Vehicle Tracking in Outdoor Environments using 3D Models

by

Nathalie El Nabbout

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2008

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

There has been a growth in demand for advancing algorithms in surveillance applications concerning moving vehicles where analysis of traffic has a potential application to security, traffic management (congestion and accident detection), speed measurement, car counting and statistics, as well as turning movement at intersections. This research focuses on multiple-vehicle detection, recognition, and tracking in urban environments based on video sequences obtained from a single CCD camera mounted on a pole at urban highways and crossroads. The proposed system integrates several modules including segmentation, object detection, object recognition and classification, and tracking. Background segmentation, based on Gaussian Mixture models, is used to extract moving objects from images using the respective foreground object information such as location, size, and color distribution. To recognize vehicles, a 3D polyhedral car model described by a set of parameters is built and mapped to the 2D edge information attained from the video sequence. The matching process is then used to classify the foreground object obtained into vehicles and non-vehicles. The output from the recognition model is used in tracking multiple cars based on a deterministic data association method that takes place between consecutive frame information.

The multiple-vehicle surveillance system developed in this thesis, based on integrating different modules, provides a novel approach for vehicle monitoring. Furthermore, the system makes use of minimal a priori knowledge about vehicle location, size, type, numbers, and pathways. The system implemented in this work functions well under various camera perspectives, background clutter, vehicle viewpoints, road types, scale changes, image noise, image resolutions, and lighting conditions.

Acknowledgements

I am deeply grateful to my thesis supervisors, Prof. David Clausi and Prof. John Zelek, for their guidance, good nature, and support during my MASc program. I would also like to express my gratitude to Prof. Paul Fieguth and Prof. Jeff Orchard who have agreed to be my thesis examiners despite their busy schedule. Furthermore, I would like to thank Vicky Lawrence who took the time to answer all my questions throughout the past two years.

I would like to extend my thanks to Miovision Technologies Inc. and Aimetis who have provided me with video sequence data sets used in this thesis.

My sincere thanks goes to my overseas friends: Samar, Sarah, and Najat, who have encouraged me throughout the years in my graduate school. Special thanks goes to Sami Khachan who inspired and believed in me; Sami, I cannot begin to tell you how much I appreciate your constant motivation and sound advice that you have provided me with all these years.

Last but not least, I would like to express my deepest thanks to my parents, sister, and brother for their love, care, and support throughout my entire life. This thesis would not have been possible without my parents' understanding and encouragement.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Problem Definition

There has been active research in recent years to develop automatic real-time surveillance systems that aim at detecting and tracking various objects such as cars in video sequences obtained from stationary cameras. The ultimate goal of the research is to replace traditional methods of video surveillance where human operators monitor a large number of screens and the amount of constant attention needed exceeds human capability [17].

Popular methods for extracting moving objects range from using optical flows [24][27] which do not return exact vehicle locations to adaptive background subtraction techniques whose accuracy depend on the number of frames available [51]. Vehicle representation techniques vary from simple blob representations that do not allow good distinction among different extracted objects to 3D model approaches that are more complex but are more robust to occlusion and permit the categorization of objects more accurately [24]. Tracking methods range from particle filtering approaches that require accurate target initialization and termination making it difficult to track an unknown number of objects in complex environments to data association methods that are computationally expensive [29].

To increase efficiency and decrease human error, an automatic visual surveillance system is proposed which uses video sequences of road traffic to detect, recognize, and track multiple cars in various urban environmental conditions (different road types, light conditions, and various camera perspectives). The multiple-object tracking system proposed uses background subtraction techniques for object detection, 3D models for vehicle representation, and data association tracking methods.

1.2 Challenges

The proposed system integrates several modules including object extraction (image segmentation and analysis), object recognition and classification, and multiple object tracking. Each of these modules has some challenges that must be dealt with in addition to the difficulties that arise when integrating the different modules together. Here is a list of challenges that must be addressed by each module to obtain a successful surveillance system.

- Light and Weather Conditions: Changes in the lighting of the road affects the way the vehicles appear. Light conditions may vary from a sunny bright day to a gloomy day where the former might result in more reflections off the metallic body of the cars and the latter might make the cars seem darker thus making them blend in with a dark surrounding. Such pixel color variations make it very challenging to segment the vehicles out of the image. Furthermore, weather conditions ranging from a sunny bright day to one that is rainy or snowy might also drastically make the segmentation process more difficult due to the reflection of the car and the water or snow causing everything to appear homogeneous.

- Image Noise: Another challenge arises from image noise. The data collected about the urban environment is done via a common video camera that might not have high resolution and might vibrate when the weather conditions are unfavorable. This increases the amount of image noise and distortions in the frames which in turn makes image segmentation and object extraction more prone to error. For example, if the noise increases substantially, edge extraction becomes more difficult since the edges do not appear clear-cut as in a noise-free image.

- Image Resolution: To save memory space, some of the cameras operate at lower resolutions. Furthermore, older cameras usually support lower resolution video recording. Therefore, a surveillance system should be able to support a range of different image or video resolutions in order to be practical.

- Background Clutter: Most real-world scenes include background clutter such as trees, buildings, traffic lights, trash cans, and so on, which makes object segmentation and extraction even more challenging, particularly in scenes where there are multiple objects.

- Camera Perspective: The camera, mounted on different poles and recording the data of the different sections of the highways and intersections, may vary in perspective, zoom, and height from one pole to another. With different camera perspectives, vehicle appearances (upper view, side view, ... etc) and size change. Furthermore, the extent to which one camera covers one particular location (field-of-view) affects the degree of appearance change of a car within one scene. For example, if a camera's field-of-view covers several kilometers of a highway, the cars will appear very small at first and become extremely large as they approach the camera closer.

- Occlusion: Depending on the camera perspective, different types of occlusion may also occur. If the camera height is low, some parts of the vehicle may be obscured by another vehicle. Additionally, some objects may exist between the camera and the road (eg. trees) that occlude the view of the car.

- Vehicle Viewpoint Changes: The car may appear different every time it changes its location with respect to the camera position. The camera records one perspective per each car which means that the car's front parts occlude its other parts. This results in many unique appearances for each vehicle view; thus, this makes car recognition a more challenging task.

- Intra-class Variation: Another main challenge for the recognition system and classification is the variety of different vehicle types: sedans, jeeps, trucks, buses, ... etc. Each type of car may share common features with other classes in geometry, appearance, and texture yet differ enough to create a need to introduce a new description for this class type in the recognition system.

- 3D Information Loss: Once the camera records the information, all 3D information is reduced to 2D. Therefore, there is no depth and height information in the 2D images.

- Multiple Tracking: When a large number of cars is in the scene, it becomes difficult to keep track of all the cars at once and retain track information per car. Problems arise due to challenges pertaining to systems that the tracker depends on — namely the recognition system that is facing occlusion, viewpoint changes, interclass variation, and information loss challenges and the segmentation system that faces problems in weather conditions, occlusion, and image noise. Furthermore, the tracker faces the problem of multiple car association which is answering the question 'Which car is which in the

previous frame?'. This can especially become hard when cars change direction haphazardly.

1.3 Thesis Objectives

Many of the challenges that are faced by automatic surveillance systems have been mentioned in the previous section. However, not all of the challenges are addressed in this thesis. Furthermore, the implemented system provides solutions to the challenges under some assumptions that are mentioned in the following list. Following the list of assumptions, a numbered list of goals that this research targets is given.

Assumptions:

1. The camera obtaining the video sequences is fixed at a height exceeding the height of a car. Once the camera is installed at a particular location, it remains fixed for the entire time of a video sequence. However, different cameras in different locations may vary in the tilt and zoom of the location being monitored.
2. The camera type and specifications (focal length) are assumed to be known.
3. The camera perspective is also assumed to be known. Therefore, the tilt of the camera and its height from the road is assumed to be a priori knowledge.
4. The camera plane with respect to the road plane is also assumed. In other words, the camera tilt with respect to the road (and not only to the pole) is assumed.

Objectives:

1. The implemented system must perform successfully under different road types such as highways and intersections.
2. The implemented system must be invariant to camera perspective.
3. The implemented system should work under varying vehicle viewpoints.
4. The implemented system should function under various types of background clutter

5. The implemented system should operate under many light conditions such as brighter days, gloomier days, evening times, and foggy weather.

6. The implemented system should be able to process noisy (point noise) images.

7. The implemented system should be capable of operating with different video resolutions as long as the resolutions maintain an image quality whereby the car is still recognizable.

8. The implemented system should work even in cases of occlusion.

9. The implemented system must be a multiple tracker that is able to track several objects simultaneously. Since humans can track up to eight objects simultaneously [2], the multiple tracker should be able to track at least eight objects at the same time to make is as efficient as the human visual system.

10. The implemented system should process each frame within a few seconds.

Moreover, a few additional constraints are targeted within this research. The first constraint is that the proposed system has no prior knowledge about the car size, type, or location. Also, there is no prior knowledge about the number of vehicles on the road and minimal information about the road structure.

Having discussed the general challenges that a surveillance system faces and specified the goals and assumptions of this research, the next section explains the overall proposed system.

1.4 System Overview

The first step in building an automatic surveillance system is the identification of moving objects based on the video sequence obtained from the camera. Therefore, after the breakdown of the video sequence into individual frames for analysis, the system should first filter the image to improve the signal to noise ratio and extract potential useful information from the frames. The first module — object extraction — uses Gaussian Mixture models to segment the different frames into foreground and background layers (refer to 1a in Figure 1.1). The foreground result is then analyzed (part 1b Figure 1.1) whereby the different foreground object information — such as pixel color distributions, size, initial orientation estimate, and position — is acquired per object from the result of the Gaussian Mixture Model as well as the original image (which is needed to acquire pixel color values). Meanwhile

the second module (refer to 2 in Figure 1.1) has a generic polyhedral wireframe model for a car. In order to recognize whether each object is a car, the information from the first module — particularly the size, orientation estimate, and location of each foreground object — is sent to the object recognition module where the edges of the different objects at each location are extracted. Edge-based vehicle detection has been chosen to be used for its enhanced performance [7][22] compared to other background removal or thresholding methods especially because edges remain salient in a variety of ambient lighting [23]. Having obtained the edges of the real object, they can be compared to the generic rescaled model. If a match occurs, then the vehicle type is classified. The classification information from the second module as well as the rest of the information extracted in the first module are sent to the final module — multiple tracker(refer to 3 in Figure 1.1). The tracker uses an association matrix to keep track of the best matches between consecutive frames — current frame and the past frame. As each new frame is received, the track per each car is updated based on this data association method.

A block diagram of the system is given in Figure 1.1. A video sequence captured by a video camera is the input. This is then split into frames that are analyzed by the three previously described modules. Each of the module boxes in the diagram contain a number to refer to in the text as well as the chapter that explains it in more detail.

1.5 Thesis Contributions

This thesis presents an alternative approach to multiple-vehicle surveillance systems by integrating various modules (Gaussian Mixture Model and temporal differencing for motion detection, 3D vehicle recognition based on Hausdorff measure, and deterministic data method for tracking which is able to deal with entering, leaving, and disappearing cars) in a way that has not been integrated before. This novel system is a simpler approach compared to other state-of-the-art surveillance systems that use 3D models (refer to Section 2.4). Furthermore, the system is capable of functioning under different camera perspectives, background clutter, vehicle viewpoints, road types, scale changes, image noise, image resolutions, and lighting conditions. Another important aspect of the proposed system is its minimal amount of a priori knowledge given to the system — no a priori knowledge of vehicle size, vehicle pathways, or detected object type is given, and the vehicle locations are automatically detected by the system.

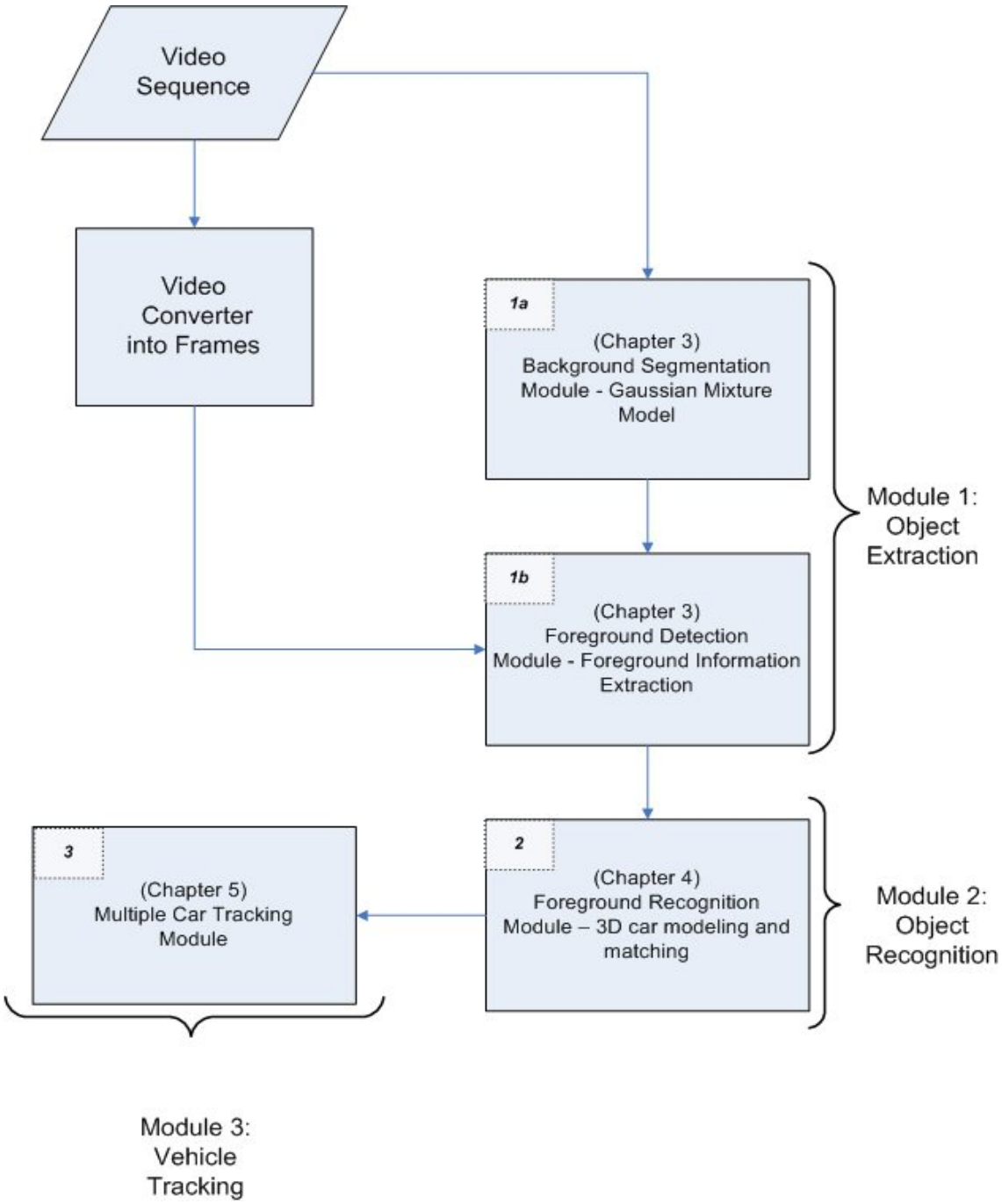

Figure 1.1: System overview diagram of the proposed system showing the major modules that make up the surveillance system.

1.6 Thesis Outline

Chapter 2 offers a survey of the different research in the fields of object detection, object classification, and tracking and also provides an overview of the state-of-the-art research and thesis contribution in the domain of video surveillance of vehicles. The first module — object extraction — is explained in detail in Chapter 3. Three approaches are described for background modeling and some results of this module are demonstrated along with the advantages and the drawbacks. Chapter 4 moves on to explain the object recognition module. The generic 3D model used is shown, model-to-data matching method is explained, and classification implementation methods are scrutinized. The last module — multiple tracker — is described in Chapter 5. The overall results of the system and the performance with regards to different challenges are shown in Chapter 6. Chapter 7 discusses the conclusions, future work, and contributions of this research.

Chapter 2

Literature Review

Recently, there has been considerable focus in advancing surveillance systems that monitor moving objects such as pedestrians and cars, which have a wide variety of applications such as human identification, congestion and car counting statistics, detection of un-natural behavior, and so on [16] [17] [23] [37] [48]. These surveillance system frameworks build upon several main components which include motion detection, object classification, and object tracking. In what follows, a review of recent developments in each of these components is presented. Finally, a state-of-the-art vehicle surveillance system is discussed followed by the contributions of this work.

2.1 Motion Detection

Different techniques have been used in various studies for the purpose of moving-object detection. Among these methods is the use of edge-complexity detection [16], optical flow methods [24][27], temporal differencing [9], and background subtraction [51]. Edge-complexity detection methods basically detect a car by the edge complexity on the road which can be computed by the use of histograms [23]. The latter three methods are more common and are thus described in more detail in the following sections.

2.1.1 Optical Flow

Optical flow (or image velocity) based motion segmentation is the perceived motion of 3D-object surface points (that are projected unto a 2D image) in terms of image

displacements or image flow vectors [5]. Optical flow methods are useful for detecting motion even if the recording camera is itself moving. However, optical flow approaches suffer from being computationally expensive as well as being sensitive to noise [27]. Furthermore, the optical flow estimated vector does not always represent the actual object movement rate in cases of optical flow discontinuities that occur when greyscale pixels disappear due to occlusion or illumination [37]. Another case where optical flow might fail is in areas where texture is not predominant and thus an object might appear to be not moving[37].

Despite the disadvantages of the method, optical flow was used by various researchers in creating a vehicle surveillance system. One of the more related works to this thesis is that by Haag and Nagel [15] where they incorporate the optical flow motion segmentation for 3D-model-based tracking of vehicles [37]. Using optical flow, the velocity of the object as well as its orientation can be computed. For more detailed information, refer to [15][37].

2.1.2 Temporal Differencing

Frame by frame differencing or temporal differencing approaches are based on subtracting consecutive frames to determine possible regions of moving objects [30]. There are several benefits from using a temporal differencing approach. First, it is computationally inexpensive. Second, it is quite effective at eliminating immobile objects from the scene and leaving only regions of moving objects. Also, when an object starts moving, it is able to detect it at the moment it starts displacement and does not need additional frames to adjust its output. Furthermore, it is adaptive to changing environmental conditions [17]. Nevertheless, the method fails to extract the whole object especially when an object has similar uniform regions as that of the background [23]. To solve the problem, a method using a two-component mixture density, where one component represents the background and the other represents the moving object, is presented in [38]. In most cases, temporal differencing has been applied to cases of stationary cameras [23] [38]; however, interframe differencing can also be applied to cases of moving cameras when joined with other algorithms such as motion compensation for background subtraction algorithms used for human silhouette extraction from dynamic backgrounds [43]. A more detailed re-evaluation of frame difference in motion detection is presented in [31].

Temporal differencing was implemented in this thesis and is explained in detail in Chapter 3. The method applied is explained and the different benefits and

drawbacks of the approach are demonstrated.

2.1.3 Background Subtraction

The simplest method to perform background subtraction is pixel-by-pixel subtraction of the current image from a previously known background image [32]. However, in most practical applications, the background image is not given as a priori knowledge. In that case, several other methods have been used to extract moving objects. On the more trivial end, there is background subtraction based on an averaged background image over N frames [23]. Although this is a straightforward method, it does not accurately represent the background — ghosting of slow moving objects is produced as well as the background histogram does not necessarily represent the histogram of the background in the current image. Other fairly more complicated methods include the use of Gaussian Mixture Models (GMM) [51] which are also used in this thesis. The Gaussian Mixture Model is explained in more detail in Chapter 3, along with its advantages, disadvantages, and results.

2.2 Object Classification

Once an object is detected, the next stage is to classify it since not all moving regions necessarily belong to the same category of objects. For example, in this thesis, movement on roads may be due to vehicles, humans, trees swaying in the wind, ...etc. According to Hu and Tan [17], there are currently two major approaches for object classification — shape based and motion based classification.

The former — shape based classification — is based on using shape information about the moving regions that constitute of boxes, silhouettes, blobs, points, or edges [17]. Various researchers use shape based classification. For instance, [44] silhouette blobs in the shape of vehicles can be used to scan the image and find any existing vehicles in the image. Yet another shape-based classification methodology uses templates which are made of silhouettes or basic shapes. Though templates are effective at describing the object since they use spatial as well as appearance information [48], they are computationally expensive because each template only records a single view of the object and recognizing the object would require matching it up against templates with all possible views. Edge extraction of detected vehicles and grouping the edges together to obtain the boundary of the detected

object is another approach. Using this method, the detection of vehicles or obstacles can vary from consisting of finding rectangles that enclose the boundary of the object to creating a more complex detected shape such as the entire car [48]. For instance, [37][24][27] use 3D generic vehicle models to describe different car categories — trucks, SUVs, sedans, and buses. Based on the edges of the detected car shapes and the degree of match between the detected car and each type of vehicle model, a classification of the detected car can take place. Edge-based vehicle detection is more robust to changes in ambient lighting and therefore performs better than background removal and thresholding approaches [48].

Motion-based classification takes advantage of periodic or repetitive motion patterns that can distinguish one object from another. The authors in [32] used motion classification to learn the behavior of the background moving door and discriminate the door from other object or human motion. As for [26], Lipton used the concept of residual flow to separate rigid objects such as vehicles that have little residual flow from non-rigid objects such as humans that have a higher residual flow.

In this thesis, the type of recognition and classification is shape-based — specifically based on edges and 3D models. The details of the method of object recognition is included in Chapter 4. Extra features are also combined to the shape based classification such as size of detected object and bounding box to create a more accurate classification system. The latter features are useful as the first round elimination of illogically large objects or small objects that could not possibly represent a car. The remaining car candidates then undergo the second stage of classification based on 3D models.

2.3 Tracking

The purpose of most surveillance systems is to monitor a target object by tracking it over a sequence of frames. Yilmaz et al. [48] details the different categories of object tracking currently being used. According to the categories outlined, there are three main ones: point tracking, kernel tracking, and silhouette tracking. The former tracking method is based on associating different detected points' information (position and motion) from one frame to another. Kernel tracking tracks objects by calculating the motion of an object's shape and appearance in successive frames. Silhouette tracking uses information inside the silhouette's region in the form of edge maps to track the object using shape matching [48].

In this thesis, the tracking is based mostly on point tracking; thus, a brief

overview of point tracking will be covered and the tracker used in this thesis will be mentioned.

Point tracking faces various challenges that include misdetections, occlusions, exit and entries of the objects [48]. According to Yilmaz et al., there are two categories of point tracking: deterministic methods and probabilistic models. Deterministic methods formulate the frame-to-frame object association problem as an optimization case whereby a solution consists of minimizing the correspondence cost resulting in one-to-one correspondences among all possible associations. Among the algorithms that employ this method is the Hungarian method [48]. The Hungarian approach can use several constraints to define the problem including proximity and maximum velocity constraints that state that the object will not move drastically from one frame to another and there is an upper bound on an object's velocity. In [39], the authors solve the optimization problem by proposing a greedy approach after using optical flow to obtain the initial correspondences. This method however does not work in cases of entry or exit of objects. A variant of the Hungarian approach proposed in [20] for the purpose of stem-cell tracking allows entry and exit of the cells. Another work by [41] uses graph approaches to formulate the correspondences among objects and find the best path in the graph.

Statistical methods, on the other hand, estimate the object state as a function of the measurement and model uncertainties. Among various statistical methods, the most common ones are the Kalman filters [21] (where a state is assumed to be a Gaussian) and Particle filters [3] (which overcomes Kalman's Gaussian distribution assumption). However, those two filters are usually applied to track single objects. If they are to be extended to track multiple objects, their most likely measurements should be associated using deterministic approaches previously explained [48]. An additional method for multiple vehicle tracking is that proposed in [29] where a stochastic framework based on Monte Carlo Markov Chain Association (MCMCDA) is used to develop a system capable of automatically initializing and terminating an unknown number of vehicle tracks in scenes of occlusion and clutter. Other statistical approaches to track multiple objects using data associations include Joint Probability Data Association Filtering (JPDAF) and Multiple Hypothesis Tracking (MHT) [4]. JPDAF is able to assign objects to tracks using probabilistic methods, but it is not able to function under entering and leaving objects in the image [48]. Multiple Hypothesis Tracking stores several hypotheses for each object at every frame and at the end of the observation of a track, the best possible object track is created based on these hypotheses. Though it is able to handle occlusions, and entering and leaving objects, the algorithm consumes

large memory spaces and requires exponential processing times to run due to the amount of potential hypotheses required to be searched in order to determine the most possible set of assignments [10].

2.4 State-of-the-Art

The previous sections have provided an overview of different available methods to handle the different stages of a surveillance system: object detection, object recognition/classification, and object tracking. This section provides a brief overview of the most recent systems developed using 3D-model classification which is the method employed by this work. Afterward, the advantages and disadvantages of the system are outlined.

Based on Hu et al. [17], there is no significant number of references to *3D-model* vehicle tracking since Haag and Nagel [15]. 3D wireframe vehicle models where a ground-plane constraint (GPC) is assumed to reduce the degrees of freedom of a vehicle's pose have been used [27]. In [47], a localization algorithm is proposed where a pose evaluation formula is used to compute the match between the image edge points and the projected model [17]. The most recent known advancement is that by Nagel (2007)[37], where a fully automatic system is proposed which uses edge-element and optical flow to determine locations of motion. A 3D automatic classification system is used to categorize the vehicles automatically into four different vehicle types. The tracking takes place using a Kalman filter with a motion-model that assumes a straight line or circular movement [37] which limits the application to these types of vehicle behaviors (straight or circular).

With respect to other types of approaches used to build surveillance systems, 3D model-based surveillance systems have several advantages. First, prior knowledge of the 3D shape of the object results in better performance under occlusion. Also, once the 3D world to image projection is established, the 3D pose of the objects can be acquired easily [17]. Furthermore, 3D models can be applied even if objects change their orientation during the motion. Another benefit of using model based models and edges to describe the object detected is the ability of edges to be robust to lightning changes. On the other hand, model based approaches can have higher computational costs than methods that do not involve model matching.

2.5 Contribution

Sections 2.1 to 2.3 are a brief review on the various methods used by different researchers for each module of the system outlined in Figure 1.1. Section 2.4 provides an overview of the state-of-the-art vehicle surveillance systems developed by other researchers. This section compares the method used in this thesis to several other recent approaches and underlines the contribution of this work.

Most of the 3D-model based vehicle surveillance systems developed are subdivided into modules that include a motion detection module, object recognition module, and a tracking module. However, various surveillance systems differ by having different approaches used per each module. Next, a comparison between the method used in this thesis to the most recent known surveillance systems based on vehicle modeling is presented.

For the first module — motion detection, Nagel et al. [15][37] as well as others [24] use optical flow methods to detect the regions of vehicle motions. On the other hand, Yiu et al. [49] use adaptive background subtraction models to detect the approximate location of the vehicle. This thesis builds upon using the background subtraction method — specifically Gaussian Mixture Models — as the main mode of object detection over video sequences with an implemented adjusted temporal differencing system which obtains the differenced result of images and enhances the output by using the OR operator and filling in gaps (Chapter 3). The latter system is used as a backup method whenever the Gaussian Mixture Model fails by returning extensively noisy results.

For the second module — object recognition, [15][24][27][37][47] use similar 3D wireframe models to model a vehicle; however, [49] uses simple 3D cuboid models with a width, height, and length description per model. To match the models to the vehicles in the 2D images, [15][37] use the Mahalanobis method to match the model to the vehicle edge information, [27][47] use the point-to-line segment distance to match the model to vehicle edges, and [49] use the Chamfer distance to compare the detected shape of the vehicle to the contour vehicle model. The system implemented in this thesis uses a generic 3D model similar to [24]; however, it is slightly simplified by eliminating an extra vector that describes the 3D model vehicles distance from base to ground. Various experiments demonstrated that this vector only increases complexity of the model and does not increase the accuracy. For this reason, it was eliminated. Furthermore, the matching method used between the projected 3D model and image vehicle edge segments is the Hausdorff distance.

For the tracking module, all of [15][27][37] use Kalman filtering methods for the purpose of tracking. [37] assume that the motion of vehicles are either circular or straight in accordance to the data sets that they use. This thesis on the other hand, investigates extended deterministic point tracking methods where the optimization problem constitutes of minimizing the cost function with minimal amount of constraints imposed. The cost minimization function takes into consideration several vehicle features that are associated among frames. Among those features is the general color of the vehicle, the position, and velocity. The algorithm proposed is able to handle entering and leaving cars. Also, the tracker deals with cases of error (such as a car disappearing in mid-road). Furthermore, it is computationally inexpensive and is able to function in real-time. In addition, no motion models are given a priori to predict where the vehicle will be next based on knowledge of potential vehicle pathways for a particular road (Chapter 5).

Therefore, this thesis has several contributions as listed below.

- Different methods for each module have been integrated in a novel manner to build a system that is based on background subtraction (GMM)/temporal difference module, 3D vehicle models with Hausdorff matching method, and optimization-based vehicle tracking that uses minimal amount of constraints to solve the tracking problem. Therefore, this work provides an alternative approach to other state-of-the-art methods for vehicle monitoring which integrates different modules that have not been integrated in such a manner before. Furthermore, the method used in this work is overall simpler compared to other state-of-the-art surveillance systems using 3D models.

- The background subtraction (GMM), which is adaptive and robust to light changes, was introduced with an enhanced 'OR' temporal difference motion detection method that can be used as backup system for the former method.

- The system is automatic with minimal need for input from the user. The vehicles are automatically detected, recognized, and tracked as they enter and leave the scene. No vehicle size, vehicle types, vehicle numbers, vehicle location, or vehicle travel pathway is known a priori. Also, the system functions well under different camera perspectives, vehicle viewpoints, road types, scale changes, image noise, image resolutions, and lighting conditions. However, the assumptions needed for this system are based on prior knowledge of fixed camera location, camera type, and camera tilt with respect to the road, which is acceptable for the applications targeted in this thesis.

In the next chapters, the methodology of the proposed and implemented automatic surveillance system is explained in details.

Chapter 3

Motion Detection

Among the initial steps of any automatic surveillance system is distinguishing target objects from a given video sequence. To detect moving objects, motion detection can be implemented by several methods such as image difference and background subtraction. In this research, both approaches are investigated; the former method is explained in Section 3.1, whereas the latter method which uses Gaussian mixture models for background subtraction is discussed in Section 3.2. Furthermore, an image segmentation approach — statistical region merging — for the purpose of object detection in explained briefly in Section 3.3. The advantages and disadvantages of all methods are discussed and the three methods are briefly compared to each other in Section 3.4. Section 3.5 explains the process of vital information extraction per detected object.

3.1 Image Difference

One of the methods for motion detection is temporal differencing which is adaptive to changes in dynamic backgrounds [9]. The implementation of temporal differencing, namely image differencing, in this research is discussed in the next section followed by a section summarizing the advantages and drawbacks of this approach.

3.1.1 Methodology

Image differencing is the difference between consecutive frames, frame at time t and another frame at time $t-1$. Subtracting consecutive frames from each other results in an image showing any changes or motion that has occurred between the

two frames while eliminating the stationary background. Consecutive frames have the advantage of having similar backgrounds since not enough time has passed for the background to change significantly. This allows more accurate removal of non-moving objects. For this reason, temporal differencing is effective in motion detection in dynamic environments.

Image differencing suffers from several drawbacks. First, it extracts only part of the object features. When an object moves from one coordinate in the first frame to the another coordinate in the next frame, there is an overlapping region of the object that is still common between the two frames. The overlap is removed in the subtraction of the two frames and only the increment of the moved object remains in the result. The second problem with image differencing is ghosting [31]. After the object moves and the images are subtracted, only the parts of the car that have moved remain in the result — as mentioned earlier. However, some of the parts correspond to the location of the car in the past. This region is referred to as a ghosting effect. Both phenomena discussed are illustrated in Figure 3.1. In the figure, Frame 1 is considered to be equivalent to time $t-1$ and Frame 2 is at current time t.

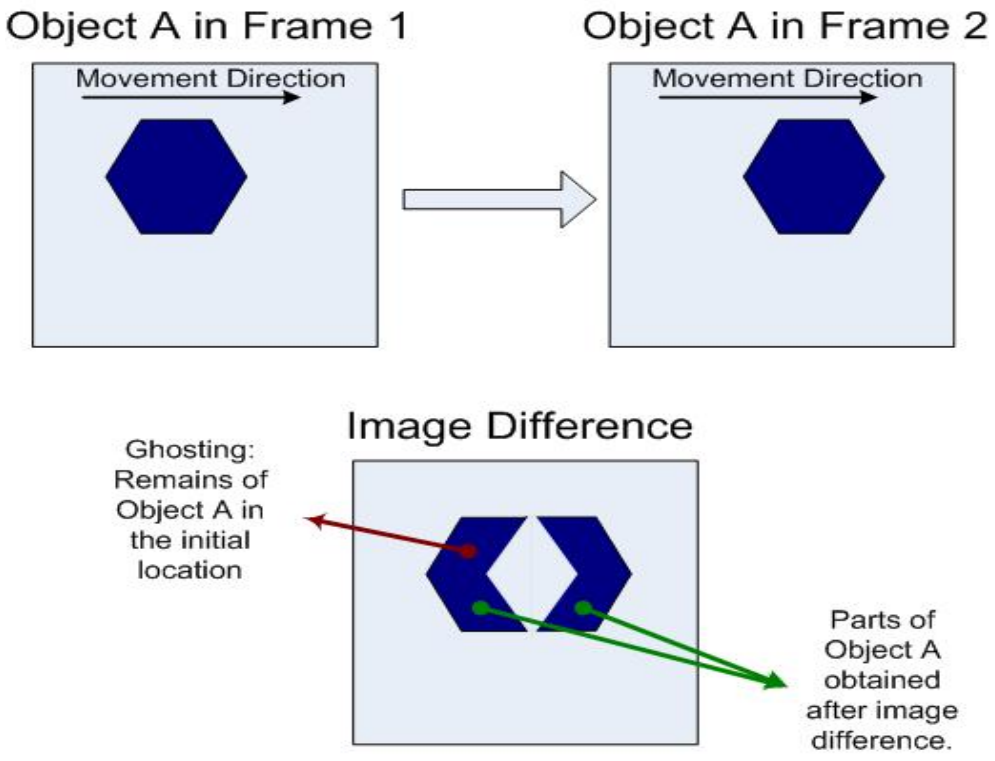

Figure 3.1: Image Difference: Partial object extraction and ghosting phenomena.

The initial algorithm for the image differencing method implemented in this research project targets to solve the partial object extraction problem shown in Figure 3.1. The algorithm proceeds to obtain the individual greyscale images of both frames at time t and at time $t-1$. This can be done by converting the RGB images into greyscale using Otsu's method [36]. Each of the images undergo a pre-processing stage where individual pixels are removed and singleton background pixels in dense foreground neighborhoods are set to belong to foreground as well. Such pre-processing eliminates potential noise and ensures that minimal noise is propagated to the image-differenced result. The second step is to undergo frame-to-frame subtraction. This involves subtracting the previous frame from the current frame. Let $P(x, y)$ represent a previous frame and $C(x, y)$ represent the current frame where x and y are the row and columns of pixels coordinates and both images are of MxN size. The equation below outlines the image subtraction.

The general frame-to-frame subtraction is illustrated by equation 3.1 where the previous frame is removed from the current frame.

$$R(x, y) = C(x, y) - P(x, y) \forall (x, y) \in \{1, ..., M\} \times \{1, ..., N\} \tag{3.1}$$

Such image differencing will result in values ranging from negative to positive. The negative values belong to the previous frame that are not found in the current frame. Therefore, given that the same objects are moving slightly from one frame to the other, the negative values pertain to the locations where the objects were in the previous frame and are no longer there in the current frame. For this reason, the negative values are isolated and called the 'Past Change'. On the other hand, positive values in the differenced image result belong to the moving object in the current frame at locations where the object was not yet previously. In other words, positive values indicate the number of pixels moved since the previous frame. The object movement in this case is referred to 'Present Increment'. Both situations are summarized in equations 3.2 and 3.3.

$$PastChange(x, y) = \begin{cases} |R(x, y)| & \text{when } R(x, y) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

$$PresentIncrement(x, y) = \begin{cases} R(x, y) & \text{when } R(x, y) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

A note can be made that such image differencing retains the information of object movement and not the whole object itself. This means that common areas with

some features of the object in the previous and current frames are subtracted out, which explains the partial object problem discussed earlier. The image differences are then converted into binary images. Conversion into binary images occurs by setting all pixel values that are less than a threshold T to 0 (black) and anything above T to 1 (white). The binary results of the image differences illustrating the past change and current increment are shown in Figure 3.3 a and b.

To solve the partiality problem, the two binary image differences obtained are passed through an OR operator. The logic behind this is that by obtaining the OR of both image differences, the common area is engulfed by the change made by the object. This engulfment causes a capsule that can be easily filled using morphological operators (such as *imfill* in Matlab) to obtain a homogeneous area representing the object. The result of the OR operation is shown in Figure 3.3 c.

3.1.2 Conclusions, Advantages, and Drawbacks

The image differencing method used in this project is based on subtracting consecutive frames from each other and obtaining the result of applying OR operator on the differenced image. After filling the gaps in the result of the OR operator, the moving objects are extracted as illustrated in Figure 3.3 d.

This method has many advantages:

- The method is straightforward to implement.
- The computational complexity of the algorithm is low and therefore runs quickly (within a second per frame).
- Using previous frame to approximate the background in the image is more accurate since there is high probability that the background did not change within the limited time frame that has passed between one frame and the other (1/30 second). On average, the displacement of the vehicles from one frame to its consecutive frame can vary from being one pixel if the camera perspective is zoomed out to ten pixels if the camera perspective is a closeup of the road. Frame-to-frame differencing can have temporal sampling of every 3 to 4 frames (every 3/30 or 4/30 of a second) to effectively capture the vehicle movement while the background remains constant. Such sampling would decrease computational times for processing the entire video sequence.
- Image differencing technique does not need training or large sequence of images to obtain a solution.

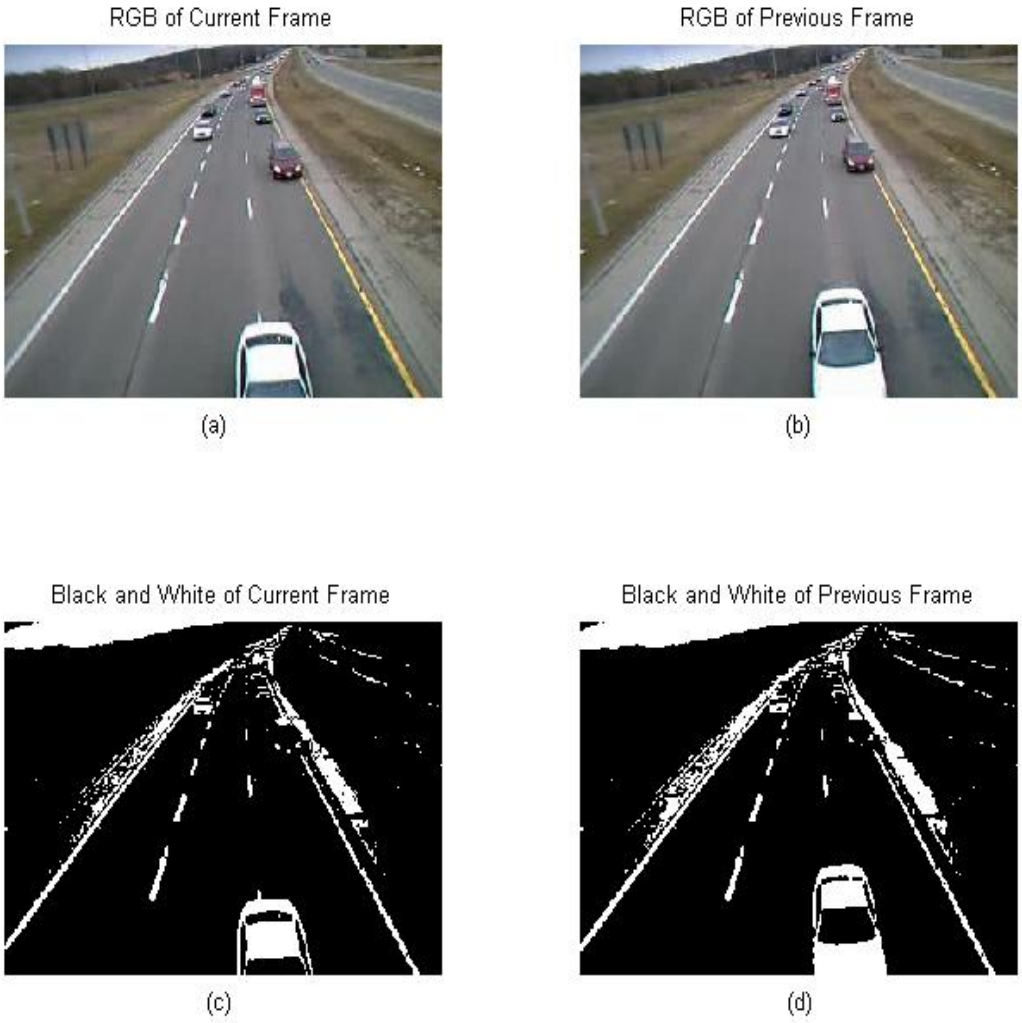

Figure 3.2: a) The current frame in RGB. b) The previous frame in RGB. c)The black and white results of the current frame. d) The black and white version of the previous frame.

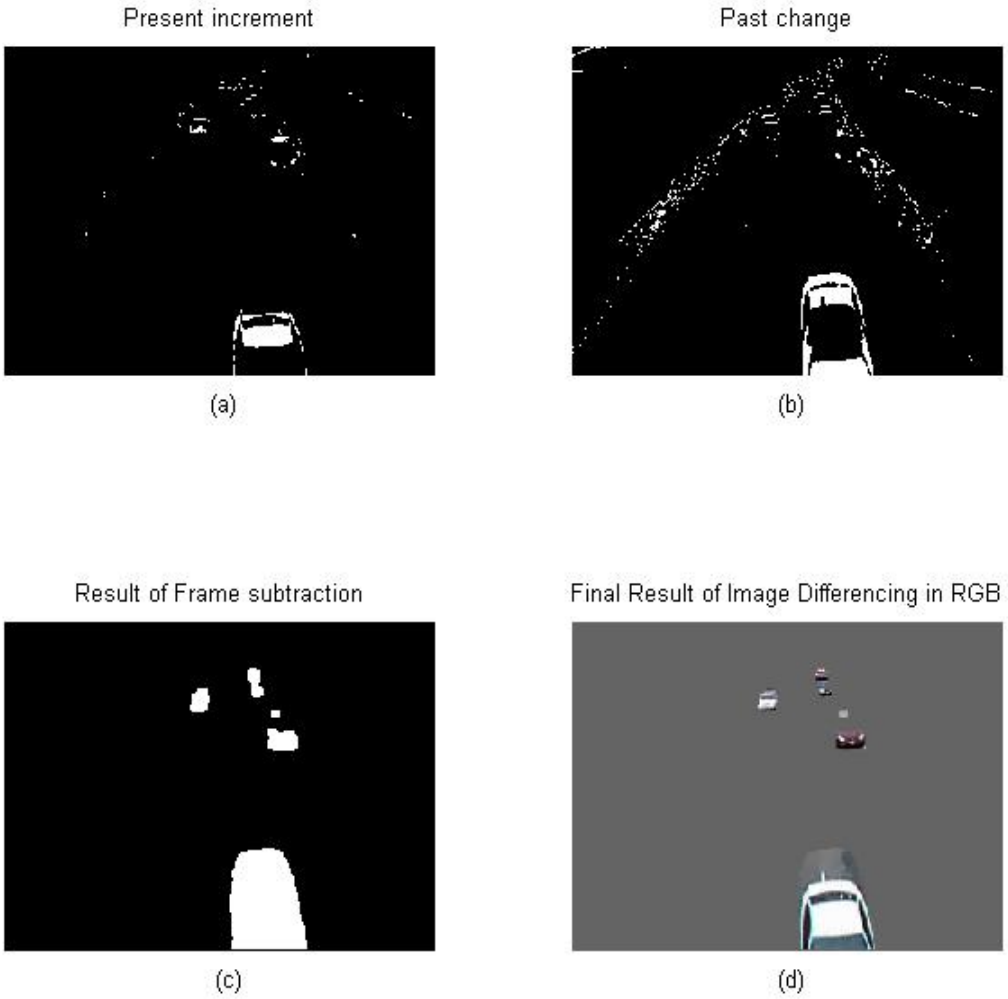

Figure 3.3: a) The Past Change that results from the image difference of the images shown in Figure 3.2. b) The Present Increment that results from the image difference of the images in Figure 3.2. c) The result of using an OR operator and filling in the gaps. d) The equivalent result of 3.3c in RGB.

- Image differencing can capture the movement of objects that were not moving for a long time. For example if a car has been still for 60 frames and suddenly starts moving, the image difference is able to capture that movement.

There are also several disadvantages to this method:

- The size of the extracted object includes some of the past history and is therefore sometimes inaccurate. Furthermore, when the history is included, the shape of the object also changes which could increase probability of error in classification of the detected object.
- Stationary vehicles are not detected using temporal differencing because the method captures movement which is not present with stopped vehicles. However, this disadvantage is not considered vital in this thesis because the work primarily deals with detection, recognition, and tracking of moving vehicles.
- Some of the very small objects such as a car at a very far distance may be missed by the algorithm when pre-processing of the images occurs using morphological processes. However, this disadvantage is not important because distant cars eventually drive toward the camera and grow in scale at which point they can be detected.

To solve some of the disadvantages of this method, other techniques for object extraction were investigated. Among those techniques are the statistical region merging and background subtracting using Gaussian mixture models. Both methods are described in more details in the next section.

3.2 Gaussian Mixture Model Method

Background subtraction, the difference between current frame and a reference background image, is a common approach to extract moving objects from surveillance videos [30]. The most straightforward implementation of background subtraction is removing a priori known background from the current scene. However, such a solution is not feasible for many surveillance problems where the background is not known a priori. This research uses the Gaussian Mixture Model (GMM) method for background subtraction explained next.

3.2.1 Background

By using the GMM technique, the values of each particular pixel in the image is modeled as a mixture of Gaussians. Based on the repetitiveness and variance of each of the Gaussians of the mixture, Gaussians that correspond to the background can be determined [45].

Mathematically, a history of each pixel, $\{X_1, \ldots, X_t\}$, can be modeled by K Gaussian distributions [51]. The probability of the value of the current pixel X_t is:

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} * \eta(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{3.4}$$

where K is the number of distributions, $\omega_{i,t}$ is the weight of the amount of data represented by the i^{th} Gaussian in the mixture at time t, $\mu_{i,t}$ is the mean of the i^{th} Gaussian at time t and $\Sigma_{i,t}$ is the i^{th} covariance at time t. η is the Gaussian probability density function such that

$$\eta(X_t, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma|^{1/2}} e^{-1/2(X_t-\mu_t)^T \Sigma^{-1}(X_t-\mu_t)} \tag{3.5}$$

Each time a new sample is obtained, the set, $\{X_1, \ldots, X_t\}$, is updated and the distribution is re-estimated [51]. To simplify computations, the covariance matrix is assumed to be $\Sigma_{k,t} = \sigma^2\mathbf{I}$ which assumes that the RGB values of the pixels are independent with identical variances [45].

K is usually selected based on the memory and computational power available [45]. However, in this research, a GMM method is used where the model's parameters are constantly updated in a recursive manner and the appropriate number of components for each pixel is selected based on an adaptive method developed by Zivkovich [50],[51]. To update the parameters of the Gaussian Mixture Model (weight, mean, and variance), a Mahalanobis distance is obtained from the sample to a particular component. The threshold that determines whether the Mahalanobis distance between the sample to a component is close or not was chosen to be three similar to the authors who proposed this GMM method [50][51]. 'Closeness' of a sample to a component is represented by o which is set to 1 if the Mahalanobis distance is less than 3 and thus the sample is considered to be 'close' to the component and is set to 0 otherwise. If there are no 'close' components, a new component is created. Based on whether it is close or not to the component, the parameters

are then updated as illustrated in the equations below. The Mahalanobis distance can be formulated as in equation 3.6.

$$D_i^2 = \frac{(x_t - \mu_i)^T (x_t - \mu_i)}{\sigma_i^2} \tag{3.6}$$

If $D_i^2 > 3$, weight, mean, and variance are updated as in equation 3.7.

$$\begin{aligned} \omega_i &\longleftarrow \omega_i + \alpha (o_{i,t} - \omega_i) \\ \mu_i &\longleftarrow \mu_i + o_{i,t} (\alpha/\omega_i) (x_t - \mu_i) \\ \sigma_i^2 &\longleftarrow \sigma_i^2 + o_{i,t} (\alpha/\omega_i) \left((x_t - \mu_i)^T (x_t - \mu_i) - \sigma_i^2 \right) \end{aligned} \tag{3.7}$$

where $\alpha = 1/T$ is an asymptotically decaying function that decreases the influence of old samples. Having updated the parameters, all components are sorted according to decreasing weights and the background model is approximated using the first B largest clusters [50].

$$B = argmin_b \left(\sum_{i=1}^{b} \omega_i > (1 - c_f) \right) \tag{3.8}$$

where c_f is the maximum amount of data that can belong to the foreground without influencing the background model.

3.2.2 Sample Results and Conclusions

One of the advantages of using Gaussian Mixture Models (GMM) for background subtraction is its ability to adapt to changing illumination such as weather and day conditions in the video sequence as well as entering and leaving moving objects from the scene [50]. Furthermore, it has the advantage of returning the entire object extracted compared to the image differencing technique discussed in section 3.1. The result of GMM for the same current frame used for the image differencing illustration is shown in the Figure 3.4.

GMM needs to learn many frame samples before the background subtraction actually provides good accuracy. Cases exist when only a few frames are available. For example, if the system is running in real time, at startup, it has only a few frames of the video sequence available that it can use to process from. Another

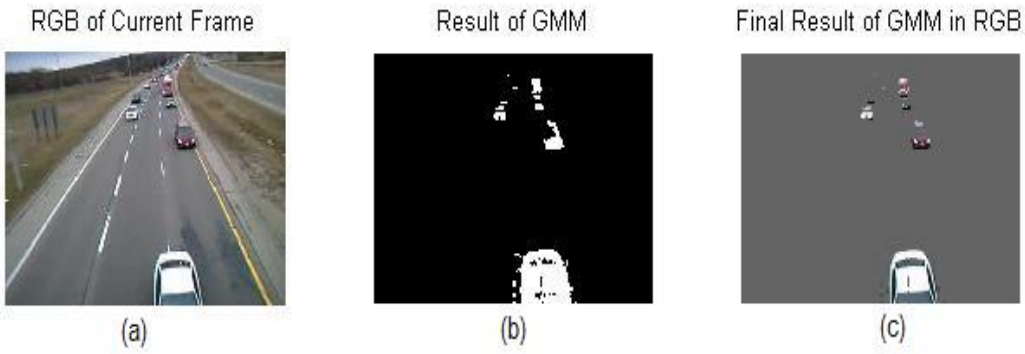

Figure 3.4: a) The current original RGB image. b) The result of GMM in black and white. White locations indicate the detected foreground objects. c) The equivalent of the BW result in RGB.

example is providing the system with a video sequence of tens of frames to process. When only a few frames are provided, results tend to be noisy as shown in Figure 3.6a.

Figure 3.5: Original Frame.

Therefore, to eliminate such extra noise in the frames, some post-processing of the GMM results is needed. Morphological operators such as removing isolated pixels, dilating detected objects to enlarge them, filling the holes inside the objects, and finally eroding the objects by same operator as used by dilation to shrink them back to original size are applied to clean the image. Figure 3.6b shows the result of post-processing.

The next section explains the statistical region merging method followed by a section of comparison of the three different methods.

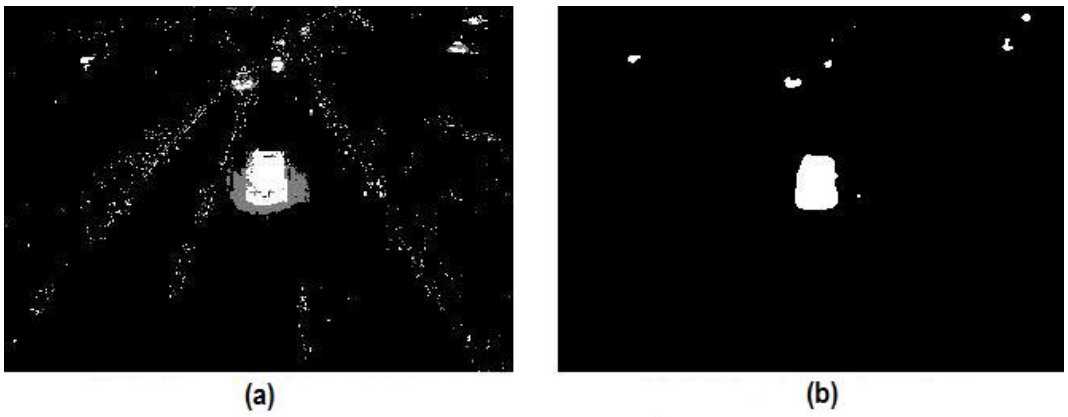

Figure 3.6: a) The result of the Gaussian Mixture Model after only a few first frames. Note that the results appear to include a lot of noise. b) The result of GMM after post-processing the noisy GMM results using morphological operators.

3.3 Statistical Region Merging

Nock and Nielsen in [34] proposed the method of statistical region merging which is a statistical image segmentation method that merges regions in a particular chosen order. A theoretical unknown (true) image is assumed with pixels referred to as "statistical pixels" which are characterized by distributions from which sampling of the observed image occurs. Merging of regions is based on the similarity of different pixel statistical expectations. Each pixel color channel is replaced by the number of independent positive random variables specified by parameter Q. The statistical complexity of the scene is modified by using the parameter Q which also controls the coarseness of the image segmentation [34].

Nock and Nielsen provide a C-code implementation of their algorithm which was used to test on the data sets used in this thesis. Although the algorithm decreases the number of undermerged regions, the amount of overmerging is excessive. Furthermore, the merging frequently merges distinct objects together as well as merges foreground targets with the background. To test the implementation, it was applied on intersection images with varying Q values: 6, 20, 32, 72, 218, and 686. The smaller Q is, the more merging occurs and the less numerous the regions are in the output [34]. According to [34], Q=32 seems to be sufficient to 'nicely' segment many of the images [35]. For this reason, Q=32 was selected as one of the values to test the implementation along with random values that ranged from small (Q=6) to high (Q=686) that would illustrate the algorithm's segmentation performance into a few regions as well as numerous. Figures 3.7 to 3.10 illustrate the results.

Figure 3.7: Original image used for testing SRM.

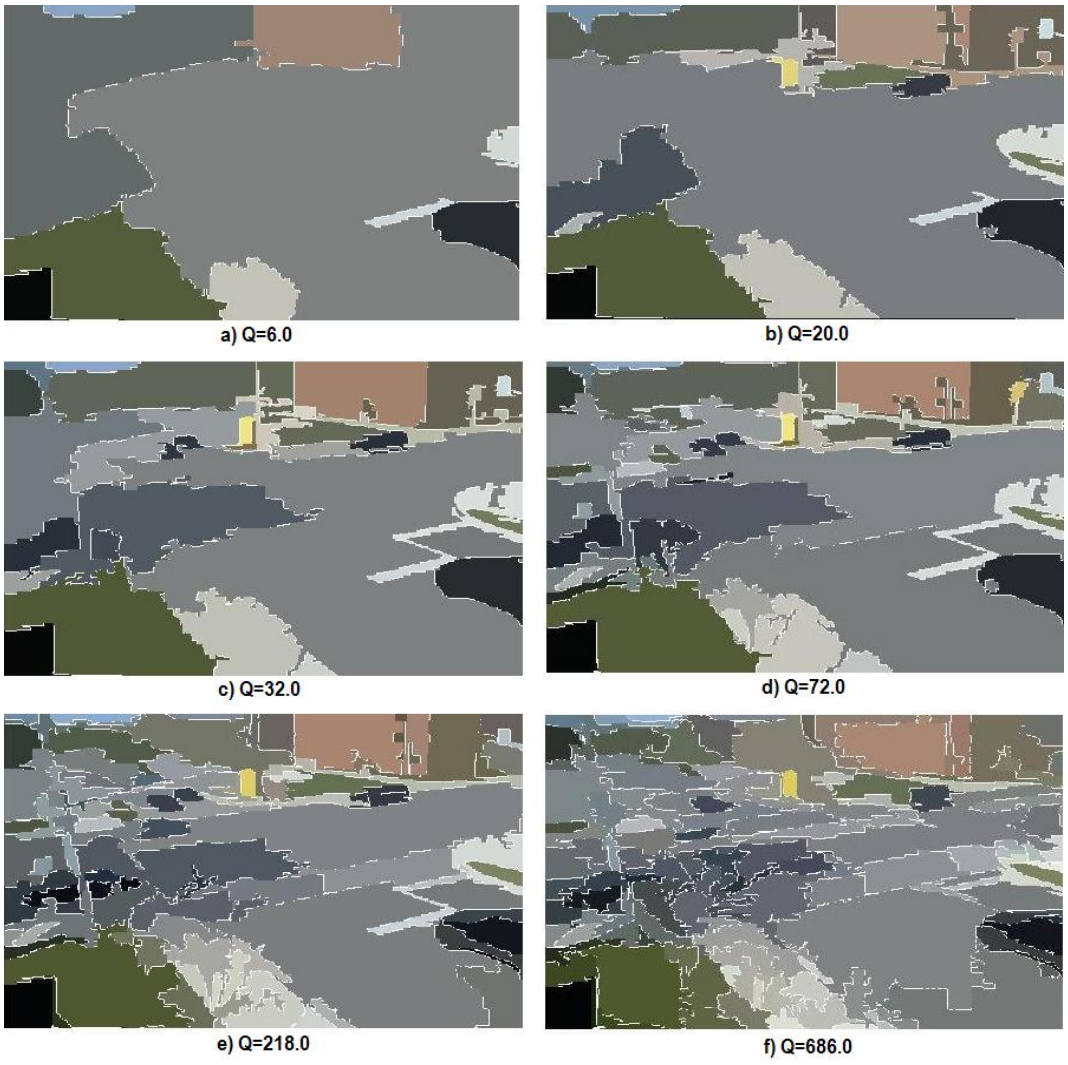

Figure 3.8: Results of SRM based on image in Figure 3.7 with varying Q: a) Q=6, b) Q=20, c) Q=32, d) Q=72, e) Q=218, and f) Q=686.

Figure 3.9: Another original image used for testing SRM.

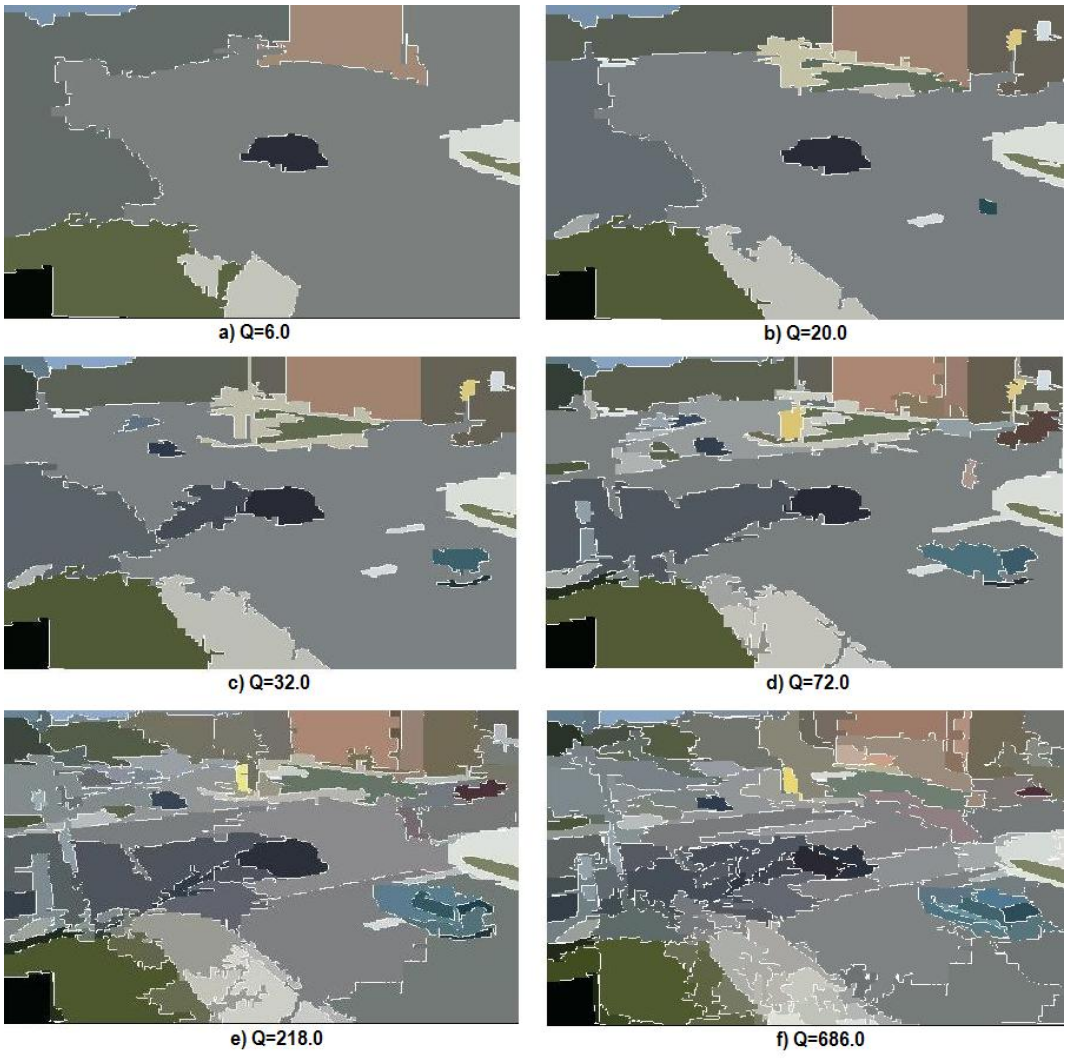

Figure 3.10: Results of SRM based on image in Figure 3.9 with varying Q: a) Q=6, b) Q=20, c) Q=32, d) Q=72, e) Q=218, and f) Q=686.

3.4 Comparison of Methods

Three different methods for object detection were discussed in the previous sections: image differencing, background subtracting using Gaussian Mixture Models, and statistical region merging. Based on the results in the images of Figures 3.7 to 3.10, one can note that statistical region merging results in merging that does not provide any significant information on what regions belong to the foreground or background as well as overmerges the regions in most cases. Also, the target foreground objects are usually not segmented as individual objects; instead, they are partly merged with the background, subdivided into several regions, extracted only partly, or not detected at all. For all the aforementioned reasons, the statistical region merging method was abandoned and approaches using time-differencing and background were considered.

Table 3.1 compares the two remaining methods by referring to their advantages and disadvantages that have been separately mentioned in the previous sections. The first column just numbers the rows, the second and third columns indicate the advantage or disadvantage for each method. A plus or minus sign is used to indicate whether the point is an advantage or disadvantage with regards to the method.

Table 3.1: Comparison of image difference and GMM

Comparison of Methods				
No.		Image Difference Proposed	Gaussian Mixture Model	
1	+	Simple Implementation	More Complex Implementation	–
2	+	Low Computational Complexity	More Computationally Complex	–
3	+	Robust to Light Changes	Also Robust to Light Changes	+
4	+	Needs Only two Frames to Function	Performance Depends on Number of Frames	–
5	+	Detects Movement Instantly	Needs Several Frames to Detect Sudden Movement	–
6	–	Results Include Ghosting	Results Do Not Include Ghosting	+
7	–	Does Not Return Entire Object	Returns Entire Object	+
8	–	Does Not Detect Very Small Objects	Does Not Detect Very Small Objects	–

Although the Gaussian Mixture Model for background subtraction approach is computationally more expensive and requires more a priori knowledge than image differencing method, it provides more accurate results when extraction of the whole moving object is required. However, GMM needs to learn many frame samples (reaching up to fifty frames) before it can provide good accuracy. If only a few frames are provided, results tend to be noisy. Thus, the image differencing method is useful whenever the video sequence is short because image differencing does not

face any problems with regards to the number of frames provided (as long as there are more than 2 frames). Furthermore, when an object remains inactive in a scene for a long time, the GMM method will label it as part of the background. When the object actually starts moving, the GMM requires several frames to re-adjust the object's labels to foreground. This problem is not faced by the image differencing technique which will simply capture the movement of the object by consecutive frame subtraction.

Since in this research the size of the object extracted plays an important role for resizing vehicle models (as will be explained in following chapters), the Gaussian Mixture Model for background subtraction was used as a default motion detection technique. In cases where the result of the GMM was too noisy or inaccurate, the image differencing result was substituted for GMM result for that particular frame. An image can be assessed as too noisy when an excessive number of small objects are detected which were not present in the previous frame and which have not appeared at the sides of an image possibly conveying potential entering vehicles.

3.5 Analysis of Detected Object

In Figure 1.1 of chapter 1, the object extraction module was divided into two different processes: background segmentation and foreground detection (or foreground information extraction). The former — namely time-differencing and Gaussian Mixture Models for background subtraction — have been discussed in the previous sections of this chapter. The latter consists of using the information obtained from the background subtraction algorithm along with the original images in order to assemble vital information regarding the extracted object's size/area, position, histogram, major and minor axes, initial orientation estimate, as well as the minimum bounding box needed to encapsulate the foreground object. This is done by labeling all the foreground detected objects in the black and white (BW) images returned by the GMM or temporal differencing methods. Labeling takes place by scanning the BW images column-by-column for 8-connectivity pixels. For each object found, a number is given greater than 1 if it is foreground (white) and 0 if it is background. A Matlab function *regionprops* can then be used to extract different information about each labeled detected object: bounding box, orientation, centroid, area, and minor/major axes. The bounding box, the smallest rectangle containing the detected region, is described in terms of [corner width] where corner is the upper left corner of the bounding box in the form of $[x\ y]$ for the 2D images used, while width

is in the form of $[width_x\ width_y]$ which specifies the width of the rectangle along the x and y axes respectively. Once the bounding box is obtained, the colors of the object can be extracted within the region of interest (ROI) constrained by the bounding box as shown in equation 3.9. I is the original RGB image, $[x\ y]$ is the upper left corner of the bounding box, and $[width_x\ width_y]$ are the width of the bounding box along x and y axes.

$$R = I(i, j, 1) \forall (i \in \{x, ..., x + width_x\}, j \in \{y, ..., y + width_y\}) \tag{3.9}$$

$$G = I(i, j, 2) \forall (i \in \{x, ..., x + width_x\}, j \in \{y, ..., y + width_y\}) \tag{3.10}$$

$$B = I(i, j, 3) \forall (i \in \{x, ..., x + width_x\}, j \in \{y, ..., y + width_y\}) \tag{3.11}$$

Figure 3.11 illustrates the various aspects of the object under scrutiny.

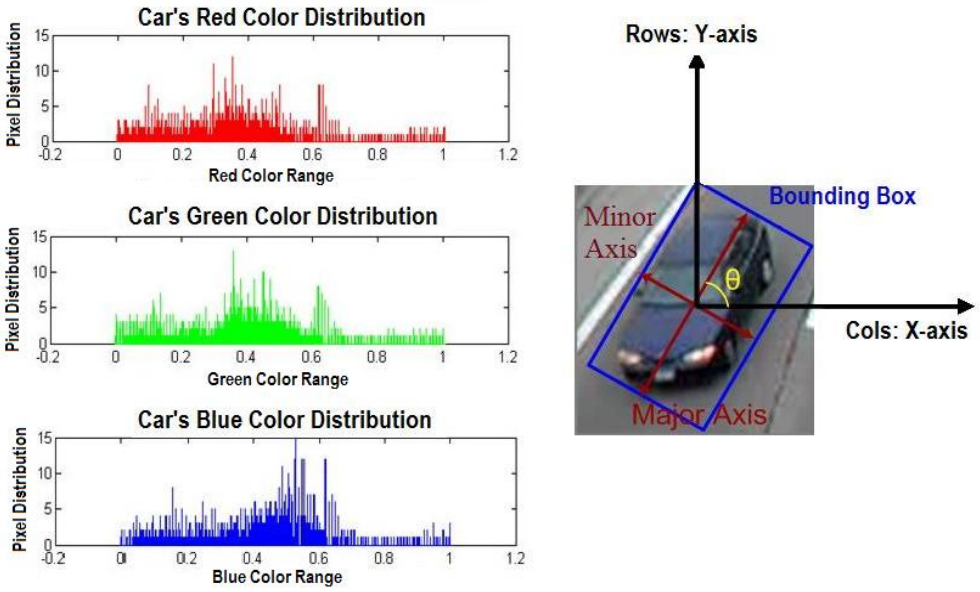

Figure 3.11: Different aspects of every detected car are extracted and stored. The left side of the image illustrates that the histograms of each color distribution of detected object pixels is a useful component to describe the object extracted. The right side of the image shows other important aspects to be extracted about the detected object: bounding box, centroid, orientation, area size inside the bounding box, as well as the major and minor axes.

3.6 Chapter Summary

Chapter 3 described the first module — object extraction — of the automatic surveillance system implemented. This module is made up of two major components: foreground detection and foreground information extraction. Three methods for foreground detection — temporal differencing, Gaussian Mixture Models for background subtracting, and statistical region merging — are investigated and discussed in sections 3.1, 3.2, and 3.3 respectively along with their advantages and disadvantages. Section 3.4 compares the three investigated methods and concludes that the statistical region merging method is not suitable for the construction of an automatic surveillance system due to its tendency to overmerge and segment the image into incorrect regions. The remaining two methods are then compared against each other and the Gaussian Mixture Model for background subtraction is chosen as the primary method for object detection in this work. Section 3.5 discusses the second component of the foreground extraction module which involves extracting useful information about each detected object — such as the color histograms, the minimum bounding box that can engulf the object, area, approximate orientation, centroid, and major/minor axes. After the information per object is obtained, it is passed along with the original image to the next module: foreground recognition which is explained in the Chapter 4.

Chapter 4

Foreground Modeling

In the previous chapter, the different methods for the purpose of image differencing, background subtraction, and target object information extraction that are used in the first module — object detection — were explained. This chapter focuses on the second module of the automatic surveillance system — object recognition. The different steps for vehicle modeling, object recognition, and classification are explained in the following sections.

4.1 Three-dimensional Models

There are various approaches in the literature that have been used to represent an object — namely rigid models and non-rigid models [6] [14] [24]. Rigid models, whereby a large amount of templates are stored and compared to the detected object, are extremely computationally expensive [24]. On the other hand, by using the method of non-rigid models, a parametrized generic vehicle model can be constructed and used to determine whether an object in the scene is a vehicle or not. Parametrized models can vary from being a 3D contour model such as in [49] to a more detailed vehicle model such as in [17]. In this work, a 3D model car is used similar to one proposed by Koller [24] and used by Nagel and others [15] [37]. The model will be explained in the next section followed by a discussion of 3D model mapping to 2D image information.

4.1.1 Parametric Car Model

The parametrized polyhedral model that is used as a generic vehicle model constitutes of a vector with 11 different distance parameters unlike Koller's version [15] [24] where 12 parameters are used. Based on experiments, it was found that the distance from ground to car base does not contribute to the overall recognition process and was thus eliminated. Each of the 11 parameters represent the length, height, or width of a particular car sector as shown in Figure 4.1.

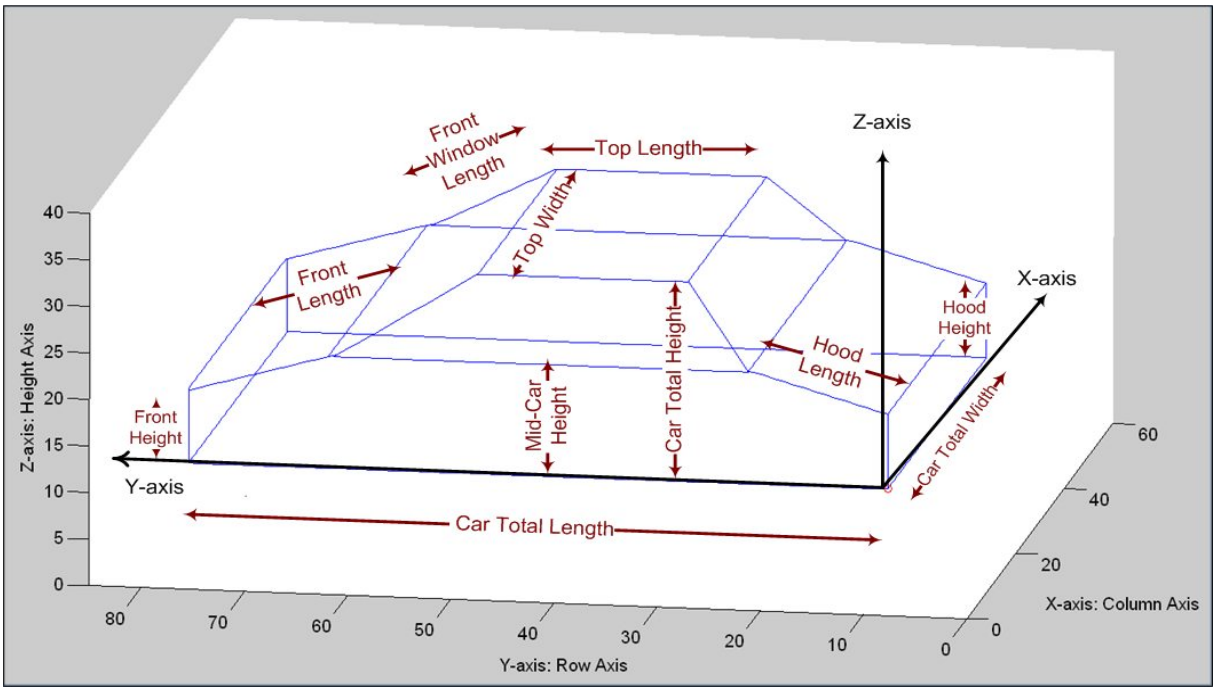

Figure 4.1: Parametrized Polyhedral Generic Vehicle Model Used in this thesis.

Thus, the vehicle-parameters vector used to represent the generic car model can be expressed as shown in equation 4.1.

$$v = \{bl, bw, tl, tw, th, fwl, fl, fh, hl, hh, mh\} \tag{4.1}$$

where each of the vehicle parameters are then defined in terms of the three major variables: H, L, and W. These variables represent the total height, length, and width (respectively) of the 3D car. By defining the vehicle parameters in terms of these three variables, it becomes possible to create categories of vehicles (for e.g. sedan, SUV... etc) by altering the ratios per category. The representations of each

of the vector, v, parameters along with the definitions in terms of H, L, and W are shown in the following list.

- $bl = L$: length of vehicle's base (total vehicle length)
- $bw = W$: width of vehicle's base (total vehicle width)
- $tl = 2.5L/10$: length of vehicle's top
- $tw = 1.6W/2$: width of vehicle's top
- $th = H$: height of vehicle from base till top
- $fwl = 5L/10$: length of vehicle front added to length of front window
- $fl = 2L/10$: length of vehicle's front
- $fh = 2H/4$: height of vehicle's front to base (thickness of vehicle at front)
- $hl = 1.5L/10$: length of the vehicle's rear hood
- $hh = 2.5H/4$: height of vehicle's rear hood to base (thickness of vehicle at the back)
- $ah = 2.5H/4$: height of vehicle from base to windows on the side (mid-car height)

Each of the three variables — H, L , and W — are used to scale the 3D generic model of a vehicle. Those variables are then matched to the car information obtained from the first module — object extraction. Toward the end of chapter 3, the different data obtained about each object extracted were explained. From the information that was attained about the foreground object were the size (area of object), major and minor axes of the ellipse encompassing the object, and the minimum bounding box needed to overlap the foreground object. This information aids in calculating the length and width of the object in 2D. Once the measurements are available for the 2D detected object, the 3D car model can be mapped and rescaled to match the size of the 2D vehicle. The process of model mapping to 2D is explained in the next section.

4.1.2 3D-2D Projection

To map the 3D vehicle model to 2D, a pinhole perspective is assumed to simplify the perspective projection computations. Although the pinhole model is quite a straightforward method, it provides an acceptable approximation for the imaging process [11]. The 3D model can be treated similar to a real-world 3D vehicle and thus use the same projection matrices to map a real-world object to image coordinates.

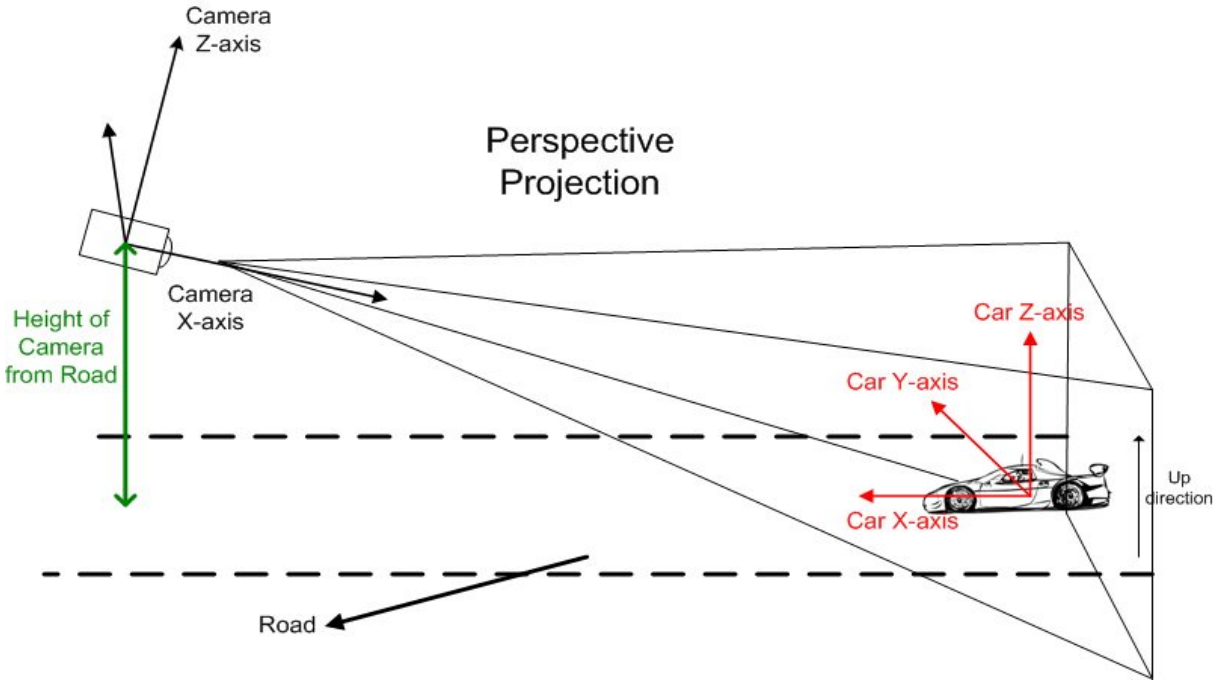

Figure 4.2: 3D model to image coordinates illustration.

Similar to Figure 4.2, the wireframe vehicle model consists of a group of edge endpoints in the world coordinates (Car X-axis, Car Y-axis, and Car Z-axis) that need to be projected unto the camera coordinates (Camera X-axis, Camera Y-axis, and Camera Z-axis) and then project the latter to image coordinates (x,y) whereby one dimension is lost. To project world coordinates to camera coordinates, some translations and rotations should take place.

Let's assume there is a point on an object with coordinates ($X1$,$Y1$,$Z1$) with respect to the world coordinates (X_W,Y_W,Z_W). If this point is translated by dx,dy, and dz respectively, the resulting new coordinates of the point ($X2$,$Y2$,$Z2$) will be as shown in equations 4.2 to 4.4 [42].

$$X2 = X1 + dx \tag{4.2}$$

$$Y2 = Y1 + dy \tag{4.3}$$

$$Z2 = Z1 + dz \tag{4.4}$$

Those equations can be re-written in matrix form as shown in equations 4.5 and 4.6.

$$\begin{bmatrix} X2 \\ Y2 \\ Z2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X1 \\ Y1 \\ Z1 \\ 1 \end{bmatrix} \tag{4.5}$$

$$\begin{bmatrix} X2 \\ Y2 \\ Z2 \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} 1 & 0 & 0 & \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{bmatrix} X1 \\ Y1 \\ Z1 \\ 1 \end{bmatrix} \tag{4.6}$$

Assume a segment with the endpoint of ($X1$,$Y1$,$Z1$) is rotated about an angle of θ around the Z-axis. The final rotation matrices are shown in equations 4.7 and 4.8 [42].

$$\begin{bmatrix} X2 \\ Y2 \\ Z2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X1 \\ Y1 \\ Z1 \\ 1 \end{bmatrix} \tag{4.7}$$

$$\begin{bmatrix} X2 \\ Y2 \\ Z2 \\ 1 \end{bmatrix} = R_\theta \begin{bmatrix} X1 \\ Y1 \\ Z1 \\ 1 \end{bmatrix} \tag{4.8}$$

The final general equation that is needed to translate the world coordinates to the camera coordinates as illustrated in Figure 4.3 is shown in equation 4.9.

$$\begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix} \tag{4.9}$$

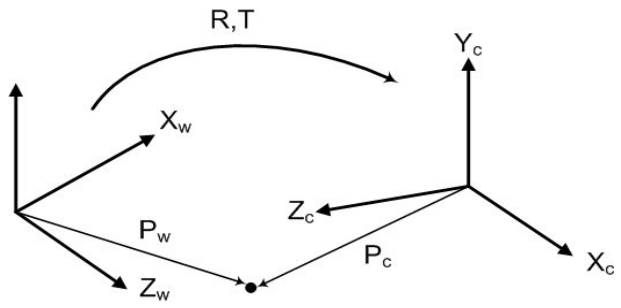

Figure 4.3: 3D Object Point converted from World Coordinates to Camera Coordinates.

Having converted the world coordinates to camera coordinates, the next step is to project the camera coordinates unto the image. Let point P in camera coordinates be represented by (x,y,z) for pinhole camera. Let f be the focal length of the camera and x',y' be the equivalent image coordinates of (x,y,z). Figure 4.4 shows a diagram of the projections and equations 4.10 and 4.11 state the final projections from the camera to image coordinates [11][46][42].

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1/f & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{4.10}$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = C \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{4.11}$$

Thus, concatenating the previous matrices for world to camera and camera to image projections would result in the general equation that is used to project real world objects to image coordinates as stated in equation 4.12.

$$[x'] = C\,[R|T] \begin{bmatrix} X \\ 1 \end{bmatrix} \tag{4.12}$$

Equation 4.12 is the central form that was applied in the work presented in this thesis where X is the 3D vehicle model being projected to a 2D image. The focal length of the camera was assumed to be known because it is easily obtainable from the camera specifications as long as the camera is known. This solves the camera

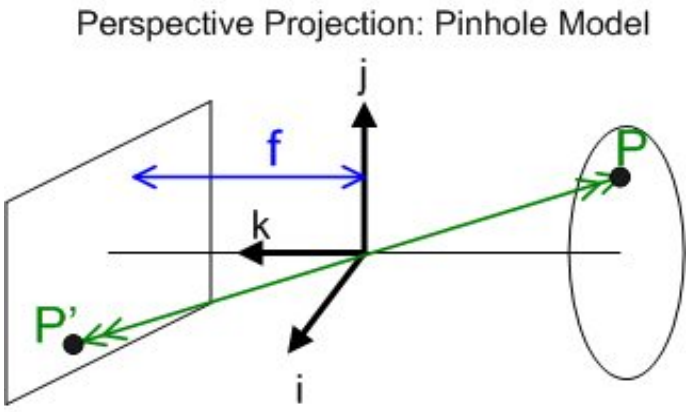

Figure 4.4: 3D Camera Coordinates to 2D Image Coordinates.

to image projection. As for the rotations and translations needed for the world to camera projections, there are basically three different rotations — θ around Z-axis (which has been mentioned in equation 4.7), β around Y-axis, and α around X-axis. The most vital one to compute is the rotation around the Z-axis (θ) because it determines the direction which the car is traveling in. β configures the amount of tilt of the car to its side (refer to Figure 4.1). As for α, this adjusts the amount the car is tilting forward or backward (refer to Figure 4.1). In general, the least that changes among different data sets is the rotation around X-axis, α, because the camera is mounted on a pole looking downwards with mostly almost the same range of tilt. Thus, this angle is assumed. Furthermore, to simplify the amount of iterations needed, the rotation around Y-axis, β is also assumed. The only angle that is determined per frame when matching the best 3D model to the car is the θ angle. The matching process will be explained in more details in the next section. Therefore, the following equations 4.13, 4.14, and 4.15 illustrate the rotations around Z-axis, Y-axis, and X-axis respectively used in the MATLAB code to compute the image coordinates of the 3D car model.

$$R_\theta = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.13}$$

$$R_\beta = \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.14}$$

$$R_\alpha = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.15}$$

The above rotations are then used to calculate the translations needed to project the 3D car model unto the detected object in the image obtained in the first module — object extraction. The object extraction module detects the object and returns the object's location. The location of the object attained by the first module is the destination at which the model projection should be overlapped. The 3D model location is known because it was designed from scratch. Therefore, the distance increment needed to translate the 3D model object to the location of the 2D detected object can be derived by subtracting one of the rotated coordinates of the 3D model from the final destination of the detected object in the image. Those translation and rotations are then used to map the model to image.

4.2 Model Instantiation

In the previous section, the 3D generic wireframe model was explained and the 3D to 2D projection equations used were stated. In this section, the edge information extracted from the image per detected object is discussed and the best-model-to image matching process is explained.

4.2.1 2D Vehicle Edge Detection

The generic polyhedral 3D model used to describe a vehicle is essentially composed of a group of connected vectors. Therefore, to match the 3D wireframe vehicle model to the detected object, the best route would be by comparing the edge elements obtained of the foreground object since it is only logical to compare features of the same type.

Once an object is detected by the first module, the object detected within the bounding box is converted from a color image to a greyscale image using Otsu's method [36]. A Canny edge filter [8] is used on the greyscale equivalent of the object within the bounding box to extract the different edges. An edge detector detects different major local changes in an image's intensity levels which are quantified by the image gradients [1]. In the case of a Canny filter, the gradient is computed through the use of a Gaussian Filter derivative. The reason for using the Canny edge detector is due to its good noise invariance and minimal error [8]. The Canny edge detector that was applied during implementation was that of range between two thresholds, Υ_1=0.1 and Υ_2=0.2, whereby the thresholds are required to find the strong and weak edges which are kept in the output only if they are attached to strong edges [8]. This method ensures that very weak independent edges are eliminated from the result. The sigma used to specify the standard deviation of the Gaussian filter was chosen to be 2 throughout the implementation because based on experimentation, 2 yielded the most satisfactory results for the purpose of this work. Figure 4.5 illustrates an example of an original image of a car and the result of a Canny edge detector applied to the image.

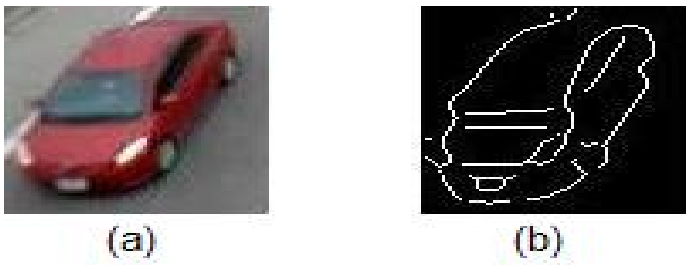

Figure 4.5: a) Cropped image the size of a bounding box around the detected car. b) Canny edge detector result of greyscale of left image.

After edge detection, the next step is to work on linking and grouping related edges and separating the rest. This process makes the matching of the model to image data more feasible. The process of edge linking was implemented through the use of Kovesi's functions for *Edge linking and line segment fitting* [25]. The function links the edge pixels to each other to form a list per each edge contour. Contours that amount to being smaller than five pixels long are eliminated. Each independent contour is then colored in a unique color as shown in Figure 4.6.

The obtained contours can then be fitted using line segments (curves are approximated by lines) and stored to be later compared with the model lines. The reason for fitting the contours with lines is simply to make comparison of the model

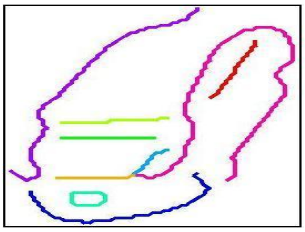

Figure 4.6: Edges obtained in right image of Figure 4.5 are linked and distinguished as unique contours.

edges more practical with that of the image. The function used to obtain the line filled segments are also part of Kovesi's functions [25]. Figure 4.7 depicts the result of line segmenting the image shown in Figure 4.6.

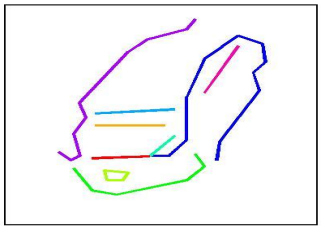

Figure 4.7: Line fitted segments.

With regards to model scaling, the 3D projected model is scaled with respect to the bounding box attained around the detected foreground object by the first module. Therefore, whenever there are scale changes within a test set, the detection phase identifies the object's size and based on this information the projected 3D model is changed with accordance. This method ensures that the system functions under scale changes unlike works where the sizes are assumed constant.

To conclude, this sub-section elaborated on the methods used to extract the detected object edge information and link them in such a manner as to make the edges ready to be matched against the projected 3D car model edges. In the next sub-section (4.2.2), the model matching technique is investigated.

4.2.2 Model to Data Matching

To match the best 3D model to the detected object edges, the 3D model is projected, scaled, and then iterated over a range of rotations around the Z-axis until the best match to the detected object's edges is found. The approach investigated

for matching is Hausdorff method [40]. The method is discussed in the following section. Toward the end, the model matching results are presented.

Hausdorff Method Theory

The method examined for the purpose of matching is the Hausdorff distance measure. The Hausdorff distance is characterized by being the distance between two point sets where one is the model edge point set and the other is the image edge point set [40]. The Hausdorff measure computes the extent to which each point of an image set is close to a point on the model and vice versa [19]. The Hausdorff distance can be applied to this work to find translated and scaled instances of a model vehicle even if the image includes multiple objects, noise, occlusion, and specious features [40]. The 3D projected model is iterated over a range of different Z-axis rotations while computing the Hausdorff distance at each angle. The minimal Hausdorff distance to the detected object can be considered the best candidate for the foreground car. The basic form of the Hausdorff distance can be defined as in equations 4.16-4.17 [18].

Let M be the model point set and I be the image point set.

$$D_H(M, I) = \max(h(M, I), h(I, M)) \tag{4.16}$$

$$h(M, I) = \max_{m \in M} \min_{i \in I} |m - i| \tag{4.17}$$

where $\|.\|$ is the L^2 norm.

The basic meaning behind the above equations is that $h(M, I)$ can be calculated by taking every point in M and getting its distance to the closest point in I and obtaining the maximum of the set of obtained distances. Then, going over a similar procedure for computing $h(I, M)$. To get the Hausdorff distance, the maximum of the two, $h(M, I)$ and $h(I, M)$, is returned [40].

Model Matching Results

The model and image edge points were used as features to be matched up by the Hausdorff distance method previously covered. The location and scale of the detected object was obtained from the first module and therefore the translation that the scaled 3D model had to undergo could be calculated. The vital point

was to match the vehicle model to the image car with the correct orientation. Furthermore, the matching process must also be able to distinguish between vehicle and non-vehicle foreground objects.

A total of 90 images were used to test the Hausdorff measure performance. The reason why only 90 images were used to test the method is because it is further tested later on in Chapter 6 when the overall system is examined. The Hausdorff distance was applied on different training images prior to the testing to determine an estimate for a threshold that would separate the foreground objects into vehicles and non-vehicles. Other images were then used to test the Hausdorff matching process on vehicle and non-vehicle objects. Figure 4.8 illustrate some sample test sets used.

Table 4.2 reports the results obtained from testing the 90 images on Hausdorff matching method. The first and second columns state the type of images (non-vehicle and vehicle) and their number, respectively. The third and fourth columns declare the number of images as a percentage that were classified as vehicle or non-vehicle.

Table 4.1: Hausdorff Matching Results

Image Type	Total Image No.	Classified as Vehicle(%)	Classified as Non-vehicle(%)
Vehicle	60	86.7%	13.3%
Non-vehicle	30	10.0%	90.0%

Figure 4.8: a) Sample vehicles. b) Sample non-vehicles.

Table 4.1 reflects some advantages of using the Hausdorff method. Besides being a convenient method to compare sets of edge points and being robust in spite

of occlusion, noise, and multiple objects (as previously mentioned), it also has an overall performance of successfully classifying objects into vehicle and non-vehicle of around 88% and is also able to overlap the vehicle model with the correct orientation. Figure 4.9 illustrates additional images of successfully overlapped models based on the best chosen model by Hausdorff.

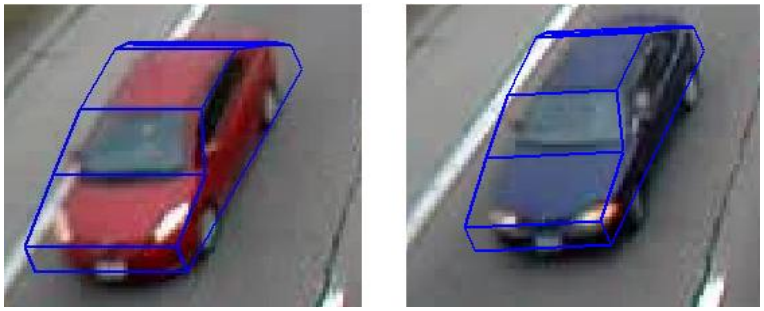

Figure 4.9: Additional Examples.

However, the major drawback of the Hausdorff method is its computational complexity. If two polygons are being matched with n and m vertices, then the total computational complexity will be $\mathbf{O}(n+m)$ [13]. The system can still be made to work in real-time if frames are skipped (not every frame is processed). Skipping frames would decrease the amount of processing needed as well as eliminate the need to repeat the processing for almost identical images (since the change between consecutive frames is very minute). In other words, if there are F frames in a video sequence and each frame has P polygons to match to the car-model where each polygon match has a computational complexity of $\mathbf{O}(n+m)$, then the total computational complexity for the video sequence will be a function of F,P,n, and m. Since P,n, and m, can not be altered (they are all needed for comparison), F can be decreased — for instance to $F/3$ — to decrease the overall time it takes to process the video sequence. Furthermore, by skipping frames, the system has more time to finish processing of the current frame before it receives a new frame to process. In what follows, the process of iteration around the z-axis will be depicted graphically (Figure 4.10) and then a graph that shows the computational times versus the degree range covered in the search for the best match angle will be included in Figure 4.11.

To illustrate the process of iterating over different θ values around the z-axis and returning the best match based on Hausdorff method — the best match is the one that has the smallest Hausdorff distance — Figure 4.10 is a series of images showing the different plausible rotations and the best match chosen for a single car.

The default angle is the angle that can be computed from the initial orientation estimate of the foreground object obtained from the first module (refer to chapter 3). A search around the default angle takes place over a search area ranging from default–ρ to default+ρ where ρ is specified by the user. Experimentally, the best search range was found to be in the span of 35° around the default angle which was set as a constant in the rest of the surveillance system implemented.

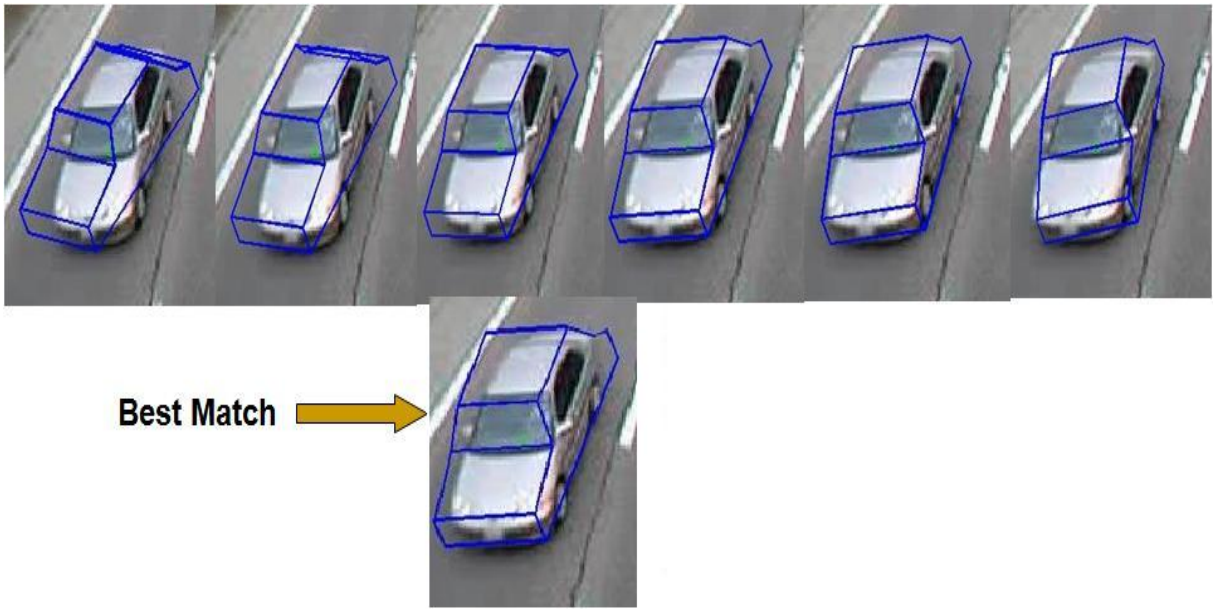

Figure 4.10: Iteration around z-axis and best chosen match.

If more than one car is in the image, the same process of matching the projected 3D model to the edges of a vehicle in the 2D image takes place for each detected vehicle. The system loops throughout the total number of detected vehicles in the current image [$i = 1 : M$, where M is the total number of cars in the current image], at each iteration performing the Hausdorff matching procedure for the current car (i), classifying, and assigning the best found orientation of the projected 3D vehicle model to the vehicle edges of the current car. Then, the loop increments, $(i + 1)$, until all the vehicles are matched with a model.

Different search ranges were attempted and the computational times needed to process the first and second modules of the system are recorded. As the search area increases, the computational time in seconds per frame also increases. Figure 4.11 shows the relationship. A span of 0° means that the default value was processed only. The other ranges are the search areas around the default value and the default value inclusive.

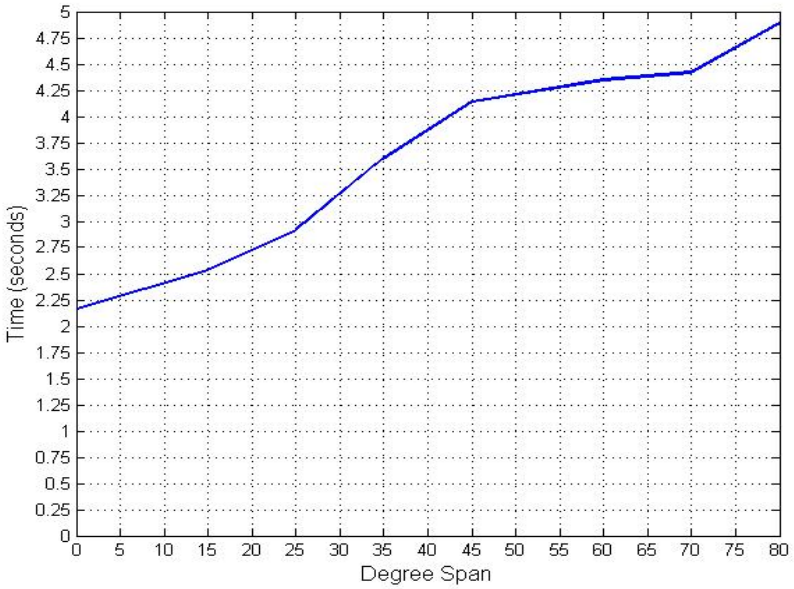

Figure 4.11: The graph illustrates the time in seconds needed to process each frame in Matlab (1.73 GHz single core processor) with respect to the range of degree values tried in the Hausdorff matching process before arriving at the best orientation. The recorded times include the processing of the first module — object extraction — and is thus the total of the time needed to process the first and second modules. According to the graph, as the search for the best angle increases, the processing time also increases reaching up to 5 seconds for a large search area of 80 degrees around the initial default angle.

4.3 Chapter Summary

This chapter elaborated on the second module of the system — foreground recognition module. The first section (Section 4.1) explicated the generic 3D vehicle model used and the needed projections to convert it to 2D. The 3D vehicle model used in this work is based on a generic wireframe model constructed from using 11 vectors to describe the length, height, and width of different sections of the car. The 3D model is then projected onto a 2D image using the equations stated in sub-section 4.1.2. Afterward, the information extracted (namely edges) about the foreground object was discussed in section 4.2. The Canny edge detector that was used in this work was briefly explained. Finally, the Hausdorff matching method used to match the 3D projected model to the image edge information was clarified in section 4.2.2 and the results of the matching were shown. In Chapter 5, the third module (tracking) will be examined.

Chapter 5

Object Tracking

In the previous chapters, the first two modules — object extraction and object recognition — responsible for detecting, recognizing, and classifying the object were discussed. In this chapter, tracking, the third and final module of the surveillance system, is explained. In the first section, the tracking algorithm implemented is explicated. Later, the results of the tracking alone are briefly illustrated with a short discussion.

5.1 Tracking Algorithm

Since multi-object tracking is basically an assignment problem [20], the central concept used for multi-vehicle tracking in this surveillance system is based on matrix associations that relate previous car information to the cars found in the current frame. The problem is formulated similar to that of an optimization problem whereby the overall cost function is being minimized. Next, the different inputs are first mentioned, the different information taken into account in the optimization process are discussed, and finally the general cost function is stated. Figure 5.1 is a summary of the algorithm described.

After the second module — object recognition — is processed, the classification information pertaining to each foreground object is available. The foreground objects that are classified as vehicles are kept to be processed in the third module (tracking). For each of these remaining vehicles, information from the first module regarding the vehicles' numbers, areas, RGB histograms, and locations is passed along with the information attained from the second module regarding the best chosen orientation of the vehicle model and the classification type (vehicle). This

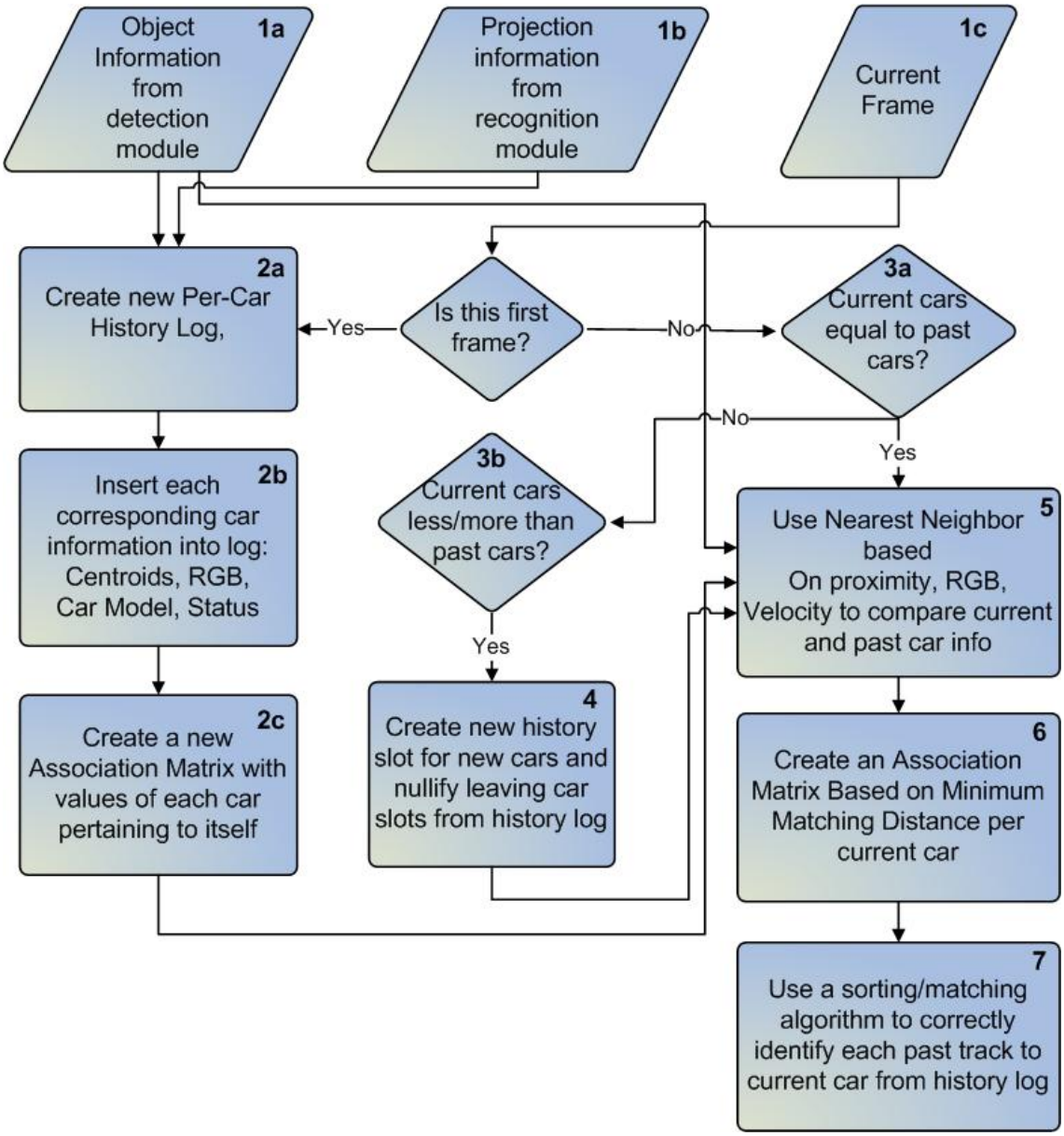

Figure 5.1: Diagram of the tracking algorithm.

process takes place for every new frame. The information obtained from each of these modules is shown in blocks 1a and 1b in Figure 5.1.

For the current frame, the system asks itself whether this is the first frame it receives or not (block 1c in Figure 5.1). If the current frame is the first frame, then a new history log is created per each detected object classified as a vehicle (block 2a). The history log is essentially a matrix of size $m \times n \times p$, where for each point in time (i.e. per frame), each row belongs to a particular car and the columns store the different information of the car (RGB, centroid, type, car status (is the car still active? did the car leave? is the car just entering?), ... etc). An example of a history log is illustrated in Figure 5.2. Since this is the first frame, there would be no past information and the only information that is entered in the history log is the current information (block 2c).

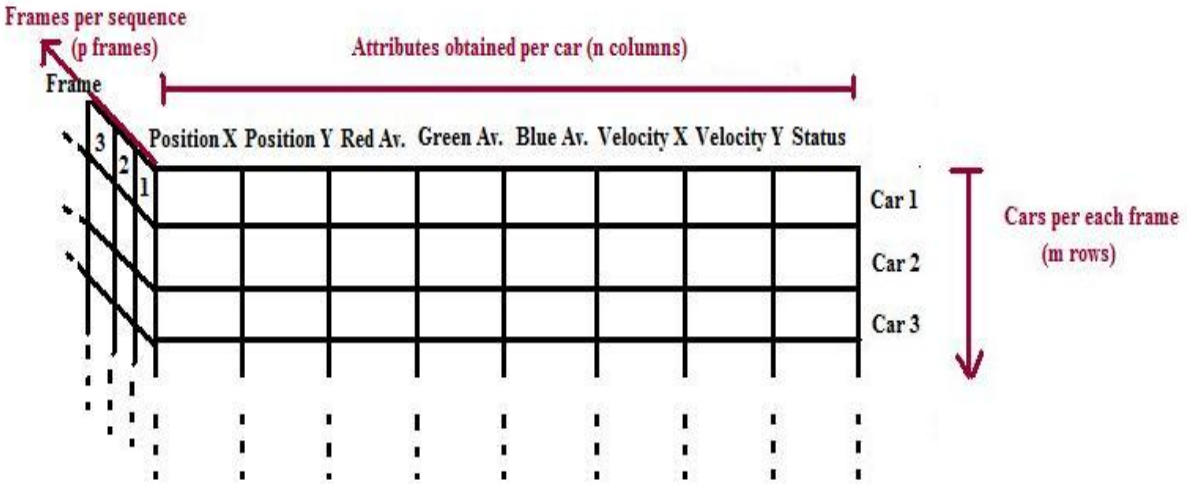

Figure 5.2: Example of a history log illustrating the attributes inserted in the columns corresponding to each car (row). This is assimilated for all frames (p frames).

If the current frame is not the first frame, the current cars along with their information are detected and stored in a temporary matrix. At this point, it is important to find out whether there are any new cars in the scene or any cars that have left the scene. The current number of detected vehicles is compared to the number of vehicles in the previous frame. If the cars are equal in number and no cars have simultaneously just entered and left the scene (block 3a), then the next step would be minimizing the cost function by applying nearest neighbor classifier (block 5). Otherwise, if the cars in the previous frame are not equal to the number detected in the current frame, this means that there should be some new cars that

entered the scene or left the scene. The car that has left the scene is identified from the previous frame information and is eliminated from the matching process with the current cars. Furthermore, the car that just entered the current frame is also eliminated from the matching process to the previous cars since it is assigned a new slot in the history log.

There are only limited ways a car can enter or leave the scene and those constitute of crossing the edge of the image such that it is out of camera view. Thus, the locations of the cars in the previous frames are scanned to determine if there were any entering cars or leaving cars. To simplify the problem, prior knowledge about where the cars leave the scene and where they enter the scene (i.e. prior knowledge about the image extremities as shown in Figure 5.3) was assumed. Similar assumptions were made about two-lane roads.

Figure 5.3: Example of entering and leaving regions.

If a car is found to be leaving the scene, its status is changed to inactive to eliminate it from the association process in the next frame. If a car is entering, a new slot is added for it in the history log with all the new information about it. Its status is recorded as 'entering car' in order to eliminate it from the association process for the current frame (since it did not exist before). Only the cars that existed previously undergo the association process (block 5).

In case there are fewer cars in the current frame than the previous frame and no car had left the scene since the previous frame, it means that there is a possibility that there was a car that got occluded currently. For this reason, the car that has disappeared from the previous frame is identified by having the worst match to all the current cars. This car is saved aside along with its information in case an occlusion is occurring. Later on, when the car reappears, a case occurs where more cars are in the current frame than in the previous frame. If no car had entered the scene through the edges of the image, the new car is isolated and labeled as the

occluded car. This is then linked to the car that was labeled as occluded when a car suddenly disappeared. Such linking compensates for the occlusion over a few frames even if there have been failures in the detection module (did not detect car) and ensures that the tracks are continuous over the video sequence. However, if a car does not reappear after a few frames, the information about the occluded car is deleted.

The association process is a minimization process of the cost function that takes place by using nearest neighbor. A Euclidean distance taking into consideration all the weighted car attributes is applied on each pair of cars — one past car and one present car — in order to determine which two cars are associated. Assume there are P past cars and C current cars.

The cost function to be minimized that was created and implemented in this thesis is shown in equation 5.1. X and Y are the coordinates of the vehicle's centroid, R,G and B are the average colors, V_x and V_y are the velocity components, and av means average of color intensity for vehicle i or j. The position of the vehicle gets the highest weighing, followed by the RGB values, and then by the velocity.

$$\begin{aligned} cost(i,j) = (w_{1,1}X_i + w_{1,2}Y_i + w_{1,3}R_{avi} + w_{1,4}G_{avi} + w_{1,5}B_{avi} + w_{1,6}V_{xi} + w_{1,7}V_{yi})^2 \\ -(w_{2,1}X_j + w_{2,2}Y_j + w_{2,3}R_{avj} + w_{2,4}G_{avj} + w_{2,5}B_{avj} + w_{2,6}V_{xj} + w_{2,7}V_{yj})^2 \end{aligned} \tag{5.1}$$

where $i \leq P$ and $j \leq C$

Additional constraints are as follows:

$$P \geq 0; C \geq 0 \tag{5.2}$$

$$0 \leq w_{1,1}, w_{1,2}, w_{1,3}, w_{1,4}, w_{1,5}, w_{1,6}, w_{1,7} < 1 \tag{5.3}$$

$$0 \leq w_{2,1}, w_{2,2}, w_{2,3}, w_{2,4}, w_{2,5}, w_{2,6}, w_{2,7} < 1 \tag{5.4}$$

$$\sum_{k=1}^{7} w_{1,k} = 1 \tag{5.5}$$

$$\sum_{g=1}^{7} w_{2,g} = 1 \tag{5.6}$$

Therefore, the constraints state that the summation of the weights should be maximum 1 (with none of the weights being 1) and that the number of cars cannot

logically go below 0. The cost function is solved by using a nearest neighbor search (or proximity search) which basically looks for the closest point in the metric spaces. The associations found are later recorded in a temporary matrix (block 6).

For each pair of associated vehicles (current and past), the past vehicle's information is retrieved from the history log. Since the association basically means that it is the same car in a new position, the new (or updated) information about the current position of the car is added in the correct history slot. This searching and sorting step is vital for the process of keeping the tracks correct. If the vehicle is recorded in the wrong slot, the plotted tracks would end up being incorrect (block 7).

A pseudo code for the tracking algorithm is outlined in Algorithm 1.

Algorithm 1 Tracking Module Pseudo code

Require: Current image, number of vehicles detected, information extracted about each vehicle
  if first frame of video sequence **then**
    for each detected car in current frame **do**
      INSERT car information (centroid, RGB values, velocity) into a row of the history matrix
    end for
  else
    if Number of detected cars in current frame = Number of detected cars in previous frame **then**
      RUN ALGORITHM FOR EQUAL CASE
    else if Number of detected cars in current frame > Number of detected cars in previous frame **then**
      RUN CURRENT CARS MORE THAN PREVIOUS ALGORITHM
    else
      RUN PREVIOUS CARS MORE THAN CURRENT ALGORITHM
    end if
  end if

Algorithm 2 EQUAL CASE ALGORITHM

CHECK if any cars are entering or leaving
if No cars are leaving or entering **then**
  for each detected car in current frame **do**
    FIND best match to car in previous frame using minimization of cost function
    INSERT current car information to matched car information in history matrix
  end for
else if A car has left the image and another car entered the image **then**
  FIND the cars that entered the image
  LABEL those cars as 'Entering Car'
  INSERT current entering car information to new row in the History matrix and eliminate them from matching procedure
  FIND the cars that left the image from previous frame
  LABEL those cars as 'Inactive' and eliminate them from matching procedure
  for each detected car in current frame **do**
    MATCH all remaining current detected cars to cars from previous frame based on minimization of cost function
    INSERT remaining current car information to matched car information in history matrix
  end for
end if

Algorithm 3 CURRENT CARS MORE THAN PREVIOUS ALGORITHM

```
CHECK if any cars entered the frame
if A car entered the scene then
    FIND the cars that entered the image
    LABEL those cars as 'Entering Car'
    INSERT current entering car information to new row in the History matrix
    for each detected car in current frame do
        MATCH all remaining current detected cars to cars from previous frame
        based on minimization of cost function
        INSERT remaining current car information to matched car information in
        history matrix
    end for
else if A car did not enter the scene then
    A possible case of occlusion took place in previous frame
    for each detected car in previous frame do
        MATCH all previous detected cars to cars from current frame based on
        minimization of cost function
        IDENTIFY car with worst match and label it as 'OCCLUDED'
        CHECK OCCLUDED-matrix to check if this car was OCCLUDED previ-
        ously
        if The car was occluded then
            Match the occluded cars to each other to create a continuous track
        else
            An error has occurred.
        end if
        INSERT remaining current car information to matched car information in
        history matrix
    end for
end if
```

Algorithm 4 PREVIOUS CARS MORE THAN CURRENT ALGORITHM

CHECK if any cars left the frame
if A car left the scene **then**
  FIND the cars that left from the previous image
  LABEL those cars as 'Leaving Car'
  IDENTIFY the car that has left from previous cars and eliminate it from matching process
  for each detected car in current frame **do**
    MATCH all current detected cars to remaining cars from previous frame based on minimization of cost function
    INSERT remaining current car information to matched car information in history matrix
  end for
else if A car did not leave the scene **then**
  A possible case of occlusion is taking place in current frame
  for each detected car in current frame **do**
    MATCH all current detected cars to cars from previous frame based on minimization of cost function
    IDENTIFY car with worst match and label it as 'OCCLUDED'
    INSERT OCCLUDED car information into OCCLUDED-matrix
    INSERT remaining current car information to matched car information in history matrix
  end for
end if

5.2 Brief Discussion and Results

In the previous section, the algorithm was described in detail. In this section, the general advantages and drawbacks are outlined followed by a few results.

A few of the advantages are listed below.

- The tracker is a multi-object tracker since it deals with proper object association.
- The tracker incorporates several important conditions. First, it deals with cars that are entering and leaving the scene by updating the history log appropriately. Furthermore, it is able to conclude that some false detection has occurred when it finds that the number of current cars are more or less than previous cars yet no car has just entered or left the scene. In this case, it matches up all the cars and the odd one is monitored like a new car.
- Using these different conditions (outlined in detail in pseudo code in section 5.1), the tracking module is able to identify if an occlusion has occurred for a few frames. At that point, it links the conditions of the car before total occlusion to the conditions when it leaves total occlusion state to form a continuous track.
- Since there is no prior prediction model embedded in the system, the tracker should theoretically deal with 'unpredictable' behavior such as changing lanes (although this case has not been tested because no such data was available).

Some drawbacks also exist.

- If there is a dense traffic with similar looking cars (similar RGB) at a close location and traveling at similar speeds, the association matrix might incorrectly assign one car to another.
- The tracking system depends on the previous two modules. Therefore, if the first module does not detect the vehicles or the second module misclassifies an object as a vehicle, then the tracker might also function incorrectly. Although the tracker does compensate and form continuous tracks for objects that undergo full occlusion for a few frames, if the full (100%) occlusion extends over several frames (eg. 5), then the tracking module will eliminate the object from being tracked. Note that the condition is full occlusion. As long as

partial occlusion of the car reappears, the tracker can continue tracking the object.

- Though the prediction models can be a drawback because using them would mean one is assuming knowledge of vehicle behavior, they also provide some support for cases of occlusion. If a tracker predicts that there should be a car yet detects none, it can compensate in cases of full occlusion if prediction is used. On the other hand, the recognition module is the one that handles cases of partial occlusion.

Figures 5.4 to 5.7 illustrate some sample results of the module. A few results are shown because more test evaluations are made in the next chapter.

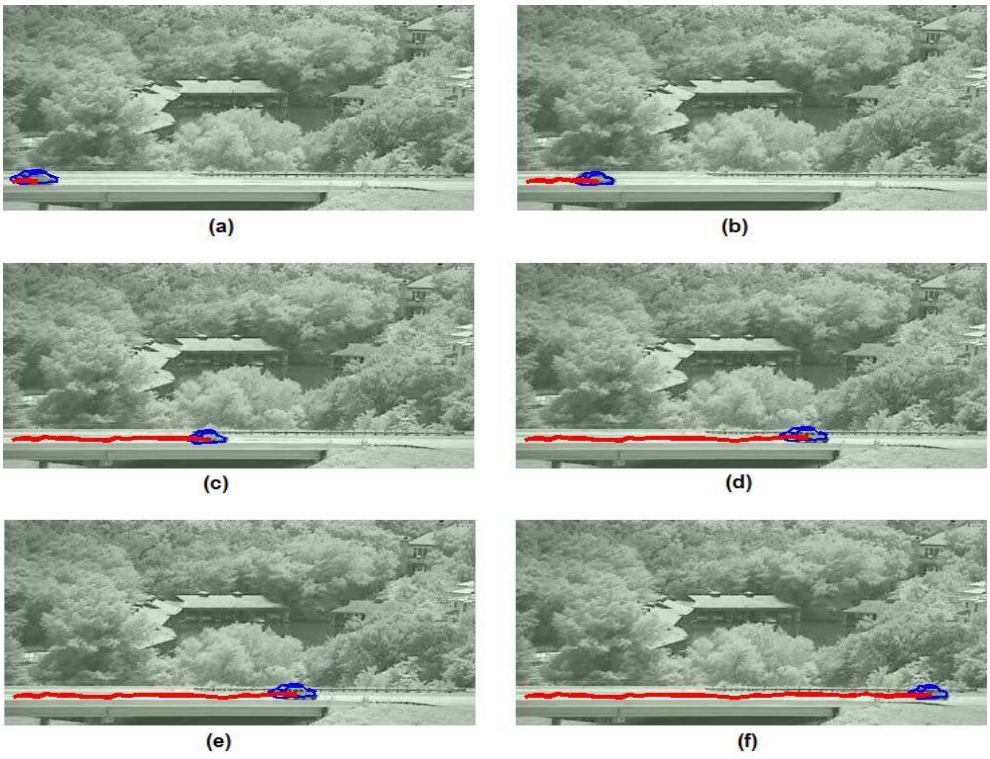

Figure 5.4: Sample frames from a video sequence of successful tracking of a far car in a monotone background.

5.3 Chapter Summary

In this chapter, the third major module — tracking module — of the implemented surveillance system was discussed in details. The method used to track is based on

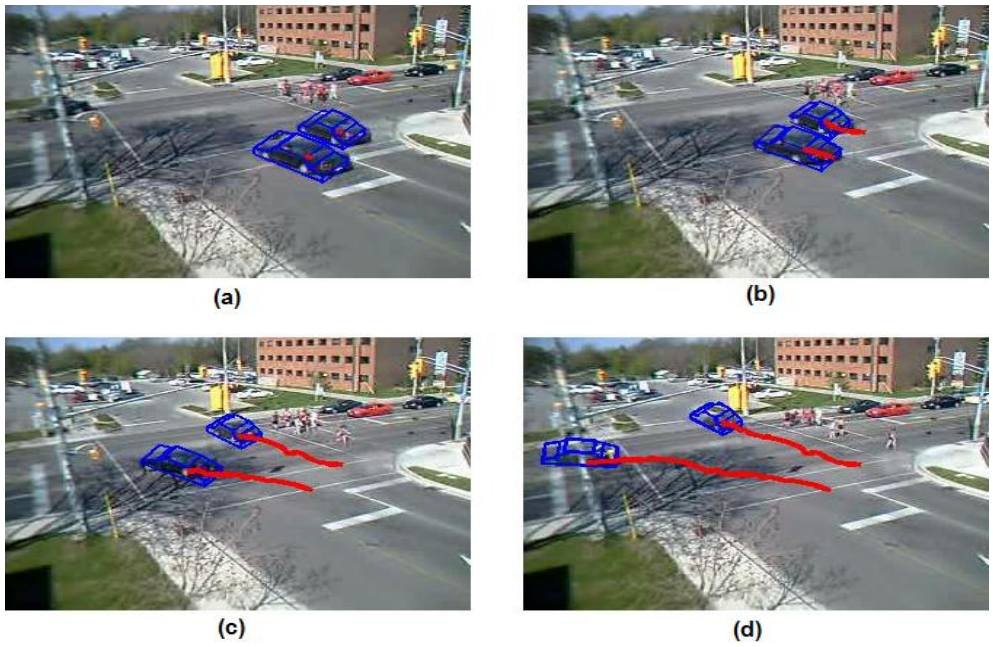

Figure 5.5: Sample frames from an intersection video sequence of successful tracking of cars.

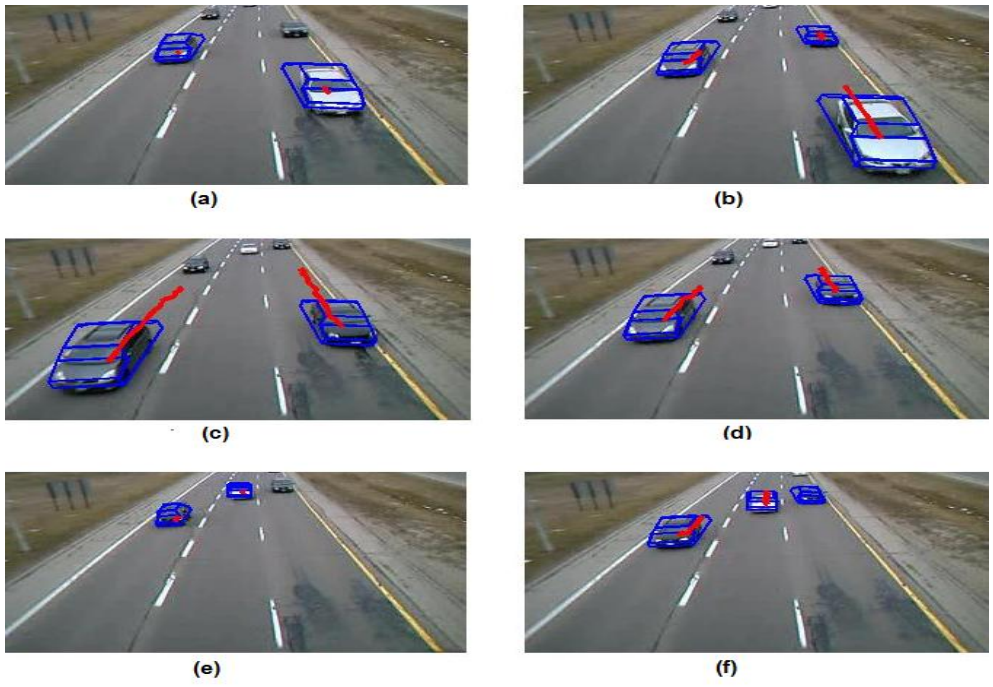

Figure 5.6: Sample frames from a zoomed in video sequence of successful tracking of cars.

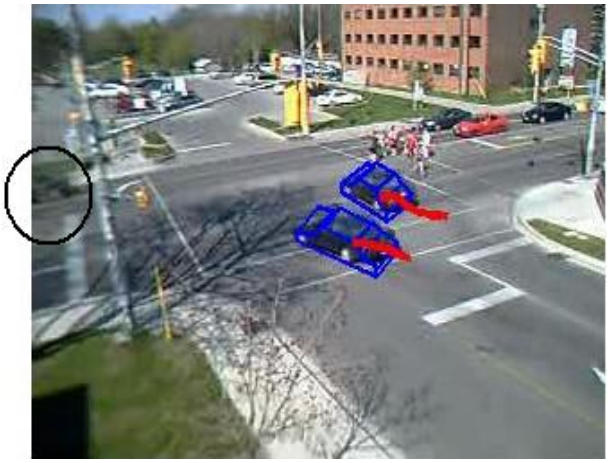

Figure 5.7: The vehicle in the circle was not tracked because it was not detected over several frames. Therefore, it was eliminated from being tracked by the tracker.

a deterministic matrix association approach whereby a cost function is minimized. The algorithm used was based on minimizing the euclidean distance of different vehicle attributes such as position, velocity, and average RGB values. The algorithm takes into consideration various conditions that might occur during tracking and solves each of those using the method explicated in the pseudo codes included in Section 5.1. Section 5.2 outlines the main advantages and drawbacks of the system and illustrates some examples of tracking.

Chapter 6

Overall System Results

In the previous chapter, the different modules of the surveillance system — object extraction, object recognition, and object tracking — were explained in detail. Some results relating to each individual module were shown in their respective chapters. This chapter focuses on presenting the results of the overall developed surveillance system. The first section of this chapter explains the types of data sets used and the second section discusses the grading scale used to assess the results. The third section presents the results that are evaluated based on the success rates in both normal conditions (daylight with original image noise and resolution) as well as altered conditions with respect to light conditions, image resolution, and image noise. Furthermore, the proposed system is also tested with respect to various road types, camera perspectives, and occlusion. The chapter ends with a revisit to the objectives of the surveillance system and a discussion of results.

6.1 Data Sets Used

The data sets that were used mainly constituted of video sequences of different types of roads at different camera perspectives. The camera was mounted on a pole above the different road varieties as depicted in Figure 6.1. The types of roads included highways and intersections. The camera perspective varied between being very close range or wide range for the highway data sets and high and middle altitude for the intersection video sequences. The images that follow illustrate sample images from each data set used.

The first set of images (Figure 6.2) illustrates the case where the camera is directed toward the highway at a wider angle away from the horizontal plane parallel

Figure 6.1: Example of a camera mounted on a pole above highways and recording video sequences (Source: Texas Transportation Institute).

to the road. In other words, the camera's viewpoint covers a shorter distance on the road and cars appear relatively larger than in cases where a larger area is viewed by the camera as shown in Figure 6.3.

Figure 6.2: Sample Frames from Video Sequence of Close-Range Perspective of a Highway (Source: Miovision Technologies Inc.).

The second set of images (Figure 6.3) show the data sets where the camera viewpoint covers a wider section of the highway. Therefore, in this case, the car scale changes from being very minute to clearly visible as it approaches the location of the camera.

The following sample images (Figure 6.4) are part of a data set in which the

Figure 6.3: Sample Frames from Video Sequence of Far-Range Perspective of a Highway (Source: Miovision Technologies Inc.).

camera tilt and viewpoint are very similar to that of the second data set; however, the highway design is more complex.

Figure 6.4: Sample Frames from another Video Sequence of Far-Range Perspective of a Highway (Source: Miovision Technologies Inc.).

In the following two data sets, the road type is an intersection. Two different camera heights are used. The first intersection case has the highest camera height (Figure 6.5), whereas the other one has medium camera height respectively (Figure 6.6)

The proposed system was also tested on a data set of a bridge at a distance. In that case, the cars appear small (few tens of pixels in length) and the viewpoint of

Figure 6.5: Sample Frames from Video Sequence of an Intersection from a high-mounted camera perspective (Source: Miovision Technologies Inc.).

Figure 6.6: Sample Frames from Video Sequence of an Intersection from a medium-height mounted camera perspective (Source: Miovision Technologies Inc.).

the bridge is sideways (Figure 6.7).

Figure 6.7: Sample Frames from Video Sequence of a bridge from a side-view perspective (Source: Aimetis).

The total number of test sets amounted to 114 different test sets with a total of around 19,000 frames . Those tests included 26 test sets with normal conditions, 4 test sets with different resolutions, 13 test sets with 6 different light conditions (78 test sets total), and 6 different test sets with varying image noise.

6.2 Grading Scale

For each of the following sections, the developed system is tested using various test sets with various road types, camera perspectives, scale change, image resolution, light conditions, and image noise. In order to determine the performance of the system, a grading system is developed and used for all the different tests. Several key points are crucial to examine: background subtraction (first module of the system), object recognition (second module of the system), tracking (third module of the system), overall performance, point of breakdown of the system, and reasons for a test failure. For each of the three modules, a 4 point grade system is used to determine the module's performance. Slightly similar to Nagel's grading system in [37], results are assessed as 'very good' (denoted as '4'), 'good' (denoted as '3'), 'tolerable' (denoted as '2'), and 'failure' (denoted as '1'). It is important to note that the data sets used did not have ground truth for the vehicles. Therefore, the tracking was assessed based on visual feedback. Table 6.1 illustrates the four grade scales in terms of each module.

Table 6.1: Grading System Explanation

Grading Scale				
Grade	Module 1: Background Subtraction	Module 2: Object Recognition	Module 3: Tracking	Overall Performance
'Very Good' or 4	Foreground segmentation encapsulates the whole vehicle and each vehicle is represented by an individual blob.	The car model candidate covers the visible vehicle very well.	A vehicle is correctly tracked throughout all the frames of a test sequence.	If all the previous modules take 4 or two out of the three modules take a 4 grade and one takes a 3 grade, then the overall performance of the system for the test set will amount to being 4.
'Good' or 3	Foreground segmentation encapsulates the majority of the vehicle and each vehicle is represented by an individual blob.	The car model candidate covers the visible vehicle with slight discrepancies or even a few larger discrepancies that have been corrected automatically.	A vehicle is correctly tracked over 85% of the test set frames.	If all the previous modules take 3 or two out of the three modules take a 3 grade and one takes a 3 grade, then the overall performance of the system for the test set will amount to being 3.
'Tolerable' or 2	Foreground segmentation encapsulates at least half of the vehicle and each vehicle is represented by an individual blob.	The car model candidate covers at least half of the visible vehicle with slight discrepancies or even a few larger discrepancies that have been corrected automatically.	A vehicle is correctly tracked over 65% of the test set frames.	If all the previous modules take 2 or two out of the three modules take a 2 grade and one takes a 1 grade, then the overall performance of the system for the test set will amount to being 2.
'Failure' or 1	Foreground segmentation does not detect the vehicle or is merged with another vehicle throughout the video sequence.	The vehicle has not been recognized and given a model.	A vehicle is correctly tracked less than 65% of the test set frames.	If all the previous modules take 1 or two out of the three modules take a 1 grade, then the overall performance of the system for the test set will amount to being 1.

Each of the previously explained grades are illustrated in Figures 6.8 and 6.9 where cases for each grade level are shown for the three different modules.

Figure 6.8: The left image is a case where the grade would be 4. The vehicles that are visible were well detected individually and recognized with the model well encapsulating them. Furthermore, the tracking has been correct. The right image is an example of a case when the grade would be 3. Both vehicles are well tracked (grade 4 for module 3) but the segmentation module gets 3 because the size of the model is not covering the whole vehicle and recognition model gets a grade 3 because the model overlap is slightly moved sideways.

Figure 6.9: The left image is an instance of a video sequence where the grade would be 2. The vehicles that are visible were well detected individually, however the size detected did not cover the whole vehicle (grade 2). The vehicles were recognized and a model was overlapped, but one of the vehicles was given a wrong model orientation (grade 1). The tracking is correct, yet slightly jagged (grade 3). However, the model corrects itself in later frames. The right image is a failure case due to the undetected car behind the red vehicle and the incorrect overlap of the model for the front most car.

6.3 Test Results

The previous sections have covered the data sets that are used to test the proposed system and the grading scale used to evaluate the system. In this section, the actual test results will be stated in the form of tables accompanied by sample frames from the test sets. Seven parts make up this section to assess the surveillance system with regards to different road types, different camera perspectives, scale changes, image resolution, light conditions, as well as image noise and occlusion. Test results reported in what follows are based on one generic vehicle model and thus classification results are not shown in those tables.

6.3.1 Different Road Types

Three major different road types were investigated in this thesis: highways, intersections, and bridges. The results obtained for the different road types are under normal conditions (no major illumination and day-light conditions). The image noise found in the test images are due to the real camera movement and resolution. Such type of noise is treated as part of the normal conditions. The image resolution used for the highway and intersection test sets is 320 by 240 pixels, whereas the bridge test set is of 352 by 208 pixels. In total 26 test sets were used for this part of the performance evaluation: 14 highway tests, 10 intersection tests, and 2 bridge tests.

Table 6.2 shows the results obtained for this part of testing. For each road type test, the frames of every case are assessed with respect to modules 1, 2, and 3. The number of frames that fall under each grade level (1 to 4) is recorded in the table. Then the overall performance is calculated based on the number of frames found per grade. If the majority of frames are under a certain grade (for example 4), then the general grade is given in the overall performance as 4 and the specific percentage of frames that have this grade is stated in parentheses. In other words, the overall performance is number of frames that obtained 'Very Good' divided by the total number of frames. Finally, if any of the frames failed, the general reason is given in the last column of the table.

Table 6.2: Different Road Types Results per Video Sequence

Different Road Types Test Results																
Road Type	Test Set No.	Total Frame No.	Module 1 Grade (in frames)				Module 2 Grade(in frames)				Module 3 Grade(in frames)				Overall Performance	Reason of Failure (if applicable)
			4	3	2	1	4	3	2	1	4	3	2	1		
Highway	Case 1	128	127	1	0	0	96	30	3	0	128	0	0	0	4 (91%)	N/A
Highway	Case 2	180	132	48	0	0	138	30	12	0	180	0	0	0	4 (83%)	N/A
Highway	Case 3	128	124	4	0	0	83	44	1	0	120	8	0	0	4 (85%)	N/A
Highway	Case 4	78	78	0	0	0	70	8	0	0	78	0	0	0	4 (96%)	N/A
Highway	Case 5	165	160	5	0	0	138	16	11	0	165	0	0	0	4 (93%)	N/A
Highway	Case 6	150	150	0	0	0	140	10	0	0	150	0	0	0	4 (98%)	N/A
Highway	Case 7	108	107	1	0	0	60	39	9	0	108	0	0	0	4 (85%)	N/A
Highway	Case 8	141	134	7	0	0	131	7	3	0	141	0	0	0	4 (96%)	N/A
Highway	Case 9	123	123	0	0	0	115	8	0	0	123	0	0	0	4 (98%)	N/A
Highway	Case 10	88	76	12	0	0	72	16	0	0	80	8	0	0	4 (86%)	N/A
Highway	Case 11	168	164	2	2	0	144	12	12	0	168	0	0	0	4 (94%)	N/A
Highway	Case 12	150	140	6	4	0	135	6	9	0	150	0	0	0	4 (94%)	N/A
Highway	Case 13	198	185	13	0	0	155	30	12	1	198	0	0	0	4 (91%)	Failure in car model overlap.
Highway	Case 14	63	60	3	0	0	50	9	4	0	63	0	0	0	4 (91%)	N/A
Intersection	Case 1	288	277	2	0	9	264	12	3	9	279	0	0	9	4 (95%)	Failure in first module led to failure in second two modules.
Intersection	Case 2	129	121	5	0	3	83	28	6	12	129	0	0	0	4 (86%)	Failure due to car model size and orientation.
Intersection	Case 3	288	280	8	0	0	144	144	0	0	280	8	0	0	4 (81%)	N/A
Intersection	Case 4	288	247	4	4	33	189	68	31	0	252	6	0	30	4 (80%)	Failure in module 2 and 3 was a result of failure in the first module.
Intersection	Case 5	168	168	0	0	0	136	32	0	0	168	0	0	0	4 (94%)	N/A
Intersection	Case 6	88	80	0	4	4	64	8	9	7	88	0	0	0	4 (88%)	Failure in first module led failure in second module.
Intersection	Case 7	247	243	0	0	0	240	3	0	0	235	8	0	0	4 (98%)	N/A
Intersection	Case 8	126	107	2	3	14	108	4	5	9	126	0	0	0	4 (90%)	Failure in first module led failure in second module.
Intersection	Case 9	272	240	0	0	32	192	48	0	32	272	0	0	0	4 (86%)	Failure in first module led failure in second module.
Intersection	Case 10	150	150	0	0	0	148	2	0	0	145	5	0	0	4 (98%)	N/A
Bridge	Case 1	210	210	0	0	0	200	10	0	0	189	21	0	0	4 (91%)	N/A
Bridge	Case 2	220	198	5	5	2	190	10	7	3	217	1	0	2	4 (92%)	Failure in second two modules due to first module.

Therefore, overall the results demonstrate that the system works well under different road types. In turn, this makes the system practical as it may be mounted on any road type without any significant alterations needed for the system to run. Out of the 4275 frames tested over 26 different cases, the overall performance for all test cases in this section is around 95% with 90.8% being in the 'very good' range. The frames that have failed were mostly due to incorrect foreground segmentation. This undetected or incorrectly segmented vehicle resulted in an incorrect model instantiation and overlap. Furthermore, tracking (third module) is based on the previous two modules and thus also failed sometimes. Usually, failure in tracking occurs when a vehicle is not detected at all for many frames. As for failure in the second module, it can be due to several reasons: incorrect result from first module, car model orientation chosen to best represent the car is well off the actual car orientation, or the location/size of the car model is not overlapping the actual vehicle in the image.

6.3.2 Camera Perspective

For each of the different roads — highway and intersection — various camera perspectives are evaluated. The highway road ranged from long range to short range and the intersections varied from high mounted camera to medium height camera. The difficulty with the long range cameras is that at some point the cars are so small that they are not detected. For this reason, the detection, recognition, and tracking only starts after the vehicles become of visible size (toward third of the image in the long range case). As for the short range, the vehicles appear quite large; however, the complexity arises because with closeup perspectives, the cars pass through the scene at a high speed and many cars enter and leave the scene. Therefore, those cases are required to be taken care of. As for the high-mounted camera above the intersection, although the cars appear more distinct from each other, they are also small and harder to detect. As the altitude of the camera decreases, the view of the intersection becomes skewed and car-to-car occlusion increases significantly where in some tests a car may occlude another almost entirely.

For this portion of the testing, 8 different test sets were used to test long range highway cases, 6 test sets for short range perspectives, 6 various test sets for high mounted camera over intersection, and another 4 test cases for medium height camera perspective over intersection. Table 6.3 summarizes the results over the variety of tests. As in table 6.2, each module is subdivided into 4 different grades and the number of frames that fall under each grade level is stated. The overall

Table 6.3: Different Camera Perspectives Result

Different Camera Perspective Test Results															
Perspective	Total test sets	Module 1 Grade (in frames)				Module 2 Grade(in frames)				Module 3 Grade(in frames)				Overall Performance	Failed Frames (%)
		4	3	2	1	4	3	2	1	4	3	2	1		
Long Range Highway	8	989	44	6	0	862	127	49	1	1031	8	0	0	4 (92%)	0.03%
Short Range Highway	6	771	58	0	0	665	138	26	0	821	8	0	0	4 (91%)	0%
High Intersection	6	1173	19	8	49	880	292	49	28	1196	14	0	39	4 (87%)	3%
Medium Intersection	4	740	2	3	46	688	57	5	41	778	13	0	0	4 (93%)	5.5%

performance for each camera perspective is then stated with respect to the grade attained by the majority of the frames (and percentage of frames that attained that grade). The last column reports the average overall percentage of failed frames in each of the case types.

Based on the results in table 6.3, a note can be made that the general overall percentages are relatively close to each other ranging from 87% to 93% of all frames tested for being 'very good'. However, the difference between the tests shows when examining the number of failed frames. More failure occurs in the intersection tests than the highway cases. The intersection tests in general include more complex backgrounds; thus it is harder for the background subtraction module to extract the foreground object correctly. If analyzed with respect to different camera perspectives, the most change in failure occurs between the two camera perspectives in the intersection cases. As the camera height decreases, the error increases due to significant increase of car-to-car occlusions and change scale changes. Therefore, camera perspective is a vital component with surveillance systems. The higher the camera view, the lower error occurs due to a distinct view of the vehicles.

Figures 6.10 to 6.13 illustrate the different system results obtained with respect to different camera perspectives. Figure 6.10 shows the case of a closeup view of a highway. There are two cars that enter the scene and are correctly detected, recognized, and tracked until they leave the camera view. With such a perspective, the vehicles appear larger and the additional clutter due to background surrounding the roads is avoided. However, scale changes quickly from the beginning of the scene to the end. Scale change will be tested in more details in the next section.

Figure 6.11 is the case when the camera perspective of the highway becomes

long-range. It should be noted that for the first third of the scene the vehicles are ignored because they are too far away and their correct detection becomes extremely tedious. The background in this scene is more difficult than the first case (Figure 6.10) and the scale change is more drastic. However, the cars remain in the scene for larger amount of frames.

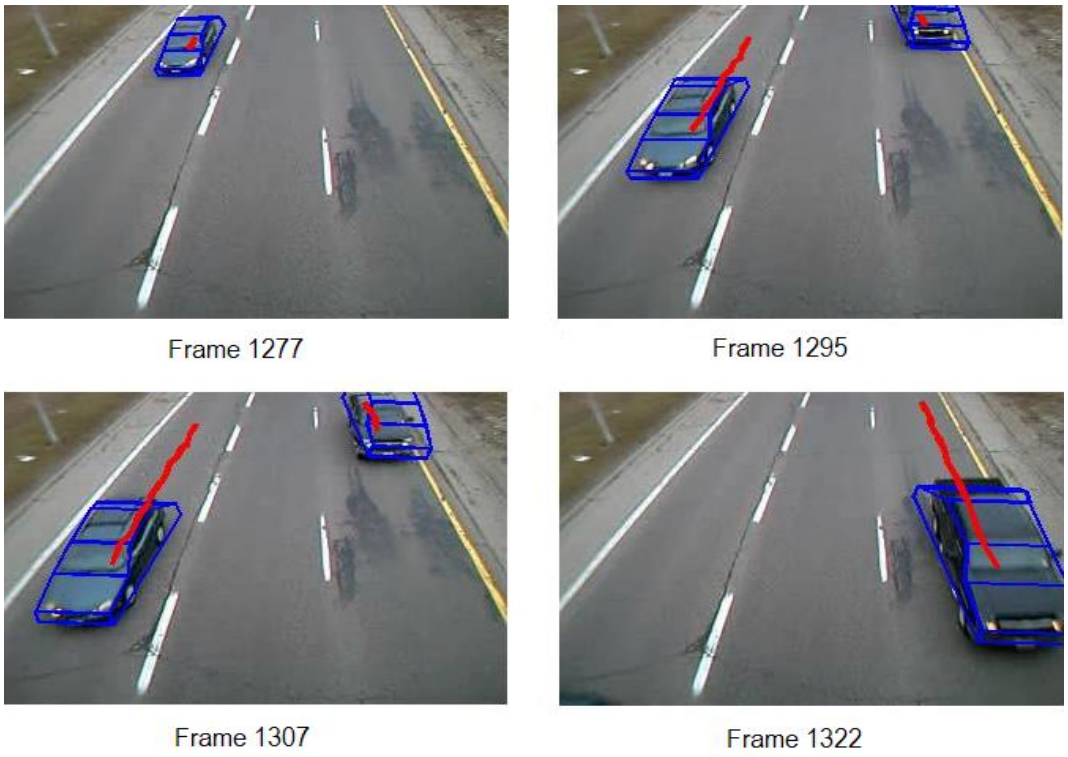

Figure 6.10: Close Range Camera Perspective Example.

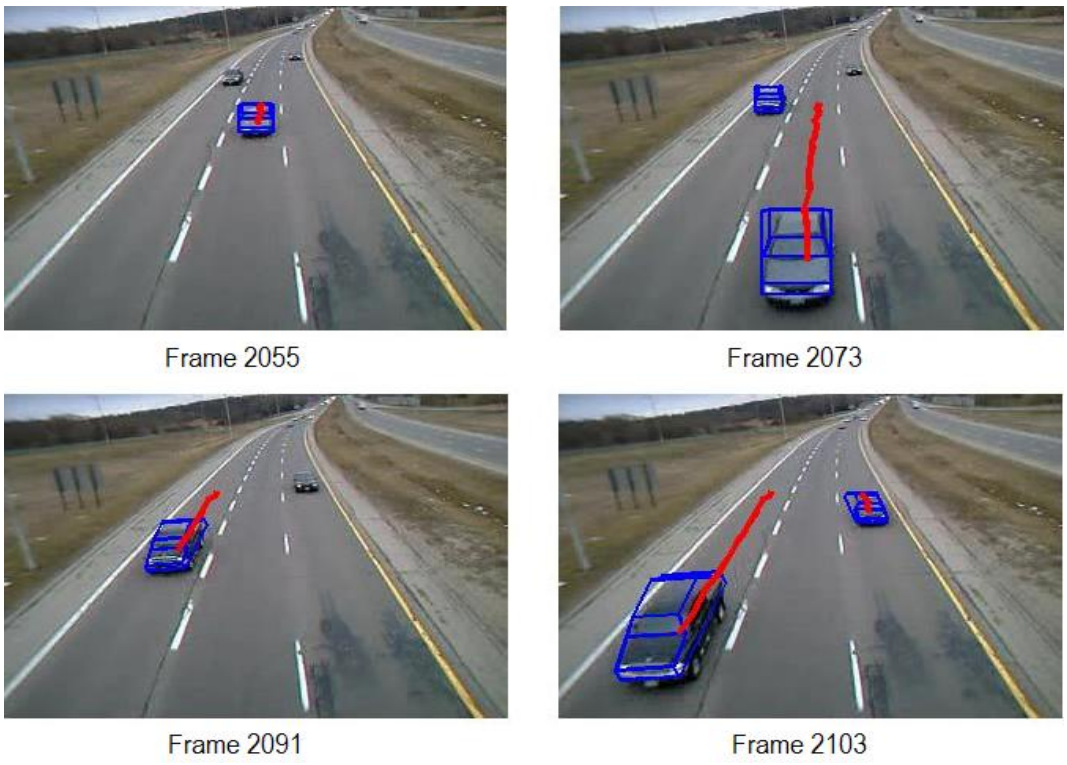

Figure 6.11: Long Range Camera Perspective Example.

Figures 6.12 and 6.13 are both of intersection type. In both cases the background is more complex than the first set of images. However, among the two intersection cases, more occlusion appears to be in Figure 6.13 (naturally from the trees, poles, and urban construction) as well as when two cars pass each other. For example, in Frame 1811 of Figure 6.13, the white car is occluded by the tree branches. In Frame 1835, the white car is approaching to occlude the red car. On the other hand, two cars can pass each other with less occlusion in the high mode case (Figure 6.12). Therefore, camera position (height) and angle (long or short range) is important to take into consideration during testing because each camera is mounted differently on the road and the surveillance system should function properly under various perspectives.

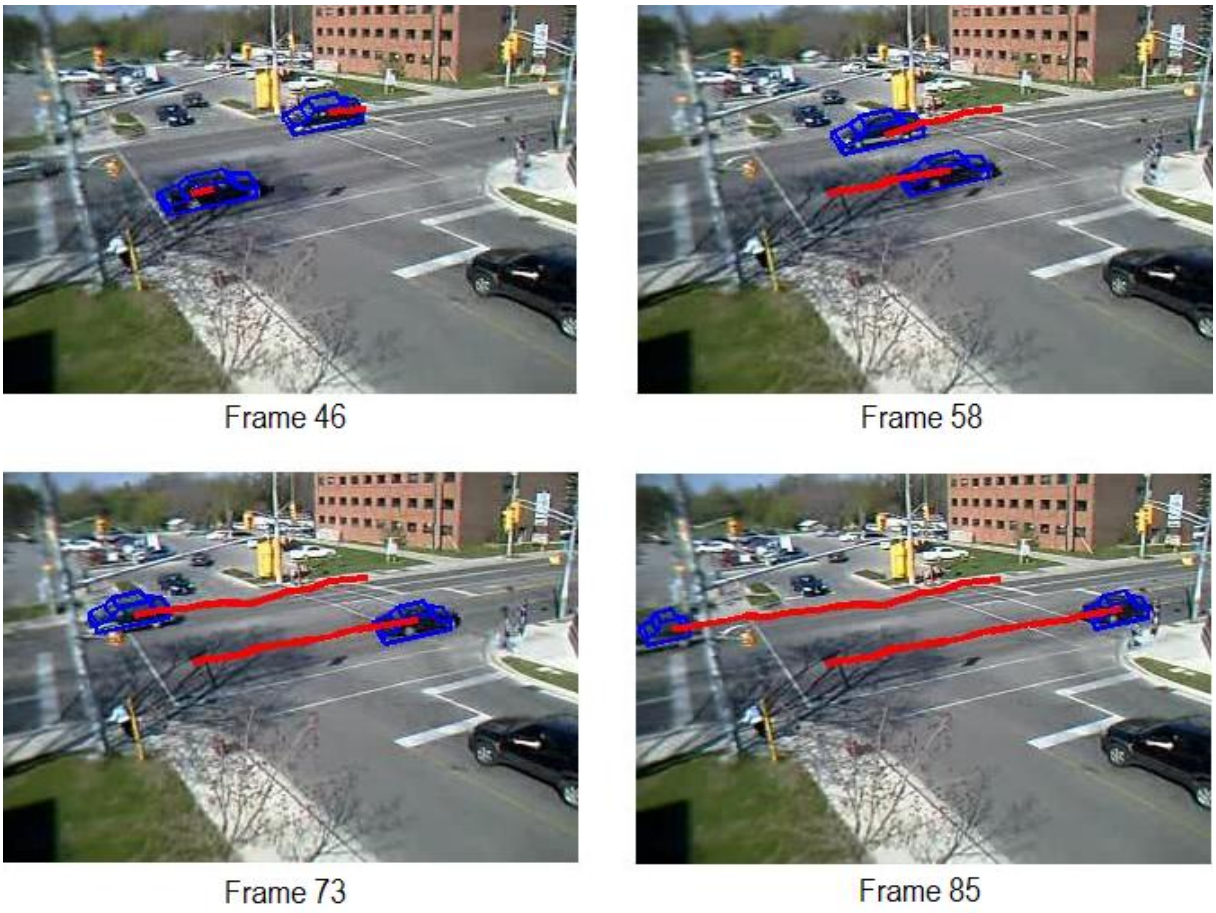

Figure 6.12: High height Camera Perspective Example.

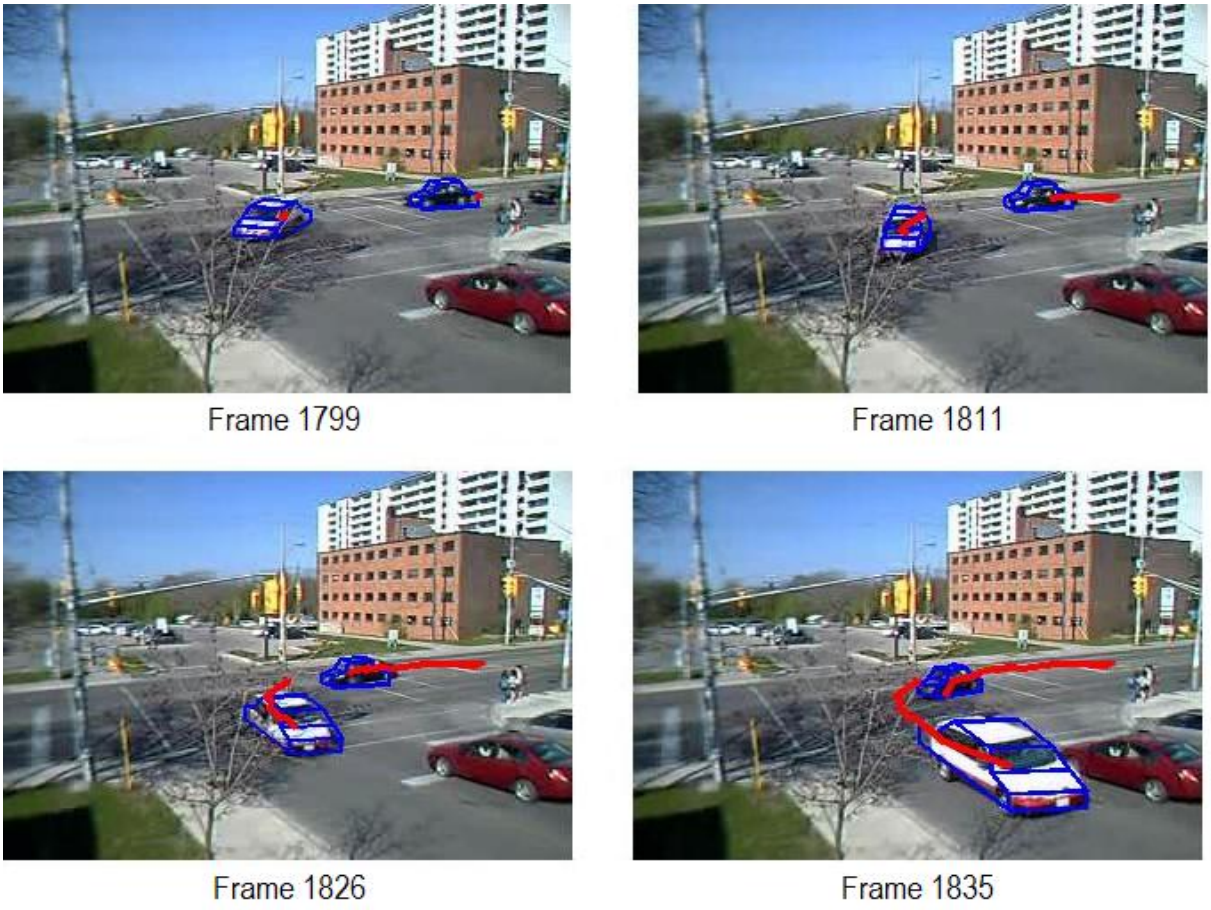

Figure 6.13: Medium Height Camera Perspective Example.

6.3.3 Scale Change

Many of the test sets used for testing the surveillance system constituted of video sequences of highways. The camera mode on these highways varied from close range (covering a small part of the highway) to far range views (covering a large area of the highway). In each of the different test set types, the car's scale changed as it traveled from the beginning of the highway to the last point seen by the camera. This scale change is dramatic at times reaching up to 500% change of original car size. Table 6.4 contains some results showing the scale change effect based on detection, recognition as well as tracking. The tests that are shown in the table are similar to those in table 6.3; however, the average scale change per car in each case is shown in the table.

In the following table, scale change is computed as the difference between the final vehicle size (in pixels) and the initial vehicle size, divided by the initial size.

Table 6.4: Scale Change Results

Scale Change Test Results per Video Sequence															
Road	Test Set No.	Module 1 Grade (in frames)				Module 2 Grade(in frames)				Module 3 Grade(in frames)				Overall Per-for-mance	Scale Change (increment)
		4	3	2	1	4	3	2	1	4	3	2	1		
Highway	Case 1	127	1	0	0	96	30	3	0	128	0	0	0	4 (91%)	328%
Highway	Case 2	132	48	0	0	138	30	12	0	180	0	0	0	4 (83%)	528%
Highway	Case 3	124	4	0	0	83	44	1	0	120	8	0	0	4 (85%)	518%
Highway	Case 4	78	0	0	0	70	8	0	0	78	0	0	0	4 (96%)	220%
Highway	Case 5	160	5	0	0	138	16	11	0	165	0	0	0	4 (93%)	190%
Highway	Case 6	150	0	0	0	140	10	0	0	150	0	0	0	4 (98%)	77%
Highway	Case 7	107	1	0	0	60	39	9	0	108	0	0	0	4 (85%)	500%
Highway	Case 8	134	7	0	0	131	7	3	0	141	0	0	0	4 (96%)	400%
Highway	Case 9	123	0	0	0	115	8	0	0	123	0	0	0	4 (98%)	425%
Highway	Case 10	76	12	0	0	72	16	0	0	80	8	0	0	4 (86%)	540%
Highway	Case 11	164	2	2	0	144	12	12	0	168	0	0	0	4 (94%)	420%
Highway	Case 12	140	6	4	0	135	6	9	0	150	0	0	0	4 (94%)	300%
Highway	Case 13	185	13	0	0	155	30	12	1	198	0	0	0	4 (91%)	325%
Highway	Case 14	60	3	0	0	50	9	4	0	63	0	0	0	4 (91%)	53%(decrease)

Based on the performance and scale change stated in table 6.4, a plot to illustrate the relationship between the variables is shown in Figure 6.14. On average, the scale change that falls between 0.50 (50% decrease in size over frames) to 5 (400% increase over original size) perform in the range of 90–95%. However, as the scale change exceeds 5, the performance starts decreases to around 90% and significantly starts dropping toward 80% after scale change of 6 (the cutoff % indicated in Figure 6.14). This is quite logical because when extreme scale changes occur, it is expected that initially the size of the vehicle was very small. When foreground objects to be detected are only a few pixels in length, the background subtraction module would have difficulty detecting it resulting in lower overall performance. Furthermore, extreme scale changes requires the recognition module to continuously alter the size of the model, orientation, and overlap. The more alterations needed, the higher the probability that an error might occur which also decreases the overall performance. All in all, the proposed system's performance remains good as long as the scale

does not change drastically (more than 6 times original size). A note should be made that the reason the plot only ranged to 6.5 is because no data exceeding scale change of 6.5 was available for this work.

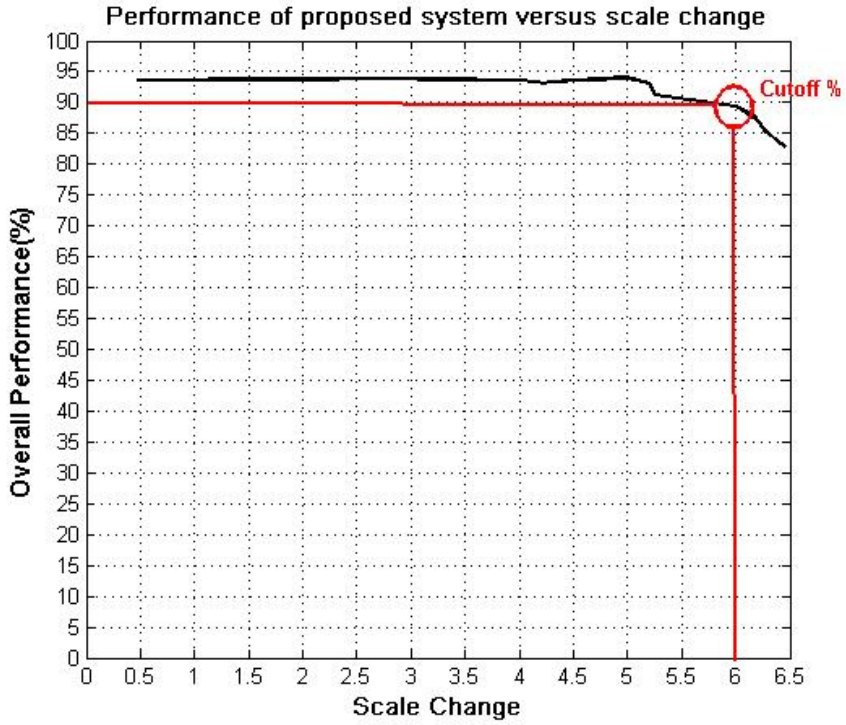

Figure 6.14: Plot of System Performance with respect to Scale change. The circle indicates scale change of 6 as being the cutoff before which the performance exceeds 90%.

6.3.4 Image Resolution

Three different image resolutions were examined: original size(320 by 240 pixels), 80% of the original size (256 by 192 pixels), and 50% of the original size (160 by 120 pixels). The importance of testing of lower resolutions is to check whether the system proposed is viable under different video camera shooting options. If the system works with low resolution images, then the cameras can record with lower resolution to save up memory space and money.

Based on the results attained from table 6.5, as the image resolution decreased, the overall system performance also decreased from 91% to 72 % (for 'Very Good' grades) and the failed cases increased from almost nill to 7.8% for the 160 by 120 pixel resolution. This is because as resolution decreases, size of foreground objects

Table 6.5: Image Resolution Results

Image Resolution Results						
Resolution Size	Overall Performance (in frames)				Overall Performance (%)	Failed Frames (%)
	4	3	2	1		
320 by 240 pixels	1713	128	27	0	4 (91%)	0 %
256 by 192 pixels	206	27	16	6	4 (81%)	2.3 %
160 by 120 pixels	183	34	18	20	4 (72%)	7.8 %

decrease also by 80% and 50% respectively. This makes foreground detection more difficult. Furthermore, image quality decreases making the edges harder to detect for the foreground recognition module. Thus, the 3D model matching will result in less accurate matched outputs and at times even failures.

Although performance decreases with resolution reduction, there are several advantages as well. Besides that less memory storage is needed to save the images, the computational speed is also significantly reduced. From an average of 3.6 seconds per frame for the normal case, the computational speed drops down to 3.2 seconds per frame for the first resolution (80% of original image size) and to 1.9 seconds per frame for the second resolution (50% of original image size). This dropdown is significant especially when large number of frames are being processed in the system. A note should be made that all computational speeds recorded were based on a 1.73 GHz single core processor and the application used was MATLAB.

6.3.5 Light Conditions

In order to test the system under daylight, foggy, or evening conditions, numerous tests were developed with histograms that have more white color bias (to mimic a bright or foggy day) as well as histograms with bias toward the black in order to test the system under conditions similar to evening lights.

In details, six different light conditions were tested excluding the normal test conditions; the first case (denoted as Lighter 1 in table 6.6), converted the original image histogram from a range of [0 1] to [0.3 1]. The second case (Lighter 2) also brightened the image by biasing the original image histogram from [0 1] to [0.5 1] making sure that only brighter colors were in the image. In both cases, the gamma was left at 1. Figure 6.15 illustrates Lighter 1 and Lighter 2.

Lighter 3 is the case when the histogram range remains the same but the gamma value is decreased to 0.5 making the image significantly brighter. Darker 3, on the other hand, represents gamma 2 and same histogram as original [0 1]. Figure 6.16 is

Figure 6.15: Lighter 1 and Lighter 2 Sample Frames.

an example of the two different gammas. Sample frames of each type are depicted in Figure 6.16.

Figure 6.16: Lighter 3 and Darker 3 Sample Frames.

Darker 2 denotes when the histogram range changes from [0 1] to [0 0.5] while keeping the gamma at 1. This makes the picture significantly more biased toward black. Darker 1, on the other hand, alters the original histogram to the range of [0 0.7]. Both of the cases are shown in Figure 6.17.

Based on the results obtained from the total 78 different test sets (total of 13,662 frames) that had different light conditions (refer to Table 6.6), the proposed system generally performs better with darker images than brighter ones. To evaluate the pattern as a whole, the system performs at 'very good' or 'good' (grades 4 or 3)

Figure 6.17: Darker 2 and Darker 1 Sample Frames.

Table 6.6: Light Condition Results

Light Condition Results								
Light Condition	Histogram Range	Gamma Value	Overall Performance (in frames) and in (%)				Frames with Good or Very Good Grade (3 or 4) (%)	Failed Frames (%)
			4	3	2	1		
Lighter 1	[0.3 1]	1	1207 (53.0%)	329 (14.4%)	384 (16.9%)	357 (15.7%)	67.4%	15.7%
Lighter 2	[0.5 1]	1	858 (37.7%)	747 (32.8%)	285 (12.5%)	387 (17.0%)	70.5%	17.0%
Lighter 3	[0 1]	0.5	1559 (68.5%)	233 (10.2%)	291 (12.8%)	194 (8.5%)	78.7%	12.8%
Darker 1	[0 0.7]	1	1906 (83.7%)	151 (6.6%)	152(6.7%)	68(3.0%)	90.3%	3.0%
Darker 2	[0 0.5]	1	1795 (78.8%)	221 (9.7%)	201(8.9%)	60(2.6%)	88.5%	2.6%
Darker 3	[0 1]	2	1703 (74.8%)	205 (9.0%)	164 (7.2%)	205 (9.0%)	83.8%	9.0%

ranging from 90% to 83% of the frames with the varying degrees and types of dark light conditions. The failure for the darker images span over 3% to 9%. On the other hand, the lighter images produce significantly lower amount of frames with 'very good' or 'good' results reaching down to 67% of the frames only. Furthermore, error rates increase up to 17%. In all cases, heightened error rates occur due to increased amount of false negatives. The first major reason for that is lack of foreground detection. When histograms are brightened or darkened, white cars and darker cars respectively become more difficult to extract due to their extreme similarity to the background. Another important reason for lower grade results for all cases is restriction of the histogram. When the histogram is adjusted such that to contain a more restricted colormap biased towards the white (for instance 0.3 to 1 or 0.5 to 1) or black (eg. 0 to 0.7 or 0 to 0.5), intensity variance defining

the edges becomes less evident. This leads to misdetection of edges and, in turn, worse model matching outcomes. As for the diversity of results between lighter and darker images, the lighter (lighter 1 and 2) results are inferior to the darker (darker 1 and 2) image results because vehicles that should be tracked are mostly dim on a gloomy grey road. Their colors lean towards black and if they are clipped by adjusting the histogram to be lighter (0.5 to 1), the intensities found within the clipped range (0 to 0.5) are eliminated. This causes less foreground detection as well as edge detection. Conversely, this occurs less when brighter ranges are clipped down to become darker since the brigher pixels are principally found in the sky and other minor objects.

With regards to images with intensity values from 0 to 1 (Lighter 3 and Darker 3), but varying gamma correction values (0.5 and 2 respectively), the image histograms are remapped to have higher(brighter) output values if gamma is less than 1 and lower(darker) output values if gamma is higher than 2. In gamma variation tests, the results are quite similar: 83% vs 78% for frames with 'very good or good' grades and 9% vs 12.8% failure rates for Darker 3 and Lighter 3 respectively. In both cases the results are not as good as normal conditions because foreground objects become more blended into the background when it is too dark or too light.

Lastly, the proposed system performs with acceptable failure rates (min. 3% to max. 17%); in other words, it has above 'tolerable' results in the range of (max. 97% to min. 83%). However, the 'very good' results drop down as image intensity values are increased/decreased (especially increased). Thus, the proposed system is functional in different light conditions (slightly brighter days, darker days, foggier days, evenings) but should be improved if it is to be used in extreme conditions (eg. whiteouts, blizzards...).

6.3.6 Image Noise

Another group test set that took place involved changing noise levels in the images. Given that the images were already sufficiently noisy (especially the intersection images), only a few additional tests were performed with regards to image noise alterations. Two types of image noise were added to the images: salt and pepper noise and Gaussian noise. Specifically they were added to see how the background subtraction module would behave and propose a simple solution in case the results were not satisfactory. The focus is on the background subtraction module because image noise mostly impacts the foreground extraction process. If the foreground

objects are not detected well, the rest of the modules will possibly fail at their tasks. Thus, it is important to ensure that the first module functions properly.

Figure 6.18 is an example of two frames with salt and pepper noise (density of 0.02) and Gaussian noise with 0.005 variance and 0 mean that have been applied respectively on the original images. Figure 6.19 on the other hand shows the result of the Gaussian Mixture Model background subtraction method when such the two types of noise are added respectively.

Figure 6.18: a) Frame with Salt and Pepper image noise of density 0.02. b) Gaussian noise with variance 0.005 and mean 0.

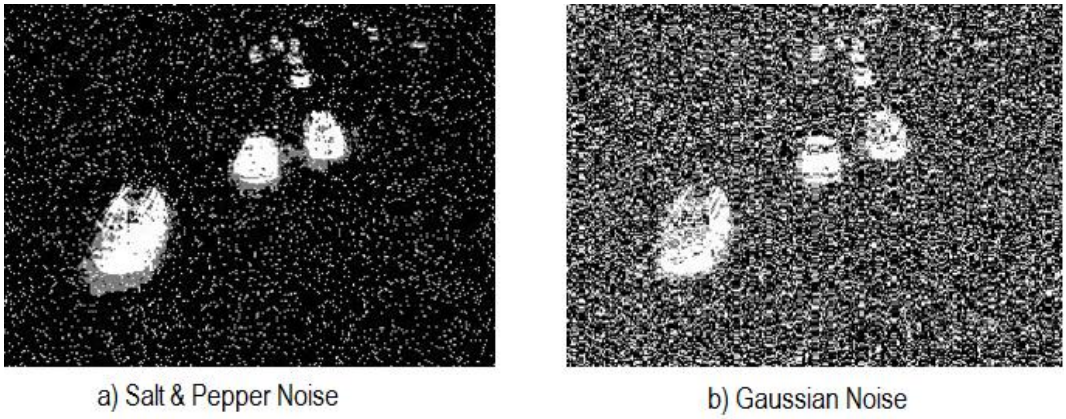

Figure 6.19: a) GMM result Frames when processed with Salt and Pepper image noise. b) GMM result Frames when processed with Gaussian image noise.

When the same morphological operators are applied on both Gaussian Mixture Model (GMM) background subtraction results, it becomes evident that the salt and pepper can be easily cleaned out due to their random nature; contrariwise, these

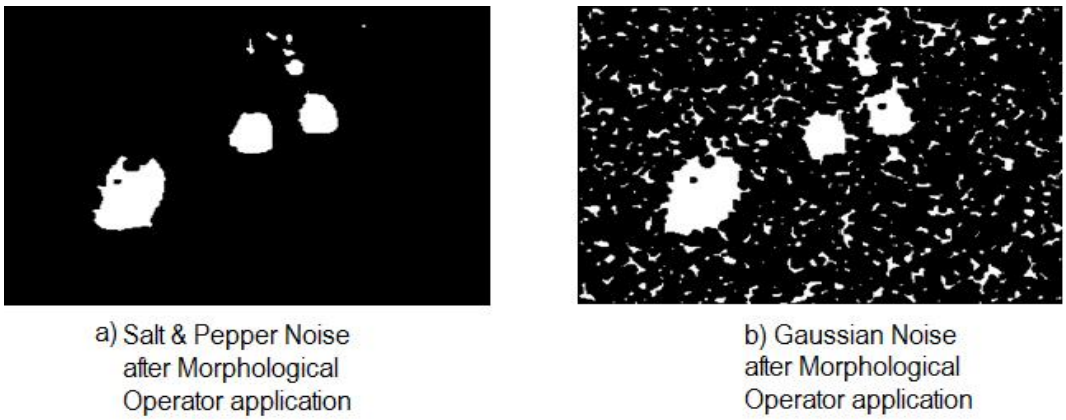

Figure 6.20: a) GMM result of Salt and Pepper noise after morphological processing. b) GMM result of Gaussian noise after morphological processing.

same morphological processes are not able to remove the Gaussian noise. Both results are shown in Figure 6.20.

In spite that the result after the morphological post-processing for the salt and pepper case is quite good (distinct almost whole vehicles that can easily be recognized and tracked), it is not the case with Gaussian noise (refer to Figure 6.20 b). Hence, a simple solution to overcome such a problem is to include a Wiener filter in the system. By including such a filter, the Gaussian noise is removed from the image and the foreground subtracted image can be used as well as images with no added initial noise. Figure 6.21 illustrates the result of the Wiener filter and morphological post-processing for the Gaussian noise case.

Figure 6.21: GMM result of Gaussian added noise image after morphological post-processing and image filtration with Wiener filter.

6.3.7 Occlusion

As previously mentioned in the camera perspective section, depending on the camera viewpoint, different kinds of occlusion might occur. In the test sets that were used, two types of occlusion are prevalent: car-to-car occlusion and scene objects and/or nature (eg. poles and trees) obscuring the camera view of the car.

A total of 560 instances of occlusion were used to test the proposed system. Table 6.7 states the result of these occlusion instances with respect to the camera perspective being high above the intersection or medium height. The second column records the total number of instances per camera viewpoint followed by a column separating the performance of the system at each of these instances (in terms of the number of instances per grade and the percentage). Finally, the last two column report the performance for 'very good' and 'good' cases and mentions the failure rate.

Table 6.7: Occlusion Results

Occlusion Results							
Camera Perspective	Total No. of Instances	Overall Performance (in instances) and in (%)				Instances with Good or Very Good Grade (3 or 4) (%)	Failed Instances (%)
		4	3	2	1		
High	316	196(62.0%)	60(19%)	8(2.5%)	52(16.5%)	81%	16.5%
Medium	244	164(67.2%)	32(13.1%)	20(8.2%)	28(11.5%)	80.3%	11.5%

When occlusion occurs, it becomes more difficult to detect the orientation of the edges in order to correctly match the best candidate for the vehicle. This explains the rates under 'Good' (3), 'Tolerable'(2), and only a few in the failed cases. As long as the occlusion is not severe, parts of the edges can be detected to orient the model. However, most of the failed instances that occur are actually due to failure in background subtraction. If the object of interest has been in full view previously before getting occluded, the drawback of the background subtraction module can be compensated by having the knowledge of the approximate size and number of cars in the previous frame.

6.3.8 Additional Results

In this section, a few additional tests were performed on video sequences used in [24][15][37] obtained from [33]. Two greyscale video sequences were tested — *dt* and *rhein* — with total frames of 132 frames.

The implemented system in this thesis depends on RGB values (or at least 3-channel images) for performing the Gaussian Mixture Model for background subtraction as well as requires the color-space as information for the tracking. Therefore, since the first video sequences tested constitutes of a single channel of greyscale 512 by 512 frames, the GMM was not applied. Instead, temporal differencing method was used for the first video sequence as the motion detector and RGB information was eliminated from the tracker. This decreased the overall performance of the system which was designed primarily to be used with color images. The second video sequence, *rhein* consisted of 564 by 688 by 3 greyscale images. Since the frames are 3-channel, the default background subtraction method (GMM) was used for the testing process.

The first test sequence used, *dt*, is a grayscale video sequence of a traffic intersection recorded using a stationary camera. The recognition and tracking modules performed better than the detection module with the recognition module performing at 'Very Good' and 'Good' up to 90% of the time; the tracker tracked with 100% accuracy for this data sequence. However, the detection module using temporal differencing performed worse than the other two modules due to the similarity of the background color to the black car. The temporal differencing has a drawback that it does not return the complete foreground object (Chapter 3). Therefore, the black car was not detected as a whole — in fact, it was detected as a small spot that got eliminated in the morphological cleaning process due to its small detected size. Therefore, for this sequence, the detection dropped down to 88% for 'Very Good' and 'Good' detections. If the GMM method were used, such missed detections would not have occured. Figure 6.22 shows sample results from the video sequence.

(a) (b)

Figure 6.22: a) Sample frame from the video sequence showing results so far. b) Sample frame from the video sequence showing results after 24 frames.

The second test sequence used, *rhein*, is also a grayscale video sequence of a traffic intersection recorded using a stationary camera. However, the camera perspective of this video sequence is closer to ground plane where cars are traveling. The detection, recognition, and tracking modules performed with an overall performance reaching up 86% for 'Very Good' and 'Good' frame results. The detection module using GMM failed a few times when vehicles were very slow moving. An example of a missed detection is the background truck which appears to be moving a few pixels every tens of frames. Figure 6.23 shows some results of the second video sequence.

(a) (b)

Figure 6.23: a) Sample frame from the video sequence showing results so far. b) Sample frame from the video sequence showing results after 54 frames.

In conclusion, the surveillance system implemented in this work was tested extensively on the data sets described in section 6.1. This section provided a sample results based on video sequences used by other researchers — namely [49][15][37]. The proposed surveillance system performed with an average accuracy of around 88% on these two video sequences. Some of the failures that occurred in the system were mainly due to errors in detections (missed detection). The overall limitations of the proposed surveillance system along with the summary results of all previous tests is discussed in section 6.3.9.

6.3.9 Summary of Results

To summarize the various aspects of the system tested in this chapter, table 6.9 states all the test results obtained. The first column states the test condition, the second column mentions the result percentage(%) with grades above 3 out of 4(inclusive), and the final column records the failure rate. All percentages are the averages of the total tests performed per test category.

Table 6.8: Summary of Test Results

Summary of Test Results		
Test	Overall Performance: Grades (%) (Sum of 'Good' 3 and ' Very Good' 4)	Failed Instances(%)
Different Road Types	95.0%	3.0%
Different Camera Perspectives	95.0%	2.1%
Scale Change	97.0%	0.015%
Image Resolution	86.0%	5.1%
Light Conditions	80.0%	10.0%
Occlusion	80.6%	14.0%
Average of all tests	88.9%	5.7%

All in all, the proposed system performs well (above 3 out of 4 inclusive) 88.9% of the time with a failure rate on average of around 5.7%. If extreme cases are not included in the system performance, the percentage boosts up to around 95%.

In summary, an automatic surveillance system was developed for tracking multiple moving vehicles on various road types, camera perspectives, scale change, and lighting conditions based on input obtained from a stationary camera mounted on a pole. Table 6.9 showed a summary of results under different conditions. A list of the strengths and weaknesses of the system is included.

Strengths:

- System performs well (up to 95%) under different road types (highways, intersections, and bridges) and under different camera perspectives (high altitude, medium, and low altitude).
- System performs well even if there is significant scale change (up to 500% change).
- System performs well with images of resolution 320 by 240 pixels and larger and with an acceptable performance for smaller resolutions (refer to section 6.3.4). The smaller the resolution becomes, the lower the performance becomes.
- The recognition module is able to match vehicles even if partial occlusion (50%) exists. The tracker is able to compensate for 100% occlusions that occur for short time (a few frames).

Weaknesses

- System performance deteriorates with extreme weather conditions (eg. extreme whiteouts). The system should be improved to perform well under conditions of snow, blizzards, and so on.
- The system performs well with partial occlusions reaching up to 50% occlusion. However, if the vehicle is 100% occluded with current cars equal to previous cars (a car entered and another got occluded), there is probability that the tracker will fail in that case.
- If the full occlusion lasts for many frames, the tracker will eliminate the occluded object from being tracked.

Chapter 7

Conclusions and Recommendations

7.1 Conclusion

In this thesis, a surveillance system to track multiple cars has been proposed. It is made up of three major modules: object extraction module, object recognition module, and tracking module. The first module (discussed in details in Chapter 3), uses Gaussian Mixture Models to detect the foreground objects as blobs. Following, morphological processes are used to clean the image. Later, major information such as color, size, orientation, and position of the foreground object are extracted. The second module (refer to Chapter 4) uses edge detection on the detected objects and matches a 3D vehicle model to the objects. The best match is chosen and the objects are classified as vehicle or non-vehicle. Finally, the third module (Chapter 5) employs association matrices in order to match multiple cars from one frame to another.

The above summarized proposed surveillance system has several major advantages. First, it makes use of Gaussian Mixture Models (GMM) that are invariant to slightly changing light conditions. Furthermore, since the thesis targets to track moving vehicles, GMM's drawback that it associates non-moving vehicles into the background is turned into a benefit. Also, a backup background subtraction method based on frame-subtraction (discussed in Chapter 3) provides an alternative foreground detection method when GMM breaks down or is not applicable (due to technical reasons eg. video encryption). With regards to the second module (recognition module), several important pros can be mentioned. Detecting edges

to recognize a car is invariant to scale, light conditions, as well as robust to partial occlusion. Moreover, 3D models closely describe what a car actually looks like opposed to methods that arbitrarily track everything in the scene. Finally, the tracking method used incorporated several components when tracking a car from frame to frame (color, size, and velocity) and does not require specific prediction models to track a vehicle. Also, it is able to successfully track entering and leaving cars from the scene.

Additionally, several important keypoints have been tackled in this thesis. The system does not require specific car information (size, location, or type) about the vehicles in the scene. This is vital because we have no such knowledge in the real world and incorporating such information into a commercial system would be impractical.

Disadvantages also exist in the system. It does not account for shadow removal nor classify vehicles into different categories. In addition, the performance decreases with full occlusion and extreme light changes. Finally, at very small image resolutions, edges become hard to detect and model matching might become inaccurate.

In conclusion, the system proposed functions well on the various aspects investigated toward the objectives of the thesis. Table 7.1 states all the objectives of the thesis that were discussed in Chapter 1 and indicates in the second and third column of the table whether they have been met.

Table 7.1 emphasizes that the system developed is able to effectively function despite different road types (intersection,highway,and bridge) at different camera viewpoints (close range, far range, high mounted camera, low mounted camera, and so on), various vehicle viewpoints (top view of car, side view, front view, back view, and different degrees of each aforementioned view), scale changes (ideally up to 500%), background clutter (many buildings, people, traffic lights, trees, and so on), different image resolutions (with better performance with image sizes around or larger than 320 by 240 pixels), moderate light conditions (gloomy and bright), partial occlusion (reaching up to 50%), and full occlusion (lasting for a few frames). Furthermore, processing time in Matlab on a 1.73 GHz processor takes up to 3 to 4 seconds per frame. This speed can be improved by coding up the system in C++ since the code contains some loops. The system is able to track up to 10 cars simultaneously (did not test for more) whereas humans can track up to 8 objects simultaneously based on a recent study [2].

Besides meeting all of the thesis objectives, several contributions have been

Table 7.1: Objectives

Objectives Check			
Objective	Goal Met	Comments	Chapter
System is robust to different road types	Y	A variety of road tests were used to test the system (intersection, highway, and bridge). The system performed well under the three types.	Chap 6
System is invariant to camera perspective	Y	A range of camera perspectives (long range, short range, high height, medium height) were used to test the system which worked well.	Chap 6
System works under varying vehicle viewpoints	Y	When testing the camera perspectives and road types, the system is being tested inherently for varying vehicle viewpoints. Therefore, the system tolerates different vehicle viewpoints.	Chap 6
System functions under different background clutter	Y	Although no official testing took place on background clutter, the system proved forbearing to different background scenes since highway test sets have simple background whereas intersections have a more complex scene.	Chap 6
System operates under many light conditions	Y	A range of light conditions were tested for and the system performed satisfactorily (gloomy, bright, and so on).	Chap 6
System is able to process images with noise	Y	System is able to tolerate a limited amount of noise especially if noise removal filters are included. More work should be performed in the future with regards to this point.	Chap 6
System accepts diverse image resolutions	Y	System is able to process various different image resolutions (128 by 128 pixels up to 700 by 700 pixels).	Chap 6
System performs under occlusion	Y	System is robust to partial occlusion (50%) and short-lived full occlusions	Chap 6
System is capable of multiple tracking	Y	System is able to track several cars simultaneously	Chap 5
System processes each frame within a few seconds	Y	System takes an average around 3 to 4 seconds to process each frame on a 1.73 GHz single core processor	Chap 4-5-6

made by this work as outlined in Section 7.2. Nevertheless, a few additions should be included in the future work of the system in order to heighten its performance with respect to some key issues and for the purpose of expanding its applicability even further - as discussed in Section 7.3.

7.2 Contributions to the State-of-the-Art

Sections 2.4 and 2.5 states an overview of the state-of-the-art methods and compares the concept of this work with other state-of-the-art works. A summary of the contributions made to the state-of-the-art is listed next.

- The multiple-vehicle surveillance system developed in this thesis is based on integrating different modules (Gaussian Mixture Model and temporal differencing for motion detection, 3D vehicle recognition based on Hausdorff measure, and deterministic data method for tracking which is able to deal with entering, leaving, and disappearing cars) that have not been integrated before in such a manner; thus, this work provides an alternative approach to other state-of-the-art researches for vehicle monitoring which is flexible enough to function well under different camera perspectives, background clutter, vehicle viewpoints, road types, scale changes, image noise, image resolutions, and lighting conditions.

- The method used in this work is overall a simpler approach compared to other state-of-the-art surveillance systems (refer to Section 2.4) that use 3D model approaches to monitor multiple vehicles under a fixed camera assumption.

- The location of the vehicles are automatically detected by the foreground extraction module. This means that no additional extensive methods are needed to attain a more accurate vehicle position, contrary to [37] who use hough transformation methods of edge elements along with optical flow to determine the vehicle location.

- The vehicle size is also detected automatically by the foreground extraction module eliminating the need to assume the knowledge of vehicle size as a priori knowledge. Furthermore, no a priori knowledge about the pathways of the vehicles is given in this thesis, unlike [37] where models of circular and straight paths are used.

- Two methods for motion detection are investigated - Gaussian Mixture models and temporal differencing - with the latter acting as a backup system in case GMM cannot be applied. This is useful to make the system more flexible in terms of the input it receives. GMM prefers 3-channel input images. If such images are not available, the backup system can process the images.

7.3 Recommendation and Future Work

The surveillance system that is proposed in this work targets a wide range of vehicle surveillance applications since it can be applied to different road types. However, to make a system even more generic, it should be able to adapt to all kinds of environmental factors that can vary from urban background clutter to various light conditions. For this reason, the surveillance system can be expanded and tested under snowy, rainy, and night conditions.

In addition, each of the three modules can be enhanced in several ways. The foreground extraction module can be improved to incorporate automatic shadow removal from the extracted objects as well as advance the adaptive Gaussian Mixture Model to make it more sensitive at detecting very-slow moving vehicles. The second module - foreground recognition - can be advanced by including a classifier that would be able to distinguish among the different types of car categories (eg. trucks, sedans, buses, and SUVs). Information about detected car surfaces can also be merged with the edge information attained to obtain a better description of a car. As for the last module - tracking - it can be extended into a car counter as well as a tracker which would be beneficial for commercial applications (such as statistics).

List of References

[1] T. Acharya and A.K. Ray. *Image Processing Principles and Applications*. John Wiley and Sons Inc., 2005.

[2] G. Alvarez and S. Franconeri. How many objects can you track? evidence for a resource-limited attentive tracking mechanism. *Journal of Vision*, 7:1–10, 2007.

[3] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.

[4] Y. Bar-Shalom and T. Foreman. *Tracking and Data Association*. Academic Press Inc, 1988.

[5] J. Barron, D. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12:42–77, 1994.

[6] R. Basri. Recognition by prototypes. *International Journal of Computer Vision*, 19:147–167, 1996.

[7] M. Bertozzi and A. Broggi. Gold: A parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Transactions on Image Processing*, 1998.

[8] J. Canny. A computational approach to edge detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 8, 1986.

[9] R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, and O. Hasegawa. A system for video surveillance and monitoring. Technical report, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, May 2000.

[10] I. Cox and S. Hingorani. An efficient implementation of Reid's Multiple Hypothesis Tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:138–150, 1996.

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

[12] Gonzalez, Woods, and Eddins. *Digital Image Processing Using MATLAB (DIPUM)*. Prentice Hall, 2004.

[13] N. Gregoire and M. Bouillot. `http://cgm.cs.mcgill.ca/~godfried/teaching/cg-projects/98/normand/main.html`.

[14] W. Grimson. *Object recognition by computer: the role of geometric constraints*. MIT Press, 1990.

[15] M. Haag and H. Nagel. Combination of edge element and optical flow estimates for 3d-model-based vehicle tracking in traffic image sequences. *International Journal of Computer Vision*, 35:295–319, 1999.

[16] N. Hoose. Computer vision as a traffic surveillance tool. *Control, Computers, Communications in Transportation*, pages 57–64, 1990.

[17] W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, 34(3):334 – 352, 2004.

[18] D. Huttenlocher, G. Klanderman, and W. J. Rucklidge. Comparing images using the Hausdorff Distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:850–863, 1993.

[19] D. Huttenlocher and W. Rucklidge. A multi-resolution technique for comparing images using the Hausdorff Distance. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 705–706, 1993.

[20] N. Kachouie and P. Fieguth. Extended-Hungarian-JPDA: Exact single-frame stem cell tracking. *IEEE Transactions on Biomedical Engineering*, 54:2011–2019, November 2007.

[21] R. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45, 1960.

[22] W. Kasprzak. An iconic classification scheme for video-based traffic sensor tasks. *Computer Analysis of Images and Patterns*, pages 725–732, 2001.

[23] V. Kastrinaki. A survey of video processing techniques for traffic applications. *Image Vision Computing*, 21:359–381, 2003.

[24] D. Koller. Moving object recognition and classification based on recursive shape parameter estimation. *12th Israel Conference on Artificial Intelligence Computer Vision*, 1993.

[25] P. Kovesi. Matlab and octave function for computer vision and image processing. `http://www.csse.uwa.edu.au/~pk/Research/MatlabFns/`.

[26] A. Lipton. Local application of optic flow to analyze rigid versus nonrigid motion. *Proceedings of the International Conference on Computer Vision Workshop Frame-Rate Vision*, 1999.

[27] J. Lou and T. Tan. 3-d model-based vehicle tracking. *IEEE Transactions on Image Processing*, 14:2011–2019, October 2005.

[28] P.C. Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Science*, pages 49–55, 1936.

[29] K. McBride. Vehicle tracking in occlusion and clutter. Master's thesis, University of Waterloo, 2007.

[30] S. Messelodi, C. Modena, and M. Zanin. A computer vision system for detection and classification of vehicles at urban road intersections. *Pattern Analysis and Applications*, 8(1-2):17–31, 2005.

[31] D. Migliore, M. Matteucci, M. Naccari, and A. Bonarini. A revaluation of frame difference in fast and robust motion detection. *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, 2006.

[32] N. El Nabbout, J. Zelek, and D. Clausi. Automatically detecting and tracking people walking through a transparent door with vision. *Fifth Canadian Conference on Computer and Robot Vision*, pages 171–178, May 2008.

[33] H. Nagel. http://kogs.iaks.uni-karlsruche.de/image-sequences/.

[34] R. Nock and F. Nielsen. Statistical region merging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:1–7, November 2004.

[35] R. Nock and F. Nielsen. Semi-supervised statistical region refinement for color image segmentation. *Pattern Recognition*, 38:835–846, 2007.

[36] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9:62–66, 1979.

[37] A. Ottlik and H. Nagel. Initialization of model-based vehicle tracking in video sequences of inter-city intersections. *International Journal of Computer Vision (IJCV)*, 73(2):139–157, November 2007.

[38] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:266–280, 2000.

[39] K. Rangarajan and M. Shah. Establishing motion correspondence. *Conference on Computer Vision, Graphics, and Image Processing*, 54:56–73, 1991.

[40] W. Rucklidge. Efficiently locating objects using Hausdorff Distance. *International Journal of Computer Vision*, 24:251–270, 1997.

[41] K. Shafique and M. Shah. A non-iterative greedy algorithm for multi-frame point correspondence. *IEEE International Conference on Computer Vision*, pages 110–115, 2003.

[42] M. Shah. Fundamentals of Computer Vision, 1997. http://www.cs.ucf.edu/courses/cap6411/book.pdf.

[43] P. Skulimowski and P. Strumillo. Visual person tracking in sequences shot from camera in motion. *Computer Vision and Graphics*, 32:683–688, 2006.

[44] X. Song and R. Nevatia. A model-based vehicle segmentation method for tracking. *Proceedings of the Tenth IEEE International Conference on Computer Vision*, 2005.

[45] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. *In Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 246–252, 1999.

[46] E. Trucco and A. Verri. *Introductory Techniques for 3D Computer Vision*. Prentice Hall, 1998.

[47] H. Yang, J. Lou, H. Z. Sun, W. Hu, and T. Tan. Efficient and robust vehicle localization. *Proceedings of IEEE International Conference on Image Processing*, pages 355–358, 2001.

[48] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38:266–280, Decemeber 2006.

[49] B. Yiu, K. Wong, F. Chin, and R. Chung. Explicit contour model for vehicle tracking with automatic hypothesis validation. *IEEE Conference on Image Processing*, 21:582–589, September 2005.

[50] Z. Zivkovic. Improved adaptive gaussian mixture model for background subtraction. *Proceedings of the International Conference on Pattern Recognition*, 2:28–31, August 2004.

[51] Z. Zivkovic and F. Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7):773–780, May 2006.