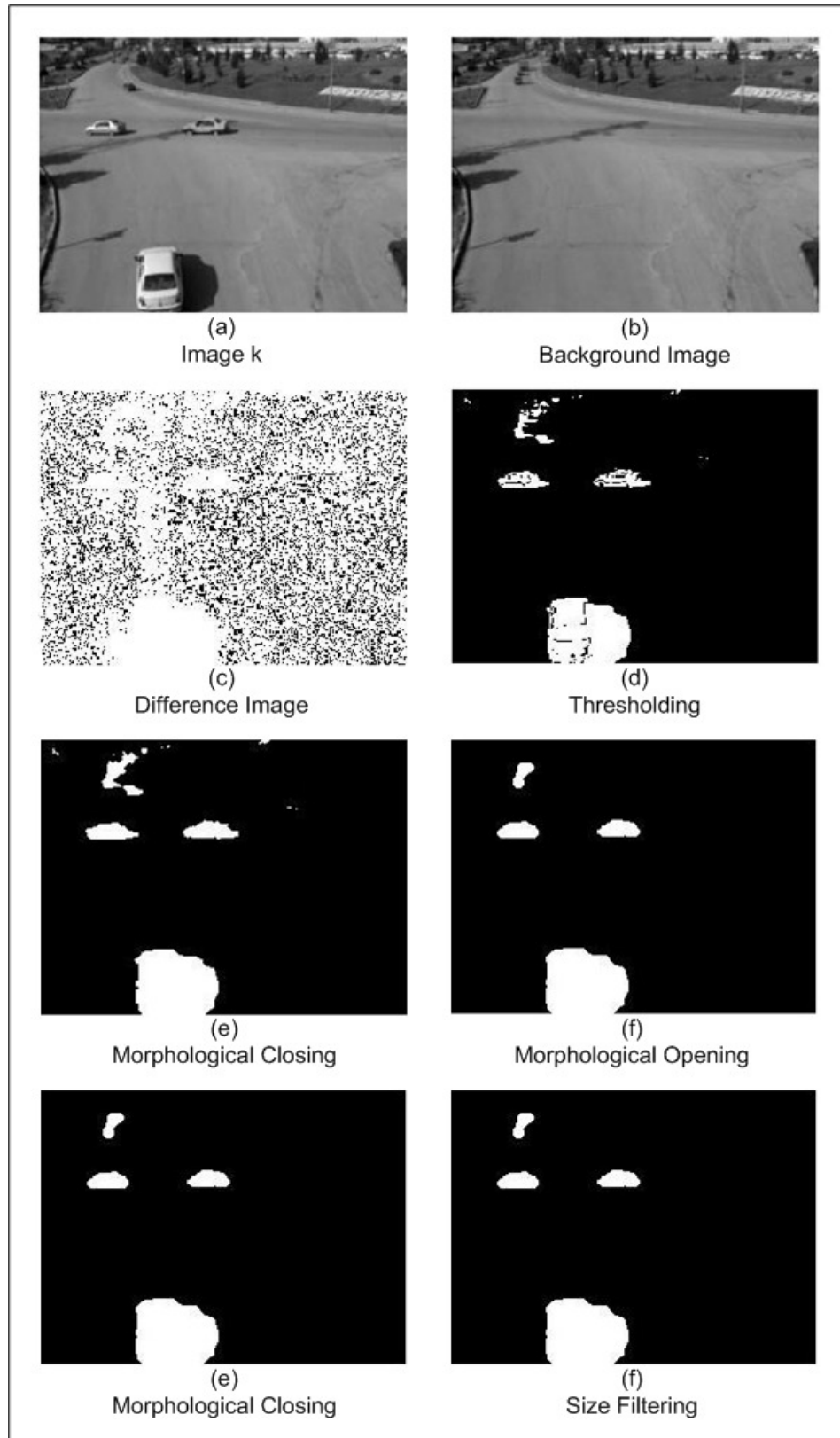
where  $D_{k-1}(x, y)$  represents the difference between the present frame and the background model and  $M_{k-1}(x, y)$  is the binary hypothesis mask of the moving objects. The gains  $\alpha_1$  and  $\alpha_2$  are based on the estimate of the rate of change of the background. *Koller et al.* [11] suggest selecting the gains as small constant values of  $\alpha_1 = 0.1$  and  $\alpha_2 = 0.01$ .

The hypothesis mask,  $M_{k-1}(x, y)$ , includes a selection process of the moving objects. A pixel of the hypothesis mask is set to one if it corresponds to a foreground pixel and zero otherwise. For detailed description of selective background update approach, we refer the reader to [11]. In this study, *selective background update* method is used for updating the background image.

The block diagram of the background subtraction method used in this thesis for moving object detection is given in Figure 3-1. As can be seen from the block diagram, morphological operator of closing and opening are applied to remove the noise and predict the vehicle location more accurately. Finally, a size filter is used to remove small regions to eliminate clutter. An example of the application of the method is given in Figure 3-2.



**Figure 3-1: Object Detection Using Background Subtraction**



**Figure 3-2: Object Detection Using Background Subtraction**

## 3.2 Frame Differentiation

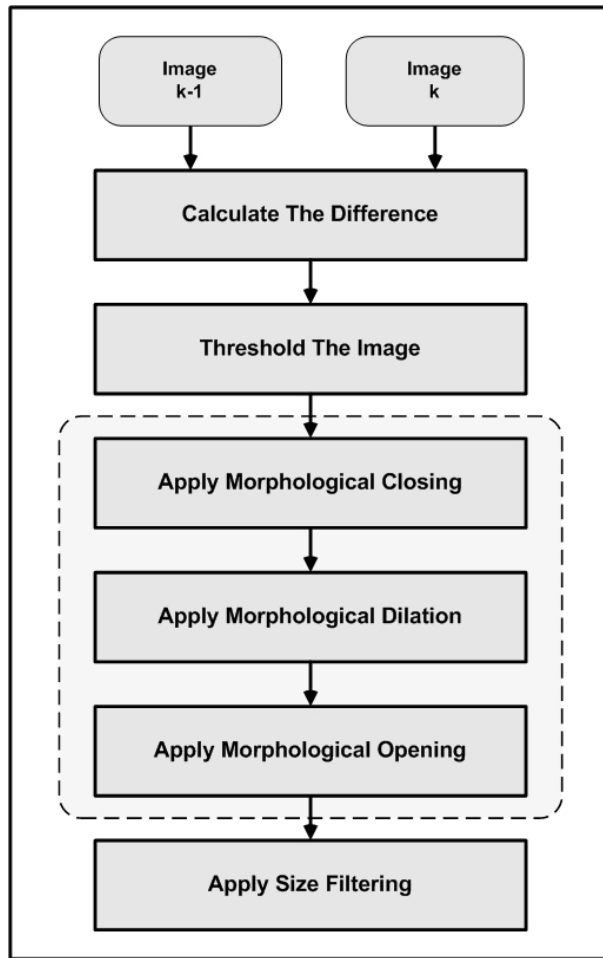
The frame differentiation method is one of the simplest methods in moving object segmentation processes. The main idea is to calculate the changes between consecutive frames by using simple frame differencing and thresholding. The moving object regions of the image can be extracted using the formula given below.

$$Bw(x, y) = \begin{cases} 1 & \text{if } |I_c(x, y) - I_p(x, y)| > T \\ 0 & \text{if } |I_c(x, y) - I_p(x, y)| \leq T \end{cases} \quad (3.4)$$

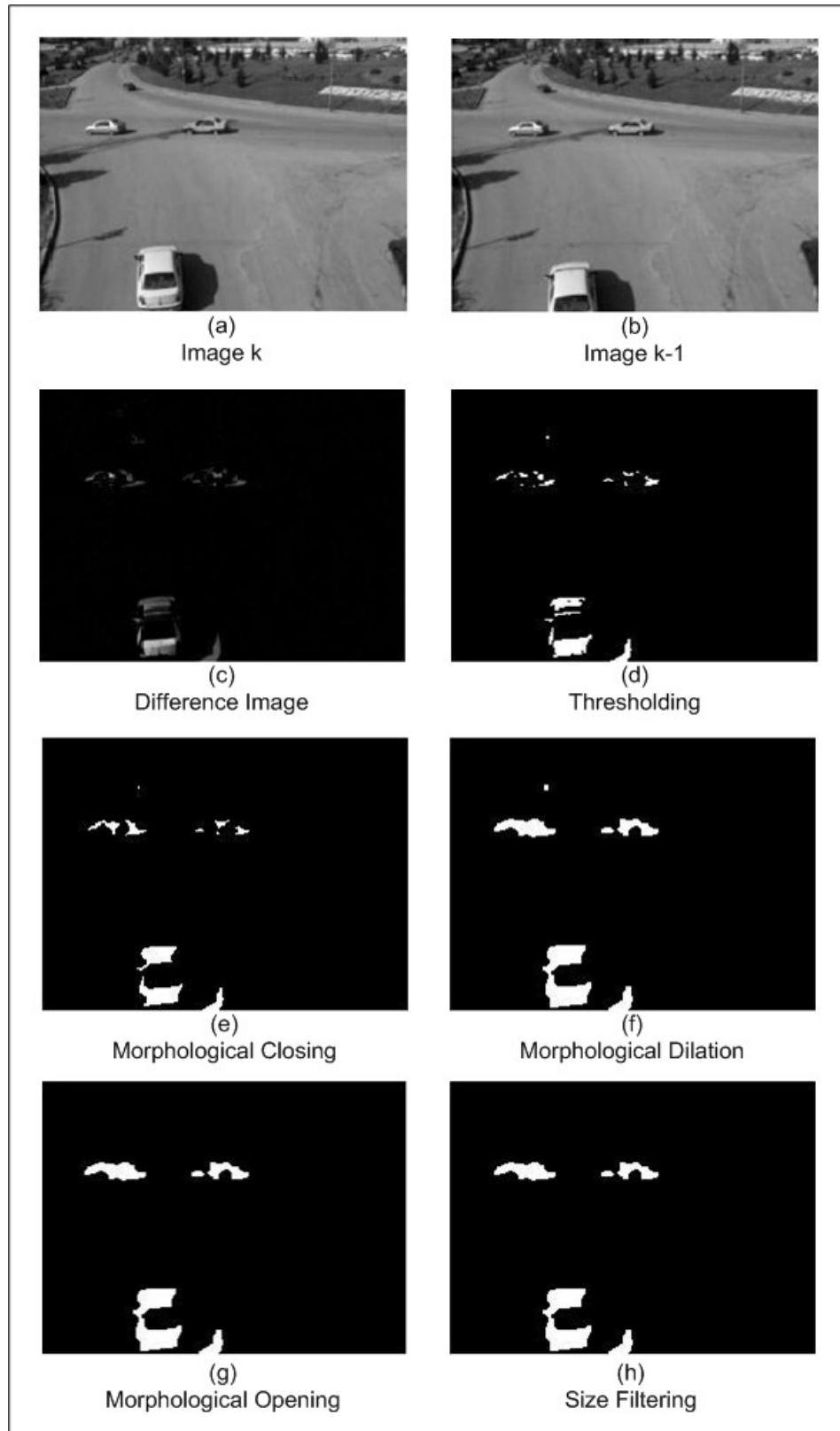
In this formula, the term  $I_c(x, y)$  is the current image,  $I_p(x, y)$  is the previous image,  $T$  is the threshold value and  $Bw(x, y)$  denotes the frame difference map. The main problem of this approach is the selection of the threshold value. Frame differentiation approach is simpler and faster than background subtraction method. Another advantage of the method is that the intensity changes do not degrade the detection systems performance since we are using the difference in one frame step.

### 3.2.1 Object Detection Using Frame Differentiation

Frame differentiation method gives the changes between consecutive frames, which represent moving objects in the scene. Due to disturbances and noises, some pixels that belong to moving objects may be classified as the ones that are in the background and vice versa. To eliminate these noise influences, binary morphological operators are usually applied on the thresholded difference image and also small sized clutters are eliminated using a size filter. The block diagram of the method used in this thesis is given in Figure 3-3. An example of the application of the method is given in Figure 3-4.



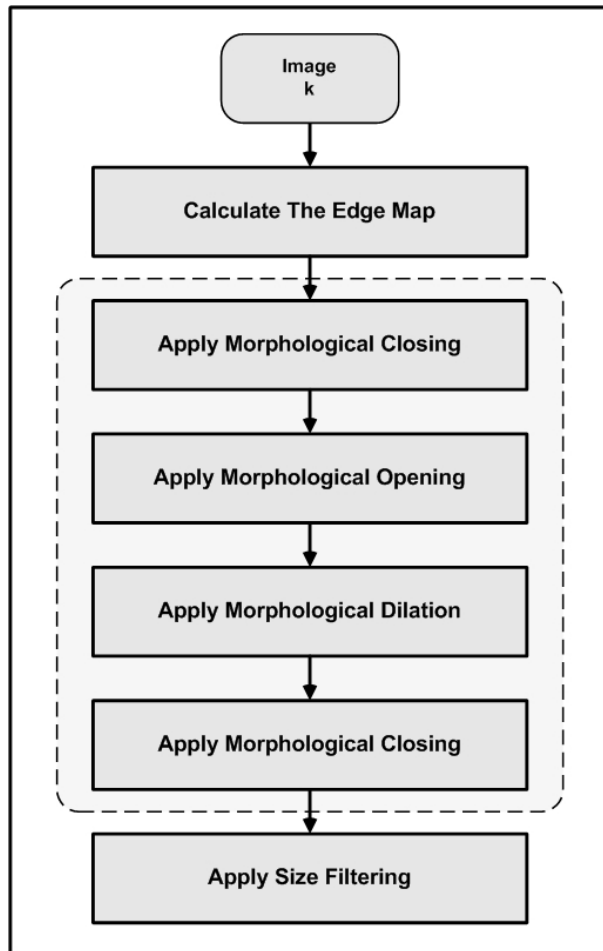
**Figure 3-3: Object Detection Using Frame Differentiation**



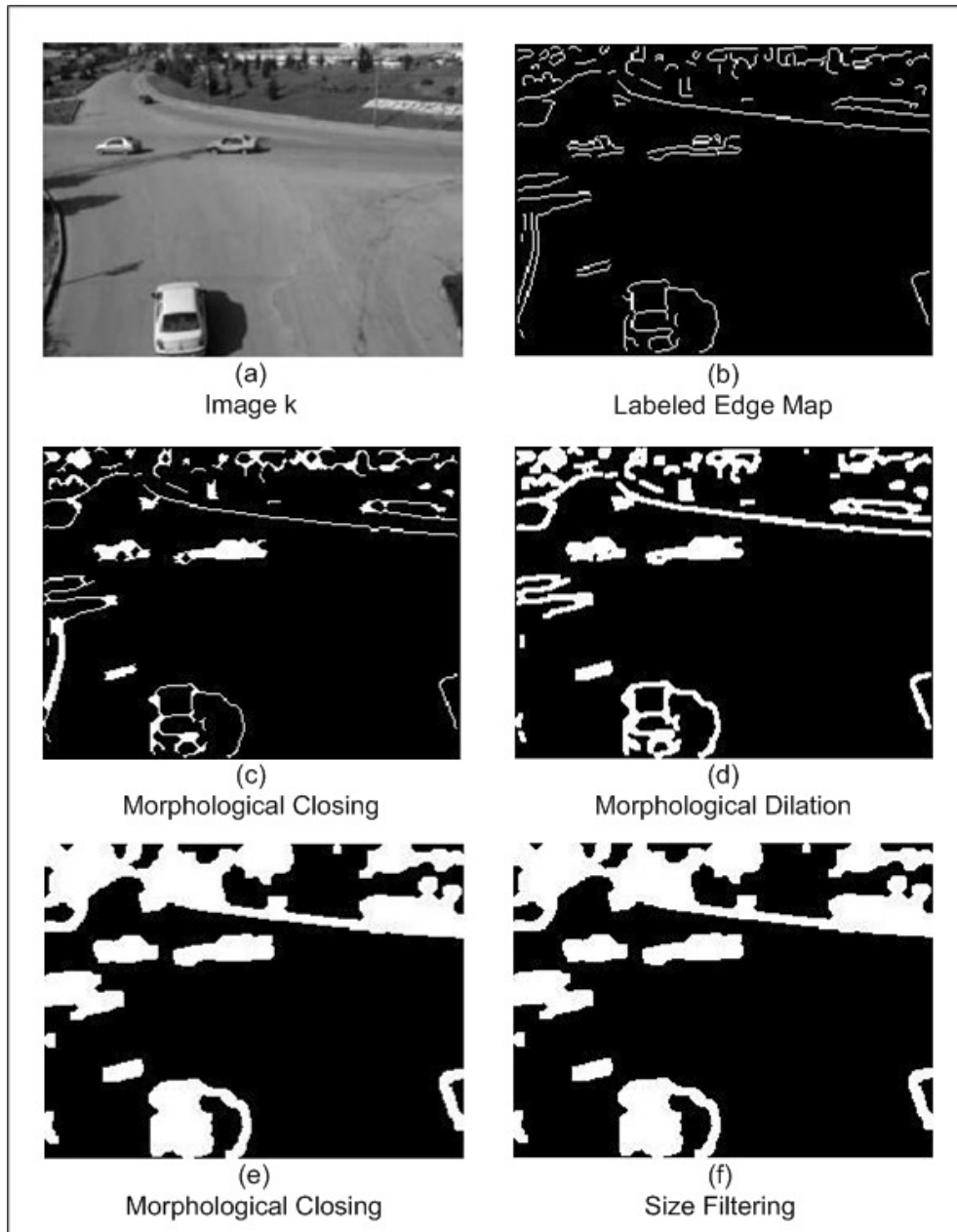
**Figure 3-4: Object Detection Using Frame Differentiation**

### **3.2.2 Object Detection Using Edge Map**

Frame differentiation method gives us information about the moving regions of the scene. It is difficult to extract the whole vehicle region from this difference mask. In this part, we propose a method to extract the vehicle location and shape more correctly. The proposed method uses the edge information together with moving object regions determined by frame differentiation method to obtain the shapes of moving objects. Edge information is used as post processing to moving object extraction part. The main idea of this post processing is to construct the shape of the object at the regions that are determined with the frame differentiation method. Since we are dealing with human made structures, i.e. our objects are vehicles, they generally have sharp boundaries, which give significant local intensity changes in edge detection. For this purpose, basic morphological operations are applied to the edge map of the image and the regions that give significant local intensity changes are obtained. The block diagram of the proposed edge map extraction method is given in Figure 3-5. An example of the application of the method is given in Figure 3-6.

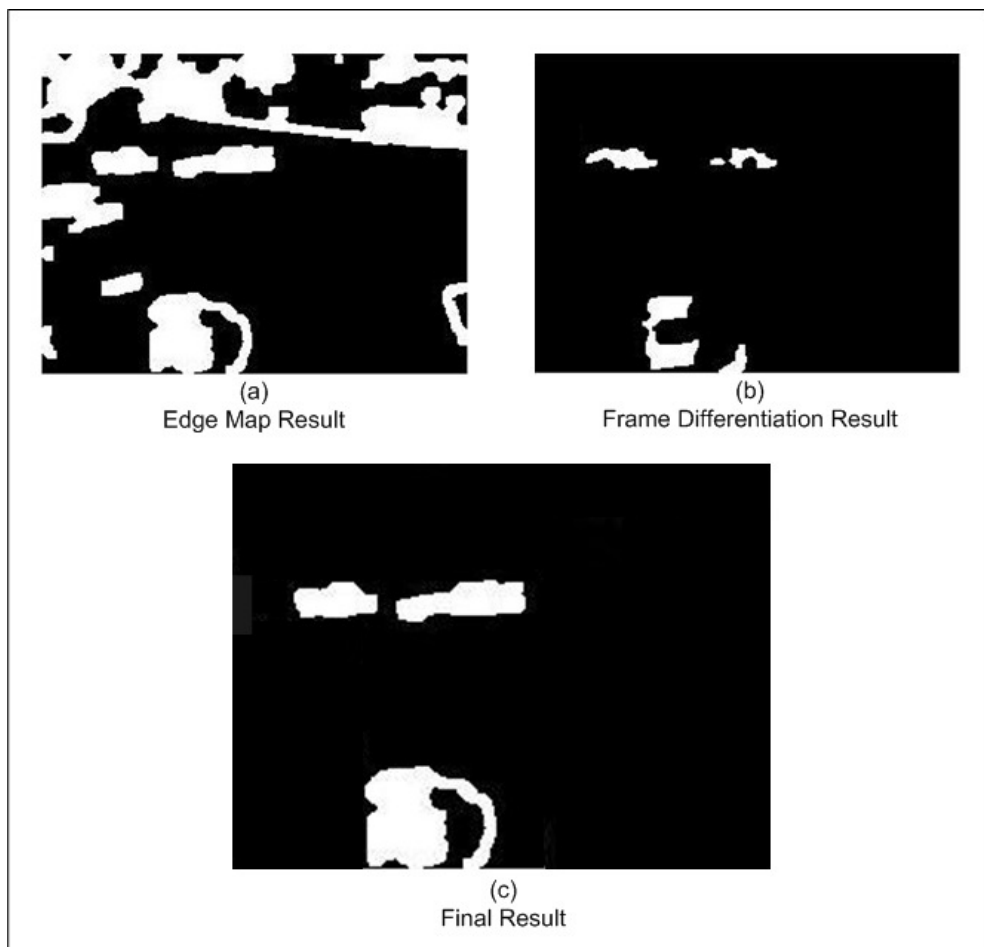


**Figure 3-5: Edge Map Extraction Method**



**Figure 3-6: An Example of the Proposed Edge Map Extraction Method**

To detect the moving objects among the ones found by edge detection based object detector, the outputs of frame differentiation and object detector are combined. The combination is done by associating edge groups and regions found by frame differentiation. Image pixels are grouped by binary image searching using 8-connected neighborhood. The binary image searching process is explained in Section 2.4. An example of the proposed method is given in Figure 3-7.



**Figure 3-7: Object Detection Using Edge Map and Frame Differentiation**

### 3.3 Proposed Vehicle Detection System

In this thesis, both *frame differentiation* and *background subtraction* approaches are used for vehicle detection problem. These two methods have both advantages and disadvantages compared to each other. For example, background subtraction provides accurate localization and shape of the vehicle, however, the background extraction and updating processes are complicated tasks. On the other hand, the frame differentiation approach is simpler and faster than background subtraction method. Furthermore, since we are interested with the changes occurred at one frame step, we get out of the importance of background currency. Frame differentiation is also robust to the small camera movements. In our application, the camera is generally placed on a pole or on a bridge, so it may be exposed to vibrations due to wind or vehicles crossing the bridge. For such cases, the currency of the background is lost but the proposed frame differentiation method can deal with small vibrations by eliminating their effect using morphological operations and edge information.

In this study, the result of background subtraction method is modified, whenever necessary by the output of the frame differentiation method to eliminate some disadvantages of it. A decision-making system is used to merge the results of the two methods. Decision-making system uses thresholded difference image obtained in background subtraction method to decide currency of the background. Thresholding the background difference image gives the moving vehicles as well as any change in the background image. In this study, we assume that if the rate of the active pixels in the thresholded difference image is less than a maximum allowable pixel change rate, then the background image is almost stationary. If the system decides that the background is almost stationary and the shadow is negligible, then the detection result of background subtraction method is directly used as the output of the vehicle detection system. However, if the above conditions are not satisfied then the outputs of two methods are combined using a simple AND operation.

The block diagram of the proposed vehicle detection system is given in Figure 3-9. The final step of the algorithm is the shadow elimination part, which is explained in the next subsection.

### 3.3.1 Shadow Removal

The background subtraction method, applied directly, may not differentiate foreground and shadows. Therefore, the detection result of the background subtraction method gives the moving objects with their shadows. The frame differencing method also extracts the shadows as a part of the moving object, since the shadow moves together with the vehicle. The main disadvantage of the shadow in vehicle tracking problem is inaccurate vehicle localization, which is a critical problem especially for wide shadow cases. The shadow removal algorithm proposed in this work has two assumptions.

- Only wide shadows will be eliminated
- The whole shadow region is not classified as foreground in the proposed frame differencing method

The pixels belonging to shadow have similar illumination values. They only give edges at shadow boundaries. Therefore, only boundaries of the wide shadow regions will appear in the morphologically determined edge map. These shadow boundaries are removed using a morphological opening. The block diagram of the shadow removal method used in this work is given in Figure 3-8. Figure 3-9 shows how the proposed vehicle detection algorithm reacts on an example. The effect of shadow removal is also demonstrated on this figure.

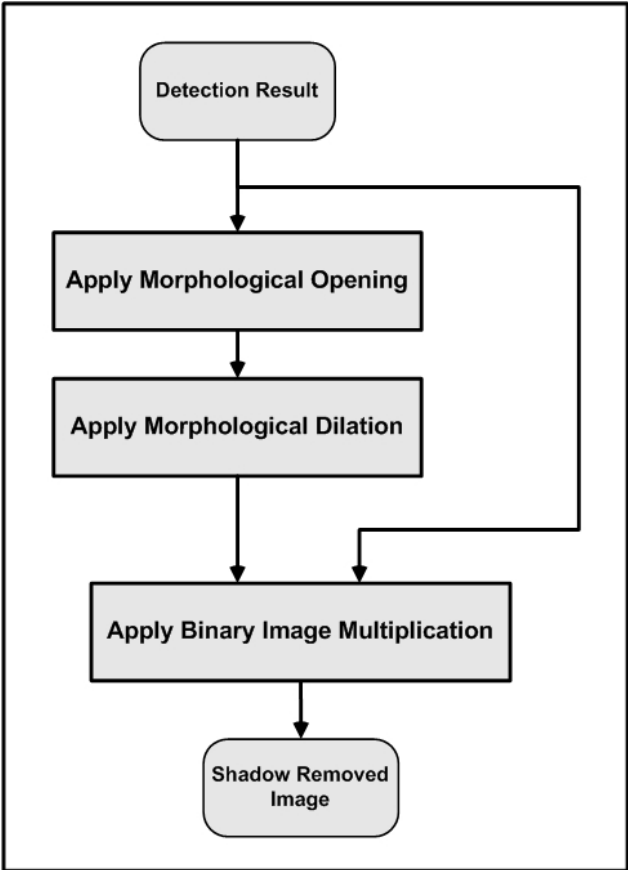


Figure 3-8: Shadow Removal

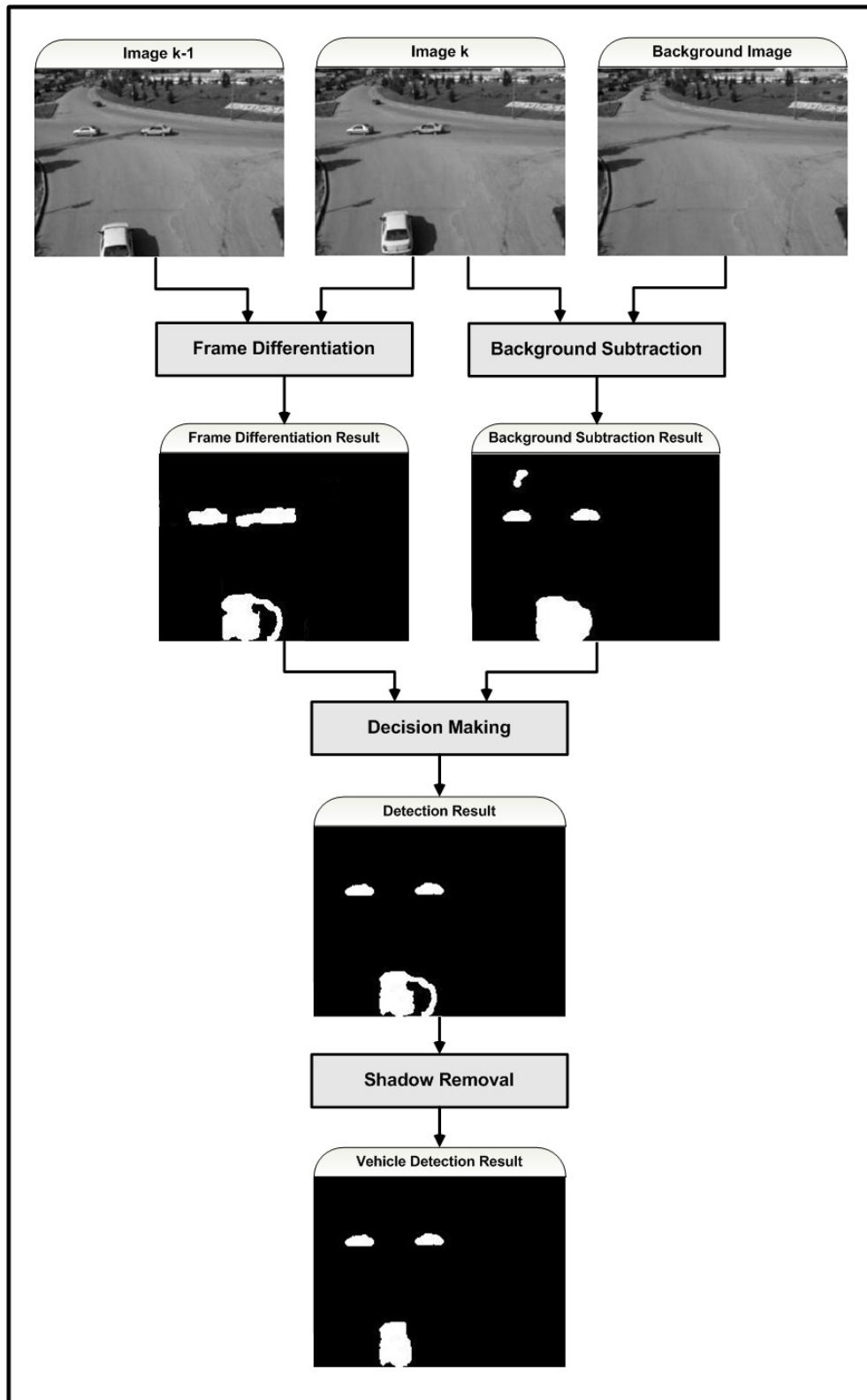


Figure 3-9: Vehicle Detection System

## CHAPTER 4

### TRAINING: MODELING OF THE SCENE AND THE MOTION

The scene inspected in this study is assumed to be stationary, which provides the ability to collect and use *a-priori information* for further processing. This information can either be provided by user or determined automatically by the system using the training data. In this study, we attempt to develop a fully automated system to obtain maximum available a-priori information about the scene and the targets. Obtaining the a-priori information about the scene is a part of the training step. The training step consists of two main parts: *parameter extraction* and *a-priori information collection*.

*Parameter extraction* system extracts the traffic information such as the vehicle count, traffic events, and traffic flows. In this thesis, a comprehensive parameter extraction system is produced that will be used by the a-priori information collection.

*A-priori information collector* uses the train data obtained by the parameter extraction system and performs analysis on this data that is necessary to obtain the scene model. *Scene model* consists of two parts: Markov chain model of the vehicle motions and linear stochastic models for the motions corresponding to different trajectories. Before mentioning the a-priori information collection process, we will first introduce the parameters that will be used in our traffic analysis system.

## 4.1 Parameter Extraction

In this thesis, the parameters are given in two classes: *image related and motion related parameters*. Motion related parameters are extracted when a vehicle is associated with a track. The parameter extraction process can be thought as a data-storage and an analysis center of the tracking system. A detailed explanation of the parameter extraction system is given in this section.

### 4.1.1 Vehicle Parameters

Vehicle parameters are obtained from the vehicle silhouettes, which are provided by the detection algorithm. Some basic vehicle parameters are illustrated in Figure 4-1.

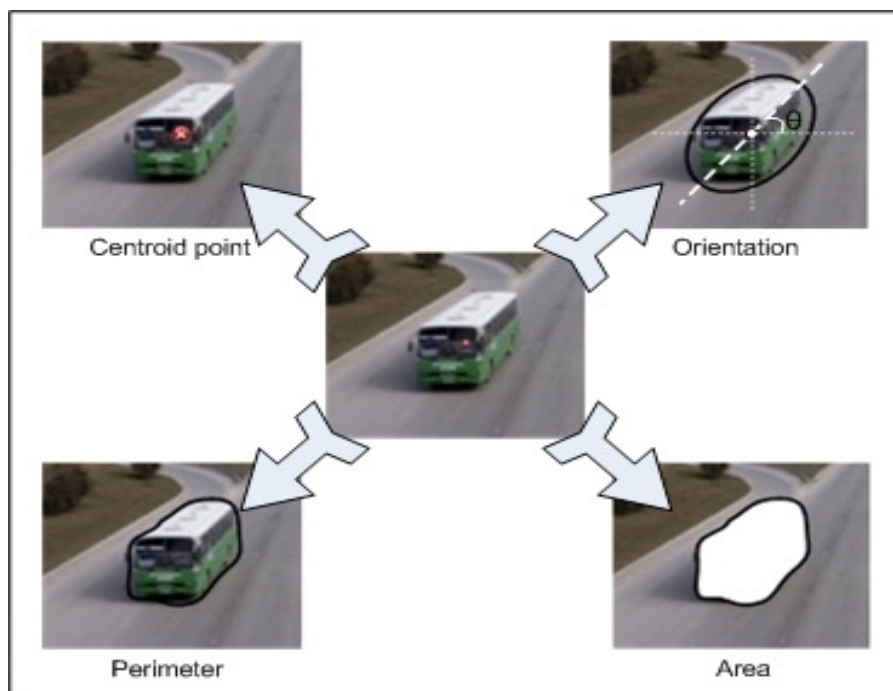


Figure 4-1: Vehicle Parameter Extraction Examples

Vehicle parameters can be discussed in two different classes: *image parameters* and *motion parameters*. The image parameters of a vehicle are extracted directly from the detected silhouette image of the vehicle. These parameters are specified in Table 4-1. The vehicle image parameters that are indicated with an asterisk are used in the tracking system of this thesis. Other image parameters are used for image representation, which will be explained in Section 5.1.1.

**Table 4-1: Image Parameters of a Vehicle**

<b>Image Parameters</b>
Image frame number *
Vehicle number in the image frame *
Position (centroid point) of the vehicle *
Area (pixel size) of the vehicle *
Orientation of the vehicle *
Perimeter (pixel size) of the vehicle
Radius of the vehicle (radius of the circular representation)
Vehicle length (major axis length of the ellipsoidal repr.)
Vehicle width (minor axis length of the ellipsoidal repr.)
Pixel distribution on major axis (ellipsoidal repr.)
Pixel distribution on minor axis (ellipsoidal repr.)

The motion parameters of a vehicle can be extracted using both the current and the previous frame's image parameters of this vehicle. Motion parameters are obtained by the help of a basic Kalman tracker explained in Section 5.3.2. The tracking information of a track is combined with the image parameters of a vehicle at each track instant to get the motion parameters of that track. As an example, the centroid information given by basic Kalman tracker is used as centroid of the vehicle associated to a track, but other parameters including velocity is computed by some other means. The computation methodology is taken from references [15, 16, 20, 24].

**Table 4-2: Motion Parameters of a Vehicle**

<b>Motion Parameters</b>
Track Label
Position (centroid point) of the vehicle
Velocity (centroid point) of the vehicle
Area (pixel size) of the vehicle
Orientation of the vehicle
Speed (magnitude of the velocity) of the vehicle
Forward speed of the vehicle
Lateral speed of the vehicle
Major axis length (Ellipsoidal representation) of the vehicle
Minor axis length (Ellipsoidal representation) of the vehicle
Predicted Position (centroid point) of the vehicle
Predicted Area (pixel size) of the vehicle
Predicted Orientation of the vehicle
Track Comments (new, merged, hidden)

The vehicle parameter extraction process is repeated for each detected object in a frame. Thus, image and motion parameters of each vehicle are labeled with its frame number and detection system vehicle labels. In addition, the label of the associated track is also added to vehicle parameters. During the tracking process, these vehicle parameters are accumulated for vehicle parameter analysis. The analysis and results of these parameters are explained in the next section.

#### **4.1.2 Traffic Parameters**

The traffic parameter extraction process provides some stochastic information about the scene and the targets. This aim is achieved by analyzing the vehicle parameters that are collected in the vehicle parameter extraction process. The result of this analysis would be used to increase the performance of both the detection and the tracking systems. The traffic parameters are given in Table 4-3. In the extraction of these parameters, we have used the methods of references [6, 8, 23, 25].

**Table 4-3: Traffic Parameters**

<b>Traffic Parameters</b>
Vehicle count
Average speed
Average forward speed
Average lateral speed
Average vehicle area
Average vehicle length (major axis length)
Average vehicle width (minor axis length)
Average vehicle orientation
Average frame length of the tracks

#### **4.2 A-priori Information Collection**

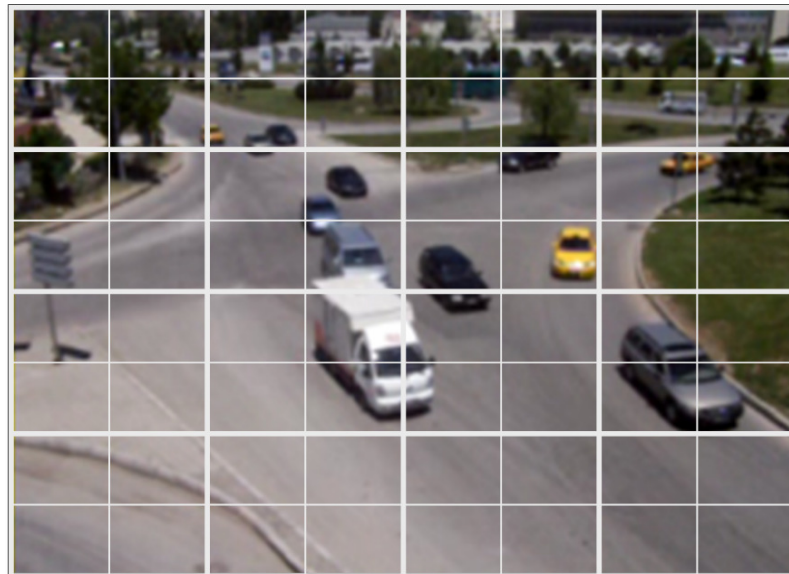
A-priori information is defined as any information that can be extracted before the operation of the system and will be helpful to improve the performance. This information can either be provided by the user or determined automatically by the system using the training data. For instance, the background image of the scene and minimum-maximum allowable target sizes are considered as a-priori information. In this work, the training data, which is explained in the previous subsection, are used to get a-priori information. The a-priori information collector system is composed of *scene modeling* and *path modeling* parts.

*The scene modeling part* produces upper level knowledge about the problem using the collected information consisting of image and motion parameters of the vehicles in the scene. In the remaining part of this chapter, we will obtain a “*model of the scene*”. Modeling of the scene will consist of obtaining the roads, starting and ending regions of the tracks in the scene, basic motions of the vehicles on the roads, velocities, sizes and orientations of the vehicles at different locations in the scene etc. To obtain such a model, first, scene is partitioned to some rectangles of sizes proportional with traffic load. Next, the average velocity, size and orientation of vehicles in each rectangle are computed together with the corresponding correlations.

*The path modeling part* of modeling is to get equations of major motions of the vehicles in the scene. This is achieved first by finding the roads and then analyzing the tracks of the vehicles. All the work is done automatically, without human interaction.

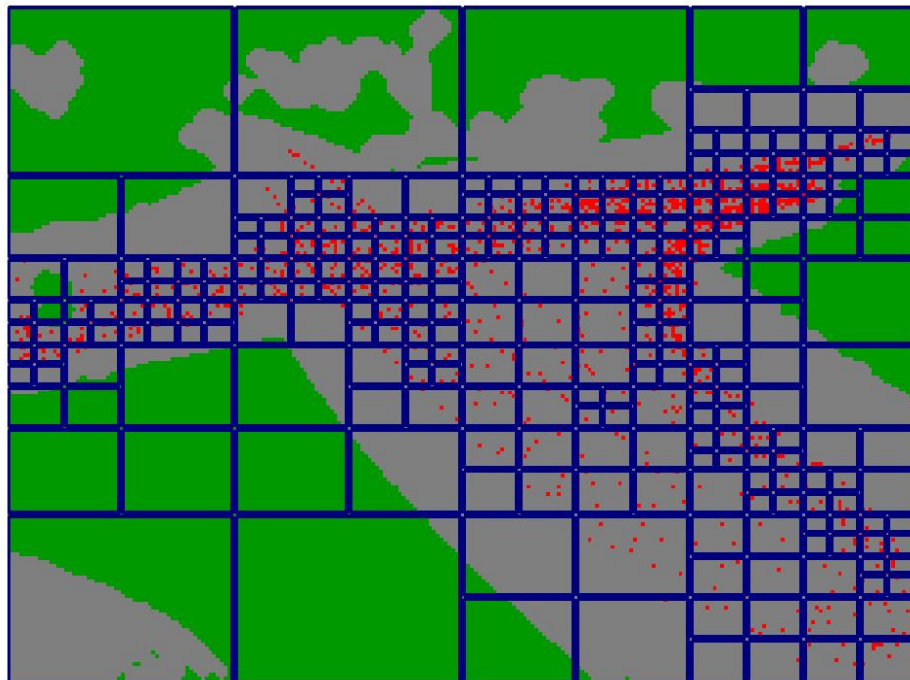
#### **4.2.1 Scene Partitioning**

Since we are dealing with uncalibrated images, vehicle and traffic parameters vary from one part of the image to another. Thus, instead of dealing with the whole scene, considering small partitions of it will provide better results for traffic monitoring. The scene is divided into small partitions to decrease the effect of non-uniform distribution of the vehicle and traffic parameters on the scene. In this work, scene partitioning is done automatically. To facilitate scene partitioning, we initially divide the scene to equal rectangles. An example of this partitioning is given in Figure 4-2.



**Figure 4-2: Scene Partitioning Example**

The mission of a-priori information collector system is to divide the scene to small partitions and gather some statistical information about the targets detected in that partition. However, the traffic is not uniform in the scene. Some regions of the scene have heavy traffic, which means that the number of frames that motion detected in that partition is high. The partitions with heavy traffic indicate important regions of the scene and representing these regions with smaller partitions will provide better representation. In addition, there are also some regions, where no vehicle is detected. It would be useless to divide them to small partitions; therefore, instead neighboring partitions should be merged at such regions.



**Figure 4-3: Partition Traffic Based Scene Partitioning**

If a partition includes the centroid of a vehicle then it is considered to belong to that partition. The traffic load of each partition is obtained by using the centroid of the vehicles in that partition. Detection system provides silhouettes of the vehicles and a simple centroid computation algorithm computes the centroid of each

silhouette. If the relative number of centroid in a partition is small, then it is merged with the neighboring partitions and if it is large, partition is divided into four rectangles.

#### 4.2.2 Partition Modeling: Statistics of Partitions

After dividing the scene to small rectangles, we obtain some statistics about the vehicles in each partition. The statistics will be based on the vehicles that are in that partition. The features for which the statistics will be obtained are listed in Table 4-4. The statistics are related with these features are their mean and variance values.

**Table 4-4: Vehicle Features Used in the Partition Modeling**

<b>Vehicle Features</b>
Position (centroid point) of the vehicle
Velocity (centroid point) of the vehicle
Area of the vehicle
Orientation of the vehicle

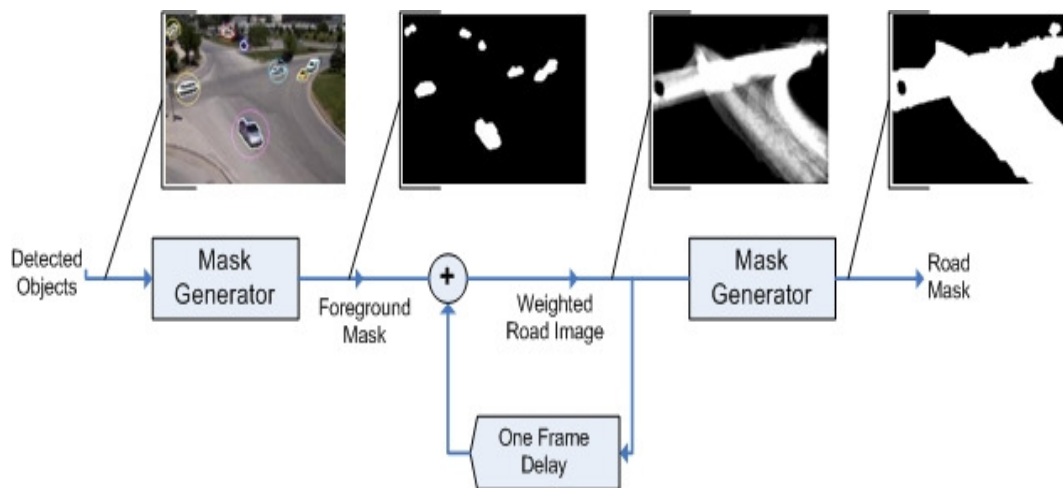
Table 4-5 gives the partition feature statistics used in this thesis. These partition feature statistics, which are also called *partition parameters*, are used to obtain the *partition model* for each partition in the scene. These statistics provide knowledge about the scene and scene partitions. For example, when a new moving object appears in the scene, the detection system may make a decision about the object class (vehicle or not) based on the a-priori information of the partition that gives an idea about minimum, maximum and average vehicle sizes in that partition. If the new detected vehicle area is smaller than the minimum vehicle area then the detection system does not consider that object as a vehicle. This provides to identify and report pedestrians walking on the road regions.

**Table 4-5: Partition Feature Statistics Used in Partition Modeling**

Partition (State) Feature Statistics
Mean of the position
Mean of the velocity
Mean of the vehicle area
Mean of the vehicle orientation
Variance of the position
Variance of the velocity
Variance of the vehicle area
Variance of the vehicle orientation
Mean value of the velocity magnitude
Max value of the velocity magnitude

### 4.2.3 Motion Region Detection

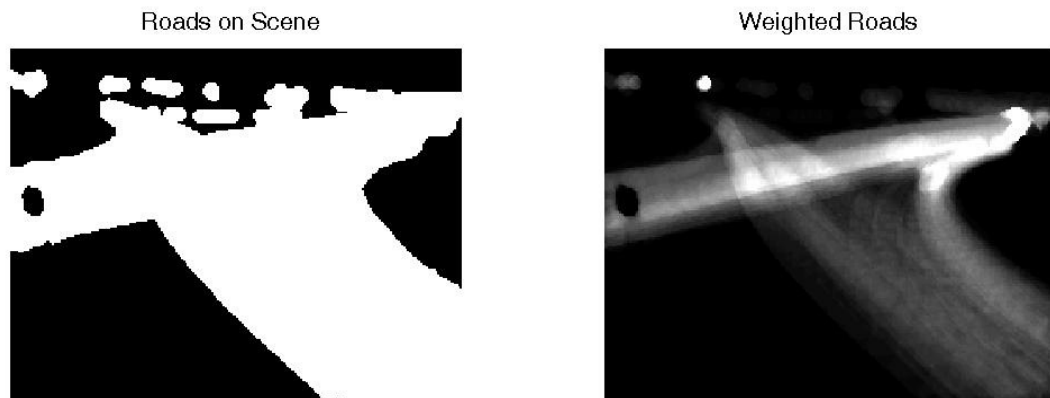
The scene (actually the background) is almost fixed because a stationary camera is used in this work. The background and foreground regions are extracted at each frame using the moving vehicle detection system.



**Figure 4-4: Motion Region Detector**

The motion region detection process, which is illustrated in Figure 4-4, uses the silhouettes of detected vehicles to get active (motion detected) pixels of the scene. By taking the union of these active pixels at each frame, we get the moving regions of the scene. The resultant map is called motion region map. Each pixel of motion region map has a value, which illustrates number of frames that it is active. If there is no detection error and none of the detected vehicles is moving out of the actual roads, we can say that the binary mask of this moving regions map roughly gives the road mask of the scene. To avoid some problems related with misclassifications, we only take the pixels that are active more than a threshold value. The processed moving regions map is binary masked and a road map is generated.

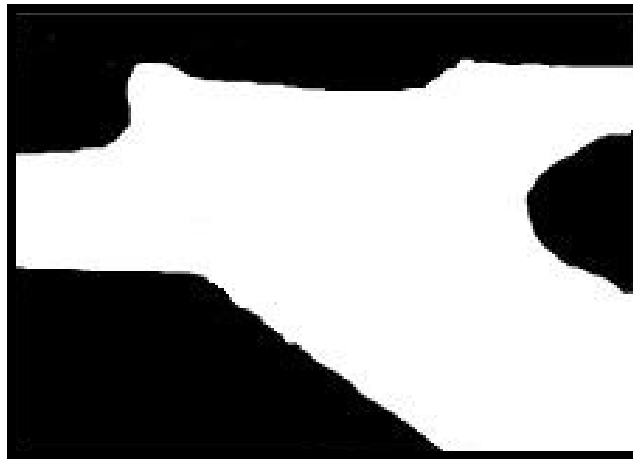
Figure 4-4 demonstrates the block diagram of the motion region detection process explained above. The output of each step of the motion region detection process is also illustrated in Figure 4-4. In our application, we have sufficient data to extract the motion region and the roads accurately.



**Figure 4-5: Motion Detected Regions of the Scene**

- (a) Motion detected regions of the scene are shown white in the figure.**
- (b) Motion detected regions of the scene are shown weighted in the figure.**

The figures above illustrate motion-detected regions of the scene. In Figure 4-5-b, the illumination value of each pixel illustrates number of frames that motion detected at that pixel. Some morphological operations are done after the thresholding to remove clutters and fill small empty regions in the moving region map. Finally, the map is morphologically enlarged to compensate the unexpected motions near roadsides. The resultant motion region map for the scene discussed in this motion region detection example is shown in Figure 4-6.

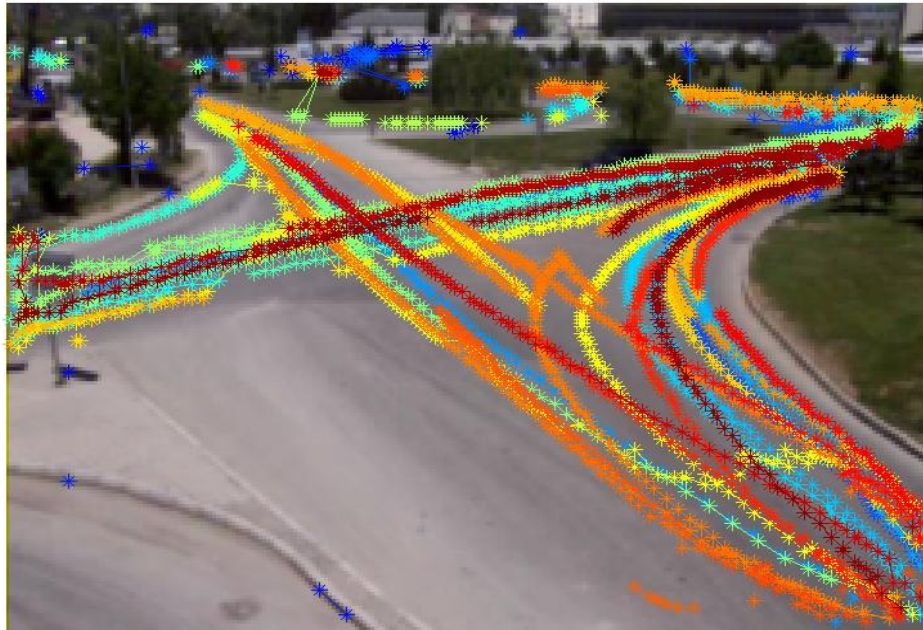


**Figure 4-6: Motion Region Map of the Scene**

#### **4.2.4 Path Detection**

The moving region map consists of all road paths in the scene. To determine each road path in the scene separately, the vehicle motion trajectories, which are shown below, are examined and similar vehicle trajectories are grouped to define a single path. This path extraction process can be done in two steps. The first step is to find out entrance and exit regions on the scene, which are called “*gates of the scene*”. The second step is to group the vehicle motion trajectories that start from the same entrance region and end at the same exit region.

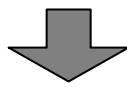
Entrance and exit gates are computed by using the tracks obtained from the training data. Because of some detection and/or tracking errors, a track may start from or end at any point of the road, but for most of the tracks starting and ending will be at gates. This basic idea is used to find the entrance and exit gates of the scene. Figure 4-8 and 4-9 explain the algorithm as well as its application to the traffic junction examined in this study. Figure 4-7 shows the trajectories of the vehicles in the junction that are used to determine the exit and entrance gates.



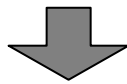
**Figure 4-7: Vehicle Motion Trajectories**



The initial trajectory point for each trajectory, which is derived from the train data, are determined and depicted with black dots in the figure left. These points indicate centroid positions of the vehicles at the moment that they just entered the scene.

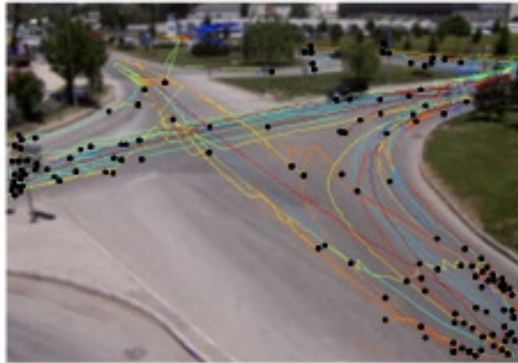


In order to define a region as an entrance gate, adequate number of vehicles must be appeared for the first time in that region. For this purpose, the detected initial points are grouped according to their positional distributions on the scene and crowded initial point regions are determined.

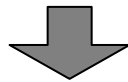


These regions are depicted according to selected region representation type and defined as entrance gates of the scene. In this work the gates are illustrated using either a rectangular gate or a group of scene partitions.

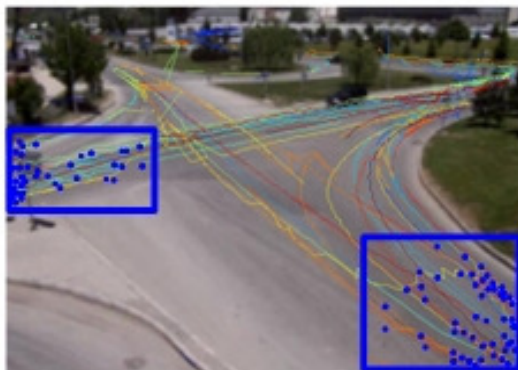
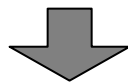
**Figure 4-8: Determination of the Entrance Gates of the Scene**



The final trajectory point for each trajectory, which is derived from the train data, are determined and depicted with black dots in the figure left. These points indicate centroid positions of the vehicles at the moment that they just leaved the scene.

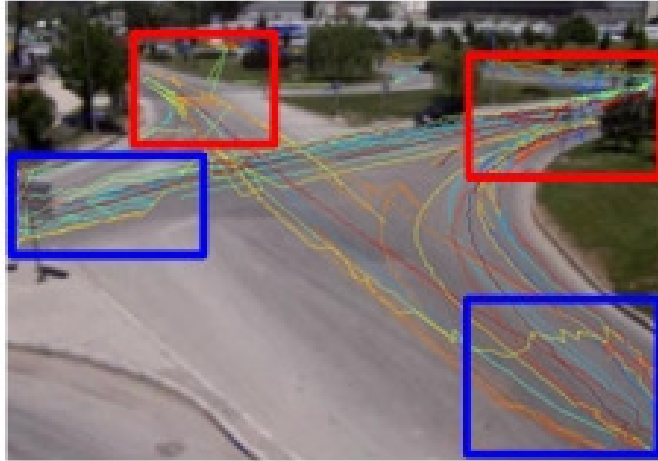


In order to define a region as an exit gate, adequate number of vehicles must be leaved the scene from that region. For this purpose, the detected final points are grouped according to scene distributions and crowded final point regions are determined.



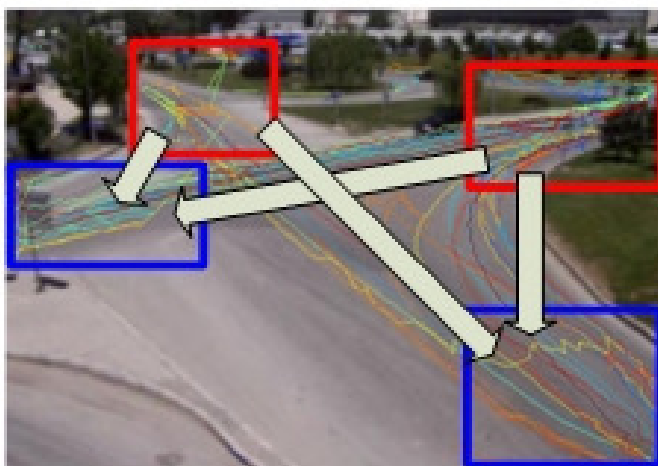
These regions are depicted according to selected region representation type and defined as exit gates of the scene. In this work the gates are illustrated using either a rectangular gate or a group of scene partitions.

**Figure 4-9: Determination of the Exit Gates of the Scene**



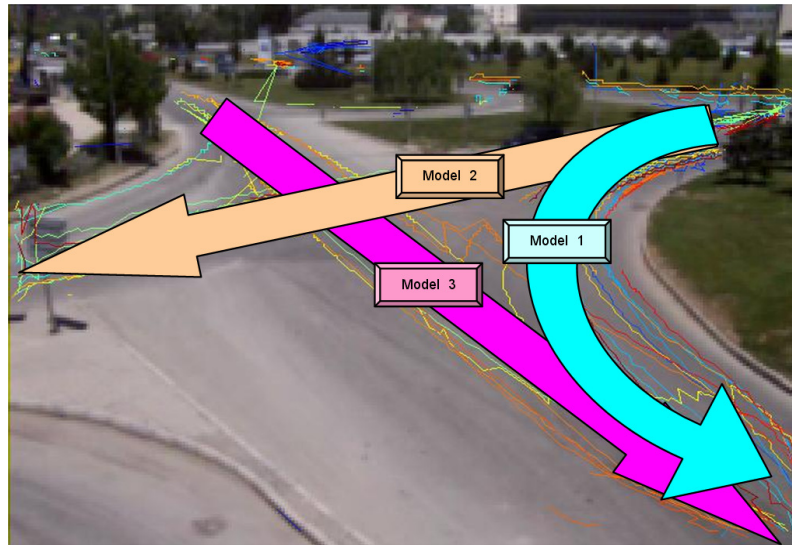
**Figure 4-10: Entrance and Exit Regions (Gates) of the Scene**

The tracks starting from one of the entrance regions and finalizing at one of the exit regions are grouped to express a path. Two entrance and two exit regions are found in our example and these entrance and exit regions are illustrated in Figure 4-10. Two entrance and two exit gates forms four path candidates. The vehicle trajectories are analyzed according to entrance-exit gates and the trajectories corresponding to each path candidate is determined. The entrance-exit gates and path candidates, which are derived from the train data example, are depicted in Figure 4-11.



**Figure 4-11: Path Candidates of the Scene**

Although the path detection system determines the four path candidates as in the figure shown above, frequently used three motion paths are considered as a path by the elimination of one. Frequently used motion paths, which are shown above, include more than 80% of the junction traffic. Therefore, the motion models at this junction are defined using these three motion paths, which are illustrated in Figure 4-12.

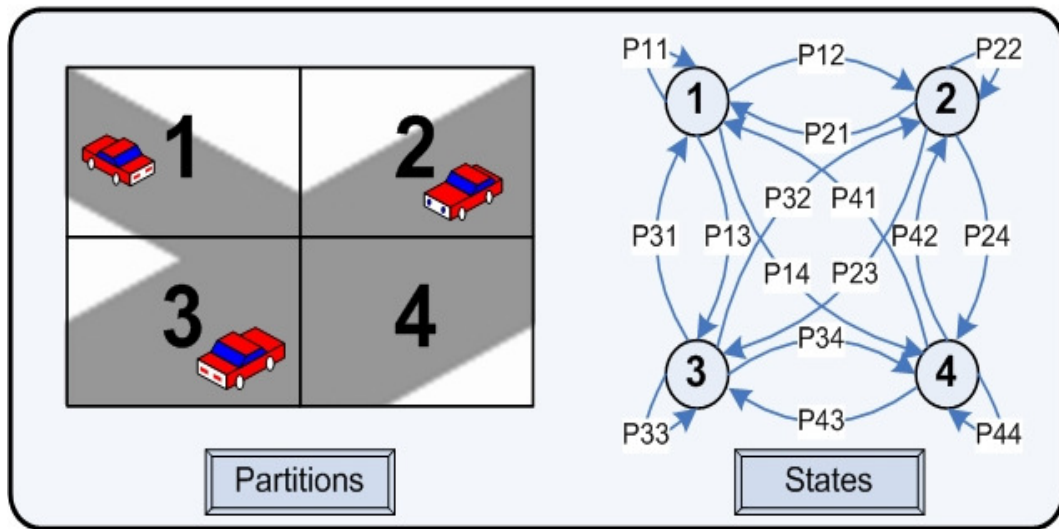


**Figure 4-12: Paths Defined in the Scene**

#### **4.2.5 Motion Model Extraction Based on Partition Statistics**

The main approach of this section is to define the motion of a vehicle as a movement from one partition to other. The partitions are taken as states of a Markov chain, that is, a partition is represented in the Markov space with a state number having that partitions label. The movements of the vehicles from one partition to the other are taken as state transitions. These transitions are analyzed for a large train data set and the Markov state transition matrix and initial probabilities are derived for each motion model. These probabilities are calculated for different motion models representing different paths in the scene. This means that for every path detected in the scene a probability transition matrix is trained.

A partition motion model training example is demonstrated below. Figure 4-13 indicates the Markov state representation of the partitions. In this demonstration, the scene is divided to four equal partitions. The motion of the vehicles between these four scene partitions can be modeled as a Markov chain. The partitions are represented as Markov states, that is, if a vehicle is in partition x then the vehicle is at state x.



**Figure 4-13 : Markov State Representation of the Scene Partitions**

The state transition probabilities of each state-to-state pairings is shown in the right figure. These probabilities are calculated from the transitions of vehicles from one partition to the other. The state transition matrix (STM) for this example will be in the form given below.

$$STM = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ P_{41} & P_{42} & P_{43} & P_{44} \end{bmatrix} \quad (4.1)$$

An example of extracting the state transition probabilities can be depicted in Figure 4-14. The scene is monitored to get the train data, and vehicle transitions between four partitions are determined. The long arrows indicate partition transitions and small arrows indicate the transitions that occur in the same partition. The state transition probabilities are given in the state transition diagram. The state transition matrix of this example is

$$STM = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ P_{41} & P_{42} & P_{43} & P_{44} \end{bmatrix} = \begin{bmatrix} 0 & 0,4 & 0,2 & 0,4 \\ 0 & 0,4 & 0 & 0,6 \\ 0,2 & 0,2 & 0 & 0,6 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (4.2)$$

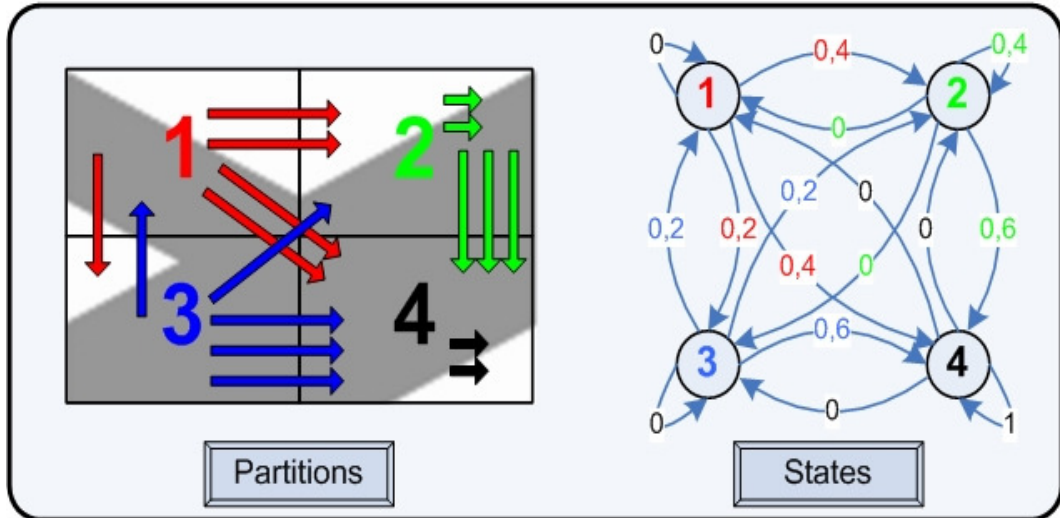


Figure 4-14 : State Transition Example

In this study, the state transition matrix is trained for each path model separately. In the path detection example given before, three different paths are extracted from the trajectories. Therefore, for that example there are only three state transition matrices and initial probability vector. The state transition matrix (STM) for path model  $m$  is defined as

$$STM(m) = \begin{bmatrix} P_{11}(m) & P_{12}(m) & \dots & P_{1n}(m) \\ P_{21}(m) & P_{22}(m) & & \\ \vdots & & \ddots & \\ P_{n1}(m) & & & P_{nn}(m) \end{bmatrix} \quad (4.3)$$

where  $P_{ij}(m)$  denotes state transition probability from state  $i$  to  $j$  for model  $m$  and  $n$  denotes number of Markov states. State transition probability also depends on velocity of the vehicle at that state.  $STM(m)$  is calculated without considering velocities in that state partition, so it can be considered as the state transition matrix corresponding to mean velocity in that state partition. The mean velocity of that partition, which is represented by the term  $\bar{v}$ , is calculated in the partition model training section. Then, state transition matrix (STM) for mean velocity is defined as

$$STM(m, \bar{v}) = \begin{bmatrix} P_{11}(m, \bar{v}) & P_{12}(m, \bar{v}) & \dots & P_{1n}(m, \bar{v}) \\ P_{21}(m, \bar{v}) & P_{22}(m, \bar{v}) & & \\ \vdots & & \ddots & \\ P_{n1}(m, \bar{v}) & & & P_{nn}(m, \bar{v}) \end{bmatrix} = STM(m) \quad (4.4)$$

More accurate state transition matrix must be velocity depended and is represented by  $STM(m, v)$ .

$$STM(m, v) = \begin{bmatrix} P_{11}(m, v) & P_{12}(m, v) & \dots & P_{1n}(m, v) \\ P_{21}(m, v) & P_{22}(m, v) & & \\ \vdots & & \ddots & \\ P_{n1}(m, v) & & & P_{nn}(m, v) \end{bmatrix} \quad (4.5)$$

Note that the diagonal entities of STM denote probabilities of staying on the same partition. Therefore, if a vehicle has a velocity greater than the mean velocity of that state then we expect that the probability of staying on the same state will be smaller than its probability computed with mean value. Since the sum of all rows of STM is equal to one, the sum of other row entities of STM must be decreased. The diagonal elements of velocity depended state transition matrix is calculated as

$$P_{nn}(m, v) = P_{nn}(m, \bar{v}_n) - \Delta P_{nn}(m, v) \quad (4.6)$$

where  $\Delta P_{nn}(m, v)$  denotes the state transition probability difference due to velocity variation. If the vehicle has a velocity smaller than the mean velocity of that state then the term  $\Delta P_{nn}(m, v)$  must be a negative quantity. The state transition probability difference is taken as

$$\Delta P_{nn}(m, v) = (v - \bar{v}_n) \alpha_n \quad (4.7)$$

where the term  $\alpha_n$  denotes the state transition probability coefficient of the velocity difference. The state transition probability coefficient of the velocity difference is taken as

$$\alpha_n = \frac{P_{nn}(m, \bar{v}_n)}{v_n^{\max} - \bar{v}_n} \quad (4.8)$$

where  $v_n^{\max}$  denotes maximum velocity of a vehicle that passes through that partition.

The velocity depended state transition probability of a vehicle for staying at the same state is explained above. Since the STM is a special matrix which satisfies

$$\sum_{j=1}^n P_{ij} = 1 \quad , \quad \forall j \quad (4.9)$$

The off-diagonal elements of P must be adjusted according to this constraint. The formulation given in the following table is used in this work.

**Table 4-6: Velocity Depended State Transition Probabilities**

Diagonal Entities	Other Entries ( $n \neq i$ )
$P_{mm}(m, v) = P_{mm}(m, \bar{v}_n) - \Delta P_{mm}(m, v)$	$P_{ni}(m, v) = P_{ni}(m, \bar{v}_n) + \Delta P_{ni}(m, v) \frac{P_{ni}(m, \bar{v}_n)}{(1 - P_{ni}(m, \bar{v}_n))}$
$\Delta P_{mm}(m, v) = (v - \bar{v}_n) \alpha_n$	$\Delta P_{ni}(m, v) = (v - \bar{v}_n) \alpha_n$
$\alpha_n = \frac{P_{mm}(m, \bar{v}_n)}{v_n^{\max} - \bar{v}_n}$	$\alpha_n = \frac{P_{ni}(m, \bar{v}_n)}{v_n^{\max} - \bar{v}_n}$
$v_n^{\max} = \frac{\bar{v}_n}{\ \bar{v}_n\ } \max\{v_n^{-\max}, v_n^{+\max}\}$	
$v_n^{-\max} = \max\{-v_n\}$	$v_n^{+\max} = \max\{v_n\}$

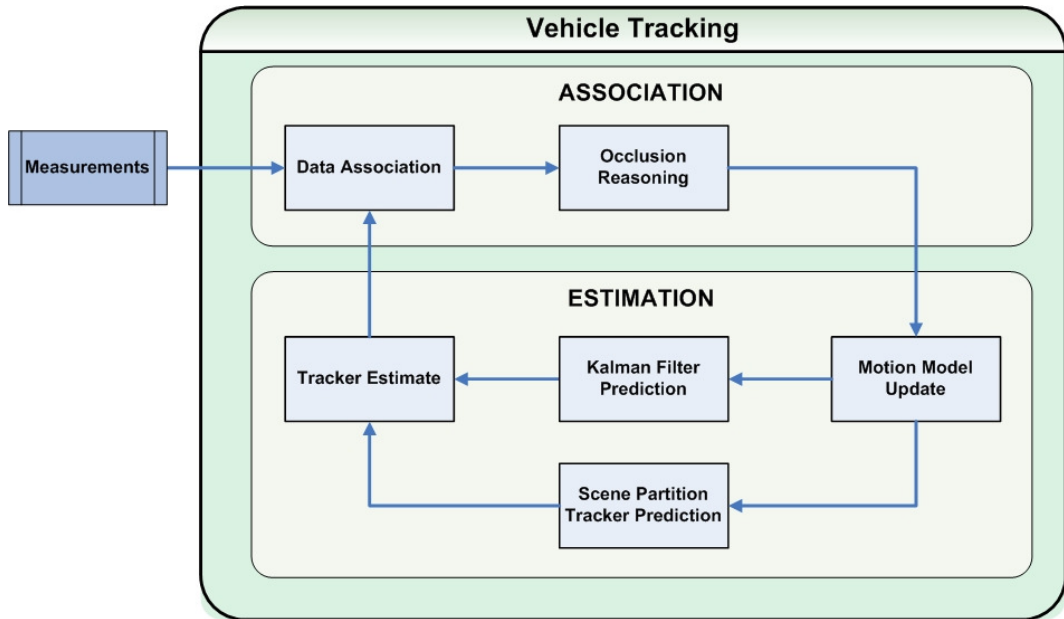
## CHAPTER 5

### VEHICLE TRACKING

In this chapter, we introduce the basic elements of any multi-target tracking system. Furthermore, the main issues, which influence the design of a multi target tracking system, are described. Firstly, some target analysis based on target shape and representation is explained. Then, a number of models are established to describe target's motion and an overview about vehicle tracking algorithm is given. Afterwards, the vehicle tracking system, which is developed in this work, is presented. The vehicle tracking system is examined in two key steps: *estimation* and *association*. The block diagram of the proposed vehicle tracking system is given in Figure 5-1.

*The estimation step* assumes a linear model for the motion of the vehicle and aims to estimate the state vector of the vehicle motion in every frame by Kalman Filtering. Additional information about the state of the system comes from the a-priori information system. The use of this information is based on modeling of the scene and the motion of the vehicles according to the Markov model explained in the previous section.

*The association step* of the tracking process is about assignment of the track estimations to the current frame measurements. This step aims to match the estimates of the tracked vehicles features with the features of the vehicles found at the current frame. Firstly, a gating technique is used to eliminate unlikely track-to-measurement pairings. Then, the association block determines which track-to-measurement assignments actually be made. As it is declared in the objectives of the thesis, the tracking system must be robust to occlusion and inaccurate detection problems. To overcome these problems, occlusion reasoning approach is used to improve the performance of the association system.



**Figure 5-1: Block Diagram of the Proposed Vehicle Tracking System**

## 5.1 Target Analysis

This section can be taken as a preparatory work before tracking which consist of target representation analysis and the coordinate systems. The target and background properties are defined in this section. A short overview about target shape and representation is given and the representations, which are suitable for selected tracking algorithm, are explained. The coordinate systems used in this thesis are also explained in this section.

### 5.1.1 Target Shape and Representation

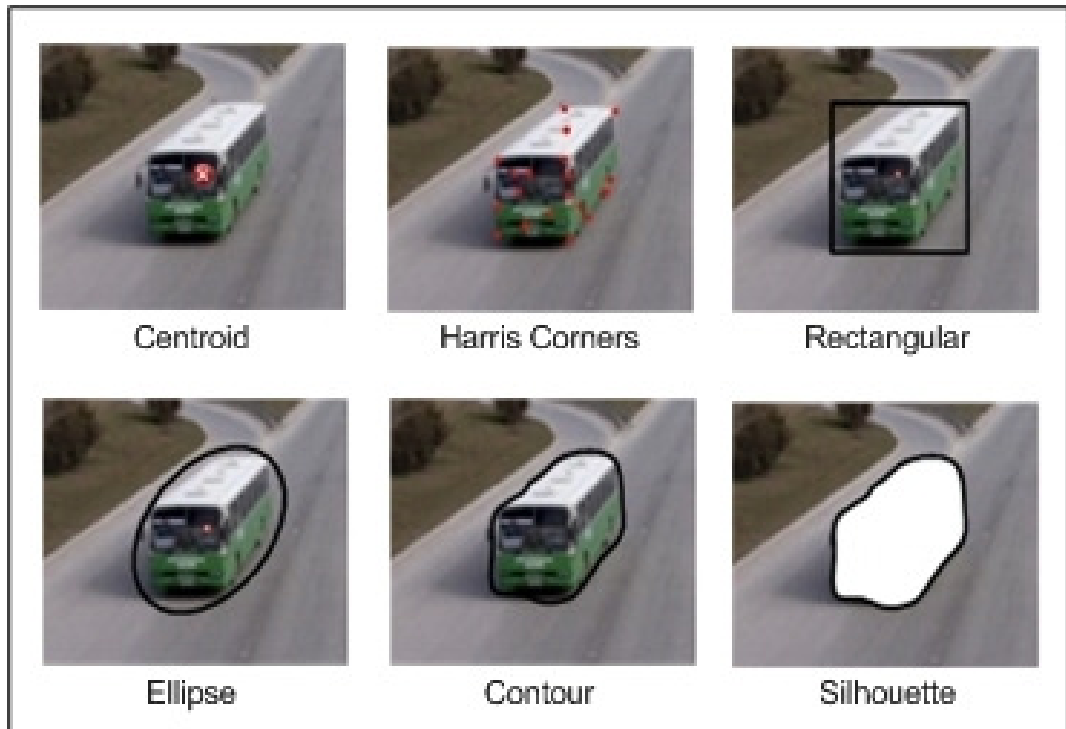
*Yilmaz et al.* [2] define target in a tracking scenario as anything that is of interest for further analysis. For instance, boats on the sea, fish inside an aquarium, vehicles on a road, planes in the air, people walking on a road, or bubbles in the water are a set of targets that may be important to track in a specific domain. In this work, we are dealing with vehicles moving on the road. A vehicle may be a car, a bus or a truck. Some vehicles concerned in this work are shown below.



**Figure 5-2: Some Vehicles Concerned in this Work**

Other moving objects that may appear in the traffic are pedestrians walking in the scene. The pedestrians walking far from the roads do not disturb the system because the detection algorithm ignores moving objects outside the moving region map defined in the previous chapter. The pedestrians walking on the sidewalk or crossing the road will be detected by the moving object detection system. It is obvious that the size of a pedestrian is much smaller than a vehicle. Thus, we can separate a pedestrian from a vehicle by comparing detected silhouette areas. The information about average vehicle size and minimum available vehicle size is already defined in traffic parameter extraction step.

The tracking system aims to track only vehicles in the scene. Thus, the pedestrians detected in the scene will not be tracked. However, the system informs to parameter extraction system about pedestrian detection and gives an alert to report the pedestrian detection in the moving region of the scene.



(a) Centroid, (b) Multiple points, (c) Rectangular patch, (d) Elliptical patch,  
(e) Complete object contour, (f) Object silhouette

**Figure 5-3: Shape and Appearance Model Representations**

The vehicles are represented using the shapes or appearance models in tracking applications. *Yilmaz et al.* [2] describe some common object shape and appearance representations and emphasizes the strong relationship between the object representation and the tracking algorithms. A vehicle can be expressed as a *point*, a *silhouette* or a *kernel*. Each representation has different tracking models. Some of shape and appearance model representations of vehicles are shown above.

- The *point representation* is generally used for objects that occupy small regions in the image. The object can be represented by its centroid point (a) or by a set of meaningful points (b).

- The *kernel representation* aims to describe the shape of the object using primitive geometric shape like a rectangular (c) or an ellipse (d).
- The *silhouette representation* directly uses the results of the detection process. The pixels belong to an object composes the silhouette (f) of the object. The boundary of the silhouette is defined as the contour (e) of the object.

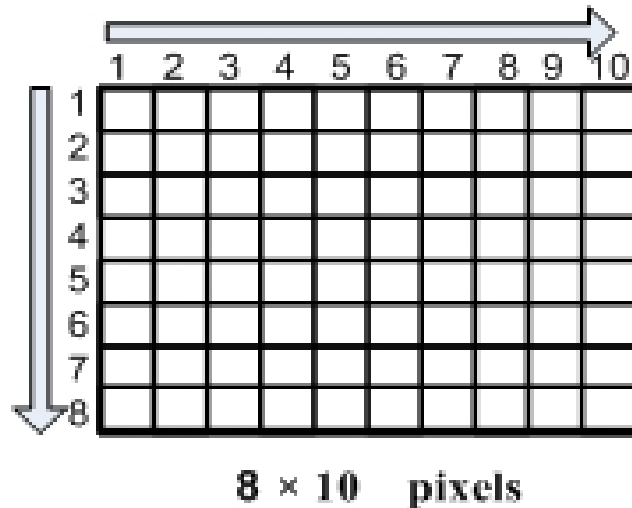
In this thesis, the detection algorithm provides the silhouette of the detected objects in each frame. Therefore, all shape models will be derived using the silhouette map and the original image. The tracking system uses the centroid point representation of the vehicles, but also utilizes shape and kernel features of the vehicle. Therefore, the tracks will be indicated using all of the model representations: centroid point, object contour and elliptical kernel. Some object model representations of a vehicle detected in the scene is demonstrated in the figure below.



**Figure 5-4: Some Vehicle Representations used in this Thesis**

### 5.1.2 Target Coordinate Systems

An image is indicated by its row and column dimensions. An  $m \times n$  image denotes an image with  $m$  rows and  $n$  columns, so the pixel coordinates changes from 1 to  $n$  in the horizontal axis and from 1 to  $m$  in the vertical axis. An example of image fixed pixel coordinate system of an  $8 \times 10$  pixel image is illustrated in the figure below.

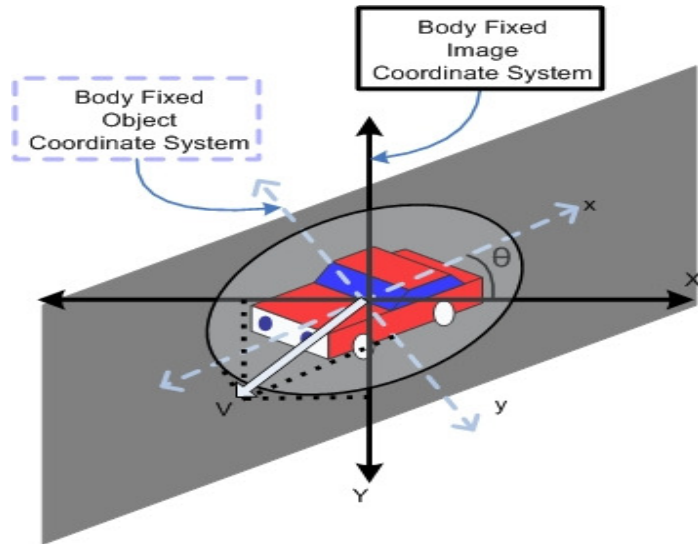


**Figure 5-5: Image Pixel Coordinate System**

In this work, three different coordinate systems are used. The first coordinate system, which is an *image fixed pixel coordinate system*, is already explained above. The other coordinate systems are *body fixed coordinate systems*, which means the origin of the coordinate system is in the centroid point of the corresponding vehicle. Figure 5-6 depicts the body fixed object coordinate systems of an object.

The second coordinate system, which is called *body fixed image coordinate system*, is obtained only by transforming the origin of the coordinate system to the centroid of the vehicle.

The last coordinate system is called *body fixed object coordinate system*. This coordinate system facilitates the processes about the vehicle motion characteristics. The horizontal axis of the body fixed object coordinate system lies on the direction of the orientation line of the vehicle. Therefore, the “x” axis is the front-back axis of the vehicle.



**Figure 5-6: Body Fixed Coordinate Systems**

In the figure above, the body fixed image coordinate system is shown by black lines and the body fixed object coordinate system is shown by dotted lines. As it is shown in the figure, the orientation angle  $\theta$  gives the rotation between the body fixed image coordinate system and the body fixed object coordinate system.

## 5.2 Motion Models

Motion models of targets in the scene are derived in this section. First, motion model of a moving point is explained. Then, advanced motion models are developed using the kinematics of vehicles. Since a moving target has time varying dynamics, a number of models can be established to describe its motion. Although increasing the number of models can give better estimates, it makes the system less efficient since more time will be required to yield the estimate. The next step is to derive road adapted motion models using road traffic parameters. The models are stochastic linear Gaussian models, so states are described by a mean and a covariance matrix. At the final step, covariance matrices, which are derived in image coordinate system, are transformed to body coordinate system using orientation transform. The derivation steps of the motion models are illustrated in Figure 5-7.

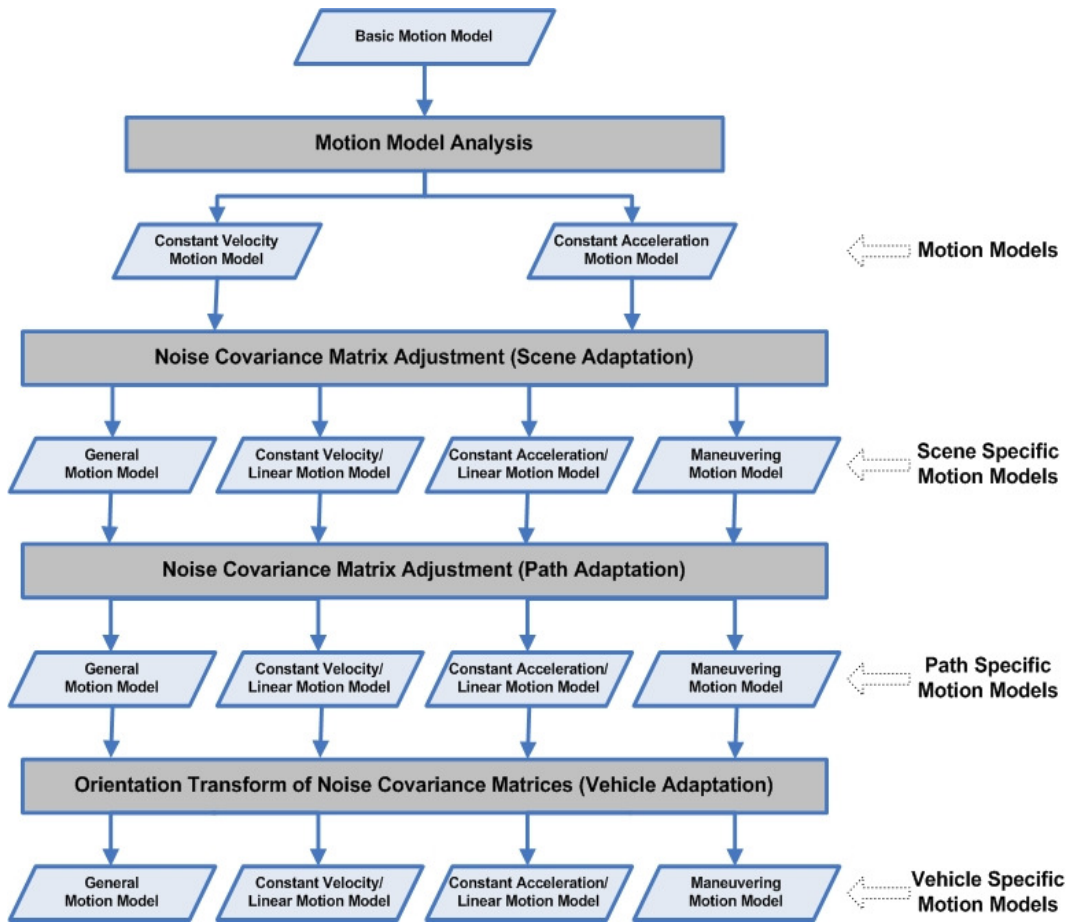


Figure 5-7: Derivation Steps of the Motion Models

### 5.2.1 Basic Motion Model

The *basic motion model* of targets in the scene is explained in this section. The motion of a point target is introduced and a general motion model is developed by assuming that we are dealing with point targets. Consider a point vehicle moving in the scene at constant velocity subject to random perturbations in its trajectory. The *state* and the *measurement* equations are defined as

$$x_{k+1} = Ax_k + G\omega_k \quad (5.1)$$

$$y_k = Cx_k + Hv_k \quad (5.2)$$

We define the state vector as a four-dimensional vector, which represents the center point positions of the target  $\{p^x_k, p^y_k\}$  and the positional change (velocity) of the center point of the target  $\{v^x_k, v^y_k\}$  per unit time interval.

$$\begin{bmatrix} p^x_{k+1} \\ p^y_{k+1} \\ v^x_{k+1} \\ v^y_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \end{bmatrix} + \begin{bmatrix} T^2/2 & 0 \\ 0 & T^2/2 \\ T & 0 \\ 0 & T \end{bmatrix} \begin{bmatrix} \omega^x_k \\ \omega^y_k \end{bmatrix} \quad (5.3)$$

$$\begin{bmatrix} P^x_k \\ P^y_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \end{bmatrix} + H \begin{bmatrix} v^x_k \\ v^y_k \end{bmatrix} \quad (5.4)$$

The measurements,  $P^X_k$  and  $P^Y_k$ , denote the detected moving objects center point coordinates. Note that, the process noise terms,  $\omega^x_k$  and  $\omega^y_k$ , denote the random accelerations of the centroid. Taking the time interval  $T=1$ , it becomes the *basic constant velocity model*.

$$\begin{bmatrix} p^x_{k+1} \\ p^y_{k+1} \\ v^x_{k+1} \\ v^y_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \end{bmatrix} + \begin{bmatrix} 1/2 & 0 \\ 0 & 1/2 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \omega^x_k \\ \omega^y_k \end{bmatrix} \quad (5.5)$$

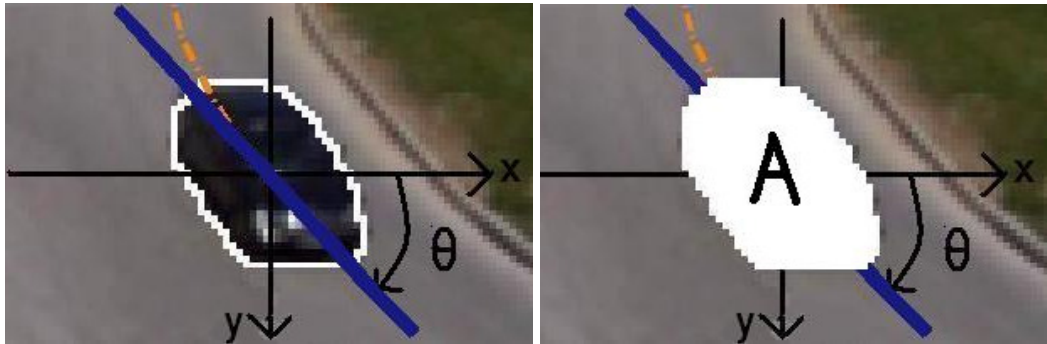
$$\begin{bmatrix} P^x_k \\ P^y_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \end{bmatrix} + H \begin{bmatrix} v^x_k \\ v^y_k \end{bmatrix} \quad (5.6)$$

## 5.2.2 Advanced Motion Models

The basic constant velocity model should be improved to get more realistic target motion modeling. To achieve this goal, first we develop general target motion models. Then, we adjust the parameters of these models for specific vehicle motions and get vehicle motion models.

There are various target motion models defined in the literature [26, 27]. Among them, two mostly known motion models, *constant velocity* and *constant acceleration* models, are introduced in this section. In this work, these two traditional model structures are modified to achieve more accurate tracking.

The first motion model, i.e., the *constant velocity model*, is build by adding two new state variables to the model given in Equation 5.5. First four states of this model are same as the basic motion model. They represent the center point positions of the target object and the positional change of the center point of the target object per unit time interval. The fifth state ( $\theta_k$ ) denotes the orientation angle of vehicle area in the scene for the body fixed image coordinate system. The orientation angle is derived from the shape of the detected objects binary mask. The sixth state ( $A_k$ ) denotes the pixel size of vehicle area in the scene. The orientation angle and the object area are illustrated in Figure 5-8.



**Figure 5-8: Orientation Angle ( $\theta$ ) and Object Area (A)**

The general discrete time state and measurement equations are given as,

$$x_{k+1} = Ax_k + \omega_k \quad (5.7)$$

$$y_{k+1} = Cx_k + v_k \quad (5.8)$$

Here, the six-dimensional state vector of the constant velocity model is defined as

$$x_k = \begin{bmatrix} p^x_k & p^y_k & v^x_k & v^y_k & \theta_k & A_k \end{bmatrix}^T \quad (5.9)$$

$$y_k = \begin{bmatrix} X_{k+1} & Y_{k+1} & \Theta_{k+1} & \widehat{A}_{k+1} \end{bmatrix}^T \quad (5.10)$$

The state and the measurement equations of the *constant velocity motion model* are defined as

$$\begin{bmatrix} p^x_{k+1} \\ p^y_{k+1} \\ v^x_{k+1} \\ v^y_{k+1} \\ \theta_{k+1} \\ A_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \\ \theta_k \\ A_k \end{bmatrix} + \begin{bmatrix} \omega^{px}_k \\ \omega^{py}_k \\ \omega^{vx}_k \\ \omega^{vy}_k \\ \omega^\theta_k \\ \omega^A_k \end{bmatrix} \quad (5.11)$$

$$\begin{bmatrix} X_{k+1} \\ Y_{k+1} \\ \Theta_{k+1} \\ \widehat{A}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \\ \theta_k \\ A_k \end{bmatrix} + \begin{bmatrix} v^x_k \\ v^y_k \\ v^\theta_k \\ v^A_k \end{bmatrix} \quad (5.12)$$

The second motion model, which is called *constant acceleration model*, is build by adding two new state variables to constant velocity model. Thus, the state variable for constant acceleration model is defined as an eight-dimensional vector. First six states and last two states are same as constant acceleration model. Constant acceleration model is used to represent curvilinear motions like maneuvering vehicles. To obtain a maneuvering motion, velocity change per unit time interval for x and y direction is included as state variables. Adding acceleration states to motion model, a curvilinear motion model may be generated.

The eight-dimensional state vector of the constant acceleration model is defined as

$$x_k = [p^x_k \quad p^y_k \quad v^x_k \quad v^y_k \quad a^x_k \quad a^y_k \quad \theta_k \quad A_k]^T \quad (5.13)$$

$$y_k = [X_{k+1} \quad Y_{k+1} \quad \Theta_{k+1} \quad \widehat{A}_{k+1}]^T \quad (5.14)$$

The state and the measurement equations of the *constant acceleration motion model* are defined as

$$\begin{bmatrix} p^x_{k+1} \\ p^y_{k+1} \\ v^x_{k+1} \\ v^y_{k+1} \\ a^x_{k+1} \\ a^y_{k+1} \\ \theta_{k+1} \\ A_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1/2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \\ a^x_k \\ a^y_k \\ \theta_k \\ A_k \end{bmatrix} + \begin{bmatrix} \omega^{px}_k \\ \omega^{py}_k \\ \omega^{vx}_k \\ \omega^{vy}_k \\ \omega^{ax}_k \\ \omega^{ay}_k \\ \omega^\theta_k \\ \omega^A_k \end{bmatrix} \quad (5.15)$$

$$\begin{bmatrix} X_{k+1} \\ Y_{k+1} \\ \Theta_{k+1} \\ \widehat{A}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \\ a^x_k \\ a^y_k \\ \theta_k \\ A_k \end{bmatrix} + \begin{bmatrix} v^x_k \\ v^y_k \\ v^\theta_k \\ v^A_k \end{bmatrix} \quad (5.16)$$

The fifth and sixth states,  $a^x_{k+1}$  and  $a^y_{k+1}$ , denotes the velocity change of the center point of the target object per unit time interval. This state model is called constant acceleration model and used for non-linear motions such as maneuvering motion.

### 5.2.3 Scene Specific Motion Models

Different tracking scenarios are observed and various vehicle motions are examined to simulate the motion of vehicles more accurately. Finally, four vehicle motion models are assigned to substitute for all possible vehicle motions. These four different vehicle motion models are developed using two motion model structures (constant velocity and constant acceleration motion model) explained before. These vehicle motion models are,

- constant velocity-linear motion model,
- constant acceleration-linear motion model,
- maneuvering motion model,
- general motion model.

The *linear motion* means the path of the vehicle on the scene is a straight line. A vehicle having constant velocity angle produce a linear motion. The linear motions have both *constant velocity* and *constant acceleration* models. The constant acceleration model is used for vehicles gaining speed or slowing down.

The *maneuvering motion model* is used for vehicles performing a turning motion, that is, a circular movement where the velocity angle is changing. These three motion models are derived for specific vehicle movements that mostly occur in the traffic. The last motion model, which is called general motion model, is utilized for motions that cannot be put into one of the classes that we have defined.

As we declared before, these vehicle motion models are developed using the constant velocity and constant acceleration motion model explained before. The first and the last motion models use the constant velocity motion model, which consist of six-dimensional state variable and the other motion models use the constant acceleration motion model, which consist of six-dimensional state vector. The main difference that distinguishes the motion models is the covariance of the process and measurement noises.

The random variables  $\omega$  and  $v$  represent the *process* and *measurement noise* (respectively). They are assumed to be independent (of each other), white, and with normal probability distributions.

$$\omega \sim N(0, Q) \quad (5.17)$$

$$v \sim N(0, R) \quad (5.18)$$

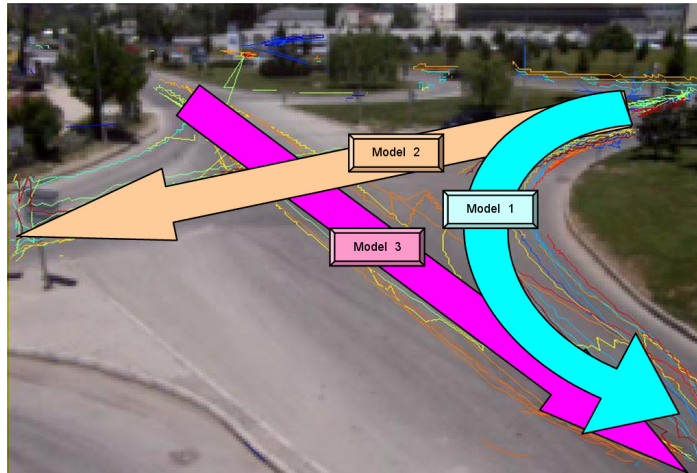
The symbol  $Q$  in the equation above denotes *process noise covariance matrix* and the symbol  $R$  denotes *measurement noise covariance matrix*. These noise covariances are initially determined by calculating variance of each state and adjusted by trial and error technique. The given noise covariances are generalized for such a tracking with similar scene conditions and camera parameters.

It is obvious that these parameters may change for each vehicle, but it is so hard to estimate them for each vehicle especially at the moment that a vehicle just enters to the scene. To cope with this problem we adapt the motion models to the scene using the extracted scene traffic parameters. Then, the motion models specific to scene is derived. The next step is to derive the noise covariance matrices for each path separately by using a parameter adjustment technique.

#### **5.2.4 Path Specific Motion Models**

In the path extraction part, we determine the most significant paths on the scene. An example of path extraction process is given in the figure below. The aim of this section is to determine the motion models for each path in this junction.

Path specific motion model derivation process for a path consists of two main steps. The first step is the selection of the most suitable motion model for that path. This step will be explained in the path model estimation phase of the tracking system, which is introduced in Section 5.3.4.1.



**Figure 5-9: Path Models Extracted from the Scene**

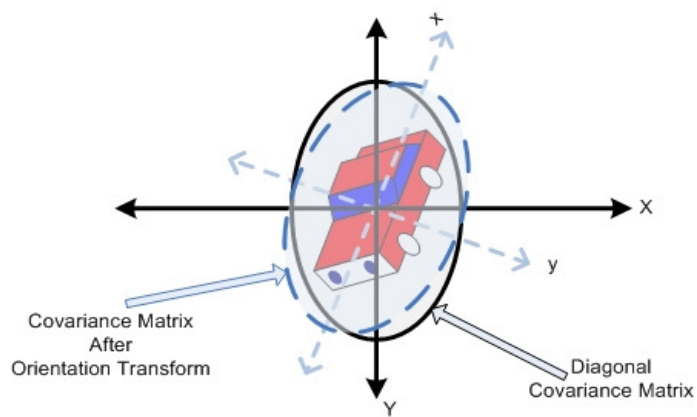
The second step is to adjust the noise covariances for the corresponding path using a-priori information collected from the partitions, related with the specific path. This process is almost same as adjusting the Markov chain probabilities for different velocities. For each parameter, the scene traffic value is compared with path traffic value and a parameter ratio coefficient is calculated. For example, the parameter ratio coefficient of the velocity noise term is the ratio of the average velocity change extracted from the path and the scene. This ratio coefficient is limited in the interval  $\frac{1}{2}$  and 2, which means that the maximum allowable change in noise covariances is halving or doubling. Then, the noise covariance matrices are determined for each path separately by simply multiplying the covariance matrix by the parameter ratio coefficient. If the path of a vehicle is known, the vehicle motion model, which is specific to that path, can be used as tracking model.

### **5.2.5 Modification of Noise Covariance Matrices**

Noise covariance matrices play a critical role in a successful tracking. Therefore, a careful adjustment of these matrices is important. It is not very reasonable to select the process and measurement covariance matrices as diagonal matrices because of high correlation between the x and y elements (position or velocity) of a motion. If body fixed image coordinates are used, then uncorrelatedness

becomes more realistic. Therefore, first we obtain the diagonal covariance matrices at the body fixed image coordinates, and then transform them to the body fixed object coordinates.

The transformation process of the noise covariance matrices from the body fixed image coordinates to the body fixed object coordinates, which is a rotational transformation, is explained in this section. For simplification, the rotational transformation matrix will be derived using only the velocity components of a vehicle.



**Figure 5-10: Transform of Noise Covariance Matrices**

The relation between velocity components of the body fixed image coordinate system and body fixed object coordinate system can be written as

$$V_x = V_x \sin \theta + V_y \cos \theta \quad (5.19)$$

$$V_y = V_x \cos \theta - V_y \sin \theta \quad (5.20)$$

where  $\{V_x, V_y\}$  denotes velocity components in the body fixed image coordinate system and  $\{V_x, V_y\}$  denotes velocity components in the body fixed object coordinate system. The relation between  $\{V_x, V_y\}$  and  $\{V_x, V_y\}$  can be put in a matrix form as follows

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} = F \begin{bmatrix} V_x \\ V_y \end{bmatrix} \quad (5.21)$$

The rotational transformation matrix  $F$  can be used to transform two axis components of a variable from the body fixed image coordinates to the body fixed object coordinates. This rotational transformation matrix can be expanded to the process noise covariance and the measurement noise covariance matrices. The expanded versions of the rotational transformation matrix of the process and the measurement noise covariance matrices are denoted as  $F_{proc}$  and  $F_{meas}$  respectively. Using the  $F_{proc}$  and  $F_{meas}$ , the covariance matrices are transformed as follows.

$$\hat{Q} = F'_{proc} Q F_{proc} \quad (5.22)$$

$$\hat{R} = F'_{meas} R F_{meas} \quad (5.23)$$

### 5.3 Estimation

Tracking is the estimation of the state of a moving object (the position, velocity, acceleration and other features of a target) based on past measurements. There are several tracking approaches with regard to multi target tracking [8, 2, 3, 28]. In this work, the targets in the scene are represented as centroid points of the detected silhouettes. Thus, the vehicles are tracked using a point tracking system [2].

The most widely used tracking approach in point tracking applications is Kalman filtering [29, 30, 31]. In this study, a new algorithm is proposed to solve the multi-vehicle tracking problem that can deal with problems such as occlusion, short period object lost or inaccurate object detection. Firstly, a multi-model Kalman tracker is developed and used for multi-vehicle tracking problem at a traffic junction. Afterwards, the performance of the tracking process is increased by using a Markov chain based tracker together with the Kalman tracker. This

section is focused on these two trackers, namely, the multi-model Kalman tracker and the Markov scene partition tracker. By combining these vehicle trackers with the developed occlusion reasoning approach, the continuity of the track is achieved for situations such as target loss and occlusion.

The proposed tracker uses a-priori information about the scene and the vehicles on the scene. The a-priori information is extracted by a-priori information collector system, which uses a tracking system to obtain vehicle motion trajectories in the scene. A basic tracker is used to follow the vehicles on the scene to get priory information about the scene.

### 5.3.1 Kalman Filtering

The Kalman filter, which is one of the most widely used state estimator, obtains an optimal estimate of the states of a linear Gaussian system. The Kalman filter involves the estimation of the state of a target. This step can be subdivided in two parts: *state prediction* and *measurement update*.

The *state prediction* step consists of forward projection of the previous state estimate to obtain the predicted estimate for the current frame. The second step, which is called the *measurement update*, consists of incorporating a new measurement into the predicted state estimate to obtain an improved estimate.

The goal of the Kalman filter is to combine the measurements taken from the target with the information provided by the motion model in order to obtain an optimal estimate of the target state. Typically, the system equations are in the form:

$$X_k = A_{k-1}X_{k-1} + w_{k-1} \quad (5.24)$$

$$Y_k = C_k X_k + v_k \quad (5.25)$$

where  $X_k$  is the state vector at frame  $k$ ,  $A_{k-1}$  is the transition matrix of the system between frame  $k-1$  and  $k$ , and  $w_{k-1}$  is a sequence of zero-mean white Gaussian noise with covariance matrices  $Q$ . In Equation 5.25,  $Y_k$  is the measurement vector,  $C_k$  is the measurement matrix and  $v_k$  is a sequence of a zero-mean white Gaussian measurement noise with covariance matrices  $R$ . Further, we assume that the process noise  $w_{k-1}$  and the measurement noise  $v_k$  are mutually uncorrelated, and the initial state  $X_0$  is Gaussian. Under these conditions Kalman filter gives the optimal estimate of the state in the sense that the error covariance of the state is minimized. For the sake of the completeness, we will give the equations of Kalman filtering with no explanation or proof.

**Table 5-1: Prediction Equations**

<b>Prediction</b>	
Predicted state estimate	$\hat{X}_k = A_{k-1} \hat{X}_{k-1}$
Predicted error covariance	$P_k^- = A_{k-1} P_{k-1} A_{k-1}^T + Q_k$

**Table 5-2: Measurement Update Equations**

<b>Measurement Update</b>	
Innovation or residual	$Inn = Y_k - C_k \hat{X}_k^-$
Innovation covariance	$S_k = C_k P_k^- C_k^T + R_k$
Kalman gain	$K_k = P_k^- C_k^T S_k^{-1}$
Updated state covariance	$\hat{X}_k = \hat{X}_k^- + K_k (Y_k - C_k \hat{X}_k^-)$
Updated error covariance	$P_k = P_k^- - K_k S_k K_k^T$

One cycle of the Kalman filter algorithm used for multi-target tracking is demonstrated in the block diagram form in Figure 5-11.

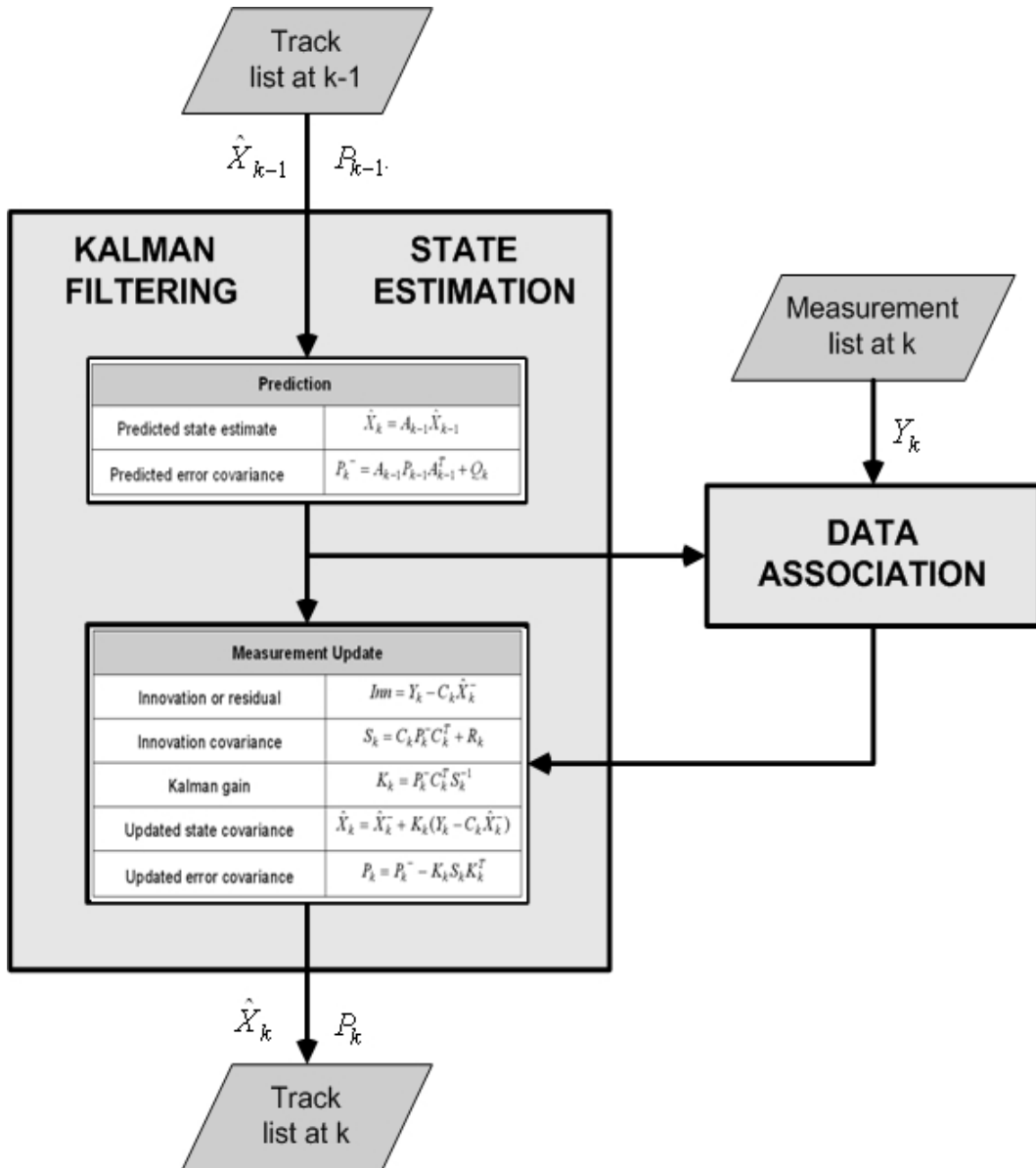


Figure 5-11: One Cycle of Kalman Filter Algorithm

### **5.3.2 Basic Kalman Tracker**

In this section, we introduce a basic tracker to track the vehicles in a frame sequence. Once we detect a moving vehicle, we can track the moving object efficiently by predicting the next center coordinate using the observed coordinate of the moving object.

Basic tracker is used in the a-priori information collecting system for tracking the vehicles in the train data set. Since we do not have any knowledge about the scene, we design a general tracker with a general vehicle motion model. Thus, the specific vehicle motions and roads in the scene cannot be considered in this design. The basic motion model, explained in Section 5.2.1 and given by Equations 5.5 and 5.6, is suitable for such a general tracker.

The basic Kalman tracker is used to estimate motions of the vehicles and associate them with new detected vehicles in the following frame to get motion trajectories of the vehicles in the train data set. In each step, information about each detected vehicle in the scene is determined and saved for a-priori information collector system.

### **5.3.3 Multi Model Kalman Tracker**

The multi-model Kalman tracker operation principal is almost same as basic Kalman tracker with the exception of model selection part. Basic Kalman tracker uses one motion model but in multi-model case, we choose most suitable model from different motion models. Once we estimate the most suitable motion model, we can apply Kalman filtering algorithm to estimate the next state of the motion.

The multi-model Kalman tracker uses path specific motion models explained in the motion modeling section. The motion model selection process is explained below.

### 5.3.3.1 Motion Model Selection

In Section 5.2.4, paths of a given junction and path specific motion models are already introduced. The aim of this section is to decide on one of these models for a given track at each frame. Multi-model approaches like IMM (interacting multiple model), GPB1 (first order generalized pseudo Bayesian) or GPB2 (second order generalized pseudo Bayesian) type of algorithms are used in the literature. However, our approach is to make the decision by using a-priori information given by Markov model of the scene greatly simplifies the computations as well as improves the performance. In this approach, the most suitable Kalman motion model is selected for each path by using Markov model of the scene developed in this thesis.

An illustrative example about the motion model selection approach developed in this work is explained below. The example mentions simplicity of the path selection process and the advantages that are provided by using Markov models. Let us consider a vehicle that is traveling in the lane of a path, which is allocated to u-turn. If the vehicle does not get out of the path, it will continue its motion and generate a rotational movement according to road shape. This motion can be tracked more accurately by using the maneuvering motion model.

The algorithms like IMM are flexible enough to choose or combine more than one model by looking to the recent past and present positions of the movement of the object. However, some sharp trajectory changes, like u-turn, may not be detected at a proper time and usually cannot be tracked by IMM or others. In our approach, the tracking system “knows” from the position of the object which path will be used in the future as well as which model should be used by the tracker.

Thus, the suitable motion model for the u-turn road segment, which is obviously the maneuvering motion model, can be used at the start of the u-turn lane. As a result, the Kalman model estimation approach developed in this thesis will increase the performance of the multi-model Kalman tracker.

The initialization of the state variables of the state prediction filter is an important issue in Kalman state estimation. The stability of a tracker depends on the quality of initialization. As an example, in the case of a constant velocity motion model, in order to identify initial velocity, the tracker must be initiated after the second appearance of the object. However, in this work, the initialization step uses the partition parameters obtained from scene modeling to get the initial velocity and the initial acceleration when the object occurs for the first time in the scene. The main idea is to initiate the track using a-priori information collected for the partition that contains the centroid of the vehicle. This initialization approach increases the initial motion estimation performance of the system, which is very important for cases where hidden vehicle tracking is needed at the beginning of the track.

#### **5.3.4 Markov Scene Partition Tracker**

The aim of this section is to explain how to track a vehicle using Markov scene partition model. The scene is taken as state space and the partitions are taken as states of a Markov chain. The movements of the vehicles from one partition to other are taken as the state transitions. These transitions are analyzed for a large train data set and the *scene partition transition* and *initial probability matrices* are derived for each path model as explained in Section 4.2.5.

Note that Markov models are path specific as explained in the training section. Furthermore, the partition transition probabilities are modified according to the last velocity magnitude estimation of the vehicle. Therefore, the partition transition probabilities depend on the estimated path model as well as the magnitude of the velocity of the vehicle. So, the next partition of a vehicle may be estimated if the current state, the velocity and the path model of the vehicle is known. The state and the velocity of a detected vehicle can be easily determined from the vehicle parameters obtained from the Kalman tracker. Thus, if the path model of the vehicle can be estimated, the state estimation can be done using the STM of that path model. The path model estimation and state estimation processes are explained below.

### 5.3.4.1 Path Model Estimation

The path detection process is already explained in Section 4.2.4. In the path detection part, we represent paths with vehicle trajectories extracted from the train data. Thus, the motion model of a road segment can be determined by examining trajectories of vehicles using that path. The training of the partition transition matrix, which is denoted by STM, is explained in Section 4.2.5. STM's are derived for each path model and they are speed dependent. In this step, we try to estimate the trajectory that a vehicle will move on, using STM and initial probabilities derived in Section 4.2.5. Since, the STM's are different for each path model, the motion path estimation is equivalent to motion model estimation.

The path model of a new vehicle entered to the scene can be estimated using the path dependent initial probabilities. Each partition has a different probability of track initialization for different paths. Path decision can be done in one-step if the largest probability of this partition is at least two times larger than all the others. If this condition is not satisfied than the general motion model is used for tracking and the next position of the vehicle is obtained. Using the Markov scene model the probability of being in the new partition is evaluated for each path and using the same decision rule, one of the paths is decided or the decision is postponed to the next frame.

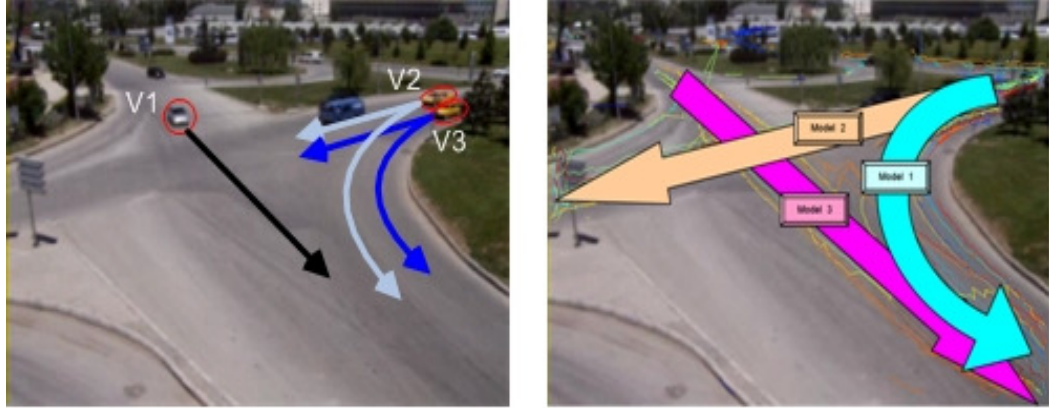
The path model estimation of a moving vehicle in such a shared path partition is performed using the partition transition history (trajectory) of that vehicle. The probabilities of specific paths (equivalently motion model) for a known trajectory is given as

$$P(\text{path} = m \mid \text{trajectory}) = \frac{P(\text{trajectory} \mid \text{path} = m)}{P(\text{trajectory})} \cdot P(\text{path} = m) \quad (5.26)$$

where  $P(\text{trajectory} \mid \text{path} = m)$  is obtained from path dependent  $STM(m)$  and  $P(\text{path} = m)$  is given as

$$P(\text{path} = m) = \frac{1}{\text{number of possible paths}} \quad (5.27)$$

Note that it is not necessary to compute  $P(\text{trajectory})$ , since it is only a scaling constant independent of “ $m$ ”.



**Figure 5-12: Path Model Estimation Example**

The path model estimation process, which is explained above, is illustrated by an example given in Figure 5-12. The scene partition of vehicle 1 selects path model 3 with probability 1. However, vehicle 2 and 3 can follow both the path model 1 and 2 with almost same probability that does not satisfy the decision criteria. Thus, the decision is delayed to next frame and a general path model is used. In addition, if the previous track length of a vehicle is not large enough to estimate the motion model accurately we also assign the general motion model to that track.

#### 5.3.4.2 State Estimation

If the path model of a vehicle is known, its trajectory can be estimated as state (partition) transitions using the STM of that path model. The state of vehicle at  $(n+k)^{th}$  frame is derived by calculating k frame STM of that model with taking the initial state as its current state. The probability of each state at frame  $n+k$  is given as

$$Pr ob(n+k) = Pr ob(n) * P^k \tag{5.28}$$

where  $P$  denotes the State Transition Matrix (STM) and  $Prob(n)$  denotes the probability of being at each state at frame  $n$ . As STM is time invariant the  $k$  frame STM can be calculated by simply taking its  $k^{th}$  power. If  $n$  is taken as the current state then the current state element of probability matrix is one and all others are zeros. So, the probabilities of each state at  $(n+k)^{th}$  frame can be estimated. Then, the state estimation, which gives the highest probability, can be selected. This allows estimating future states of the vehicle more accurately using a-priori information.

### 5.3.5 Hidden Vehicle Tracker

One of the major objectives of this work is to maintain tracking before, during and after an occlusion. The developed multi-model Kalman tracker cannot maintain tracking during occlusion due to measurement need of Kalman filtering. This problem causes loss of tracks during occlusion, which leads inaccurate associations after separation. Therefore, a *hidden vehicle tracker (HVT)* estimates the necessary measurements to maintain track motion estimation during occlusion. During the occlusion period, although a track cannot get any measurement from the detection algorithm, the tracking system will continue tracking using hidden vehicle tracker. The hidden vehicle tracker uses the statistical values of that partition determined in partition modeling and produces pseudo measurements based on these statistics. The measurements and the pseudo measurements are given in Table 5-3.

**Table 5-3: Measurement Estimates**

Measurement	Explanation	Alternative
$X_{k+1}, Y_{k+1}$	Centroid of the vehicle	-
$V^x_{k+1}, V^y_{k+1}$	Velocity of the vehicle	Average Velocity in that partition
$\Theta_{k+1}$	Orientation of the vehicle	Average Orientation in that partition
$\hat{A}_{k+1}$	Pixel area of the vehicle	Average Pixel Area in that partition

To avoid track loss due to greater uncertainty in pseudo measurements and the lack of position information, the measurement equations are changed and the covariance matrices are enlarged.

The pseudo measurements are average values of velocity, orientation angle and pixel area. The measurement model for the constant velocity and the constant acceleration model structures are given as,

$$\begin{bmatrix} AverVx_k \\ AverVy_k \\ Aver\Theta_k \\ Aver\hat{A}_k \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \\ \theta_k \\ A_k \end{bmatrix} + \begin{bmatrix} v^{Vx}_k \\ v^{Vy}_k \\ v^{\Theta}_k \\ v^A_k \end{bmatrix} \quad (5.29)$$

$$\begin{bmatrix} AverVx_k \\ AverVy_k \\ Aver\Theta_k \\ Aver\hat{A}_k \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p^x_k \\ p^y_k \\ v^x_k \\ v^y_k \\ a^x_k \\ a^y_k \\ \theta_k \\ A_k \end{bmatrix} + \begin{bmatrix} v^{Vx}_k \\ v^{Vy}_k \\ v^{\Theta}_k \\ v^A_k \end{bmatrix} \quad (5.30)$$

where  $AverVx_k$  and  $AverVy_k$  denotes horizontal and vertical components of the average velocity in that partition. The other measurements,  $Aver\Theta_k$  and  $Aver\hat{A}_k$ , denotes average orientation angle and pixel area of the vehicles that are determined in that partition.

## **5.4 Association Problem**

The association problem in video tracking systems can be defined as the assignment of the observed vehicles (observations) to vehicle tracks. A multi-vehicle tracking system must decide which of the observations should be associated with which track. For this reason, the association problem is one of the fundamental problems in multi-vehicle tracking systems.

The association problem can be introduced in two steps. The first step, which is called gating and validation, simplifies the association problem by ignoring irrelevant measurements. Then, the second step, which is called assignment, performs the matching of the observed vehicles with the tracks.

### **5.4.1 Gating and Validation**

For each observed vehicle, a multi-vehicle tracking system searches entire scene to associate it to a vehicle track. Searching the entire scene is a time consuming work especially when one is dealing with crowded scenes. To avoid this problem, a validation gate is defined. The validation region simplifies the association by defining gates to the tracks. The tracks outside of an observed vehicles validation region are called irrelevant tracks and ignored for the association process of that observed vehicle. Thus, the gating technique provides to eliminate unlikely tracks to vehicle assignments. Gating techniques are widely used in most of the data association methods in order to reduce the computational complexity.

There are several gating techniques in the literature [26, 28, 32, 33]. In this section, rectangular and ellipsoidal gates are going to be examined in details. For a detailed explanation of gates and derivation of equations, we refer the reader to [26] and [28].

#### **5.4.1.1 Rectangular Gating**

Probably the simplest gating technique is to define rectangular gates [26]. An observation is said to satisfy the gates of a given track, if the residual vector satisfy the relationship given below.

$$|y - \hat{y}| \leq K_G \sigma_r \quad (5.31)$$

where  $\sigma_r$  is the residual standard deviation as defined in terms of the measurement ( $\sigma_o^2$ ) and the prediction ( $\sigma_p^2$ ) variances:

$$\sigma_r = \sqrt{\sigma_o^2 + \sigma_p^2} \quad (5.32)$$

The prediction variance ( $\sigma_p^2$ ) is the appropriate diagonal element of the Kalman filter covariance matrix. Choice of the gating coefficients  $K_G$  is an important issue and it is suggested in [26] that taking  $K_G \geq 3.0$  gives better results in practice. This large choice of the gating coefficient is made in order to compensate for the approximations involved in modeling the target dynamics through the Kalman filter covariance matrix.

#### 5.4.1.2 Ellipsoidal Gating

Ellipsoidal gates are defined according to the residual vector and covariance that is obtained from the Kalman filter. At each scan, the normalized distance is calculated and compared with a gate. The gate ( $G$ ) is defined such that the association is allowed if the following relationship is satisfied by the norm ( $d^2$ ) of the residual vector [26].

$$d^2 = \hat{y}'S^{-1}\hat{y} \leq G \quad (5.33)$$

where  $d^2$  is typically assumed to have chi-square distribution with number of degrees of freedom equal to the dimension of the measurements [28]. Thus, the threshold  $G$  is obtained from the tables of the chi-square distribution.

Finally the volume within the ellipsoidal gate is

$$V = c_{n_z} |GS|^{1/2} \quad (5.34)$$

Note that  $n_z$  is the dimension of measurement and  $c_{n_z}$  is the volume of the unit hyper sphere of  $n_z$  dimension.

In this study, two different validation gates are used in association process. The first validation gate is a two-dimensional regional validation gate, which shows the search region of a track. This validation region is defined in the body fixed image coordinates and used to check the validation of the estimated centroid points of available tracks. The dimension of the gate indicates maximum possible movement of the vehicle in one frame step time. This is a simple rectangular gate stretched on the orientation line of the vehicle. The dimensions of the gate are determined from the maximum forward and lateral velocity information in that partition obtained from the partition modeling.

The second gate is an ellipsoidal gate, which combines the position, area and orientation state variables of the track with different weights. The second gate is defined according to the covariance obtained from the Kalman filter. Adding the area and orientation states make the validation gate more robust to occlusion situations.

#### 5.4.2 Assignment Problem

The objective of the assignment problem is to minimize the cost of track-observation assignments. This section gives a brief introduction to assignment problem and presents methods for solving the assignment matrix. The general assignment problem can be defined as an optimization problem as follows [26]:

Given the matrix of elements  $a_{ij}$ ,

Find  $X = \{x_{ij}\}$  such that

$$C = \sum_{i=1}^n \sum_{j=1}^n a_{ij} x_{ij} \text{ is minimized subject to:}$$

$$\sum_i x_{ij} = 1 \quad \forall j$$

$$\sum_j x_{ij} = 1 \quad \forall i$$

where  $x_{ij} \in \{0,1\}$ .

An optimum solution to this problem is searched for the  $x_{ij}$  values that are either 0 or 1. The  $x_{ij}$ 's in target tracking applications are the observation-to-track pairs to be found that minimizes the cost.  $a_{ij}$  Parameters are defined as distances between tracks and the observations. To compute these distances, estimated measurements are compared with the real measurements. In this work, the measurements are centroid point, area and orientation of a vehicle detected at the current frame.

The hidden vehicles (vehicles tracked during hidden tracking) are also used in the assignment process. For this case, measurement estimates are obtained from hidden tracking algorithm.

The assignment process is done in three steps. The first step covers the cost calculation of each track-observation assignment. This cost  $a_{ij}$  is the weighted difference between the real and the estimated measurements. The weights are calculated using the variances of the state variables and the importance of the corresponding state variable in the assignment process. The state variables are the centroid points, the area and the orientation of the Kalman filter estimates of the tracks and measurements of the new observed vehicles.

The second step is to form an assignment matrix, which involves the cost values of the possible assignments. The assignment matrix is generated using the assignment costs of each track-observation assignment. Then, an ellipsoidal gate is used to eliminate unlikely track- observation assignments. Afterwards, the assignment matrix is constructed using the calculated assignments costs.

In the final step, a meaningful assignment is to be performed using the assignment matrix obtained in the previous step. Several approaches are proposed to solve the assignment problem.

The simplest method is the global nearest neighbor (GNN) algorithm [26], which is most widely used method for data association problems. In the GNN data association, the track, which is located at a minimum distance, is assigned with the observed vehicle.

However, a much more complicated algorithm that gives the optimal assignment is the auction algorithm. Auction algorithm is used to solve the assignment problem. Before discussing the auction algorithm, a brief explanation on global nearest neighbor (GNN) algorithm is given to clarify the assignment problem. After that, the auction algorithm is explained in details. Finally, some necessary modifications on the auction algorithm, which are required for adaptation of the auction algorithm to the developed multi-vehicle tracking system, are clarified.

#### **5.4.2.1 Global Nearest Neighbor (NN) Algorithm**

The GNN method is the simplest and probably the most widely applied method for data association. In the GNN data association, a track can be updated by one observation at most and an observation can be assigned to one track at most. Typically, the first step for the GNN data association is to form an assignment matrix [32]. The matrix elements are the normalized distances that are defined by the measurement residual vector and covariance.

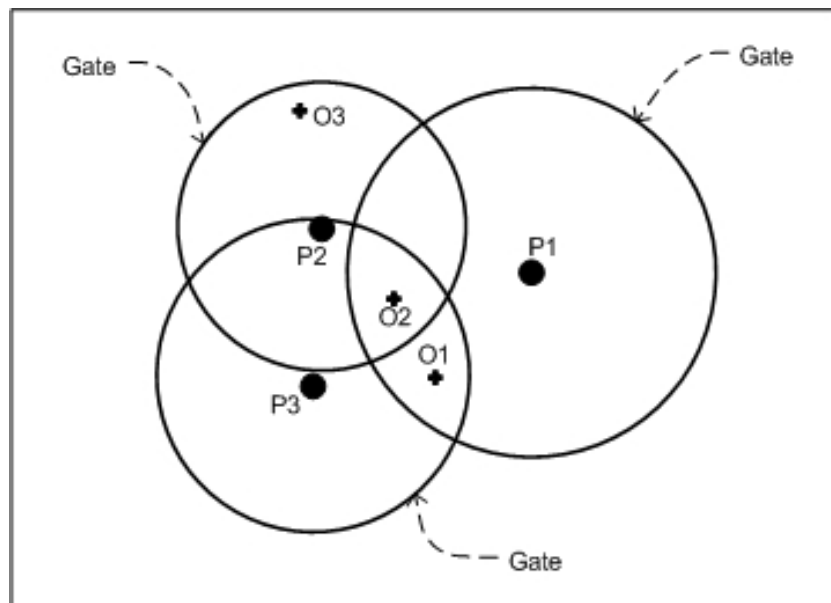
In order to assign observations to tracks the method based on searching the assignment matrix for the observation-to-track pair with the minimum distance and making the assignment. The next step is removing the row and column that corresponds to the observation-to-track pair assigned from the assignment matrix and repeating searching for the reduced matrix.

A major problem of GNN occurs when there is more than one observation in a track gate or an observation is in the gate of more than one track, which very probably produces a non-optimal assignment.

### 5.4.2.2 Auction Algorithm

In this thesis, the assignment problem is solved using the auction algorithm [26]. The aim of the auction algorithm is the same as the actual auction. The auctioneer intends to get the highest gain from the overall auction. It is an important point that we are not interested to maximize the gain of each assignment separately. The auction algorithm is concerned with the overall gain. In real auction, a product can be assigned only to one purchaser, but a purchaser can get more than one product. The auction algorithm used in this work allows only to one-to-one assignments. That means, a track can be updated by one observation at most and an observation can be assigned to one track at most.

The auction algorithm will be discussed on an association problem illustrated in the figure below. The validation gates of each track are given in the figure.



**Figure 5-13: Example of an Association Conflict Situation**

This is a three-by-three square assignment problem where an association conflict situation occurs. The first step of the observation-to-track assignment is to define the elements of this assignment matrix. Note that we will use the conversion that rows correspond to tracks and columns to observations.

**Table 5-4: Example of a General Assignment Matrix**

		OBSERVATIONS		
		O1	O2	O3
TRACKS	T1	$a_{11}$	$a_{12}$	$a_{13}$
	T2	$a_{21}$	$a_{22}$	$a_{23}$
	T3	$a_{31}$	$a_{32}$	$a_{33}$

The term  $a_{ij}$  is the gain of assigning the observation  $j$  to track  $i$ . The elements of the general assignment matrix can be calculated as

$$a_{ij} = G_{ij} - d^2_{ij} \quad (5.35)$$

where  $d^2_{ij}$  is the statistical distance between observation  $j$  and track  $i$ . The term  $G_{ij}$  is the gating limit for association of observation  $j$  to track  $i$ . The assignment matrix for example of the above figure is given as

**Table 5-5: Assignment Matrix for the Previous Example**

		OBSERVATIONS		
		O1	O2	O3
TRACKS	T1	9	6	X
	T2	X	3	10
	T3	8	4	X

After defining the elements of this assignment matrix, the auction algorithm is used to solve the assignment matrix. A brief description of the auction algorithm is given by Bertsekas [34]. Recently, Blackman and Popoli [26] examine the auction algorithm and give details about two major phases of the auction algorithm, namely, *the bidding phase* and *the assignment phase*.

*The bidding phase* consists of rising prices of the tracks to find the best track for each unassociated observation. As in real auction, the auction algorithm converges more quickly if a larger bid is used, but taking a larger bid may prevent the algorithm to achieve the optimal assignment. The aim of the bidding phase is to assign each unassociated observation with its second best track if another observation buys the best track. Therefore, it would be sensible to take the bid greater than the difference between the best and second best assignment gains for each observation. The bidding equation is given as

$$P_{ij} = P_{ij} + y_j + \epsilon \quad (5.36)$$

where  $y_j$  is the difference between the best and second best assignment gains for the observation  $j$ . The  $\epsilon$  in the bidding equation given above, which is used to pass over the gain of the best assignment, determines whether the assignment will be optimal or not. Therefore, with a view to achieve an optimal assignment, the selection of  $\epsilon$  is a critical point. As derived in [26] the total benefit of the final assignment is within  $n\epsilon$  neighborhood of the total benefit of the optimal assignment where  $n$  is given as

$$n = \min\{ \text{number of observations}, \text{number of tracks} \} \quad (5.37)$$

Therefore, to achieve an optimal assignment the selected  $\epsilon$  must satisfy the above condition

$$\epsilon < \frac{\Delta}{n} \quad (5.38)$$

where  $\Delta$  is defined as minimum possible difference between the unequal gains. If the benefits  $a_{ij}$  are all integers, that is, the maximum value of the minimum possible difference between the unequal gains is unity ( $\Delta = 1$ ), then the  $\epsilon$  in the bidding equation can be taken as

$$\epsilon < \frac{1}{n} \quad (5.39)$$

The *assignment phase* of the auction algorithm assigns a track to an observation and removes previous assignment if necessary. The assignment phase tries to find the best track for each observation and if there is an unassociated observation, the bidding phase continues raising the price of the track. This iterative process continues until a feasible assignment is reached with a maximum total benefit. A feasible assignment is the assignment of all observations to tracks.

### 5.4.2.3 Modified Auction Algorithm

In this study, the objective of the assignment algorithm is to decrease the cost of overall assignments. Therefore, we use cost table to decrease total cost instead of using gain table to increase gain. The objective of the original auction algorithm is to maximize the profit. Simple algebraic operations are used to transform the maximization to a minimization problem. For this purpose, the auction algorithm is modified to minimize cost of an observation-to-track assignment.

The initially designed auction algorithm was only applicable to square assignment matrices, but it was extended to handle rectangular assignment matrices well [34]. Since we are dealing with multi-target tracking problem in the traffic, the number of tracks and the observations may change during the tracking process. In addition, some vehicles may be lost or some new vehicles may occur in the scene. The association algorithm of a multi-target tracking system must cope with such problems. A solution of this problem may be adding a new track (a new row) for each observation and setting the assignment matrix entities of each observation-new track as maximum allowable gate dimension. The other entities is taken larger than the gate size and directly eliminated. If a new vehicle enters, it is expected that all of the entities of that column of the assignment matrix will be eliminated at the gating step and the new observation will not be assigned to any track in the scene. The modification brings the advantage of assigning this new vehicle to the new track generated for that observation.

The hidden vehicles (vehicles tracked with hidden tracking) are also used in assignment process. The measurement estimates, which are determined from hidden tracking process, are used in the assignment problem. The hidden vehicles are the tracks that could not be associated with an observation before. If a track cannot be associated with an observation, generally the association algorithm supposes that the vehicle exit the scene and finalize the track. However, if the hidden tracker system decides that this vehicle is still in the scene, then the tracking of that vehicle is continued as a hidden vehicle. An example of the modified assignment matrix is given in Table 5-6.

**Table 5-6: Modified Assignment Matrix for Auction Algorithm**

		OBSERVATIONS		
		O1	O2	O3
TRACKS	T1	$d_{11}$	$d_{12}$	$d_{13}$
	T2	$d_{21}$	$d_{22}$	$d_{23}$
	T3	$d_{31}$	$d_{32}$	$d_{33}$
	Hidden T1	$d_{H11}$	$d_{H12}$	$d_{H13}$
	Hidden T2	$d_{H21}$	$d_{H22}$	$d_{H23}$
	New T1	$G_{11}$	X	X
	New T2	X	$G_{22}$	X
	New T3	X	X	$G_{33}$

In this table,  $d_{ij}$  is the cost of assigning the observation  $j$  to track  $i$ . The term  $G_{ij}$  is the gating limit for association of observation  $j$  to track  $i$ , that is, maximum available assignment cost. Hidden  $T_i$  and New  $T_i$  denote respectively hidden track  $i$  and new track  $i$ .

### 5.4.3 Occlusion Problem

This section presents an occlusion reasoning method for detecting the occlusion of vehicles seen in a sequence of traffic images taken from a single roadside mounted camera. Occlusion is one of the most difficult problems in visual multi-target tracking applications. Occlusion problem occurs when there is overlapping of objects from the camera's point of view in a traffic-image sequence. An example of occlusion problem of two vehicles is shown in the figure below.

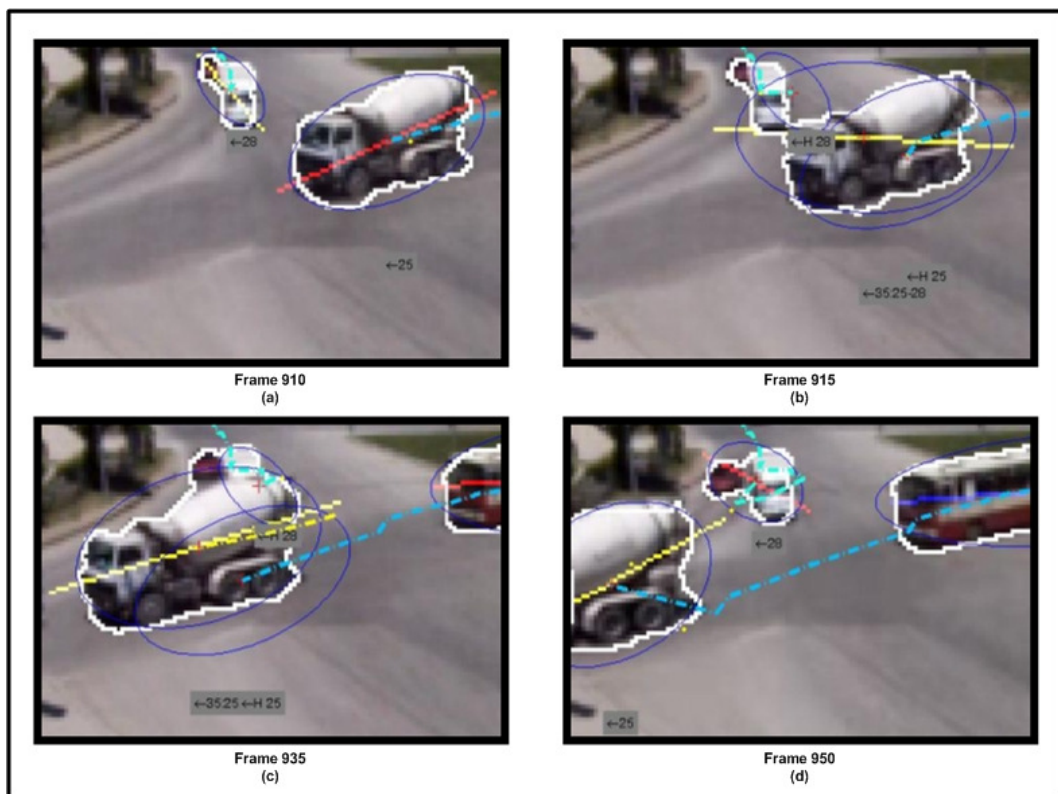


Figure 5-14: An Example of the Occlusion Problem

The occlusion situation given in Figure 5-14 presents an illustrative example of the dynamic occlusion problem. Vehicles occlude for a short time and then separate at a following frame. During the occlusion of the vehicles, the detection algorithm senses only one vehicle. Therefore, the association algorithm cannot assign the previous tracks of the merged vehicles with the new observed single vehicle. If the tracks are not matched to any observation then the tracking system cannot obtain any measurement. In this thesis, if an occlusion is detected then these tracks are considered as belonging to hidden vehicles and they are tracked using hidden vehicle motion models. When these vehicles are separated and detected as two different vehicles, each vehicle can be associated with its hidden track.

In this work, instead of resolving the occlusion problem we only consider to solve the occlusion detection problem. Before discussing the occlusion detection problem, a brief introduction about the occlusion reasoning algorithms, which is used in occlusion detection problem, is given below. The occlusion reasoning algorithms in a traffic scene can be introduced into two classes [6, 25]: explicit occlusion and implicit occlusion.

In the explicit occlusion case, multiple vehicles enter a scene separately and then merge into one moving object in the scene. In the implicit occlusion case, multiple vehicles enter the scene as merged from the beginning. Detection of explicit occlusion is easier than that of implicit occlusion since we can use the information of individual vehicles before occlusion occurs. A detailed explanation of the explicit occlusion and implicit occlusion reasoning algorithms can be found in [6]. In this work, two different occlusion detection algorithms are developed considering the explicit occlusion and implicit occlusion conditions.

If the detection algorithm determines an overlap of two vehicle regions, then the explicit reasoning algorithm creates a new hidden track for each merged vehicle. The merged object is also tracked using a new track label. The position estimates of the hidden vehicle tracker of two tracks are controlled according to motion of the merged object. If a separation occurs then the hidden tracks are associated with each separated object.

Since the hidden vehicle tracker maintains the tracking of the hidden vehicle without any real measurements, the accuracy of the tracking decreases for each motion estimation step of the hidden vehicle. The hidden track of an overlapped vehicle is finalized when a maximum tracking period of that hidden vehicle is reached. The maximum tracking period of a hidden vehicle changes according to occlusion type, path model of the occluded vehicles and average frame length of that path model. This process may cause to remove a hidden track before separation occurs but also avoid the system from false hidden track-observation assignments.

The explicit occlusion reasoning process is illustrated in Figure 5-15 and an explicit occlusion example is shown in Figure 5-16. In Figure 5-15, two vehicles, namely, A and B, occlude at frame 4 and the merged object is labeled as "C". After the separation of two vehicles, the explicit occlusion reasoning algorithm assigns the separated vehicles with the previous tracks.

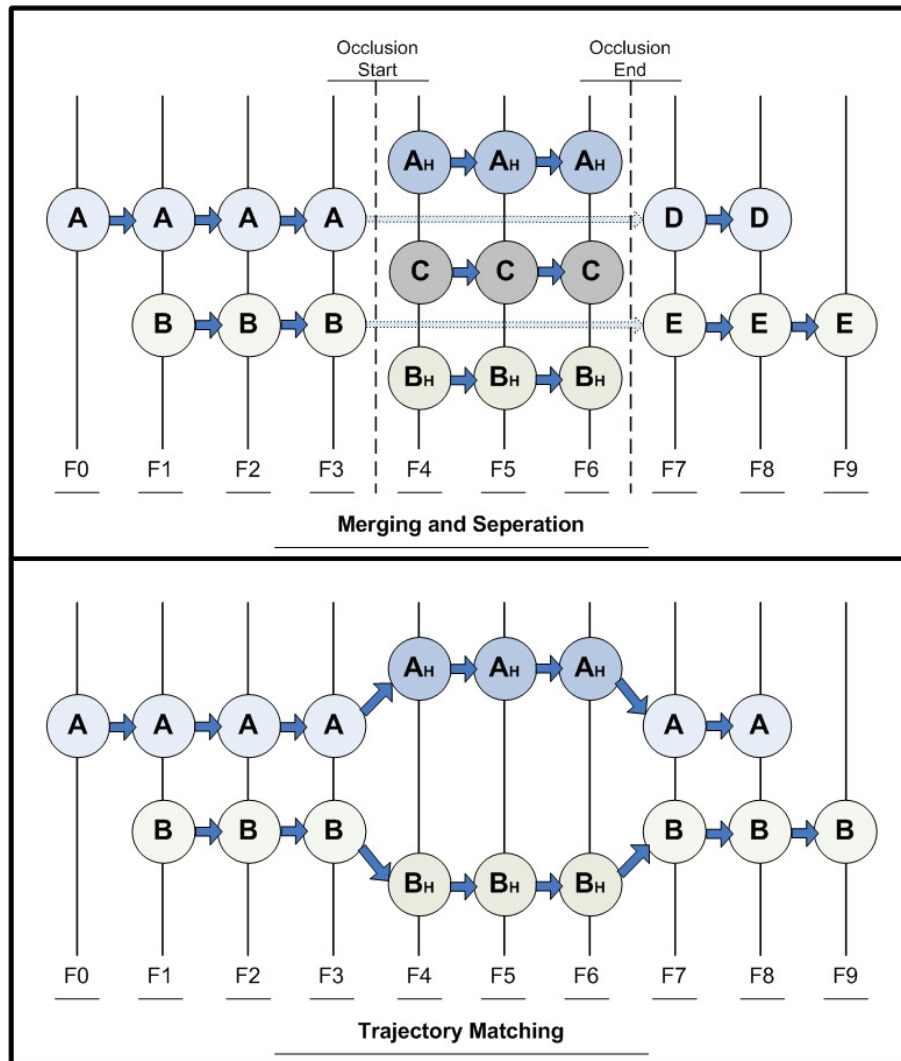
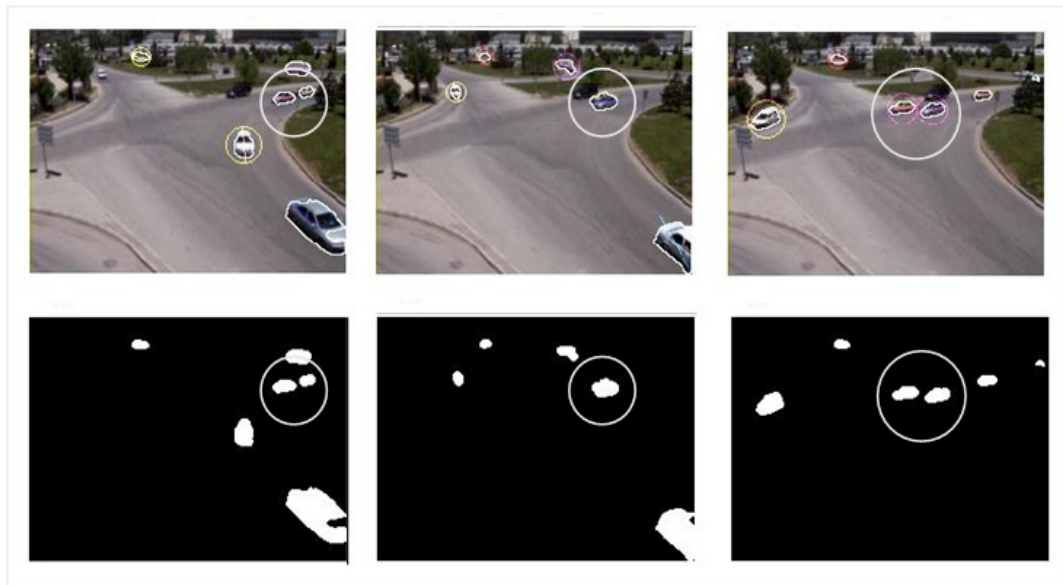


Figure 5-15: Explicit Occlusion Reasoning



**Figure 5-16: Explicit Occlusion Example**

Implicit occlusion is the entrance of occluded vehicles into the scene. In the case of implicit occlusion, we cannot obtain information of individual vehicles in the occluded region beforehand. Thus, the occlusion can only be detected if a separation occurs in the scene. In the case of a separation, the assignment algorithm selects one of the separated vehicles and associates it with the previous track. The other vehicle will be labeled as a new vehicle if it is not assigned to a different track. After labeling new tracks, the labels of the separated objects are loaded to motion parameters of the initial merged vehicle. The detection of such a separation is discussed in appearance reasoning algorithm of assignment verification section. The implicit occlusion reasoning process is illustrated in Figure 5-17 and an implicit occlusion example is shown in Figure 5-18.

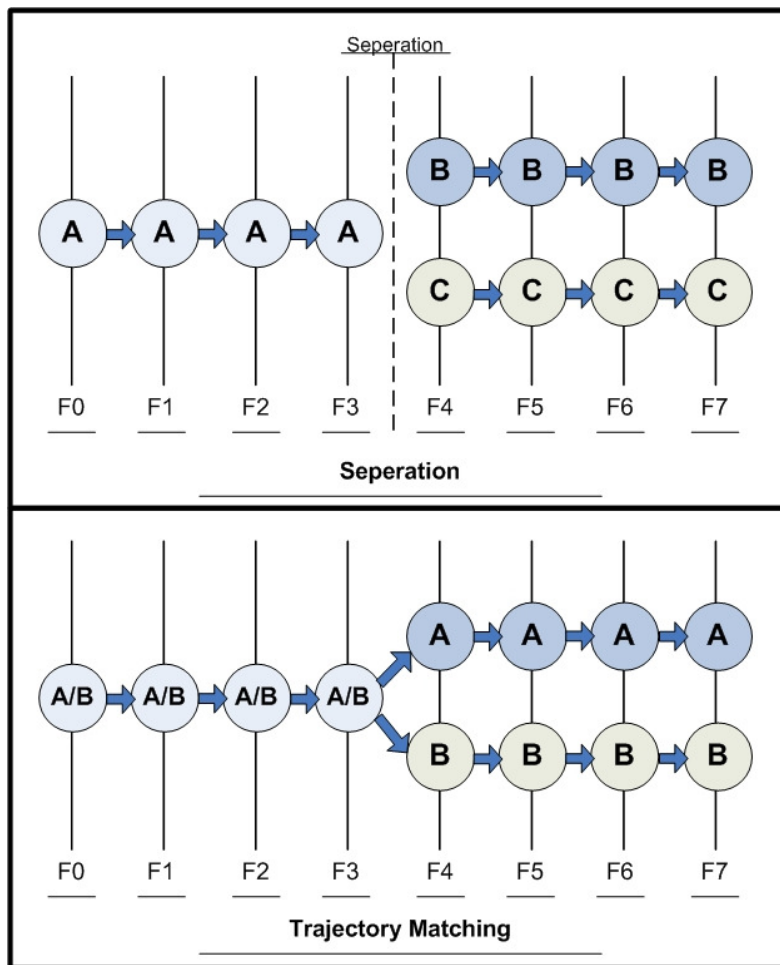


Figure 5-17: Implicit Occlusion Reasoning



**Figure 5-18: Implicit Occlusion Example**

#### **5.4.4 Assignment Post Processing**

This process is used to check the assignment results of the association algorithm and determine unassigned tracks and observations. This process has two main aims. It is used both for occlusion detection and for track initialization and finalization. By this processing, the vehicles, which leave the scene, are removed and track for new vehicles are initialized. Occlusions are determined from the results of assignment process. Each occlusion causes an unassigned track and each separation after occlusion causes an unassigned measurement. By some reasoning described below, occlusion/separation decisions are taken using this approach. The assignment post processing has two main parts, namely, new object handling and object loss handling.

#### 5.4.4.1 Object Loss Handling

If a vehicle that is labeled as a track in the previous frame is not detected at the current frame then that vehicle is called a *disappeared vehicle*. This indicates that no measurement is assigned to the track of this vehicle in the current frame. The reason of the vehicle disappearance is searched in the traffic image sequences and vehicle disappearance conditions, which are categorized into three classes, are given in Table 5-7.

**Table 5-7: Disappearance Conditions**

Case	Condition	Explanation
1	Exited	Vehicle leaved the scene
2	Merged	Vehicle is overlapped with another vehicle
3	Lost	Detectors can't found the vehicle

The first case occurs when the estimated position of the disappeared vehicle is outside the considered scene or near the scene borders. A border is assumed to have 20-pixel width. In the path extraction step, we assign entrance and exit regions of the scene. These regions are also used in vehicle disappearance reasoning. If the vehicle disappears in an exit region, then that track is finalized and removed.

The second case occurs when two vehicle regions overlap in the scene. The vehicle detection system determines one vehicle in that region instead of two. The overlapped vehicles are reported to occlusion reasoning system. The occlusion reasoning system determines the occlusion type and applies the suitable occlusion reasoning algorithm.

If these two cases are not valid then it means that the vehicle is lost in the scene. This situation occurs when the vehicle detection system cannot determine the vehicle. A reason for this situation may be the occlusion of the vehicle by a background object. For this case, the vehicle is labeled as hidden vehicle and tracked using hidden vehicle tracker. The flowchart of the algorithm for object loss handling is given in Figure 5-19.

#### 5.4.4.2 New Object Handling

The second problem that we deal in this part is appearance of a new vehicle in the scene. If a detected vehicle cannot be assigned to an existing track then that vehicle is called an *appeared vehicle* and initially labeled as a new vehicle.

The reason of the appearance of a new vehicle is searched in the traffic image sequences and new vehicle appearance problem is categorized into three classes that are given in Table 5-8.

**Table 5-8: Appearance Conditions**

Case	Condition	Explanation
1	Entered	Vehicle just enters the scene
2	Separated	Occluded vehicles are separated
3	Found	A previously lost vehicle is found

In the first case, the position of the new appeared vehicle is checked at the current frame and if it is in the border region it is decided that the vehicle is entered into the scene in this frame. In the path extraction step, we assign entrance and exit regions to the scene. These regions are also used in detecting new entered vehicles. If a vehicle appears in the entrance region, then a new track is initialized.

The second case occurs when two overlapping vehicles separate in the scene. The detector determines two vehicles in the validation region of the previous track and the assignment algorithm associate one of the detected vehicles with the previous track. Thus, the other detected vehicle is labeled as a new vehicle and a separation is reported to occlusion reasoning system. The occlusion reasoning system determines the occlusion type and applies the suitable occlusion reasoning algorithm.

If the previous two cases are not valid, then it means that the vehicle is appeared in the middle of the scene, which can take place because of the following two reasons. The first possibility is that, the vehicle had not been seen up to the current frame so the detector could not detect it. Then, this appeared vehicle remains as a new vehicle and a new track is initialized. The second possibility claims that this new appeared vehicle has seen in its history but lost more than one frame up to the current frame. For such a case, the new appeared vehicle can be assigned to a hidden track. A possible hidden track assignment of the appeared vehicle is searched and if a valid association occurs then the appeared vehicle is assigned to the hidden track and the hidden track is finalized. Figure 5-20 illustrates the flowchart of this appearance algorithm.

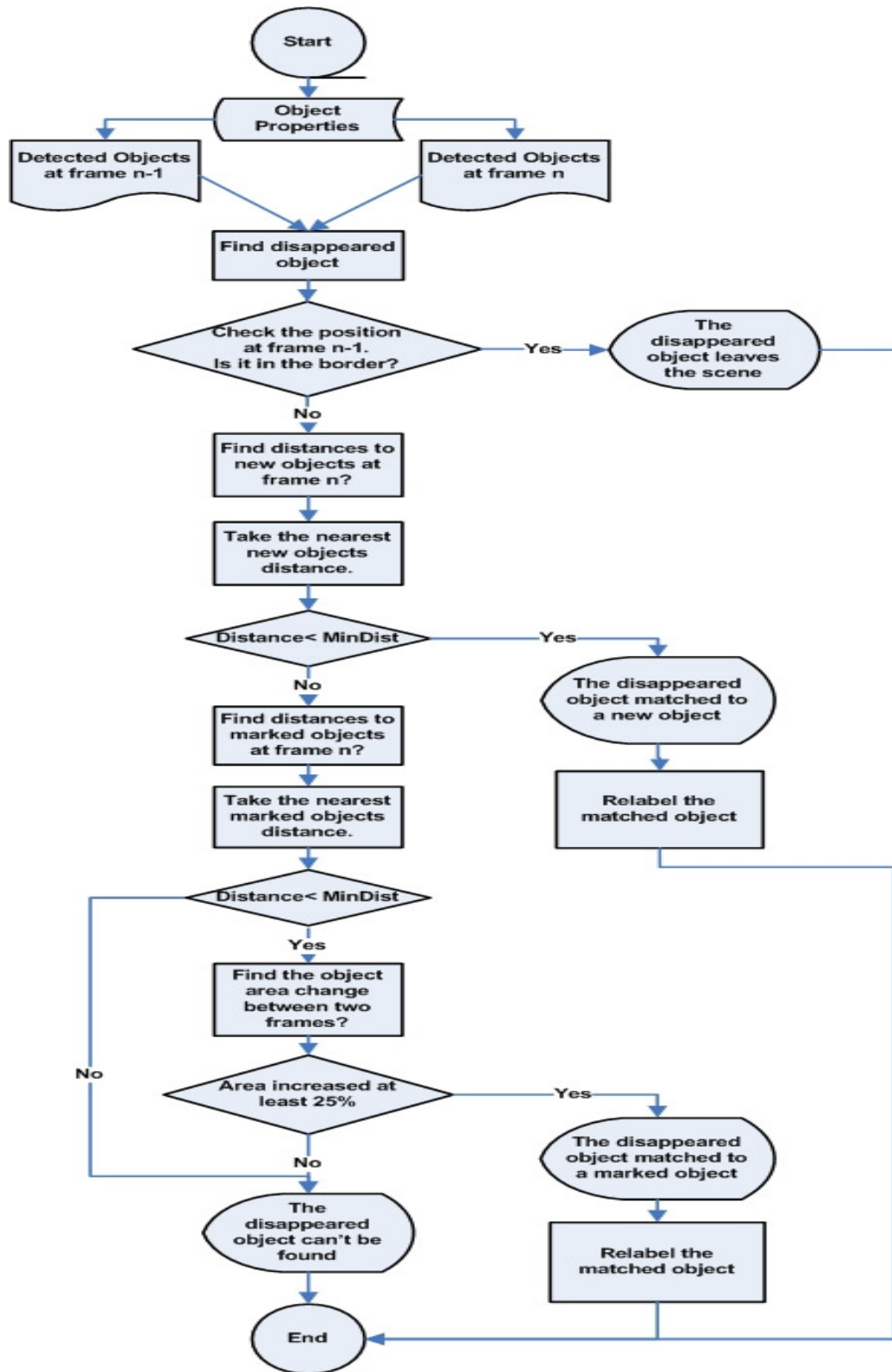


Figure 5-19: Disappearance Reasoning Algorithm Flowchart

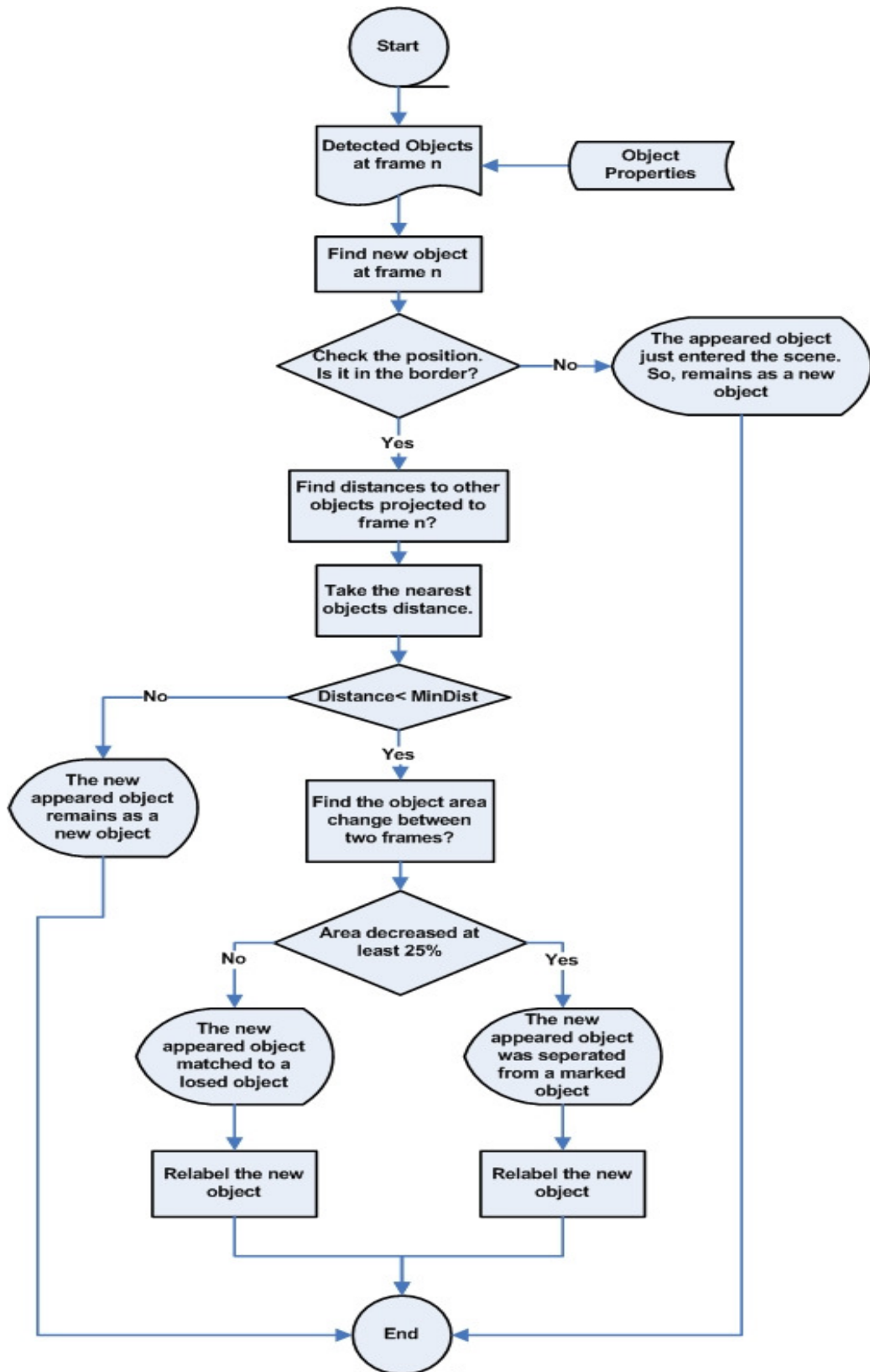


Figure 5-20: Appearance Reasoning Algorithm Flowchart

## CHAPTER 6

### IMPLEMENTATION AND SIMULATION

The developed system is tested for different scenarios to analyze reliability and robustness. We have simulated the proposed algorithm to evaluate the vehicle detection and tracking system performances on real-world image sequences. We have also examined the proposed occlusion reasoning algorithm for explicit and implicit occlusion scenarios. This chapter firstly explains the implementation of the proposed system on computer environment. Afterwards, the simulation results for real image sequences are given.

#### 6.1 Implementation Issues

The proposed system is implemented using a program written in Matlab programming language. The off-line version of the proposed system is tested on computer environment. The test images are all real-world image sequences obtained from a traffic junction by using a digital camera placed on a bridge. The video obtained from the digital camera is segmented to image frames and stored at the computer environment. Each image frame has bitmap (.bmp) format of 240x180 pixels/frame.

In the computer verification tests of the proposed system, it is determined that the off-line version of the proposed system can process real-world images sequences near real-time, if they are acquired at 5 fps. The acquiring process of the input image sequences and the loading process of the output image sequences are excluded from this time performance calculation.

## 6.2 Simulation Results

In this section, the detection and tracking system results for real-world image data sequences are introduced. The vehicle detection and tracking system is tested in an uncontrolled junction with high traffic load. The scene inspected in this study is depicted in Figure 6-1.



**Figure 6-1: Scene Inspected in this Study**

The vehicle tracking system uses 1250 frame (255 second at 5 fps), obtained at 240x180 pixel sizes, as train data sequence. 324 vehicle trajectories are extracted from this train data and 126 of them are used at scene model training phase. During the training phase, each separate road partition is trained using 20 to 50 tracks. The average tracking period of vehicles in the scene is nearly 32 frames, which corresponds to approximately 6 seconds at 5fps.

The performance test of the proposed system is performed on 1575 frames (315 second at 5fps) which are obtained from same scene with the same camera at the same position. Since we want to measure the performances of the vehicle detection and tracking systems, vehicle ground truth was manually defined for test data sequence. In the tracking performance tests, 99 vehicles moving on the road paths are examined and the vehicles detected out of the path regions are not considered.

The vehicle count estimates of the traffic parameter extraction system for different path models are given in Table 6-1. The vehicle count estimates of the proposed system are compared with manual vehicle count. The proposed system works well with 94% vehicle count performance. The vehicle count errors occur due to incorrect track associations of the occlusion reasoning algorithm.

**Table 6-1: Results of the Vehicle Count**

<b>Path Model</b>	<b>Manual Inspection</b>	<b>Proposed Algorithm</b>	<b>Performance</b>
Path 1	34	36	94.1%
Path 2	36	39	91.7%
Path 3	29	30	96.6%
<b>TOTAL</b>	<b>99</b>	<b>105</b>	<b>94%</b>

Table 6-2 gives the target tracking performances of different target tracking and association methods used in this thesis. In addition, this table demonstrates the development phases of the vehicle tracking and association methods of the proposed system. Each row of Table 6-2 indicates a development step of the proposed system and the tracking performance reached at that step. The first step uses Kalman tracker (KT) with basic motion model and the association problem is solved using nearest neighbor (NN) with basic occlusion reasoning (OR). This tracker cannot deal with difficult maneuvers like turn motion and acceleration. This problem is solved using a multi-model Kalman tracker (MMKT), which uses four different advanced motion models explained in Section 5.2.2 of this thesis. The association problem is one of the main issues in multi-target

tracking applications. For this reason, we examine the mostly used association methods in the literature and select auction algorithm (AA) for track-observation assignment problem. The tracking performance reaches a value of 80% by using MMKT with auction algorithm.

**Table 6-2: The Development of the Proposed Method**

Step	Tracking Method	Association Method	Tracking Error	Performance
1	KT	NN + OR	33	67%
2	MMKT	NN + OR	23	77%
3	MMKT	AA + OR	20	80%
4	MMKT+MSPT	AA + OR	17	83%
5	MMKT+MSPT+HVT	AA + OR	7	93%

<b>KT</b>	Kalman Tracker	<b>NN</b>	Nearest Neighbor
<b>MMKT</b>	Multi-Model Kalman Tracker	<b>OR</b>	Occlusion Reasoning
<b>MSPT</b>	Markov Scene Partition Tracker	<b>AA</b>	Auction Algorithm
<b>HVT</b>	Hidden Vehicle Tracker		

The scene, which is inspected in this study, is assumed stationary. This allows us to collect and use a-priori information for further processes about the scene. The Markov scene partition tracker (MSPT) is a Markov chain based tracker, which uses the a-priori information collected for the scene. The performance of the tracking process is increased to 83% by using a MSPT together with a MMKT.

The developed multi-model Kalman tracker cannot maintain tracking during occlusion due to measurement need of Kalman filtering. Therefore, a hidden vehicle tracker (HVT), which produces pseudo measurements for MMKT during occlusion, improves the performance of the tracking system. The auction algorithm and the occlusion reasoning are modified to handle hidden vehicle tracking. The final phase of the proposed system is reached a tracking performance of 93%, which is quite satisfactory for such a complicated traffic case.

The vehicle tracking system proves its reliability for the no occlusion case. In all test data for non-occlusion case, no target loss is observed. The simulation results give us that the hidden target tracking method gives successful results for occlusion handling and long-term target loss scenarios. Another important observation on the performance of the developed system can be the improvement of the system performance using Markov scene partitioning tracking. Markov scene partitioning tracking especially provides fast estimation of the motion model and checks the partition transitions obtained from Kalman tracker at long-term periods.

The performance of the occlusion detection and reasoning algorithms are given in Table 6-3. Most of the tracking errors occurred at long occlusion cases. The hidden track of an overlapped vehicle is finalized when a maximum tracking period of that hidden vehicle is reached. This cause to remove a hidden track before separation occurs. The remaining tracking errors occur at such cases as when two or more vehicles occlude and one of these vehicles changes its motion model during occlusion. The solution of this problem is proposed as a future work.

**Table 6-3: Occlusion Detection and Reasoning**

	<b>Explicit Occ.</b>	<b>Implicit Occ.</b>	<b>Total</b>
Number of Occlusions (Actual)	51	23	<b>74</b>
Number of Occlusions (Detected)	49	23	<b>72</b>
Occlusion Detection Error	2	0	<b>2</b>
Occlusion Detection Performance	<b>96%</b>	<b>100%</b>	<b>97%</b>
Number of Occlusions (Solved)	42	23	<b>65</b>
Occlusion Reasoning Error	7	0	<b>7</b>
Occlusion Reasoning Performance	<b>86%</b>	<b>100%</b>	<b>90%</b>

Table 6-3 indicates that all tracking errors arise from explicit occlusion situations. Therefore, the explicit occlusion scenarios are examined in Table 6-4 by considering two main properties of occlusion: number of occluded vehicles and type of occlusion (partial or full).

**Table 6-4: Explicit Occlusion Reasoning**

Number of Occlusions in a Track	Partial Occlusion		Full Occlusion		Total	
	Number of Errors/Vehicles	Performance	Number of Errors/Vehicles	Performance	Number of Errors/Vehicles	Performance
<b>0</b>	-	-	-	-	<b>0 / 50</b>	<b>100%</b>
<b>1</b>	1 / 27	96%	2 / 9	77%	<b>3 / 36</b>	<b>92%</b>
<b>2</b>	0 / 3	100%	2 / 5	60%	<b>2 / 8</b>	<b>75%</b>
<b>3+</b>	1 / 3	67%	1 / 2	50%	<b>2 / 5</b>	<b>60%</b>
<b>TOTAL</b>	<b>2 / 33</b>	<b>94%</b>	<b>5 / 16</b>	<b>69%</b>	<b>7 / 99</b>	<b>93%</b>

The simulation results of the occlusion detection and reasoning algorithms for some occlusion scenarios occurred in real-world image sequences are demonstrated in Figure 6-2, Figure 6-3 and Figure 6-4. These occlusion scenarios are depicted as key frames of output image sequences. To increase visibility of the output image sequences, key frames are enlarged (zoomed) and focused on the occlusion region. The simulation results illustrates that, the proposed multi-vehicle tracking system is capable of tracking a target in a complex environment and able to overcome occlusion and inaccurate detection problems as well as abrupt changes in its trajectory.

Figure 6-2 illustrates the maneuvering turn motion of two vehicles, which are using the path 1 shown in Figure 4-12. During the turning motion of the vehicles, they are exposed to partial occlusion and their motion is tracked using hidden vehicle tracker. At frame 1355 (b), only one moving object is determined in the detection phase since they are occluded and the occlusion reasoning algorithm detects the explicit occlusion of track 57 and 62. The merged objects are

assigned as a new hidden track and motion estimation process is continued using the previous motion parameters and statistical partition parameters of the scene. The Kalman tracker uses the pseudo measurements of the hidden vehicle tracker during state estimation process. The hidden tracking of the vehicle is illustrated in Figure 6-2 (b, c, d and e). At frame 1420 (e), two vehicles are separated and the detection system extracted two separate vehicle regions. The occlusion-reasoning algorithm detects the separation and assigns each separated vehicle to the corresponding hidden track. Finally, the track of the merged object is finalized and deleted.

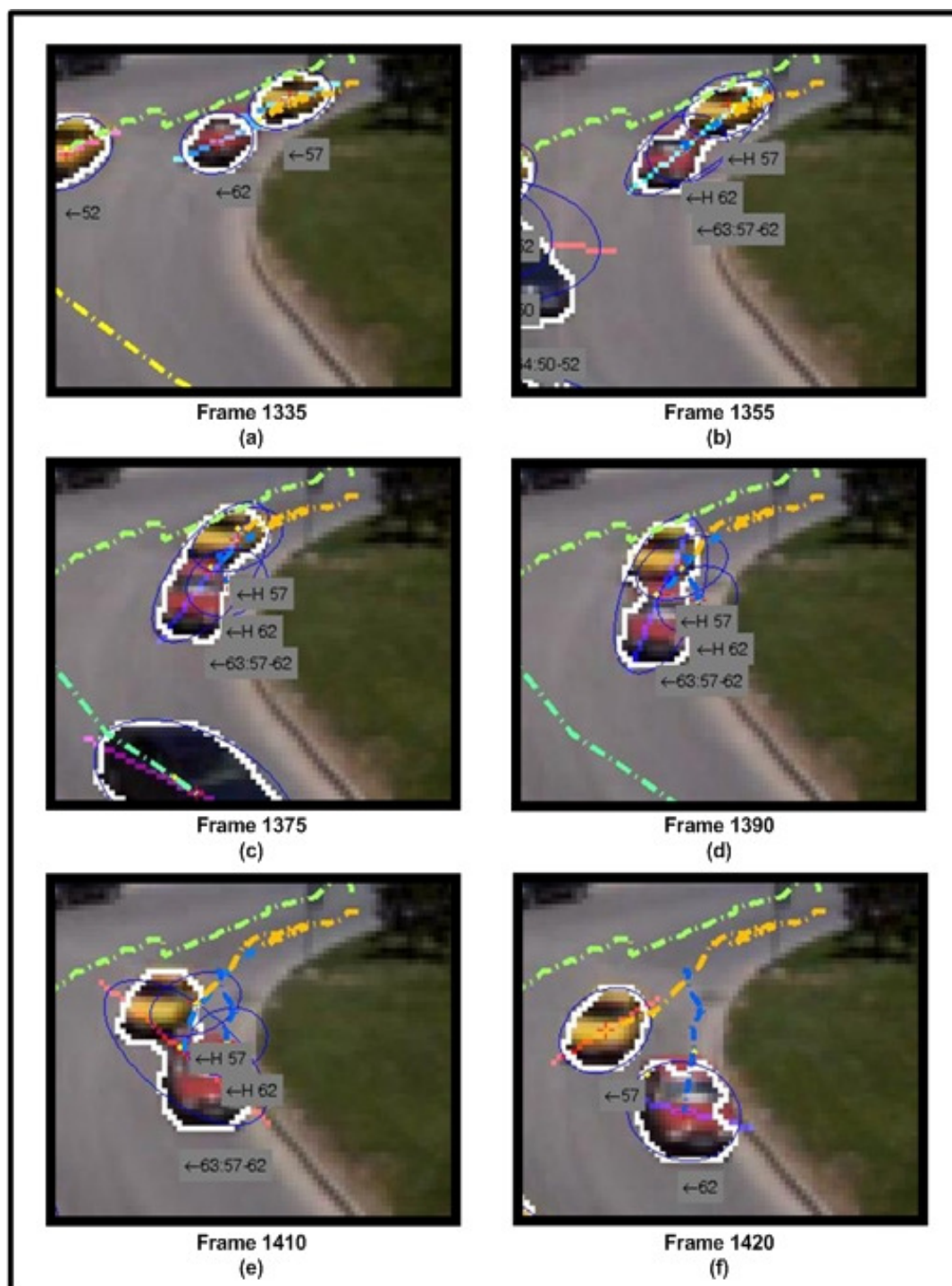
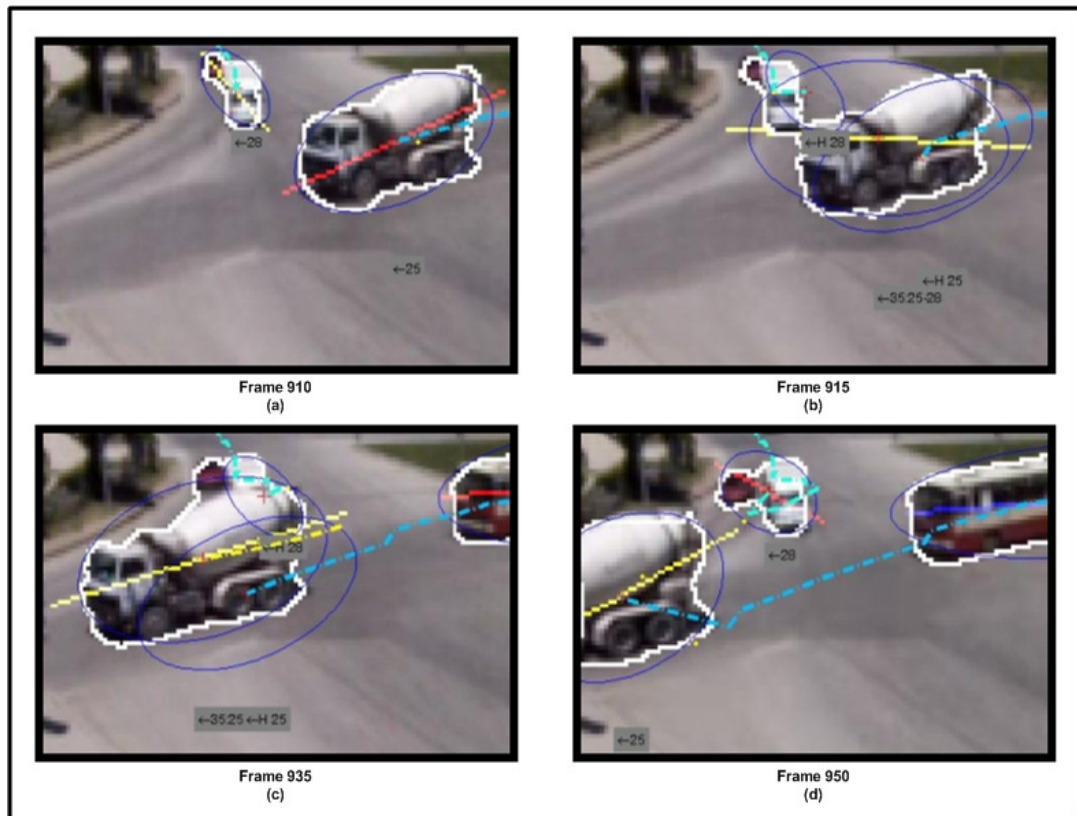


Figure 6-2: Key Frames 1 – Occlusion Detection and Reasoning



**Figure 6-3: Key Frames 2 – Occlusion Detection and Reasoning**

Another occlusion example is given in Figure 6-3. In this example, two vehicles, which are entered the scene at the same time from different gates, are tracked using the track label 28. Since, they are not separated during selected frame period, they are considered as a single vehicle in this example. The small lorry, which is labeled as 28, is traveling on path 3 and the cement lorry, which is labeled as 25, is traveling on path 2. These vehicles come across in the middle of the junction and an occlusion occurs in (b). During occlusion, the merged object is tracked using a new track (35) and the tracks of the occluded vehicles are continued using hidden tracks. In the final frame of this example, the merged objects are separated and a successful assignment is achieved.

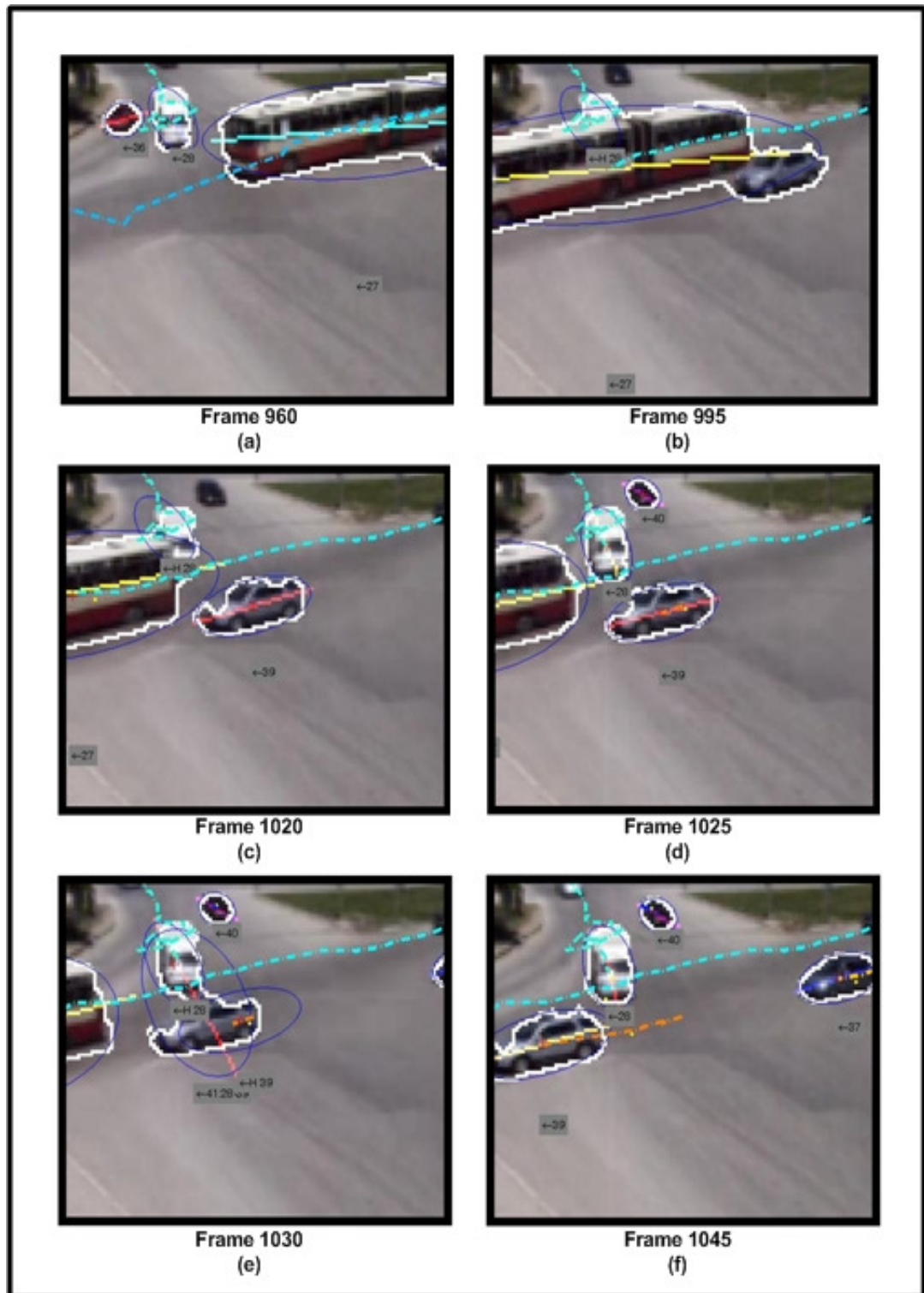


Figure 6-4: Key Frames 3 – Occlusion Detection and Reasoning

Figure 6-4 shows a successful tracking of implicit and explicit occlusions. There are two implicit (28-36 and 27-39) and two explicit occlusion (28-27 and 28-39) in this figure. We have already examined the track 28 in the previous example. In Figure 6-4, the merged object 28 is separated and two new objects are labeled according to implicit occlusion reasoning algorithm explained in Section 5.4.3. The same process is repeated for track 27. The first explicit occlusion is occurred between track 28 and 27. The small lorry (track 28) traveling in path 2 is overlapped with the bus (track 27) traveling in the path 3 at frame 995. The occlusion reasoning algorithm performs a successful separation and assignment of these vehicles at frame 1025. Afterwards, the track 28 is overlapped with the track 39, which was separated from merged track 27 just two frames ago. This occlusion is also finalized with an accurate trajectory matching. There is an important detail in the first explicit occlusion example. After the occlusion of track 28 and 27 detected, the track 28 is continued as a new hidden track but the track 27 is assigned to the merged object. This is a special case of the explicit occlusion reasoning algorithm used for small vehicle-large vehicle occlusions. In this example, since the small lorry is much smaller than the bus, the occlusion does not affect the vehicle localization and the trajectory so much. This approach provides a decrease in the size of the hidden tracks, which simplifies the tracking and association process.

## CHAPTER 7

### CONCLUSIONS

In this study, a video tracker system for traffic monitoring and analysis is developed. The developed system detects moving vehicles in the scene and maintains tracking of these vehicles during they are in the scene. This provides to perform traffic analysis about the scene inspected in this study. In this study, a new algorithm is proposed to solve the multi-vehicle tracking problem that can deal with problems such as occlusion, short period object lost or inaccurate object detection. The block diagram of the proposed video tracker system is illustrated in Figure 7-1.

The developed system is a system that collects a-priori information about the junction, models that a-priori information and uses it in the tracking stage. In addition, the a-priori information is analyzed to get traffic parameters, such as vehicle count, vehicle density and average vehicle speed. An important property of our system is that the a-priori information collecting stage of the system is done automatically. Thus, the whole system can work autonomously.

The algorithms were tested for several scenarios to examine the reliability and robustness of the proposed methods in the problem of multi target tracking in an un-controlled junction with high traffic load. The results indicate that the developed system is capable of segmenting vehicles from the background accurately and maintain tracking while the vehicles are in the scene. The occlusion detection and reasoning systems worked well with the hidden vehicle tracker in both implicit and explicit occlusion cases. As a result, we can claim that, the proposed multi-vehicle tracking system is capable of tracking a target in a complex environment and able to overcome occlusion and inaccurate detection problems as well as abrupt changes in its trajectory.

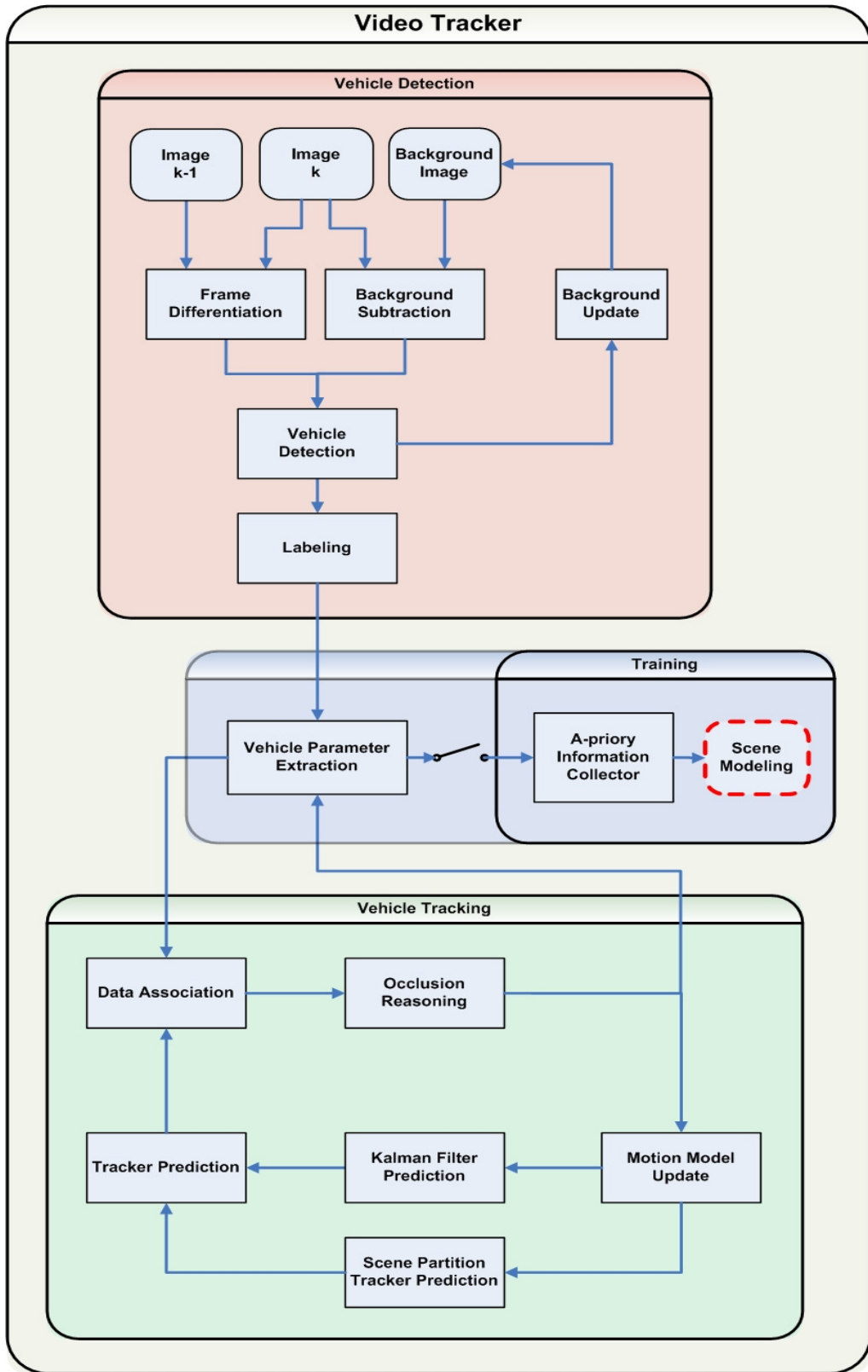


Figure 7-1: Block Diagram of the Proposed Video Tracker System

## 7.1 Contributions

The major contributions of this thesis can be summarized as follows:

### ***Vehicle Detection***

- Implementation of a system for moving object segmentation in Matlab by combining the advantages of both frame differentiation and background subtraction methods. The performance of the frame differentiation method is increased using the edge information of the vehicles for accurate vehicle localization and shape.

### ***Scene Modeling***

- Definition of partition modeling, which gives the a-priori information about the scene.
- Definition of Markov model of the scene, which gives the transition probabilities between scene partitions. By extracting the scene partition probabilities, we can estimate the motion of a vehicle between partitions.

### ***Estimation***

- Development of motion models for each path and each vehicle that will be used in multi-model Kalman tracker. This improves the estimation performance of the tracks.
- Definition of a Markov scene partition tracker, which uses scene partition transition probabilities to estimate the motion of a vehicle and checks the validation of Kalman tracker especially for cases like occlusion or detection system error.
- Definition of a hidden vehicle tracker, which is used to estimate the pseudo measurements of the Kalman tracker. The hidden vehicle tracker is used when the measurements cannot be obtained from the vehicle because of occlusion or detection system errors.

## ***Association***

- Adaptation of the auction algorithm according to hidden tracking and occlusion reasoning algorithm needs. The adapted association algorithm can cope with occlusion problems.
- Adaptation of the occlusion reasoning algorithm according to hidden vehicle tracking systems to cope with difficult occlusion problems.
- Development of the occlusion detection algorithm, which is based on searching the assignment results of the tracking system. The main idea behind the occlusion detection is to find disappeared-appeared vehicles, which are not assigned in the association algorithm.

## **7.2 Future Work**

Based on this thesis work, several extensions might be done. Some of these extensions are related to increase the tracking performance of the system and the others are related to developing new tools to adapt the system for new application areas.

In the future, we would like to extend our current work to extract other traffic events under more challenging capturing conditions such as capturing at night or in the rain. In addition, the traffic parameter extraction system can be developed to get more traffic information from the scene. In this thesis, we develop a detailed vehicle parameter extraction system, but in the parameter extraction section of this thesis, we only give the parameters that are used in the tracking system. These parameters include average color information and invariant moments. The average color information can be used for color vehicle tracking and invariant moments can be used in vehicle recognition system. As a future work, the above mentioned systems can be developed using these parameters and added to the proposed system. The extracted vehicle and traffic parameters can be also used to aid the decision making of an automated traffic management system.

Improvements on association phase of the tracker are also possible to increase accuracy of the association. Especially for occlusion of two vehicles traveling different directions, multi hypothesis tracking (MHT) approach can be used to increase the association accuracy. This can be done by estimating the path models after the assignment of the separated vehicles and if a path model conflict is determined then the assignment could be corrected.

A future work of this project is to develop a system that can track vehicles in real-time in order to detect situations giving rise to accidents. This would involve using the trajectories of the vehicles over time to predict future trajectories and then analyzing these trajectories. As a suggestion, the Markov scene partition tracker can be used to predict future trajectories of the vehicle and an accident can be predicted if the trajectories of two vehicles coincide in a partition. In addition, the proposed system can also be used to identify stalled vehicles and optimize traffic flows in the scene.

In this study, we use uncalibrated images for vehicle tracking. The calibration of the image sequences will improve the performances of the traffic parameter extraction and vehicle tracking systems. The traditional calibration approaches uses reference points of the image, which needs user assistance. As a future work, a new calibration approach can be proposed for the metric calibration of vehicle parameters using the partition modeling. The partition parameters of each scene partition were obtained in partition modeling. The main approach is to derive a relationship between two partitions according to partition parameter changes. This relation gives us the parameter metric change of each scene partition translation. Developing such an extensive and reliable metric transform matrix for partition translations may improve the performances of the traffic parameter extraction and vehicle tracking systems.

## REFERENCES

- [1] Arslan, A.E. "Visual Tracking With Group Motion Approach" Ms. Thesis Submitted To Grad. Sc. Of Nat. and Applied Science of The METU, 2003.
- [2] Yilmaz, A., Javed, O., Shah, M., "Object Tracking: A Survey" ACM Comp. Surveys. Vol. 38, No. 4, Article.13, 2006.
- [3] Comaniciu, D., Ramesh, V., Meer, P., "Kernel-Based Object Tracking" IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 25, pp.564-575, 2003.
- [4] Trucco, E., Plakas, K., "Video Tracking: A Concise Survey" IEEE Journal of Oceanic Eng. Vol. 31, No:2, pp. 520-529, 2006.
- [5] Cavallaro, A., Steiger, O., Ebrahimi, T., "Tracking Video Objects In Cluttered Background" IEEE Trans. on Circuits and Systems for Video Tech. Vol.15, No.4, pp 575-584, 2005.
- [6] Jung, Y. K., Ho, Y. S., "Traffic Parameter Extraction Using Video-Based Vehicle Tracking" IEEE Int. Conf. Intell. Transport. Systems, pp. 764-769, 1999.
- [7] Gil, S., Milanese, R., Thierry, P., "Feature Selection For Object Tracking In Traffic Scenes" SPIE Int. Symposium on Smart Highways, Boston, Massachusetts, 1994.
- [8] Beymer, D., McLauchlan, P., Coifman, B., Malik, J., "A Real-Time Computer Vision System For Measuring Traffic Parameters" Procs. of the Int. Conf. on Computer Vision (ICCV), 1999.
- [9] Koller, D., "Moving Object Recognition and Classification based on Recursive Shape Parameter Estimation", In Proc. of the 12th Israeli Conf. on Artificial Intelligence, Computer Vision, and Neural Networks, pp. 359-368, Tel-Aviv, Israel, December 27-28, 1993.
- [10] Koller, D., Weber, J., Huang, T., Malik, J., Ogasawara, G., Rao, B., Russel, S., "Toward Robust Automatic Traffic Scene Analysis in Real-time," in Proc. Int. Conf. Pattern Recognition, Israel, pp. 126–131, 1994.
- [11] Koller, D., Weber, J., Malik, J., "Robust Multiple Car Tracking with Occlusion Reasoning" Technical Report UCB/CSD-93-780, University of California at Berkley, October, 1993.

- [12] Kilger, M., "Video-Based Traffic Monitoring", Proc. Int. Conf. Image Processing and its Applications, pp.88-92, Maastricht, The Netherlands, April 1992.
- [13] Jain, R., Kasturi, R., Schunck, B.G., Machine Vision, 1995.
- [14] Ridler, T.W., Calvard, S. "Picture Thresholding Using an Iterative Selection Method", IEEE Trans. System, Man and Cybernetics, SMC-8, 630-632, 1978.
- [15] Davies, E. R., Machine Vision, Morgan Kaufmann Pub., 3rd Edition, 2005.
- [16] Matlab 6.5, Image Processing Toolbox, User Guide, Version 5.0.
- [17] Sharifi, M., Fathy, M., Mahmoudi, M. T., "A Classified and Comparative Study of Edge Detection Algorithms", IEEE Proc. of the International Conference on Information Technology: Coding and Computing (ITCC'02), 2002.
- [18] Ziou, D., Tabbone, S., "Edge Detection Techniques - An Overview." TR-195, Département de Math et Informatique, Université de Sherbrooke, Québec, Canada, pp: 1-41, 1997.
- [19] Canny, J.F., "A Computational Approach to Edge Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, V:8, pp:679-698, November, 1986.
- [20] Trucco, E., Verri, A., Introductory Techniques for 3-D Computer Vision, Prentice Hall, 1998.
- [21] Piccardi, M., "Background Subtraction Techniques: A Review", IEEE International Conference on Systems, Man and Cybernetics, pp. 3099-3104, 2004.
- [22] Friedman, N., Russell, S., "Image Segmentation in Video Sequences: a Probabilistic Approach," in Proc. 13th Conf. Uncertainty in Artificial Intelligence, pp. 1-3, 1997.
- [23] Yoneyama, A., Yeh, C. H., Kuo, C. C. J., "Robust Traffic Event Extraction via Content Understanding for Highway Surveillance System", IEEE International Conference on Multimedia and Expo (ICME) , pp.1679-1682, 2004.
- [24] Myler, H.R. and Weeks, A.R., The Pocket Handbook of Image Processing Algorithms in C, Prentice Hall, 1993.
- [25] Yoneyama, A., Yeh, C. H., Kuo, C. C. J., "Robust Vehicle and Traffic Information Extraction for Highway Surveillance", EURASIP Journal on Applied Signal Processing Vol. 14, pp. 2305-2321, 2005.

- [26] Blackman, S., Popoli, R., Design and Analysis of Modern Tracking Systems, Artech House, 1999.
- [27] Bar-Shalom, Y., Li, X. R., Kirubarajan, T., Estimation with Applications to Tracking and Navigation, John Wiley & Sons, 2001.
- [28] Bar-Shalom, Y., Li, X. R., "Multitarget – Multisensor Tracking: Principles and Techniques", YBS Publishing, 1995.
- [29] Welch, G. Bishop, G. "An Introduction to the Kalman Filter", UNC-Chapel Hill, TR 95-041, May 23, 2003.
- [30] Grewal, M. S., Kalman Filtering: Theory and Practice Using MATLAB, John Wiley & Sons Inc., New York, 2001.
- [31] Bar-Shalom, Y., Fortmann, T. E., Tracking and Data Association, Academic Press, 1988.
- [32] Oliver, E. D., "Multiple Target Tracking with Multiple Frame, Probabilistic Data Association", Signal and Data Processing of Small Targets, Vol. 1954, 1993.
- [33] Wang, X., Challa, S., Evans, R., "Gating Techniques for Maneuvering Target Tracking in Clutter", IEEE Transactions on Aerospace and Electronic Systems, Vol. 38, No. 3, July 2002.
- [33] Bertsekas, D. P., "The Auction Algorithm: A Distributed Relaxation Method for the Assignment Problem", Annals of Operations Research, Vol. 14, pp. 105-123, 1988.