AN EFFECTIVE STEP TO REAL-TIME IMPLEMENTATION OF ACCIDENT DETECTION SYSTEM USING IMAGE PROCESSING

By

LOGESH VASU

Bachelor of Engineering in Electronics and Communication

Anna University

Chennai, Tamil Nadu

2008

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
December, 2010

AN EFFECTIVE STEP TO REAL-TIME

IMPLEMENTATION OF ACCIDENT DETECTION

SYSTEM USING IMAGE PROCESSING

Thesis Approved:

Dr. Damon M. Chandler

Thesis Adviser

Dr. Qi Cheng

Dr. Weihua Sheng

Dr. Mark E. Payton

Dean of the Graduate College

ACKNOWLEDGMENTS

First I would like to thank the Holy God for giving me the strength and desire to pursue and complete my Masters Degree at Oklahoma State University. I would like to thank my adviser Dr. Damon Chandler for his immense support and advice throughout my Masters and research. Without Dr. Chandler's assistance, I would never have made it this far in my degree.

I would also like to thank my dear parents Mr. and Mrs. Vasu for their unconditional love and support that they gave me throughout my stay at United States to complete my Masters Degree.

I would like to thank my committee members Dr. Qi Cheng, and Dr. Weihua Sheng for taking the time and effort to review my thesis. I would also like to thank my fellow Computational Perception and Image Quality (CPIQ) lab members for their support and motivation. Lastly, I would like to thank my friends for their support throughout my Masters at Oklahoma State University.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

Table Page

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

1.1. Motivation

With the computation capability of the modern CPU processors, many complex real-time applications have been made possible and implemented in various fields worldwide. One of the widely used real-time applications is video surveillance systems. Video surveillance systems are been used for security monitoring, anomaly detection, traffic monitoring and many other purposes. Video surveillance systems have decreased the need of human presence to monitor activities captured by video cameras. And also one of the advantages of visual surveillance systems is videos can been stored and analyzed for future reference. One of the important applications of video surveillance systems is traffic surveillance. Extensive research has been done in the field of video traffic surveillance. Video traffic surveillance systems are used for vehicle detection, tracking, traffic flow estimation, vehicle speed detection, vehicle classification, etc. Video traffic surveillance has paved for numerous applications such as ATMS (Advanced Transportation Management Systems) [1] and (ITS) Intelligent Transportation Systems [2].

One of the widely used applications of traffic surveillance systems is vehicle detection and tracking. By detecting and tracking vehicles, we can detect vehicle's velocity, trajectory, traffic density, etc. and if there is any abnormality, the recorded information can be send to the traffic authorities to take necessary action. The main advantage of the video monitoring systems over the existing systems such as physical detectors [3] that uses magnetic loops is the cost efficiency involved in installing and maintaining these video systems, and also the aspect of

video storage and transmission to analyze the detected events. Table 1.1 shows the performance comparison of various incident detection technologies. Thus video-based traffic surveillance systems have been preferred all over the world.

Table 1.1: Performance comparison among existing incident detection technologies [4]

Type	Advantages	Disadvantages
Inductive loop detector	• Low per-unit cost • Large experience base • Relatively good performance	• Installation and maintenance require traffic disruption • Easily damaged by heavy vehicles, road repairs, etc.
Microwave (Radar)	• Installation and repair do not require traffic disruption • Direct measurement of speed • Multilane operation • Compact size	• May have vehicle masking in multi-lane application • Resolution impacted by Federal Communications Commission (FCC) approved transmit frequency • Relatively low precision
Laser	• Can provide presence, speed, and length data • May be used in an along-the-road or an across-the-road orientation with a twin detector unit	• Affected by poor visibility and heavy precipitation • High cost
Infrared	• Day/night operation • Installation and repair do not require traffic disruption • Better than visible wavelength sensors in fog • Compact size	• Sensors have unstable detection zone • May require cooled IR detector for high sensitivity • Susceptible to atmospheric obscurants and weather • One per lane required
Ultrasonic	• Can measure volume, speed, occupancy, presence, and queue length	• Subject to attenuation and distortion from a number of environmental factors (changes in ambient temperature, air turbulence, and humidity) • Difficult to detect snow-covered vehicles
Magneto-meter	• Suitable for installation in bridge decks or other hard concrete surfaces where loop detectors cannot be installed	• Limited application • Medium cost
Video image processing	• Provides live image of traffic (more information) • Multiple lanes observed • No traffic interruption for installation and repair • Vehicle tracking	• Live video image requires expensive data communication equipment • Different algorithms usually required for day and night use • Possible errors in traffic data transition period • Susceptible to atmospheric obscurants and adverse weather

Having said the advantages of video-based traffic surveillance systems a new paradigm can be added to the application of video surveillance systems if we can detect accidents at traffic intersections and report the detected incident to the concerned authorities so that necessary action can be taken. Figure 1.1, shows some of the scenarios depicting traffic accidents and its consequences.

Figure 1.1: Some of the scenarios depicting traffic accidents and its consequences

According to World Health Organization reports about 1.2 million lives are lost every year due to traffic accidents [5]. What is more amazing is the fact that traffic accident related deaths is one among the top ten causes of death worldwide, the list that includes tuberculosis, heart disease and AIDS as shown in Table 1.2. And also the cost of these accidents adds up to a shocking 1-3% of the world's Gross National Product [6]. In United States, it is estimated that vehicle accidents account for over 40,000 deaths and cost over $164 billion each year. Among these, passenger- vehicle crashes account for the vast majority of deaths [7]. Without any preventive measures these figures are estimated to increase to 65% over the next 20 years. Studies have shown that the number of traffic related fatalities is highly dependent on emergency

response time [8]. When an accident occurs, response time is critical, every extra minute that it takes for help to arrive can mean the difference between life and death. So there arises a need for a system that can detect accidents automatically and report it to the concerned authorities quickly by which the emergency response time can be made faster, therefore potentially saving thousands of lives. With the usage of video-based traffic surveillance systems, accidents can be recorded and be sent to the traffic monitoring center so that the incoming traffic can be warned of an occurrence of accident and be diverted to avoid traffic congestion. This brings us to the motivation of this research. The motivation of this research is to develop an accident detection module at roadway intersections through video processing and report the detected accident along with the crash video to the concerned authorities, so that immediate action can be taken and potentially save thousands of live and property.

Table 1.2: Top ten causes of death world wide

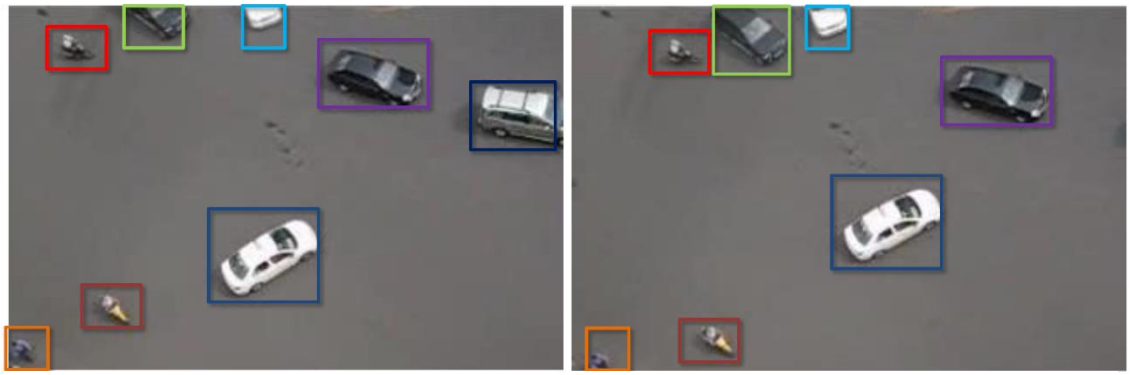
World	Deaths in millions
Coronary heart disease	7.20
Stroke and other cerebrovascular diseases	5.71
Lower respiratory infections	4.18
Chronic obstructive pulmonary disease	3.02
Diarrhoeal diseases	2.16
HIV/AIDS	2.04
Tuberculosis	1.46
Trachea, bronchus, lung cancers	1.32
Road traffic accidents #9 on the list	1.27
Prematurity and low birth weight	1.18

1.2. Research Goal and Challenges

1.2.1 Overview

Although considerable amount of research has been done to develop a system that can detect an accident through video surveillance, real-time implementation of all these systems have not been realized yet. Real-time implementation of accident detection through video-based traffic surveillance have always been challenging since one has to strike a right balance between the speed of the system and the performance of the systems such as correctly detecting accidents and also reducing false alarm rate. Ideally we want a system that could maximize the number of frames processed per second at the same time able to achieve acceptable performance rate. This brings us to the goal of this research. The goal of this research is to develop an accident detection module at roadway intersections through video processing that is suitable for real-time implementation. In this thesis we developed an accident detection module that uses the parameters extracted from the detected and tracked vehicles which is able to achieve good real-time performance.

An important stage in automatic vehicle crash monitoring systems is the detection of vehicles in each video frame and accurately tracking the vehicles across multiple frames. With such tracking, vehicle information such as speed, change in speed and change in orientation can be determined to facilitate the process of crash detection. As shown in the Figure 1.2, given the detected vehicles, tracking can be viewed as a correspondence problem in which the goal is to determine which detected vehicle in the next frame corresponds to a given vehicle in the current frame. While for a human analyst, the task of tracking can often be performed effortlessly, this task is quite challenging for a computer. Therefore in this thesis more emphasis has been given to the real-time implementation of vehicle detection and tracking.



(a) (b)

Figure 1.2: An example of vehicle tracking (a) Frame at time *t* (b) Frame at time *t+1*

1.2.1 Challenges in Vehicle Detection and Tracking

Implementing a system that does vehicle detection and tracking can be quite challenging. There are numerous difficulties that need to be taken into account, while implementing a system that performs vehicle detection and tracking. Figure 1.3 shows some of the scenarios encountered. Figure 1.3(a) shows frame captured at arbitrary time, Figure 1.3(b) shows the change in illumination conditions, Figure 1.3(c) shows change in weather conditions and Figure 1.3(d) shows change in traffic conditions. The following are the some of the challenges that can be faced while implementing real-time vehicle detection and tracking [9].

(a) (b) (c) (d)

Figure 1.3: Scenarios showing different traffic conditions (a) Example frame (b) Change in illumination (c) Change in weather conditions (d) Change in traffic conditions (Images were downloaded from http://i21www.ira.uka.de/image_sequences/)

1. Vehicles can be of different size, shape and color. Furthermore, a vehicle can be observed from different angles, making the definition of a vehicle broader.

2. Automatic segmentation of each vehicle from the background and from other vehicles so that all vehicles are detected.
3. Lightning and weather conditions vary substantially. Rain, snow, fog, daylight and darkness must all be taken into account when designing the system.
4. Vehicles can be occluded by other vehicles or structures.
5. Traffic conditions may vary, and many of the tracking algorithms degrade with heavy traffic congestion, where vehicles move slowly, and the distances between the vehicles are small.
6. Ability of the system to operate in real-time.

1.3. Methodology

The general description of the accident detection systems is presented below in the Figure 1.4.

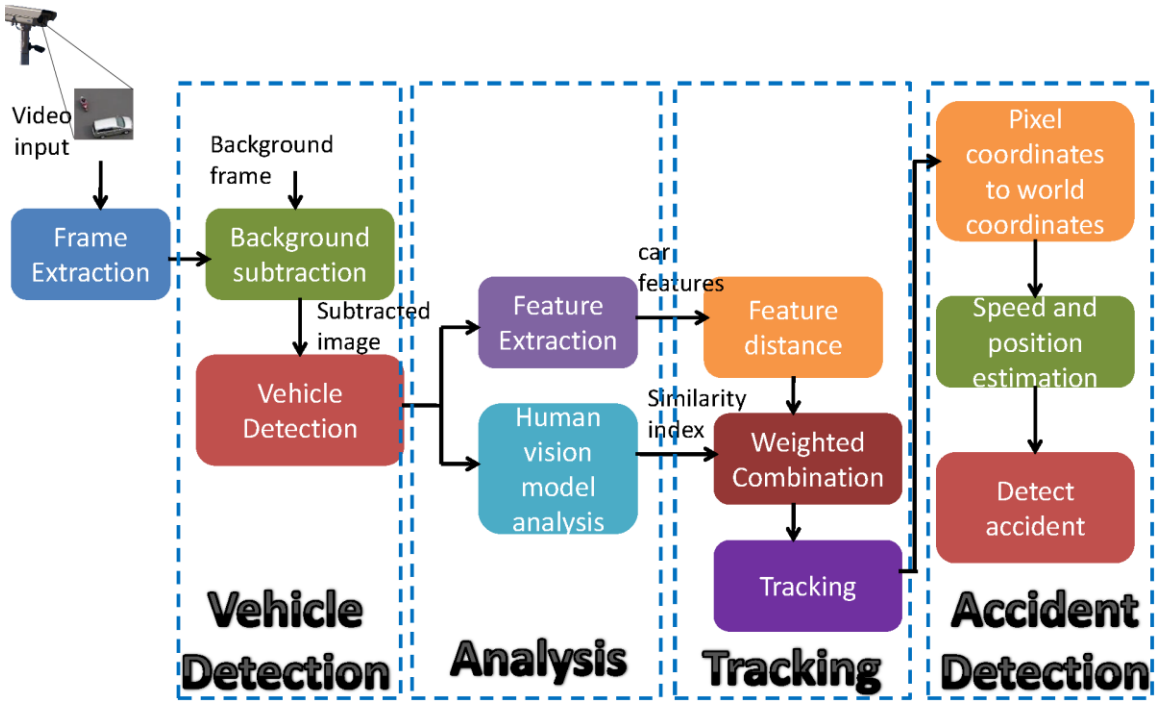


Figure 1.4: Block diagram of the proposed accident detection system

In this thesis vehicle tracking is done based on low-level features and low-level human visual-system (HVS) modeling. Low-level features (e.g., color, orientation, size) are generally used because of their low computational complexity. Our method employs a weighted combination of low-level features along with a human vision based algorithm of visual dissimilarity for vehicle tracking. Although HVS models have found widespread use in a variety of consumer image processing applications, they have yet to be used extensively for vehicle tracking.

The detail description of the entire system is as follows: The first step of the process is the frame extraction step. In this frames are extracted from the video camera input. The second step of the process is the vehicle detection step. Here the already stored background frame is subtracted from the input frame to detect the moving regions in the frame. The difference image is further thresholded to detect the vehicle regions in the frame. Hence the vehicles in each frame are detected. In the third step low-level features such as area, centroid, orientation, luminance and color of the extracted vehicle regions are computed. And also for each of the region detected in frame at time *t*, similarity index is computed with all of the regions detected in frame at time *t+1* using human vision based model analysis. In the tracking stage, Euclidean distance is computed between the low-level features of each vehicle in frame *n* and all the other vehicles detected in frame *n+1*. This Euclidean distance vector is combined with the already computed similarity index for a particular vehicle region in frame *n*. Based on the minimum distance between vehicle regions tracking was done. In the next step, the centroid position of a tracked vehicle in each frame is computed and based on this information and the frame rate; the speed of the tracked vehicle is computed in terms of *pixels/second*. Since the position of the video camera is fixed, the camera parameters such as focal length, pan and tilt angle of the vehicle remains the constant and it can be computed using camera calibration algorithm. From all this information the pixel coordinates of the vehicle in each frame is converted to real-world coordinates. By this

conversion, the speed of the vehicle in terms of *miles/hr* is computed. Based on the velocity information, position and low-level features of the tracked vehicle suitable thresholds are defined to determine the occurrence of accidents. This is the overall description of the proposed system for accident detection at roadway intersections.

1.4. Contribution

The main contribution of this thesis is the real-time implementation of vehicle detection and tracking along with accident detection at roadway intersections. As discussed earlier although considerable amount of research has been done related to vehicle detection and tracking most of the systems fail to implement them in the real-time because of the complexity of the algorithm. In this thesis a new method have been adapted to track the detected vehicles in each video frame that is suitable for real-time implementation and also accident detection module have been added to vehicle detection and tracking module that can operate in real-time. We use the low-level features such as area, orientation, centroid, color, luminance of the extracted vehicle regions to achieve reasonable tracking rate. To determine accidents, speed, area and orientation of the tracked vehicle were used. The insight of this thesis is to process maximum video frames as possible and also achieve good performance rate simultaneously.

1.5. Thesis Organization

The rest of the thesis is organized as follows: Chapter 2 reviews the previous research work being done related to vehicle tracking and incident detection. In Chapter 3, the brief overview of the about the system, its advantages and limitations are discussed. Description of vehicle detection and feature extraction are presented in Chapter 4, results of vehicle tracking and accident detection are discussed in Chapter 5. Real-time performance evaluation and results of the algorithm are discussed in Chapter 6. Finally general conclusions and steps for future work are presented in Chapter 7.

CHAPTER 2

BACKGROUND

2.1. Overview

For the last two decades researchers have spend quality time to develop different methods that can be applied in the field of video-based traffic surveillance. Some of the applications of video-based surveillance include vehicle tracking, counting the number of vehicles, calculating vehicle velocity, finding vehicle trajectory, classifying the vehicles, estimating the traffic density, finding the traffic flow, license plate recognition, etc. Of late the focus of video-based traffic surveillance has shifted to detect incidents in roadways such as vehicle accidents, traffic congestion, and unexpected traffic blocks. From researches and surveys it was found that there is more necessity to detect accidents in highways and roadway intersections, as vehicle accidents causes huge loss to lives and property. Therefore the objective of video-based traffic surveillance of present is on accident detection in highways and intersections such that necessary action can be taken promptly without losing any quality time, so that lives of the injured can be saved. As discussed earlier the another advantage of video based surveillance is that the incidents can be recorded and analyzed for future reference and also the videos can be send to the traffic monitoring center to clear traffic congestion and divert the incoming traffic. One has to say for sure that the ultimate goal of video-based traffic surveillance is to pre-determine the accidents at highways and intersections and alert the incoming traffic of the occurrence of accident, which can potentially save thousands of lives and billions of dollars.

Apart from video-based systems, researches have also been done to detect incidents in roadways using other systems such as sensors, acoustic signal and others. These systems basically use physical detectors to collect the traffic parameters and apply machine learning and pattern recognition algorithm to detect the occurrence of an incident. Some of the techniques that had been used are decision trees for pattern recognition, time series analysis, Kalman filters and neural networks [10]-[18]. Spot sensors such as inductive loop sensors were employed by Gangisetty [19]. All of these systems showed varying amount of detection performance. But all the above described systems concentrated only on few areas of incident detection and failed to address the problem of traffic crashes at intersection. Only few systems [20] have addressed the problem of detecting crashes at intersection.

Green *et al.* [21] evaluated the performance of sound actuated video recording system which was used to analyze the reasons for traffic crashes at intersections. The system automatically records incident based on the sound it receives such as horns, clashing metal, squealing tires, etc. The results of this study were used by the transportation department to enhance the safety features by the traffic department which resulted in the reduction of accidents by 14%. Another system [22] developed a method to detect and report crashes at intersection using acoustic signal generated by the crashes. An acoustic database was developed which contains the normal traffic sounds, construction sounds and crash sounds and the system would compare the captured sound signal with the signals stored in the database to determine crashes. The system produced a good performance with false alarm rate of 1%. All the above studies suggested that there was a lot of scope for improvement to make these systems robust for different traffic flow conditions.

All the above described systems were able to detect incidents to certain extent; these systems can be employed for only a particular application. However the advantage of the utilizing the vision sensors for event recognition is their ability to collect diverse information such as

illegally parked vehicles, traffic violations, traffic jams and traffic accidents. Description of some of the vision based traffic applications can be found in [23]-[25]. Lai *et al*. [26] developed a system that is used to detect red light runners at intersections. There is lot more advantages of video based traffic accident detection system since the crashes can be recorded and analyzed and these recordings can be used to enhance the safety features at roadways and intersections. And importantly once the accident have been detected, the automated reporting of accident can be used to reduce the emergency response time which on its own makes the video based traffic surveillance systems superior to other non-vision based systems.

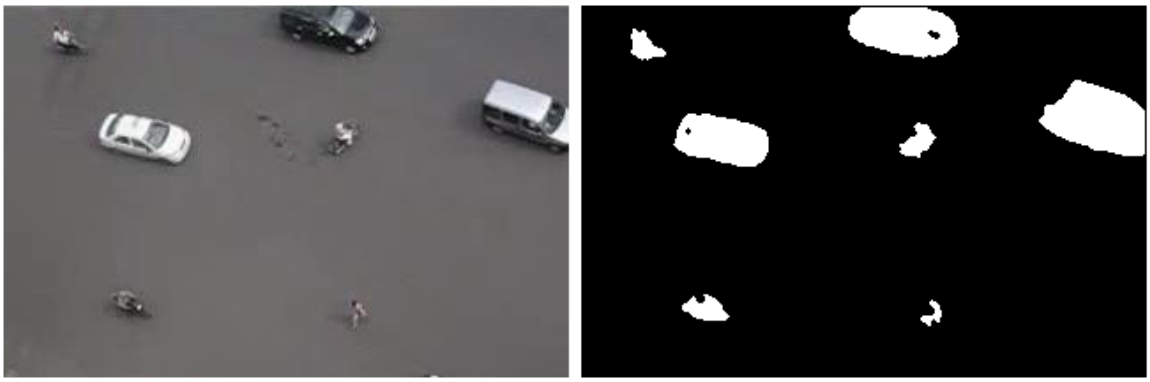
2.2. Traffic Image Analysis

There are basically four major steps involved in the video-based crash detection systems, various researches have been done on each individual section separately and good performance have been obtained. These are the following steps

1. Motion segmentation and vehicle detection
2. Vehicle tracking
3. Processing the results of tracking to compute traffic parameters
4. Accident detection using the traffic parameters

2.2.1. Motion segmentation and Vehicle Detection

Motion segmentation is the process of separating the moving objects from the background. The motion segmentation step is essential for detecting the vehicles in the image sequence. Figure 2.1 shows an example of vehicle detection. Figure 2.1(a) is the original image and Figure 2.1(b) shows the detected vehicle regions.



(a) (b)

Figure 2.1: Example of vehicle detection (a) Original image (b) Detected vehicle regions

There are four main approaches to detect vehicle regions, they are

1. Frame differencing method
2. Background subtraction method
3. Feature based method
4. Motion based method

2.2.1.1. Frame Differencing Method

In the frame difference method moving vehicle regions are detected by subtracting two consecutive image frames in the image sequence. This works well in case of uniform illumination conditions, otherwise it creates non-vehicular region and also frame differencing method does not work well if the time interval between the frames being subtracted is too large. Some of the vehicle detection methods using this technique are described in detail in [27]-[30].

2.2.1.2. Background Subtraction Method

Background subtraction method is one of the widely used methods to detect moving vehicle regions. In this step the either the already stored background frame or the background generated from the accumulated the image sequence is subtracted from the input image frame to detect the moving vehicle regions. This difference image is then thresholded to extract the vehicle regions.

The problem with the stored background frame is that they are not adaptive to changing illumination and weather conditions which may create non-existent vehicle regions and also works for stationary background. Therefore there is need to generate a background that is dynamic to the illumination and weather conditions. Various methods based on statistics and parametric model have been used. Some of the approaches [31] – [35] assumed Gaussian probability distribution for each pixel in the image. Then the Gaussian distribution model is updated with the pixel values from the new image frame in the image sequence. After enough information about model has been accumulated, each pixel (x, y) is classified either belonging to the foreground or background using the equation 2.1.

$$I(x, y) - Mean(x, y) < (C \times Std\ (x, y)) \tag{2.1}$$

where $I(x, y)$ is pixel intensity, C is a constant, $Mean(x, y)$ is the mean, $Std\ (x, y)$ is the standard deviation.

Single Gaussian distribution based background modeling works well if the background is relatively stationary and it fails if the background contains shadows and non-important moving regions (e.g., tree branches). This led the researches to use more than one Gaussian (Mixture of Gaussians) to build more robust background modeling technique. In Mixture of Gaussian methods [36] – [37] colors from a pixel in a background object are described by multiple Gaussian distributions. These methods were able to produce good background modeling. In all the above described methods several parameters need to be estimated from the data to achieve accurate density estimation for background [38]. However most of the times these information is not known beforehand.

Non-parametric methods do not assume any fixed model for probability distribution of background pixels [39]-[40]. These methods are known to deal with multimodality in background pixel distributions without determining the number of modes in the background. However these

systems does not adapt to sudden changes in illumination. So few methods based on Support Vector Machine (SVM), robust recursive learning were proposed to dynamically update the background [41]-[43]. Some methods [44] - [45] used Kalman filter to model the foreground and background and some other methods [46]-[47] used depth and color information to model the background using stereo camera. Background subtraction methods produced better segmentation results due to better background modeling, when compared to frame differencing method. But the disadvantages of background modeling to detect vehicle regions are high computational complexity making them difficult to operate in real-time and increased sensitivity to changes in lightning conditions.

2.2.1.3. Feature Based Method

Since the background subtraction methods needs accurate of modeling of background to detect moving vehicle regions, researched shifted their focus to detect moving vehicle regions using feature based methods. These methods made use of sub-features such as edges or corner of vehicles. These features are then grouped by analyzing their motion between consecutive frames. Thus a group of features now segments a moving vehicle from the background. The advantages of these methods [48] is that the problem of occlusion between the vehicle regions can be handled well, the feature based methods have less computational complexity compared to background subtraction method, the sub-features can be further analyzed for classifying the vehicle type and there is no necessity of stationary camera. Koller *et al.* [49] used displacement vectors from the optical flow field and edges as sub-features for vehicle detection. Beymer *et al*. [3] used corner of vehicles as features for measuring traffic parameters. Edges are used as features to detect vehicles by Dellart *et al*. [50]. Smith [51] used combination of corners and edges to detect moving vehicles. But the disadvantage of these systems is that if the features are not grouped accurately, then there may be failure in detecting vehicles correctly and also some of the systems are computationally complex and needs fast processing computers for real-time implementation.

2.2.1.4. Motion Based Method

Motion based approaches were also used to detect the vehicle regions in image sequences [38]. Optical flow based approaches were used to detect moving objects in the methods [52]-[53]. These methods are very effecting on small moving objects. Wixon [54] proposed an algorithm to detect salient motion by integrating frame-to-frame optical flow over time; thus it is possible to predict the motion pattern of each pixel. This approach assumes that the object tends to move in a consistent direction over time and that foreground motion has different saliency. The drawbacks of optical flow based methods are calculation of optical flow consumes time and the inner points of a large homogeneous object (e.g. car with single color) cannot be featured with optical flow. Some of the approaches used spatio-temporal intensity variations [43, 55] to detect motion and thus segment the moving vehicle regions.

2.2.2. Camera Calibration

Once the vehicle regions are detected and suitable features such as area, orientation, color, etc. are calculated, it is necessary for the 2D information obtained from the image to be mapped to the 3D information with respect to the real-world. Geometric camera calibration is the process of determining the 2D-3D mapping between the camera and the world coordinate systems [56]. Therefore there is a need to convert the camera coordinates to world coordinates using the internal (principal point, focal length, aspect ratio) and external parameters of the camera (position and orientation of camera in world coordinate system). We can classify the camera calibration techniques in two types: photogrammetric calibration and self-calibration. In photogrammetric calibration [57], an object whose 3D world coordinates is known in prior is observed by a camera to find the calibration parameters. In self calibration techniques [58], the position of the camera is changed to record static scenes and image with known internal

parameters, from which calibration parameters can be recovered. Figure 2.2 shows imaging geometry for perspective transformation.

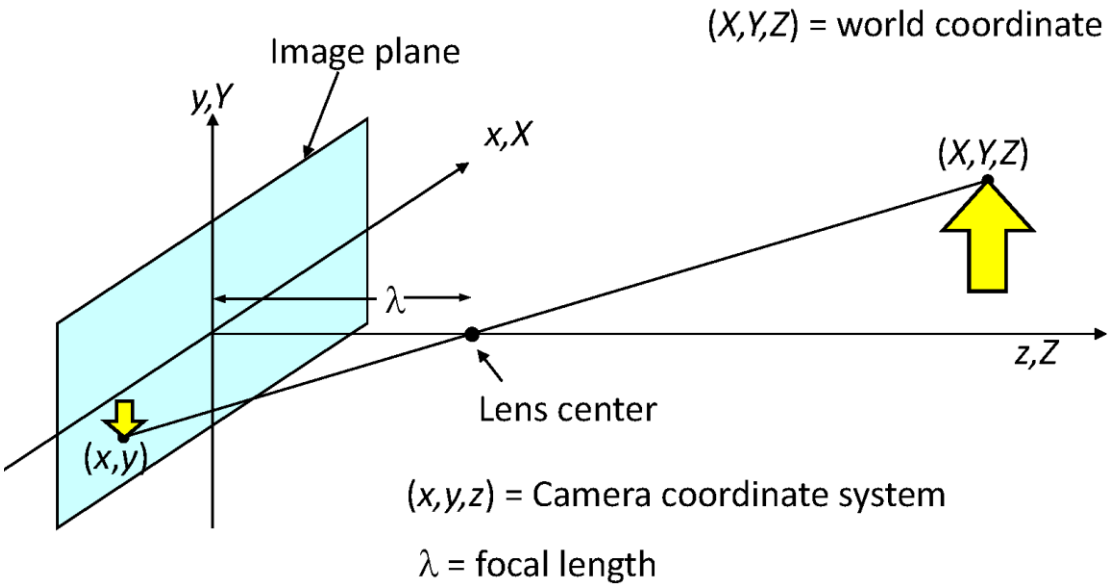


Figure 2.2: Imaging Geometry: Perspective Transformation [59]

Earlier research related to camera calibration employed full-scale non-linear optimization [60]-[62]. Good accuracy has been achieved using these methods, but the disadvantage of these methods is that they require a good initial guess and a computationally intensive non-linear search. Although the equations governing the transformation from 3D world coordinate to 2D image coordinate are nonlinear functions of intrinsic and extrinsic camera parameters, they are linear if lens distortion is ignored. By this assumption perspective transformation matrix can be computed using linear equations [63]-[65]. The main advantage of these methods is that they eliminate non-linear optimization. However, lens distortion cannot be modeled using these methods. If images are taken by the same camera with fixed internal parameters, correspondences between three images are sufficient to recover both the internal and external parameters [66]-[67]. Bu the disadvantage of these methods is they require a large number of features to achieve robustness.

Worall *et al.* employed an interactive tool for calibrating a camera that is suitable for use in outdoor scenes [68]. Wang and Tsai [69] used vanishing point technique to calibrate traffic scenes. Bas and Crisman [70] used the known height and tilt angle of the camera for calibration using a single set parallel line drawn by the user along the road edges while Lai [71] used an additional line of know length perpendicular to the road edges to remove the restriction of known height and tilt angle of the camera. Fung *et al*. [72] developed a method that is robust against small perturbations in markings along the roadside. The above described methods works well if the camera is static and fails if the position of the camera is changed.

To overcome the problems of calibration when the position of the camera is changed automatic calibration techniques were used. Daily *et al*. [73] relate pixel displacement to real-world units by fitting a linear function to scaling factors obtained using a known distribution of typical length of vehicles. Schoepflin and Dailey [74] calibrated PTZ cameras using lane activity maps which are computed by frame-differencing. Zhang *et al*. [75] used three vanishing point method to estimate the calibration parameters.

2.2.3. Vehicle Tracking

Once the vehicle regions are detected it is necessary for these vehicle regions to be tracked in the image sequences so that necessary information about the vehicle such as speed, vehicle trajectory, vehicle dimensions can be computed and used for further use. An illustration of vehicle tracking is shown in Figure 2.3. Figure 2.3(a) shows frame at time *t*, Figure 2.3(b) shows frame at time *t+1* and Figure 2.3(c) shows frame at time *t+2*. In these figures, the bounding box around each vehicle is same indicating that the vehicles are tracked correctly.

(a) (b) (c)

Figure 2.3: Illustration of vehicle tracking (a) Frame at time t (b) Frame at time t+1 (c) Frame at time t+2

Vehicle tracking is an important stage in crash detection systems. Given the detected vehicles, tracking can be viewed as a correspondence problem in which the goal is to determine which detected vehicle in the next frame corresponds to a given vehicle in the current frame. Over the years various researches have been conducted related to vehicle tracking. These approaches can be classified as follows.

1. Region-Based Tracking
2. Active Contour Tracking
3. 3D Model-Based Tracking
4. Markov Random Field Tracking
5. Feature- Based Tracking
6. Color and Pattern-Based Tracking
7. Other approaches

2.2.3.1. Region-Based Tracking

In this method image frame containing vehicles is subtracted from the background frame which is then further processed to obtain vehicle regions (blobs). Then these vehicle regions are tracked. Various methods have been proposed based on this approach [73, 76, 77]. Gupte *et al*. [76] proposed a method that performed vehicle tracking at two levels: region level and vehicle level. The method is based on the establishment of correspondences between regions and vehicles as the

vehicle move through the image sequence using maximally weighted graph. Bu the disadvantage of these methods is that they have difficulty in handling shadows, occlusion.

2.2.3.2. Active Contour Tracking

The next approach used to track vehicles was tracking the contours representing the boundary of the vehicle [78]-[79]. These are known as active contour models or *snakes*. Once the vehicle regions are detected in input frame the contours of the vehicle are extracted and dynamically updated in each successive frame. In the method used by Koller *et al*. [78] the vehicle regions are detected by background subtraction and tracked using intensity and motion boundaries of the vehicle objects. This method makes use of Kalman filters for estimating the affine motion and the shape of the contour. The advantage of active contour tracking over region-based tracking is the reduced computational complexity. But the disadvantage of the method is their inability to accurately track the occluded vehicles and tracking need to be initialized on each vehicle separately to handle occlusion better.

2.2.3.3. 3D Model-Based Tracking

In 3D model-based tracking [80]-[84] localize and track vehicle by matching a projected 3D model to the image data. Tan *et al*. [85]-[86] proposed a generalized Hough transformation algorithm based on single characteristic line segment matching an estimated vehicle pose and also analyzed the one-dimensional correlation of image gradients and determine the vehicle pose by voting. Pece *et al*. [87] presented a statistical Newton method for the refinement of the vehicle pose. The advantages of 3D model-based vehicle tracking is they are robust to interference between nearby images and also be applied to track vehicles which greatly change their orientations. But on the downside these models have high computational cost and they need detailed geometric object model to achieve high tracking accuracy.

2.2.3.4. Markov Random Field Tracking

Kamijo *et al*. [88] proposed a method to segment and track vehicle using spatiotemporal Markov random field. In this method the image is divided into pixel blocks and spatiotemporal Markov random field is used to update an object map using current and previous image. This method handled occlusions well. But the drawback of this method is that it does not yield 3Dinformation about vehicle trajectories in the world coordinate system. In addition in order to achieve accurate results the image in the sequence are processed in reverse order to ensure that vehicles recede from the camera. The accuracy decreased by a factor of two when the sequence is not processed in reverse, thus making the algorithm unsuitable for on-line processing.

2.2.3.5. Feature-Based Tracking

In feature-based tracking [78], [89]-[92] suitable features are extracted from the vehicle regions and these features are processed to track the vehicles correctly. These algorithms have low complexity and can operate in real-time and also can handle occlusions well. Beymer *et al*. [3] used feature tracking method in which the vehicle point features are tracked throughout the detection zone (entry and exit region). The feature grouping is done by constructing a graph over time, with vertices representing sub-feature tracks and edges representing the grouping relationships between tracks. Kanhere *et al*. [89] used 3D world coordinates of the feature point of the vehicle and grouped those points together in order to segment and track the individual vehicles. Texture-based features were used for tracking in [92]. Scale-Invariant features were used for tracking by Choi *et al*. [93]. The drawback of feature-based tracking is the recognition rate of vehicles using tow-dimensional image features is low, because of the non-linear distortion due to perspective projection and the image variations due to movement relative to the camera.

2.2.3.6. Color and Pattern-Based Tracking

Chachich *et al*. [94] used color signatures in quantized RGB space for tracking vehicles. In this work, vehicle detections are associated with each other by using a hierarchical decision process that includes color information, arrival likelihood and driver behavior. In [95], a pattern-recognition based approach to on-road vehicle detection has been studied in addition to tracking vehicles from a stationary camera. But the tracking based on color and pattern matching is not that reliable.

2.2.3.7. Other Approaches

The other approaches that were used for vehicle tracking includes optical flow based tracking [36, 51, 96], Markov Chain Monte Carlo based tracking [96] and Kalman-Filtering based tracking [97].

2.2.4 Accident Detection Systems

After vehicles in image sequence are detected and tracked correctly suitable traffic parameters (e.g. speed, trajectory, traffic flow) are extracted from the vehicle, the computer traffic parameters are used to detect incidents at highways and roadway intersections. Many research works have been done in the past to address this problem as there are numerous advantages in detecting accidents. Earlier researches related to traffic accident detection system involved detecting abnormal incidents such as traffic jam, detecting fallen-down obstactles, etc. Ikeda *et al*. [98] proposed an image processing based automatic abnormal incident detect system. The system is used to detect four types of incidents namely stopped vehicles, slow vehicles, fallen objects and vehicles that have attempted lane successive changes. Kimachi *et al*. [99] studied about vehicle behaviors causing incidents (e.g. traffic accident) using image processing techniques and fuzzy logic to predict an incident before it occurs. Trivedi *et al*. [100] described a method for developing distributed video networks for incident detection and management using

omnidirectional camera. Blosseville *et al*. [101] used image processing technique to detect shoulder incidents. Versavel and Boucke [102] presented a video incident detection method that used PTZ cameras. Michalopoulos and Jacobson [103] developed a method to detect incidents almost 2miles away. These methods were able to detect incidents to good extent with false alarm rate of about 3%. However most of these systems were used to detect incidents at roadways and have limited ability to detect traffic accidents at intersection.

Some of the approaches discussed above were limited to detect abnormal incidents at roadways; therefore more emphasis was given to determine accidents by later researchers. Atev *et al*. [104] used vision-based approach to predict collision at traffic intersection. In this method the vehicles are tracked using their centroid position and information about the vehicle such as velocity, position, width and height are computed and a bounding box is drawn around each tracked vehicle. Using the bounding box information around each vehicle collision is determined based on the amount to which the bounding box intersect. Hu *et al*. [105] used vehicle velocity and trajectory information to predict traffic accident using neural networking training of traffic parameters. Ki and Lee [106] used variation in speed, area, position and direction of the tracked vehicle to obtain and accident index which would determine the occurrence of accident at intersections. The method produced correct detection rate of 50% and false alarm rate of 0.00496% on their test conditions. Kamijo *et al*. [107] applied a simple left-to-right hidden Markov model to detect accident at traffic intersections. The system is able to determine three types of incident namely bumping accident, passing and jamming. However the conclusion was restrictive because there were not enough traffic accident data. Salim *et al*. [114] used speed, angle, position, direction, size of the tracked vehicle and using learning algorithm to detect collision at road intersections. Althoff *et al*. [117] used Markov chain probability model to detect collision. Zou *et al*. [118] used HMM to detect incidents at signaled intersection using modeled traffic parameters. How the disadvantage of these methods is that some algorithm need high level

learning algorithm that are computationally complex and may not feasible to be implemented in real-time.

2.3 Limitations of existing Vehicle Tracking and Accident Detection Systems

Some of the limitations of the existing vehicle detection, tracking and accident detection systems are given below

1. Most of the existing vehicle detection and tracking systems have high computational complexity
2. Some of the vehicle detection systems use sophisticated background modeling methods which adds to the complexity of the algorithm.
3. Some of the existing vehicle tracking systems need high speed processors for implementation.
4. Some of the systems make use of high level learning algorithm such as HMM, neural network to detect accidents which make the real-time implementation difficult.
5. Real-time implementation of these systems is not feasible on low speed processors

In this chapter we have briefly discussed about the existing vehicle tracking and accident detection systems and their advantages and disadvantages. Considerable number of research work has been done related to this field and few systems have been used in practical situations. However there is room for improvement in all the systems that have been reviewed

CHAPTER 3

SYSTEM OVERVIEW AND CAMERA CALIBRATION

3.1. Problem Definition

The problem addressed in this research is preliminary approach to real -time implementation of accident detection systems at traffic intersections. Therefore the main objective of this research is to design and implement an accident detection system through video processing that is suitable for real-time implementation. Figure 3.1 illustrates brief overview of the system.

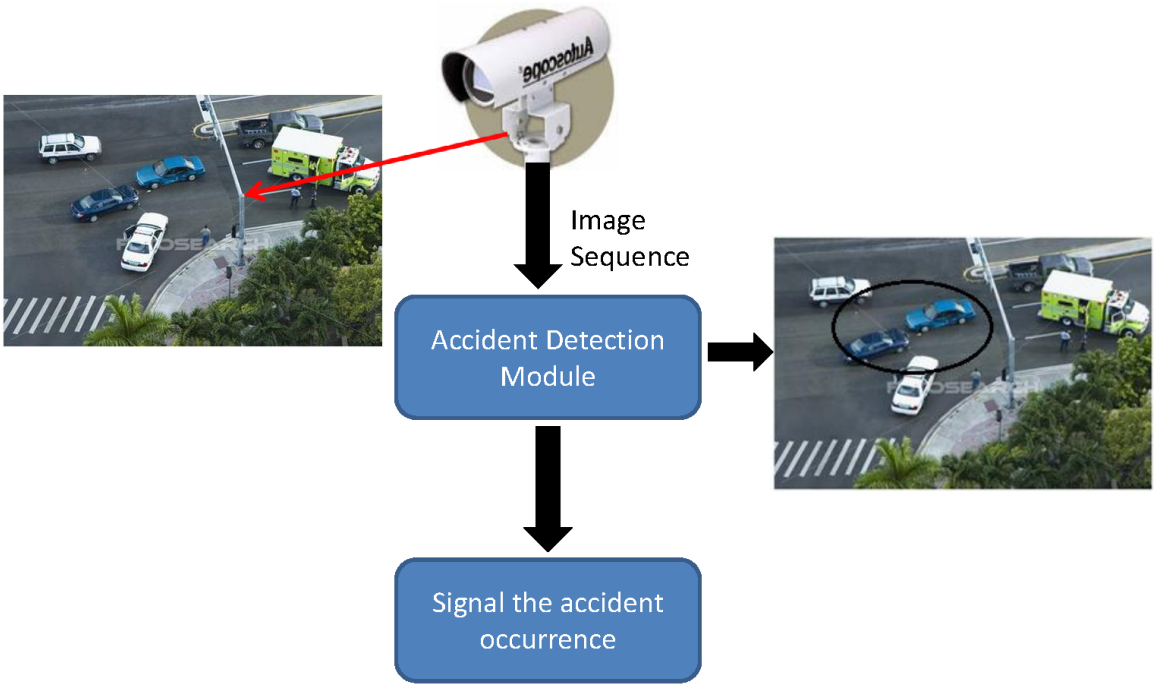


Figure 3.1: Brief overview of the accident detection system [accident image shown in the figure is downloaded from http://www.fotosearch.com/GLW051/690397494/]

3.2. Approach

As shown in the Figure 3.1, image sequences are extracted from the video camera mounted on the pole at traffic intersection. The image sequences are fed to the accident detection module system where the occurrence of the accident is determined. The accident detection system consists of vehicle detection, vehicle tracking, vehicle parameter extraction and accident detection sections. The brief overview of the accident detection module is presented in detail in chapter 1, section 1.3. In addition to the image input, some more auxiliary information is also fed as input to the system. The following information is fed as input to the system. They are: stored background image, threshold values for the image processing, information about the position and orientation of the camera, camera calibration parameters and frame rate of the video sequence. After analyzing the image sequence, the system identifies the moving vehicles in the image and tracks them using low-level features. After the vehicles are tracked correctly in each frame, the speed, orientation, position and area of the tracked vehicle are used to determine the occurrence of accident. Once an occurrence of accident is detected, the system signals the detection of accident to the user.

3.2.1. Advantages of the System

The following are the advantages of the system

1. System is able to operate in real-time with common CPU processor and with the use of high speed processor; the processing rate can be speeded up further.
2. The vehicle detection and tracking algorithm used in the system have low computational cost.
3. Vehicle tracking method used in the system makes use of low-level features which have low complexity when compared to the existing systems discussed earlier

4. The system focuses on real-time implementation of accident detection at traffic intersection where the occurrence of accidents is estimated to be more.
5. Because of the fast operating time, future enhancements can be added to the system to make it more robust.

3.2.1. Limitations and Assumptions

The following list the limitations and assumptions used in the thesis

1. The system assumes that the video camera used to record the traffic image sequence is assumed to be parallel to the ground plane.
2. The system works well only in daylight conditions. The performance of the system has not been evaluated in night conditions.
3. The system does not handle occlusion well especially if part of a vehicle is occluded with another vehicle.
4. Although the system is able to achieve real-time performance, the method had been tested and evaluated offline. Online evaluation of the algorithm has not been done.

3.3. Experimental Setup and Testing Conditions

For the purpose of testing our algorithm we used two video sequences obtained from traffic intersection at Saigon, Vietnam. These videos were used to test the performance of the tracking algorithm. These videos were used as they provided nice scenarios for busy traffic intersection. The testing was done offline and results were used to improve the performance of the tracking algorithm. Few frames from the test videos are shown in the Figure 3.2 and Figure 3.3. Figure 2.4 shows the frame sequences from the video named *saigon01.avi* and Figure 2.5 shows the frame sequences from the video named *saigon02.avi.*

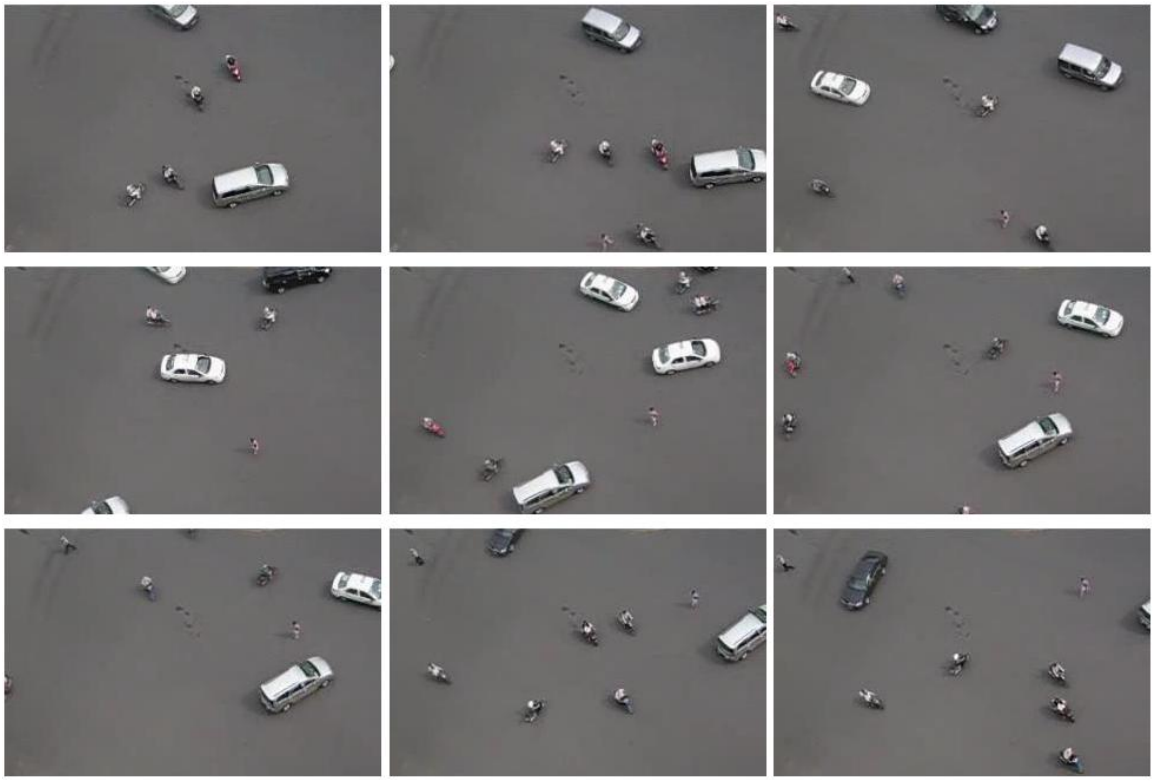


Figure 3.2: Frame sequences from test video *saigon01.avi*

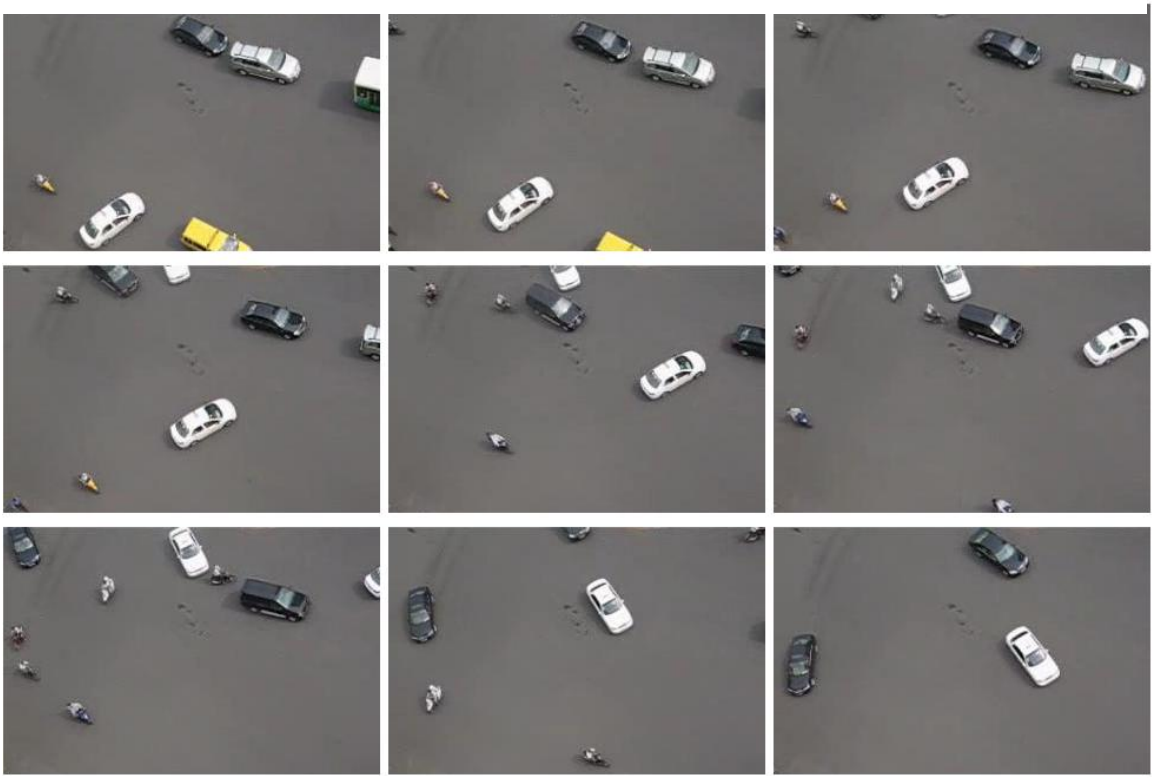


Figure 3.3: Frame sequences from test video *saigon02.avi*

3.3.1. Indoor Experimental Setup

For the purpose of testing the accident detection system, indoor experimental setup was made in the laboratory with artificial lighting mimicking daylight conditions. The video camera used for this purpose was Q24 Mobitix Camera. The camera was placed parallel to the ground plane to shoot the testing sequence. PC controlled cars were used for the purpose of tracking vehicles and creating collision between vehicles. The advantage of PC controlled cars is that the trajectory and the velocity of the vehicles can be controlled by the user. Optical tracking system was also used in the setup. The advantage of Optical tracking system is that the velocity information of the vehicles can be obtained instantaneously which is used to verify the velocity information determined by the accident detection system. Figure 3.4 shows the experimental setup used in the laboratory.

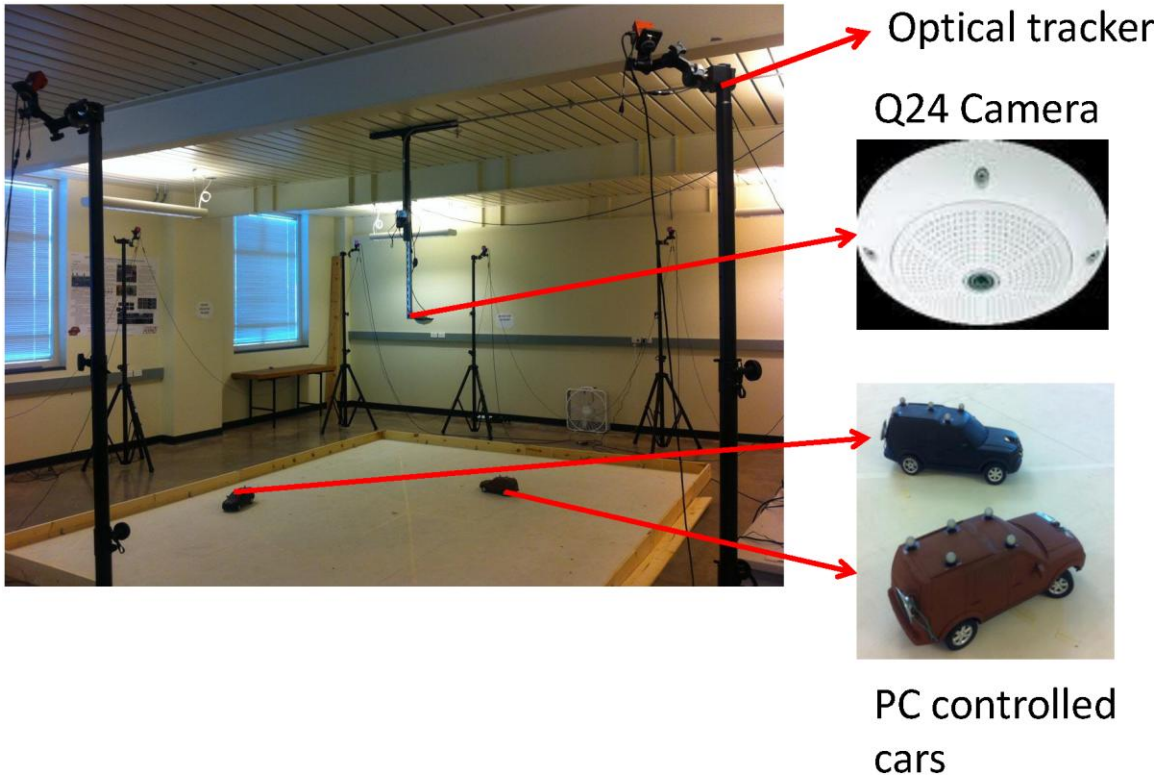


Figure 3.4: Experimental setup used in the laboratory

Figure 3.5 shows the frame sequence in the testing video obtained from the laboratory setup. The bottom row shows a scenario for occurrence of accident.

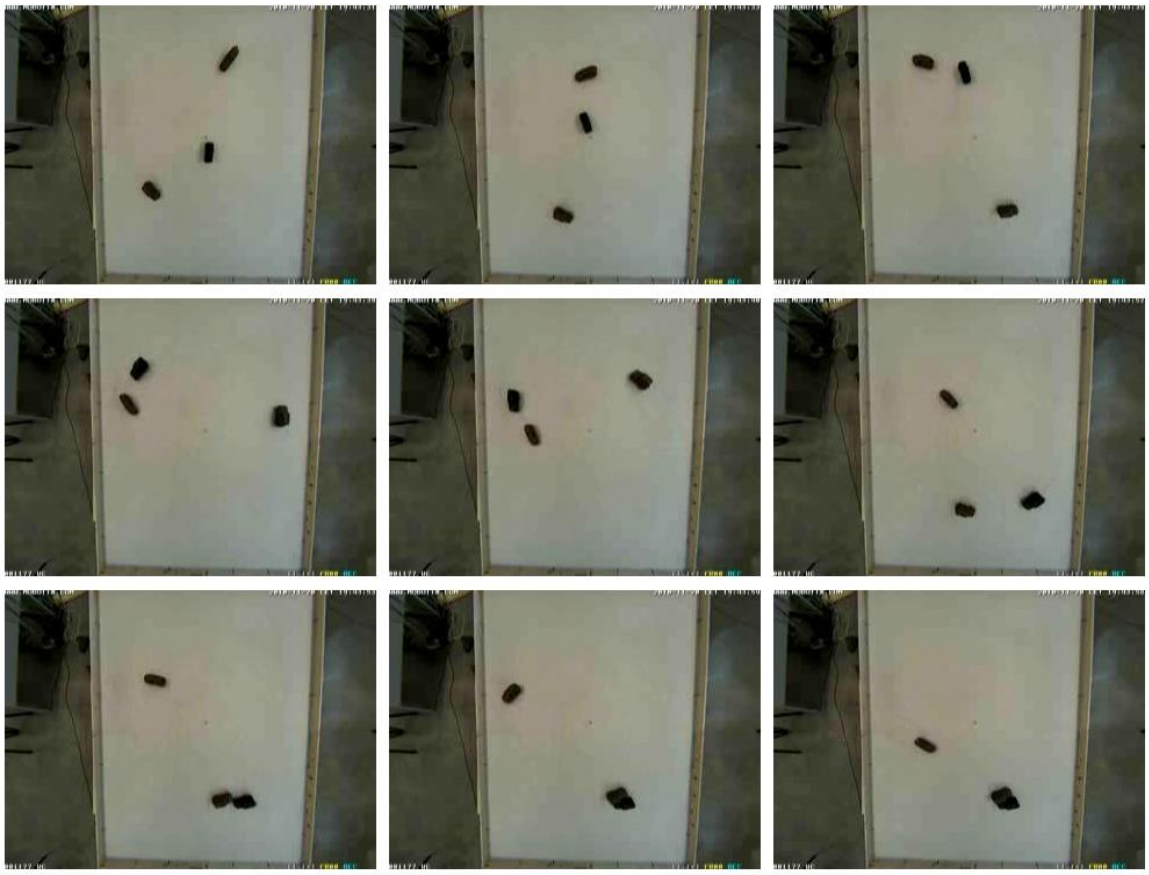


Figure 3.5: Frame sequences from the test video obtained from the laboratory setup

The algorithm for vehicle detection, vehicle tracking and accident detection system are explained in detail in the following chapters. The detail description of vehicle detection is presented in Chapter 3.

3.4. Camera Calibration

Camera Calibration is an essential step in a vision-based vehicle tracking system. Camera calibration involves estimating a projective matrix which describes the mapping of points in the world onto the image plane. A calibrated camera enables us to relate pixel-measurements to measurements in real world units (e.g., feet) which are used to handle scale changes (as vehicles

approach or recede from the camera) and to measure speeds [108]. Two types of parameters need to be estimated to find the mapping between the image coordinates and world coordinates. They are extrinsic and intrinsic parameters of the camera. The extrinsic parameters define the location and orientation of the camera reference frame with respect to a known world reference frame. The intrinsic parameters are necessary to link the pixel coordinates of an image point with the corresponding coordinates in the camera reference plane [109]. Figure 3.6 shows mapping of 3D point to 2D point.

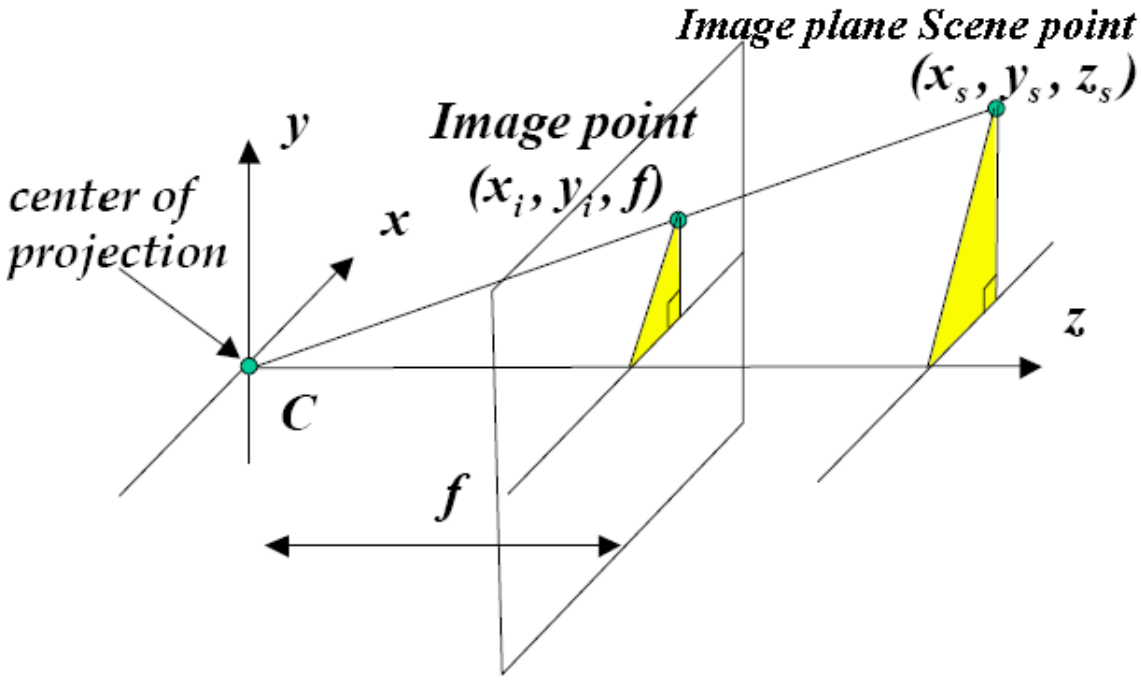


Figure 3.6: Mapping 3D point to 2D point [109]

3.4.1. Calibration for *Saigon* Traffic Videos

For the *Saigon* videos the camera is assumed to be placed parallel to the ground plane since accurate information about the camera position is not known. From close observation of the traffic video it was estimated that the camera was placed at very high angle, presenting good top-

down view of the traffic intersection. Figure 3.7 shows the scenario encountered in case of *Saigon* traffic intersection. The camera is place parallel to the ground plane.

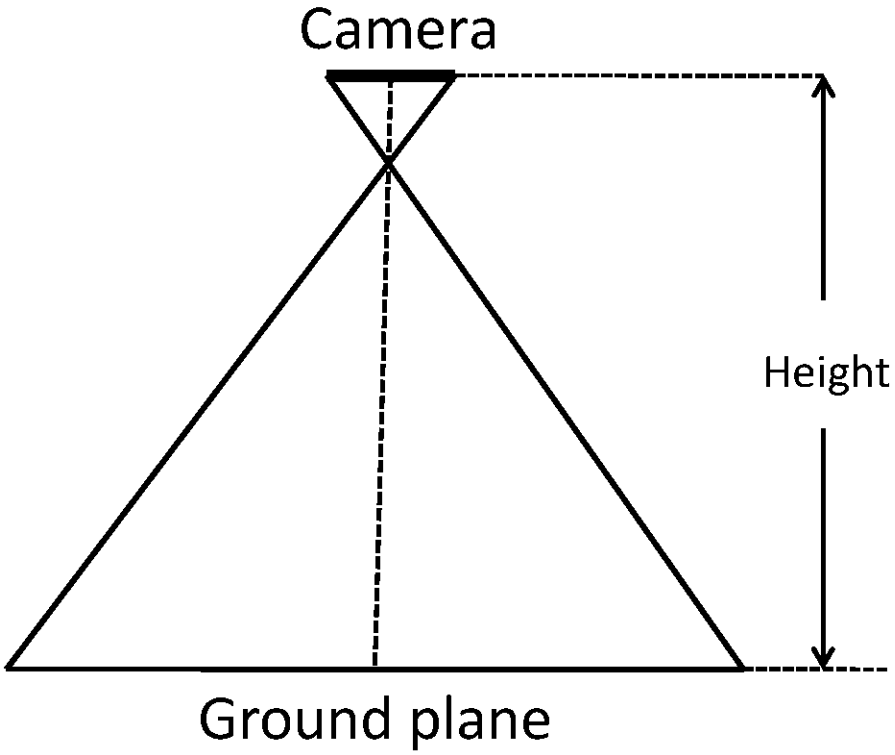


Figure 3.7: Camera placed parallel to the ground plane

3.4.1.1. Photogrammetric correction factor

Since the sizes of the objects in an image are measured in pixels, they have to be converted into units used in the real world to determine the sizes and speeds of vehicles to which they correspond [48]. The Figure 3.8 illustrates the situation where the camera is place parallel to the ground plane.

From the image geometry shown in Figure 3.7

$$\frac{y}{x} = \frac{d_2}{d_1} = constant \tag{3.1}$$

$$x = \left(\frac{d_1}{d_2}\right) \times y \tag{3.2}$$

where (d_1/d_2) = scale factor

In this case, the actual distance can be calculated by multiplying the distance in pixels by the scale factor. Similarly, the area of a blob can be calculated using the scale factor.

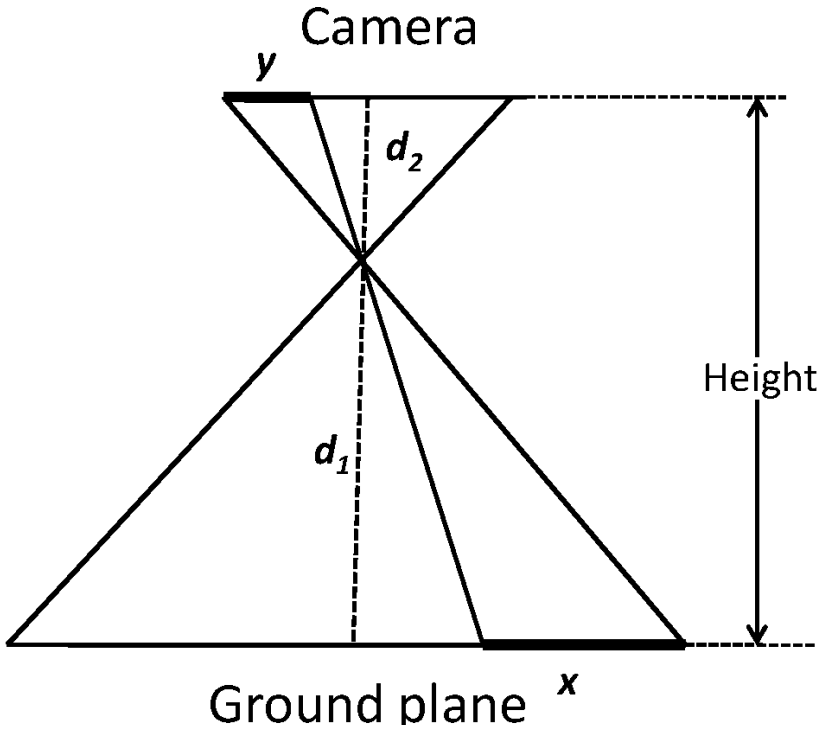


Figure 3.8: Photogrammetric correction for camera placed parallel to the ground plane

3.4.2. Camera Calibration for *Q24 Mobitix Camera*

A perspective-projective pinhole camera model is assumed. The relationship between an object point measured with respect to user-defined world coordinate system and its image plane is described by a 3x4 homogeneous transformation matrix [108]. The detailed description of the intrinsic and extrinsic parameters can be found in [109,110,111].This matrix will be referred as the camera calibration matrix *C*, where $C = C_{in}C_{ext}$. C_{in} is the matrix containing the intrinsic parameters and C_{ext} is the matrix containing the extrinsic parameters.

The matrix containing the intrinsic camera parameters is given by:

$$C_{in} = \begin{bmatrix} -f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

where f is the focal length of the camera. $[s_x\ s_y]$ correspond to the effective size of the pixels in the horizontal and vertical directions (in millimeters). $[o_x\ o_y]$ are the coordinates of the principal point (in pixels).

The matrix containing the extrinsic camera parameters is given by:

$$C_{ext} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & -{R_1}^T T \\ r_{21} & r_{22} & r_{23} & -{R_2}^T T \\ r_{31} & r_{32} & r_{33} & -{R_3}^T T \end{bmatrix} \tag{3.4}$$

where T is the translation vector and R is the rotation matrix given by

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{3.5}$$

$$\hat{p} = C\hat{P} \tag{3.6}$$

where $\hat{p} = [uw\ vw\ w]$ and $\hat{P} = [x\ y\ z]^T$ are vectors containing homogeneous coordinates of image point, $p = [u\ v]$ and world point $P = [x\ y\ z]^T$ respectively. Representing the matrix with corresponding entries we get

$$[uw\ vw\ w] = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \end{bmatrix} [x \quad y \quad z \quad 1] \tag{3.7}$$

The homogeneous transformation matrix C is unique only up to a scale factor. C by making the scale factor $c_{34} = 1$.

Expanding the above equation (3.7), yields

$$u = \frac{c_{11}x+c_{12}y+c_{13}z+c_{14}x}{w} \tag{3.8}$$

$$v = \frac{c_{21}x+c_{22}y+c_{23}z+c_{24}x}{w} \tag{3.9}$$

$$w = c_{31}x + c_{32}y + c_{33}z + 1 \tag{3.10}$$

Substituting *w* into two equations (3.8) and (3.9) gives

$$u = \frac{c_{11}x+c_{12}y+c_{13}z+c_{14}x}{c_{31}x+ c_{32}y + c_{33}z+1} \tag{3.11}$$

$$v = \frac{c_{21}x+c_{22}y+c_{23}z+c_{24}x}{c_{31}x+ c_{32}y + c_{33}z+1} \tag{3.12}$$

Equations (3.8) and (3.9) define a mapping from the world coordinates to the image coordinates.

For a point in the world, we can calculate its image coordinates if we know the location of that point in terms of the user-defined world-coordinate system and camera calibration matrix, *C*. The camera calibration matrix *C* consists of 11 unknown parameters. Knowing the world coordinates and the image coordinates of a single point yields to equations of the form (3.8) and (3.9). Six or more points in a non-degenerate configuration lead to an over-determined system:

$$\begin{bmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1z_1 \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1x_1 & -v_1y_1 & -v_1z_1 \\ x_2 & y_2 & z_2 & 1 & 0 & 0 & 0 & 0 & -u_2x_2 & -u_2y_2 & -u_2z_2 \\ 0 & 0 & 0 & 0 & x_2 & y_2 & z_2 & 1 & -v_2x_2 & -v_2y_2 & -v_2z_2 \\ . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . & . & . & . \\ x_n & y_n & z_n & 1 & 0 & 0 & 0 & 0 & -u_nx_n & -u_ny_n & -u_nz_n \\ 0 & 0 & 0 & 0 & x_n & y_n & z_n & 1 & -v_nx_n & -v_ny_n & -v_1z_1 \end{bmatrix} \begin{bmatrix} c_{11} \\ c_{12} \\ c_{13} \\ c_{14} \\ c_{21} \\ . \\ . \\ . \\ c_{33} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ . \\ . \\ . \\ u_n \\ v_n \end{bmatrix} \tag{3.10}$$

which can be solved using a standard least squares technique.

The offline calibration process depends upon the user-specified point correspondences for the calibration process. For improving the accuracy, it is desired that the world coordinates are derived from the actual measurements of the scene (e.g., having place markers at known distance). The internal and external parameters of the camera were measured using the Caltech Camera Calibration Toolbox [112]. In total 15 checker board patterns with different orientation and position with respect to the camera-centered view were used. Figure 3.9 shows some of the checker board patterns used for the purpose of camera calibration as required by the Camera Calibration Toolbox. Figure 3.10 shows an illustration of camera calibration process for direct estimation of projective matrix as done by the Toolbox. Figure 3.11 shows the estimation of extrinsic parameters with camera-centered view. The detail description of the Camera Calibration process is given in [112]. Using the dimensions of a known type of object (e.g., vehicle, marker) is an approximated method for estimating world coordinates of control points.

From the camera calibration parameters a point in three dimensional space is mapped into a two dimensional image plane. The loss of dimension result into a non-invertible mapping. Given the calibration parameters for the camera and the image coordinates of a single point, the best we can do is to determine a ray in space passing through the optical center and unknown point in the world. To measure distances in the road plane, we can substitute z=0 in above equation (3.10) to get the mapping points from the image plane (u,v) to corresponding points in the road plane (x,y):

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} c_{11} - uc_{31} & c_{12} - uc_{32} \\ c_{21} - vc_{31} & c_{22} - vc_{32} \end{bmatrix}^{-1} \begin{bmatrix} u \\ v \end{bmatrix} \tag{3.11}$$

The following intrinsic and extrinsic parameters were obtained from the Camera Calibration toolbox: The resolution of images used for camera calibration is *640* x *480 pixels*.

Intrinsic parameters:

Focal length [f_x f_y] = [492.09272 399.23243] ± [108.28708 56.24564] (*in pixels*)

Principal point [o_x o_y] = [252.55073 187.45716] ± [34.26970 17.50563](*in pixels*)

Skew = 0; *Angle of pixel axes* = 90 *degrees*.

Extrinsic parameters:

Rotation matrix $R = \begin{bmatrix} 0.21 & 0.97 & -0.099 \\ 0.79 & -0.23 & -0.57 \\ -0.006 & -0.73 & -0.69 \end{bmatrix}$

Translation vector $T = \begin{bmatrix} -162.54 \\ -271.06 \\ 3126.48 \end{bmatrix}$

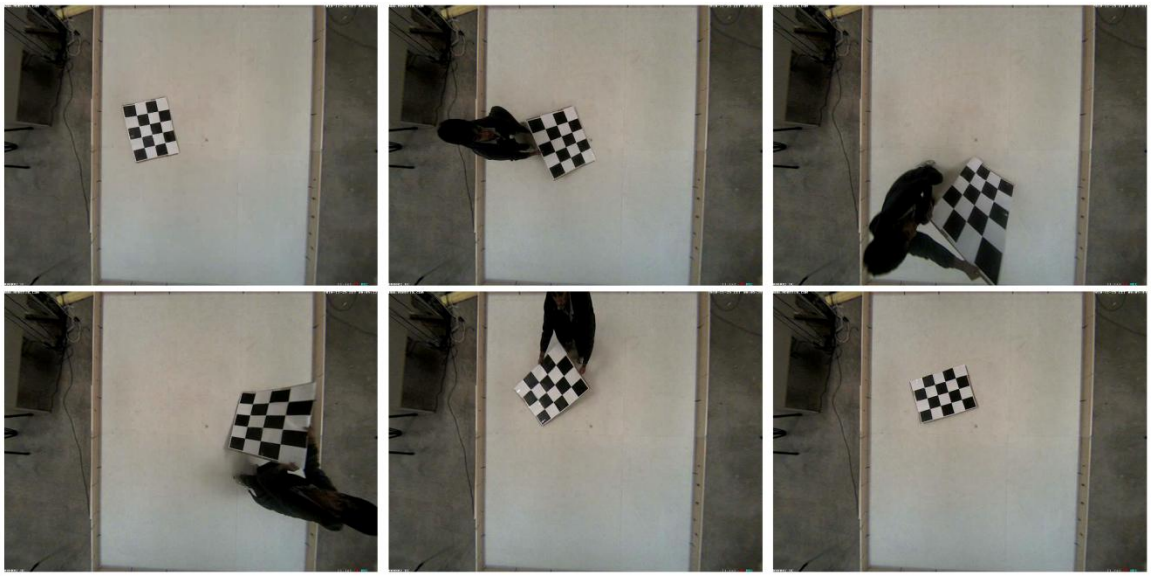


Figure 3.9: Checker board patterns used for camera calibration

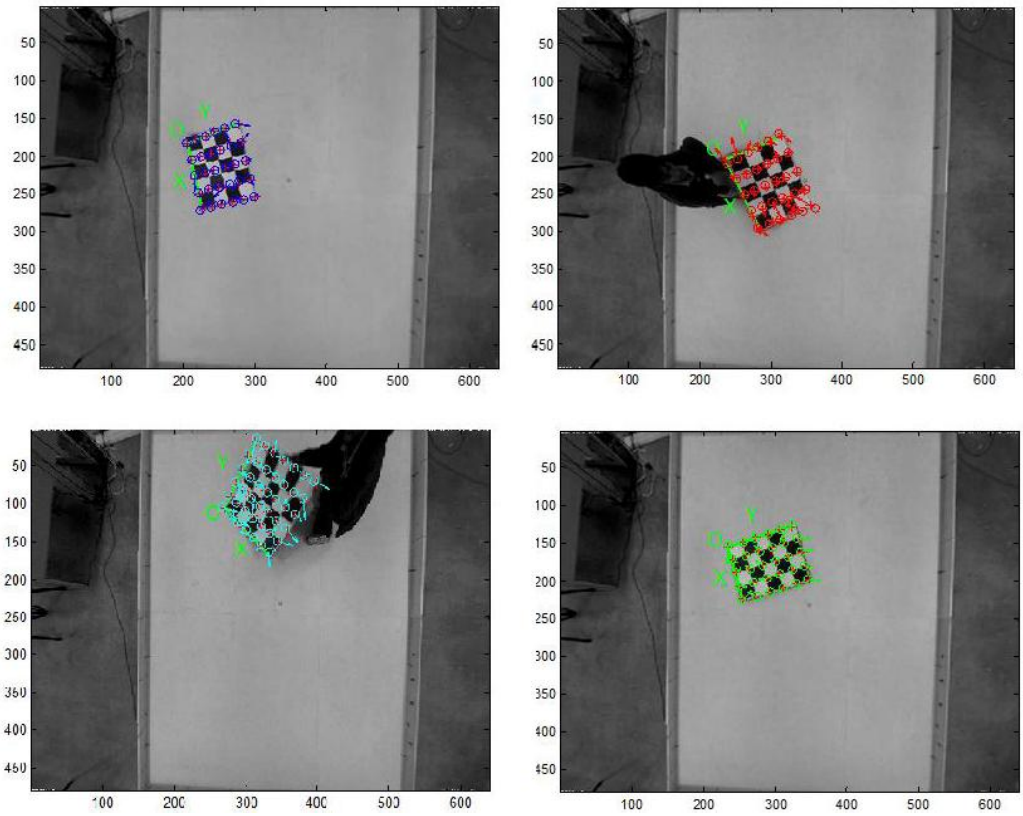


Figure 3.10: Illustration of Camera Calibration process for direct estimation of projective matrix

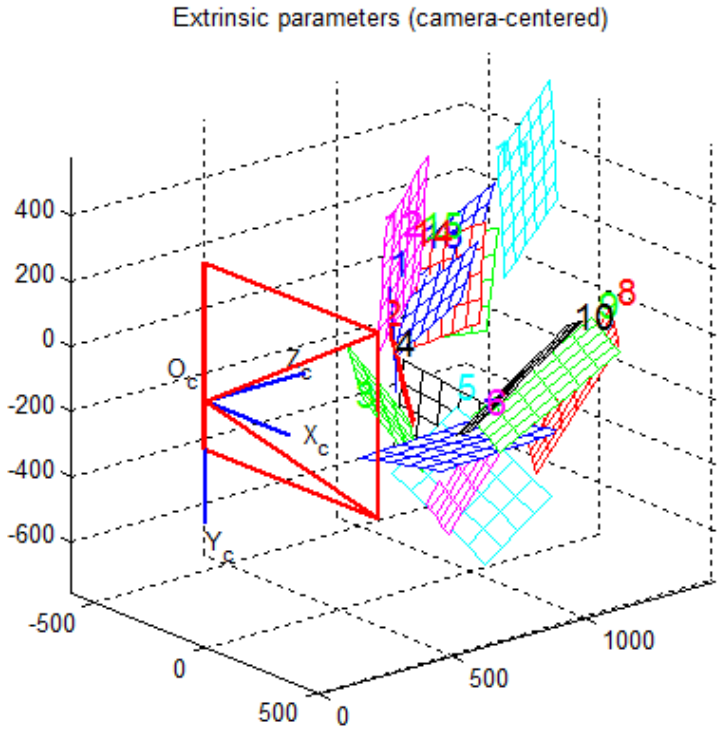


Figure 3.11: Estimation of Extrinsic Parameters

CHAPTER 4

VEHICLE DETECTION AND FEATURE EXTRACTION

4.1. Vehicle Detection

Vehicle Detection is an important stage of the accident detection system in which the moving vehicles are segmented from the background. Figure 4.1 shows brief description of vehicle detection system.

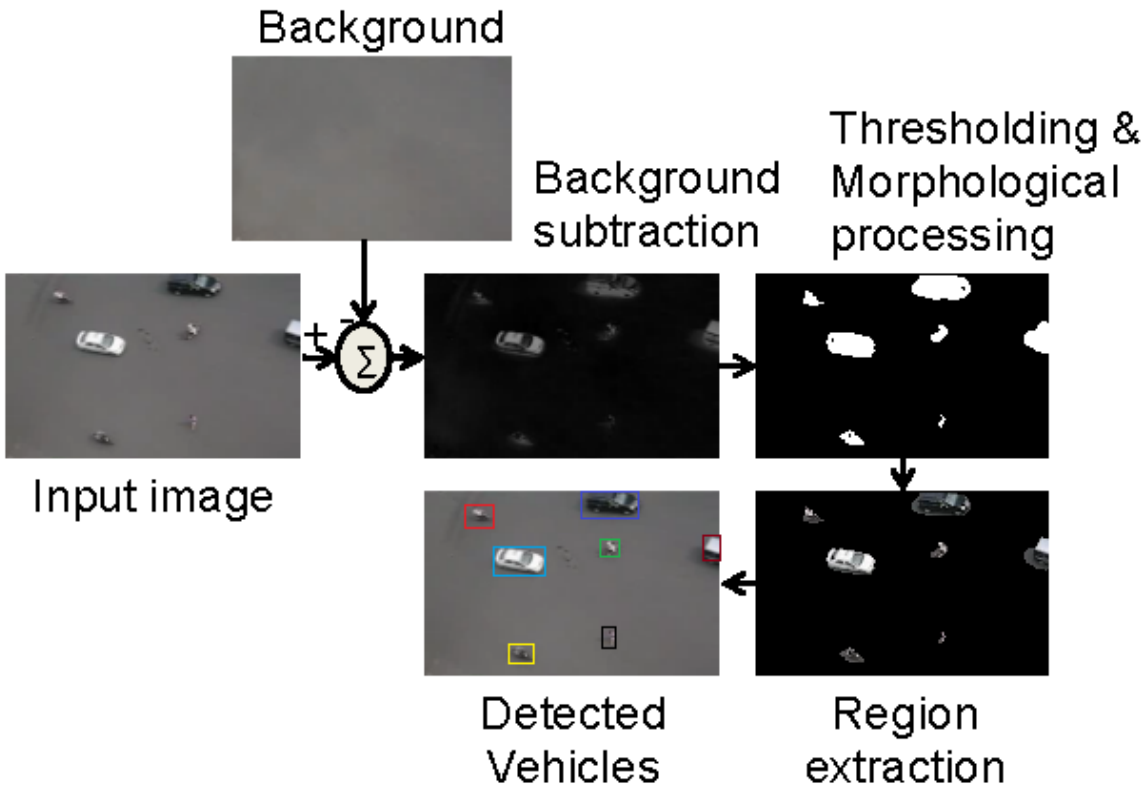
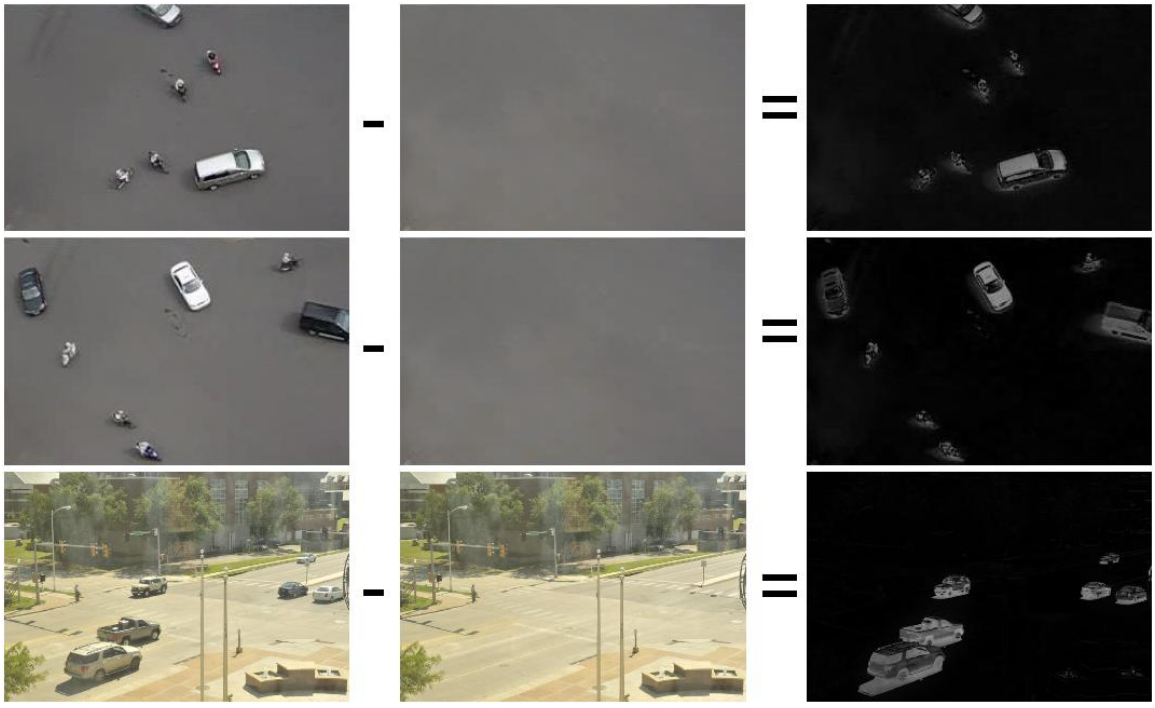


Figure 4.1: Description of vehicle detection system

The method that is used for detecting moving vehicles is background subtraction. Since the research focused on real-time implementation of the system, background modeling techniques that have high computational cost were not tried. And also since the testing of the algorithm is done offline and the position of the camera recording the video sequence is static we used a stored background frame for background subtraction. Once the vehicle regions are detected, suitable low-level features are extracted from the vehicle regions. The process of vehicle detection is explained in detail in the following sections.

4.1.1. Background Subtraction

The first step in the algorithm is to subtract the background from the current input frame to detect the vehicles. Figure 4.2 shows examples of background subtraction method. Figure 4.2(a) shows the input frame, Figure 4.2(b) shows the background frame used and Figure 4.3(c) shows the difference image.



(a) (b) (c)

Figure 4.2: Examples of Background Subtraction (a) Input frame (b) Background frame (c) Difference Image

Here frame at time *t* from the input video along with the previously acquired background frame (containing no vehicles) is fed as input to the algorithm. The algorithm subtracts the intensity value of each pixel in the frame $I_t(x,y)$ from the background image $I_{bk}(x,y)$ resulting in a difference image $I_{diff}(x,y)$ given by

$$I_{diff}(x,y) = |I_t(x,y) - I_{bk}(x,y)| \tag{4.1}$$

As mentioned, this background subtraction step is performed to detect moving objects since the static objects are part of background. Thus, we are left with the intensity values of moving objects in the difference image $I_{diff}(x,y)$.

4.1.2. Thresholding and Morphological Processing

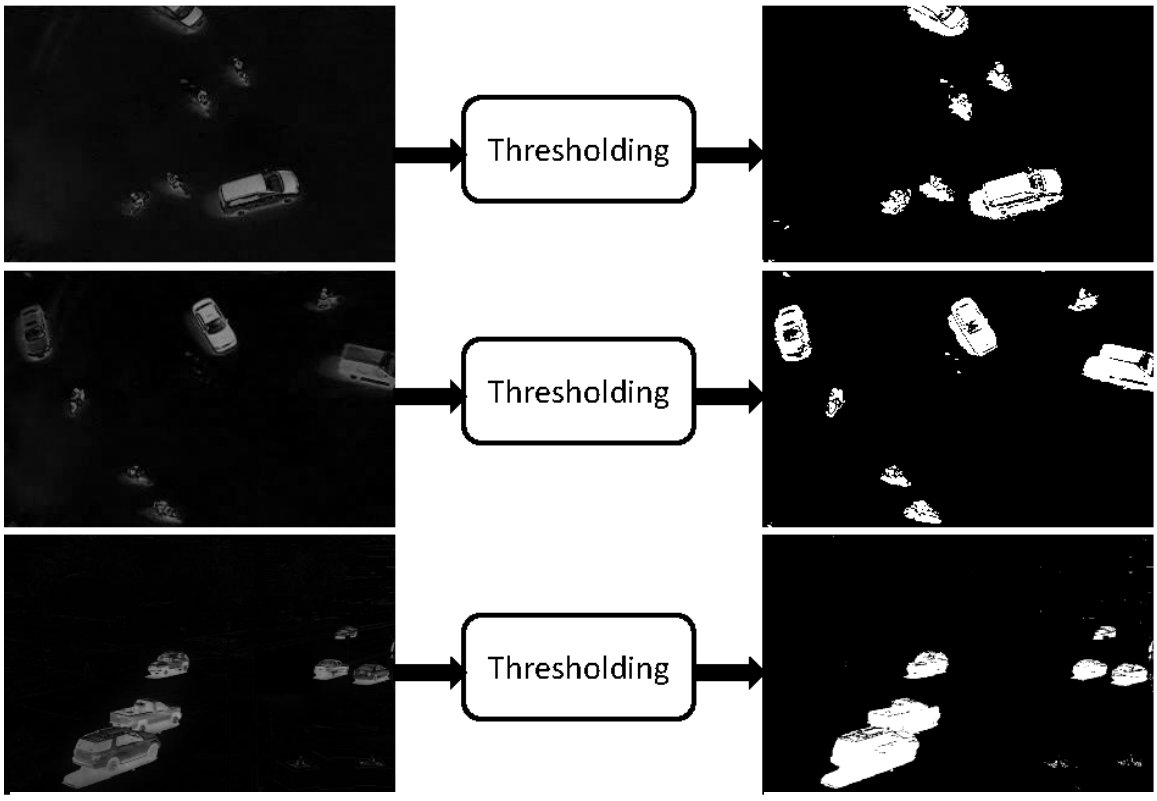


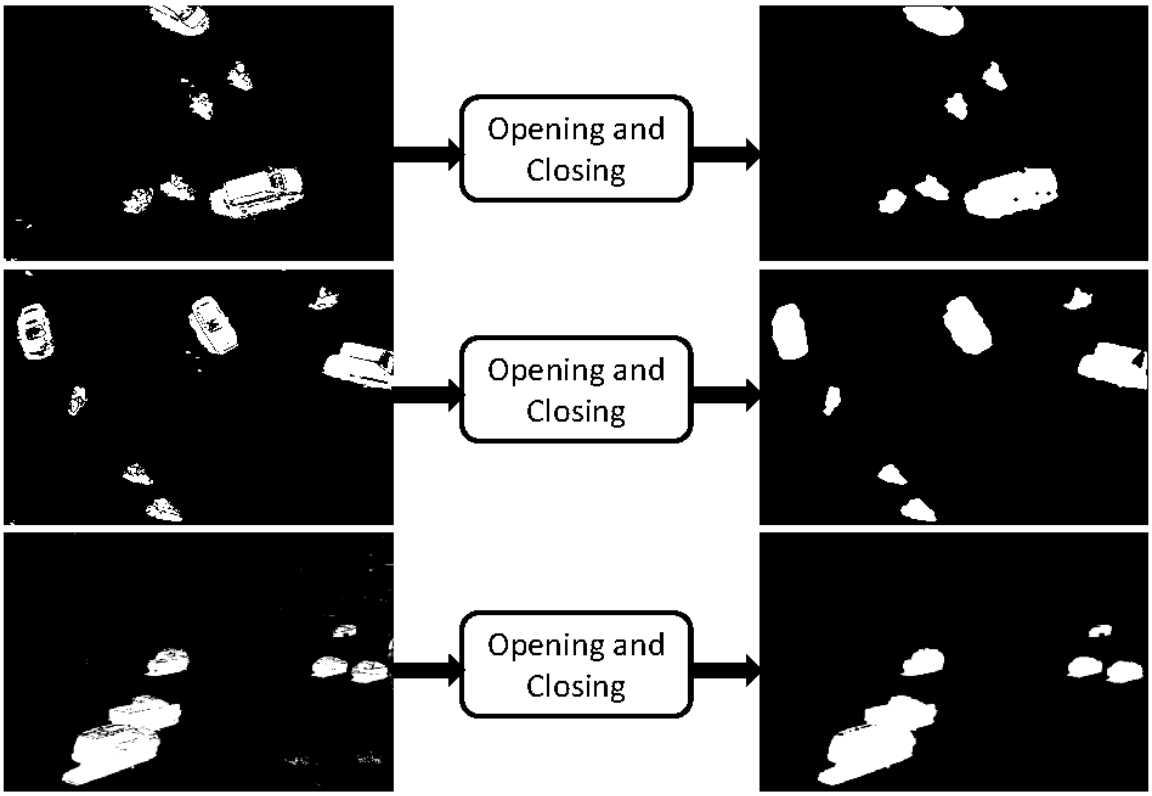
Figure 4.3: Illustration of thresholding (a) Difference image (b) Thresholded image

The difference image $I_{diff}(x,y)$ is converted into a binary image $bw(x,y)$ using a specific threshold value T as follows

$$bw(x,y) = \begin{cases} 1, I_{diff}(x,y) \geq T \\ 0, I_{diff}(x,y) < T \end{cases} \quad (4.2)$$

The value of T was empirically estimated to be 0.1 in the experiment. Figure 4.3 shows the process of thresholding. Figure 4.3(a) shows the difference image and Figure 4.3(b) shows the thresholded image.

The binary image $bw(x,y)$ obtained from thresholding suffers from noise and unwanted pixel values. Therefore morphological operations, opening followed by closing is done on the binary image $bw(x,y)$ to obtained a final cleaned image $bw_{final}(x,y)$. Figure 4.4 shows the



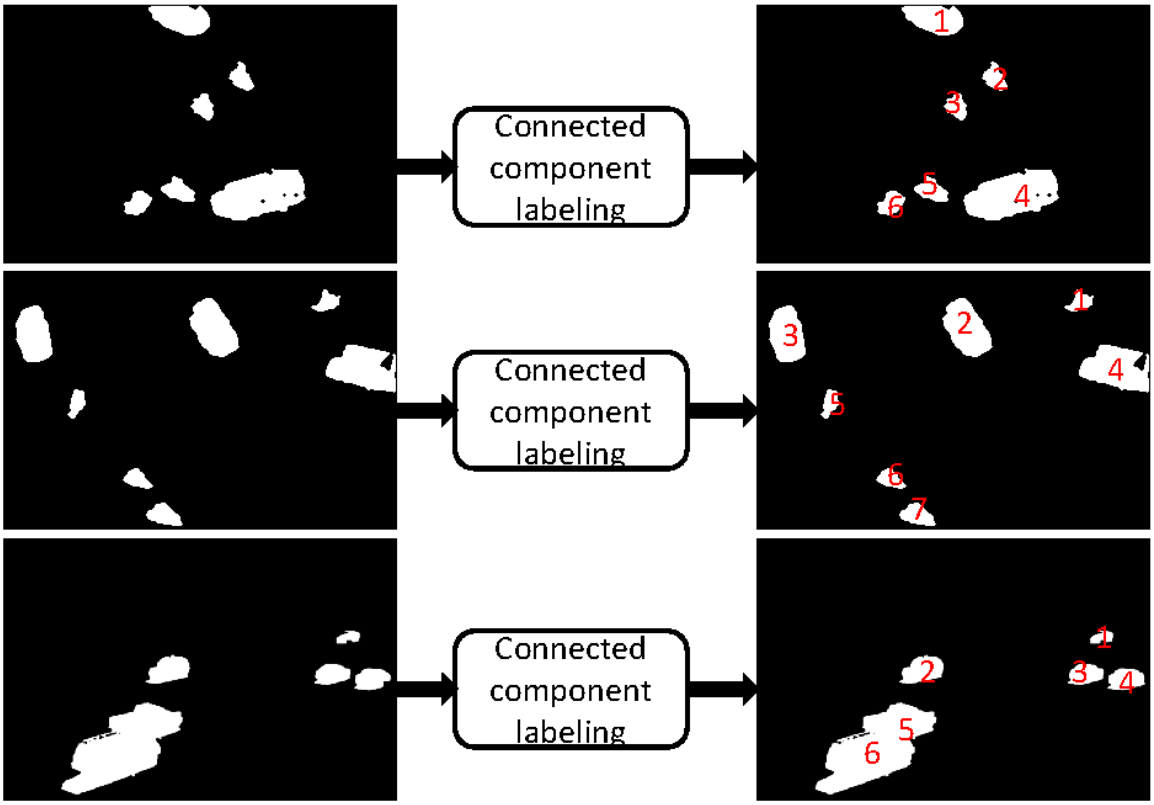
(a) (b)

Figure 4.4: Illustration of Morphological Processing (a) Thresholded image (b) Cleaned image

morphological processing. Figure 4.3(a) shows the thresholded image and Figure 4.3(b) shows the final cleaned image after morphological operations. The final binary image $bw_{final}(x,y)$ consists regions of individual detected vehicles.

4.1.3. Connected Component Labeling and Region Extraction

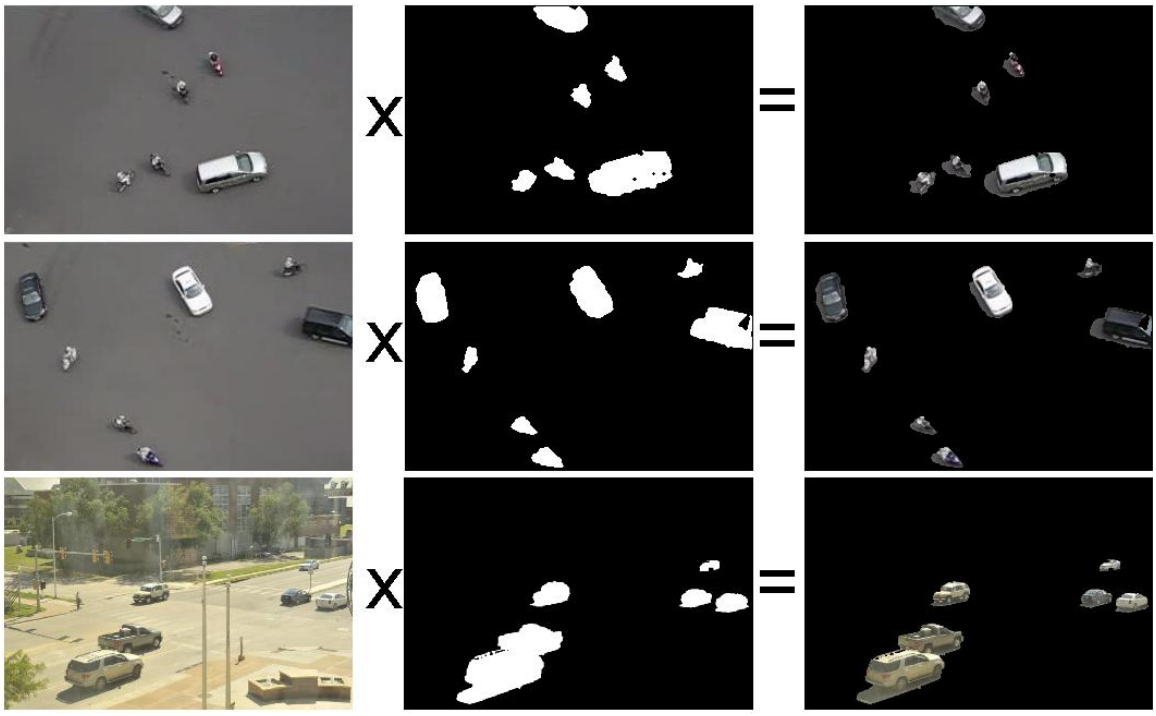
The regions in the binary image $bw_{final}(x,y)$ are labeled using connected component labeling. This process labels the regions in the binary image and yields an estimate of the number of connected components. From this process, the number of vehicles detected in the image is estimated. Figure 4.5 shows connected component labeling output. Figure 4.5(a) is the binary image and Figure 4.5(b) is the labeled image.



(a) (b)

Figure 4.5: Connected component labeling (a) Binary image (b) Labeled image

After connected component labeling, the binary map is applied on the original input frame $I_t(x,y)$ and hence we focus only on those regions in which the vehicle are detected. Figure 4.6 illustrates the process of vehicle region extraction. Figure 4.6(a) shows the input image; Figure 4.6(b) shows the binary mask of the vehicle regions and Figure 4.6(c) shows the detected vehicle regions from the input image.



(a) (b) (c)

Figure 4.6: Illustration of vehicle region extraction (a) Input image (b) Binary image (c) Detected vehicle regions

4.2. Feature Extraction

After the regions containing vehicles are extracted, suitable low-level features are extracted from the vehicle regions. The features used are area, centroid, orientation, luminance and color. These features are used because of their low computational complexity. Figure Let X_i $\{i= 1, 2, 3...\}$ denote the individual vehicle regions detected in the input image $I_t(x,y)$ and let $f_k(X_i)$ denote the k^{th} feature.

4.2.1. Bounding Box

From the connected component labeled image, the bounding box coordinates of each vehicle region is calculated. From the bounding box coordinates, the height and width information of the vehicle region is estimated. These bounding box coordinates are used to calculate the features of a particular vehicle region. Figure 4.7 shows an example of extracted vehicle regions. Figure 4.7(a) is the original image with bounding box drawn along each vehicle, Figure 4.7(b) shows the extracted vehicle regions using the bounding box information and Figure 4.7(c) shows the binary map associated with the vehicle regions.

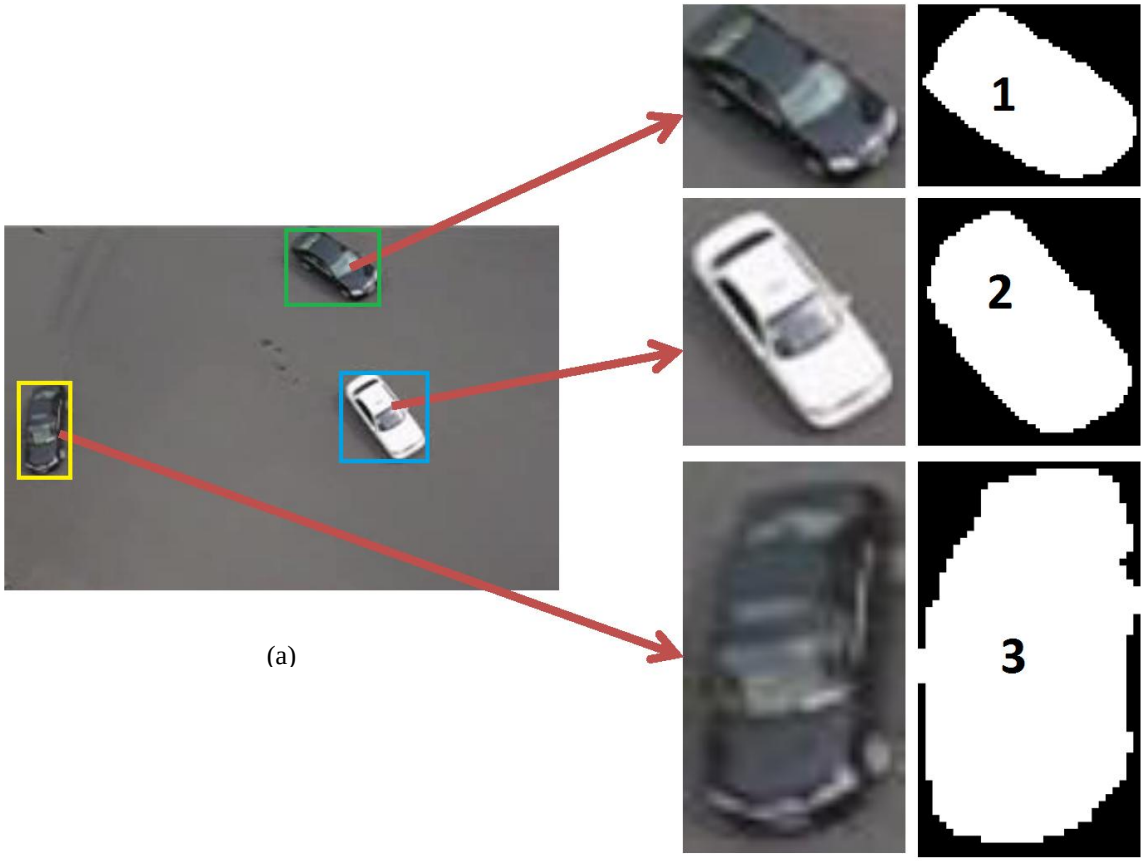


Figure 4.7: Example of extracted vehicle regions (a) Input image with bounding box around each vehicle region (b) Extracted vehicle using the bounding box (c) Labeled vehicle regions

4.2.2. Area

Let $f_1(X_i)$ denote the area of region X_i. Area is defined as the total number of pixels N in the region X_i. The expression for $f_1(X_i)$ is given by

$$f_1(X_i) = N \in X_i \tag{4.3}$$

From Figure 4.8(b), area of a particular vehicle region is given by the number of white pixels in the binary map.

(a) (b)

Figure 4.8: Example showing area of the vehicle regions (a) Vehicle region (b) Binary mask from which area is estimated

4.2.3. Centroid

Let $f_2(X_i)$ denote the area of region X_i. Centroid is defined as the centre of mass of the region X_i. The expression for $f_2(X_i)$ is given by

$$f_2(X_i) = \left(\frac{x_1+x_2+\cdots x_n}{N}, \frac{y_1+y_2+\cdots y_n}{N}\right) \tag{4.4}$$

$$f_2(X_i) = (\bar{x}, \ \bar{y}) \tag{4.5}$$

where x_1, x_2,…. x_n denote the points along the horizontal plane of the image and y_1, y_2,…. y_n denote the points along the vertical plane of the image. Figure 4.9 shows an example of location of centroid of vehicle regions.

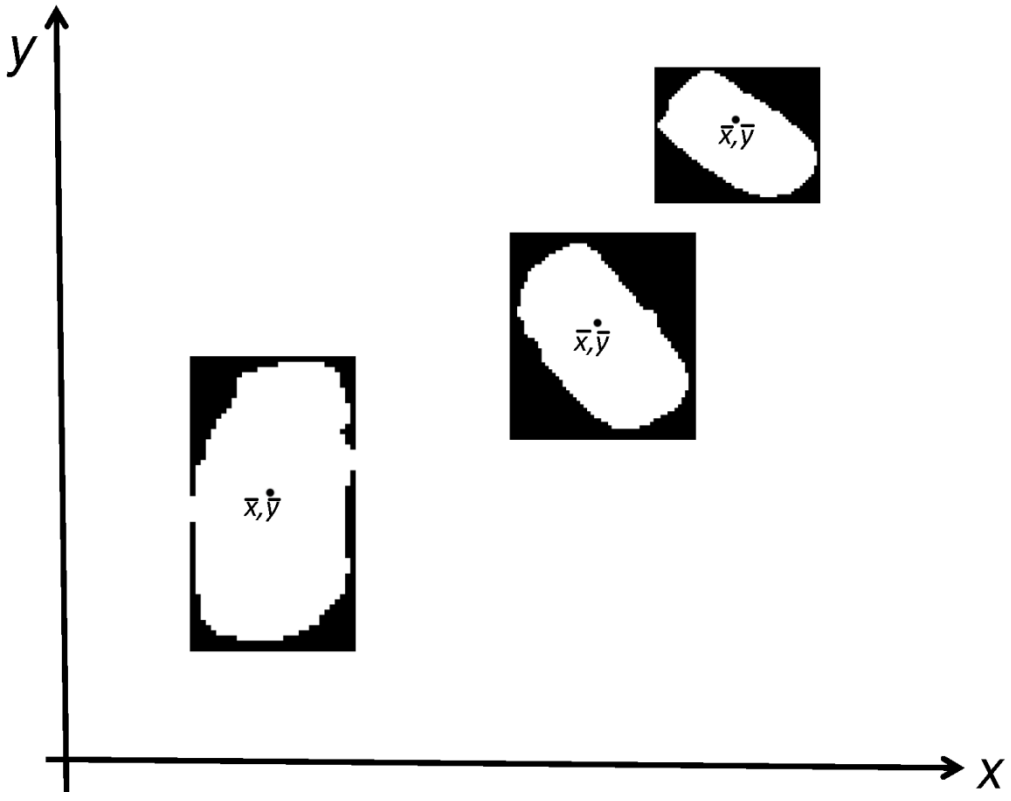
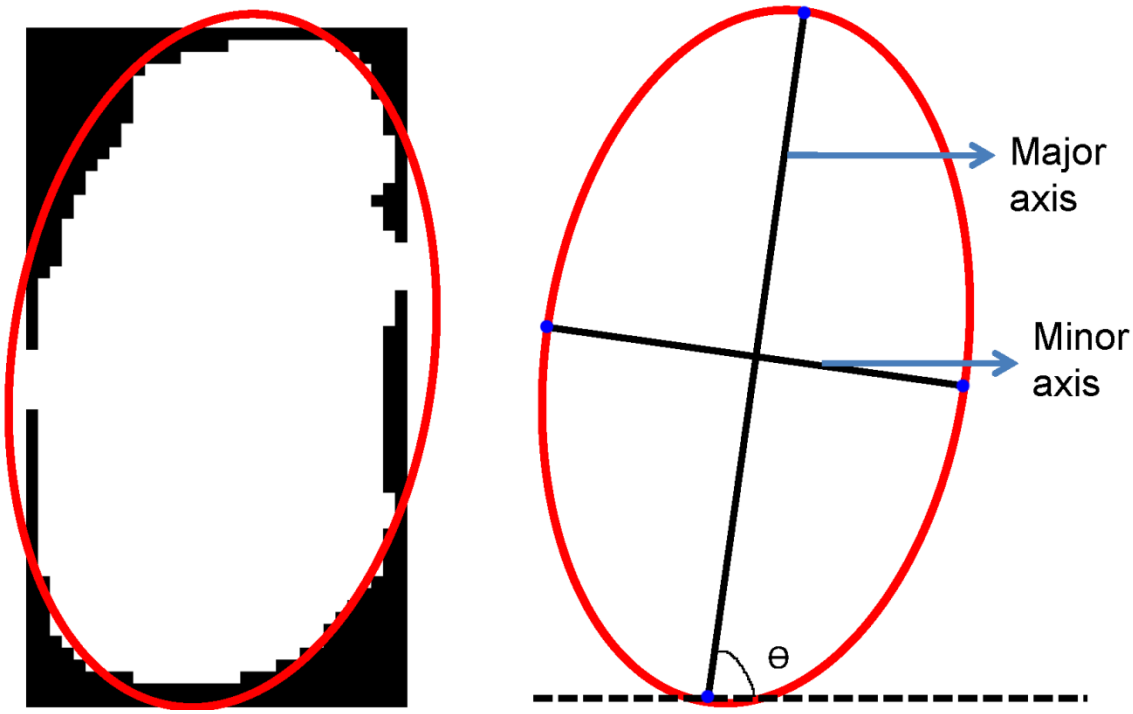


Figure 4.9: Example showing the location of centroid of vehicle regions

4.2.4. Orientation

Let $f_3(X_i)$ denote the area of region X_i. Orientation is determined by the bounding box of each vehicle region. Orientation is defined as the angle in degrees between the *x* axis and major axis of the ellipse that has the same second moments as region X_i. Orientation ranges from *90 degrees* to *-90 degrees*. Figure 4.10 illustrates the orientation of a vehicle region. Figure 4.10(a) shows a vehicle region and its corresponding ellipse. Figure 4.10(b) shows the same ellipse, with features indicated graphically. In Figure 4.10(b) the solid black lines are the axes. The orientation is given by the angle between the horizontal dotted line and the major axis of the ellipse.



(a) (b)

Figure 4.10: Example showing the orientation of the vehicle region (a) Vehicle region and its corresponding ellipse (b) Graphical representation of the ellipse

The expression for $f_3(X_i)$ is given by

$$f_3(X_i) = \begin{cases} \frac{1}{2}cot^{-1}\left(\frac{a-c}{b}\right), & b \neq 0 \; and \; a < c \\ \frac{\pi}{2} + \frac{1}{2}cot^{-1}\left(\frac{a-c}{b}\right), & b \neq 0 \; and \; a > c \end{cases} \tag{4.6}$$

where a, b is the semi-length of the of the major axis and minor axis of the ellipse, respectively and $c = \sqrt{a^2 - b^2}$.

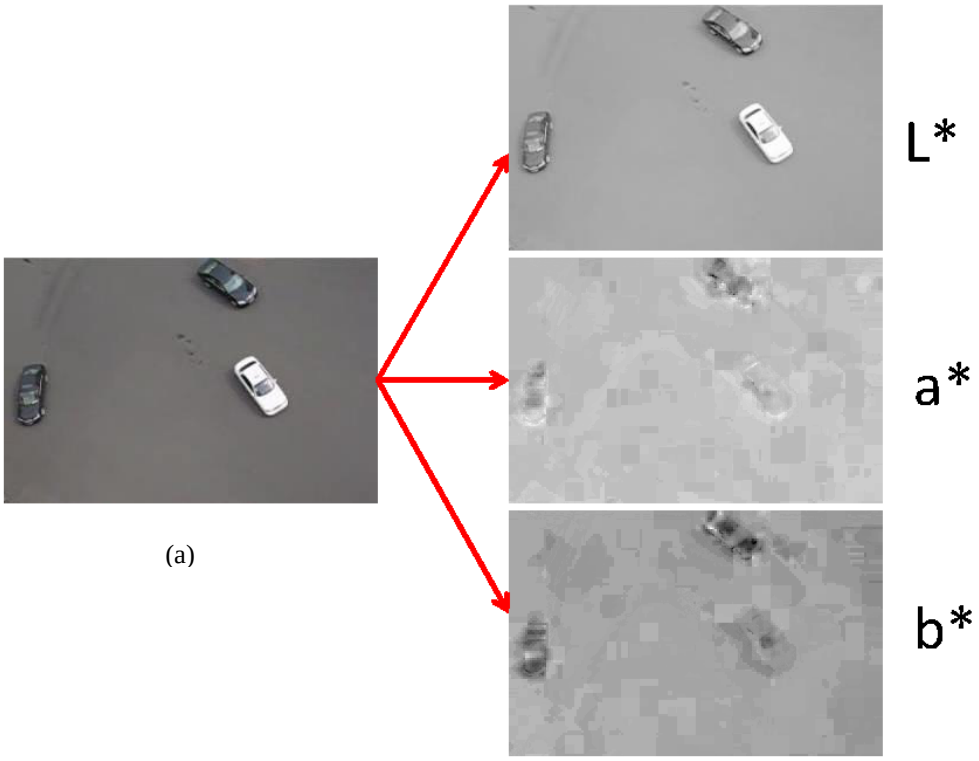
4.2.5. Luminance and Color

Let $f_4(X_i)$ and $f_5(X_i)$ denote the average luminance and average color in the region X_i. These two features are given by

$$f_4(X_i) = \bar{L^*}(X_i) \tag{4.7}$$

$$f_5(X_i) = \left(\overline{a^*}(X_i), \overline{b^*}(X_i)\right) \tag{4.8}$$

where $\bar{L^*}, \overline{a^*}, \overline{b^*}$ denote the average L^*, a^*, b^* measured in the CIE 1976 (L^*, a^*, b^*) color space (CIELAB). The value of L^* ranges between 0 and 100, while the values of a^* and b^* ranges between negative to positive values. Figure 4.11 shows the RGB image converted to L^* a^* b^* color space. Figure 4.11(a) shows the RGB image and Figure 4.11(b) shows different scales of L^* a^* b^* color space.



(a)

(b)

Figure 4.11: RGB to L* a* b* conversion (a) RGB image (b) Different scales of L* a* b* color space

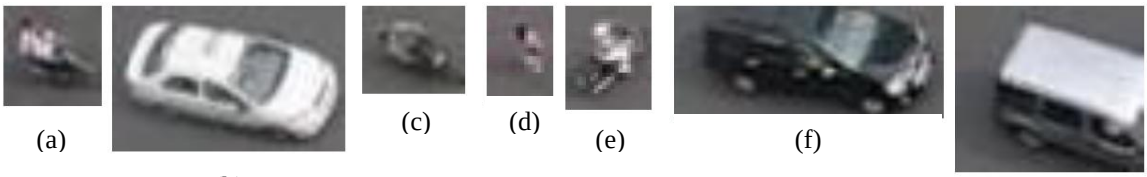
4.3. Feature Vector

All the above features discussed earlier are grouped together in a feature vector of a particular region X_i. The feature vector is given as $\boldsymbol{f}_t(X_i)$

$$\boldsymbol{f}_t(X_i) = [f_1(X_i), f_2(X_i), f_3(X_i), f_4(X_i), f_5(X_i)] \tag{4.9}$$

Examples of regions extracted from a frame are shown in Figure 4.13, Figure 4.15 and Figure 4.17 and their measured features are presented in Table 4.1, Table 4.2 and Table 4.3.

Figure 4.12: Example frame at time *t*



(a) (b) (c) (d) (e) (f) (g)

Figure 4.13: Regions extracted from Figure 4.12

Table 4.1: Features extracted from vehicles in Figure 4.13.

Features		**(a)**	**(b)**	**(c)**	**(d)**	**(e)**	**(f)**	**(g)**
Area(*pixels*)		237	1208	217	56	220	1455	1382
Centroid(*pixel*)	$\bar{x}$	44	89	84	200	195	190	299
	$\bar{y}$	25	78	178	171	73	15	67
Orientation(*degrees*)		-22.5	-10.5	-15.7	57	44	-5.1	-13
Luminance		64	83.1	61	67.7	69.8	56.5	69.2
Color	$\overline{a^*}$	1.06	-0.1	-0.21	3.72	1.3	-0.4	0.4
	$\overline{b^*}$	-0.78	-0.4	1.26	-0.83	-0.15	-2.1	-0.2

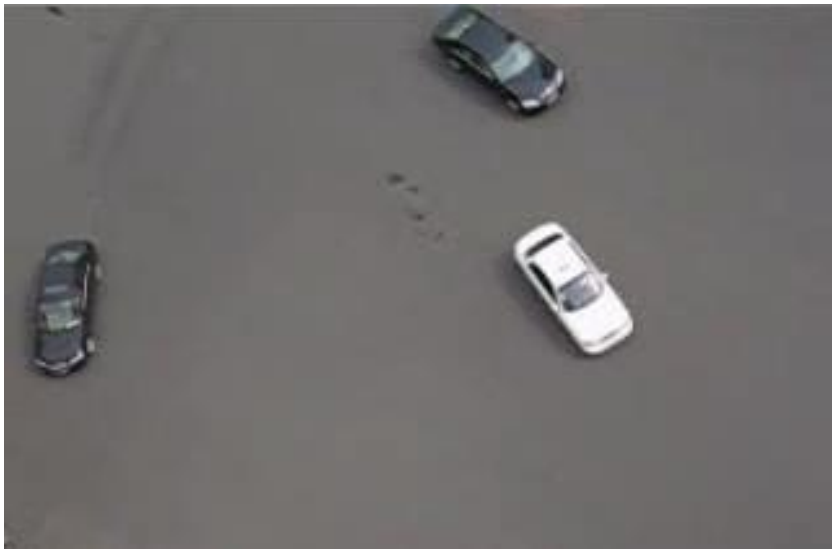
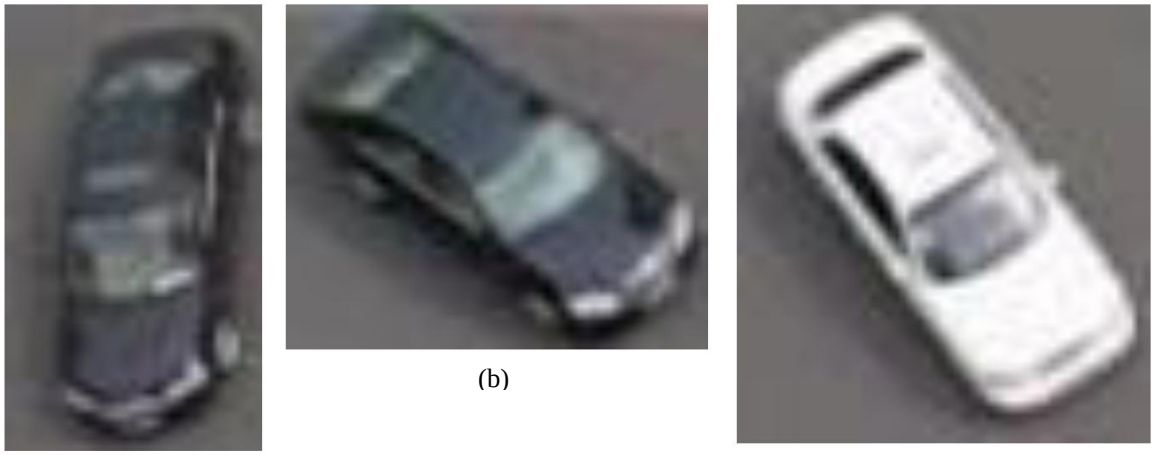


Figure 4.14: Example frame at time *t*



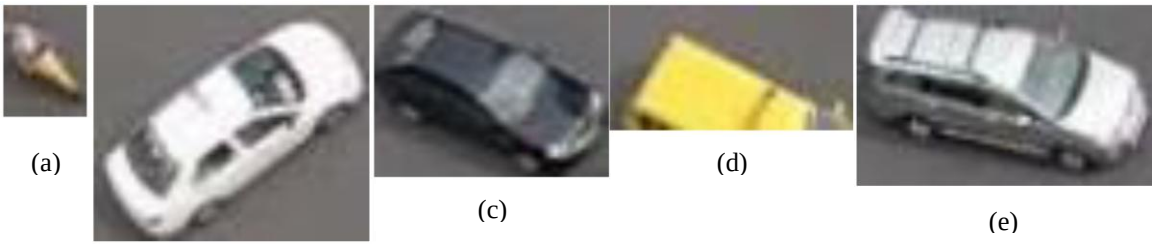
(a) (b) (c)

Figure 4.15: Regions extracted from Figure 4.14

Table 4.2: Features extracted from vehicles in Figure 4.15.

Features		**(a)**	**(b)**	**(c)**
Area(*pixels*)		1393	1472	1583
Centroid(*pixel*)	$\bar{x}$	24	190	218
	$\bar{y}$	120	24	112
Orientation(*degrees*)		81.7	-30.6	-51.7
Luminance		61.2	60.9	83.1
Color	$\overline{a^*}$	0.4	-0.8	0.8
	$\overline{b^*}$	-1.5	-2.5	-0.7

Figure 4.16: Example frame at time *t*



(a) (b) (c) (d) (e)

Figure 4.17: Regions extracted from Figure 4.16

Table 4.3: Features extracted from vehicles in Figure 4.17.

Features		**(a)**	**(b)**	**(c)**	**(d)**	**(e)**
Area(*pixels*)		180	1448	1124	511	1713
Centroid(*pixel*)	$\bar{x}$	39	105	174	185	235
	$\bar{y}$	159	175	33	202	312
Orientation(*degrees*)		-37.7	36.4	-21.8	-9.6	-10.2
Luminance		69.2	85.9	57.7	92.0	75.8
Color	$\overline{a^*}$	1.45	0.6	0.2	-6.83	0.0
	$\overline{b^*}$	6.6	0.0	-2.5	33.8	-0.6

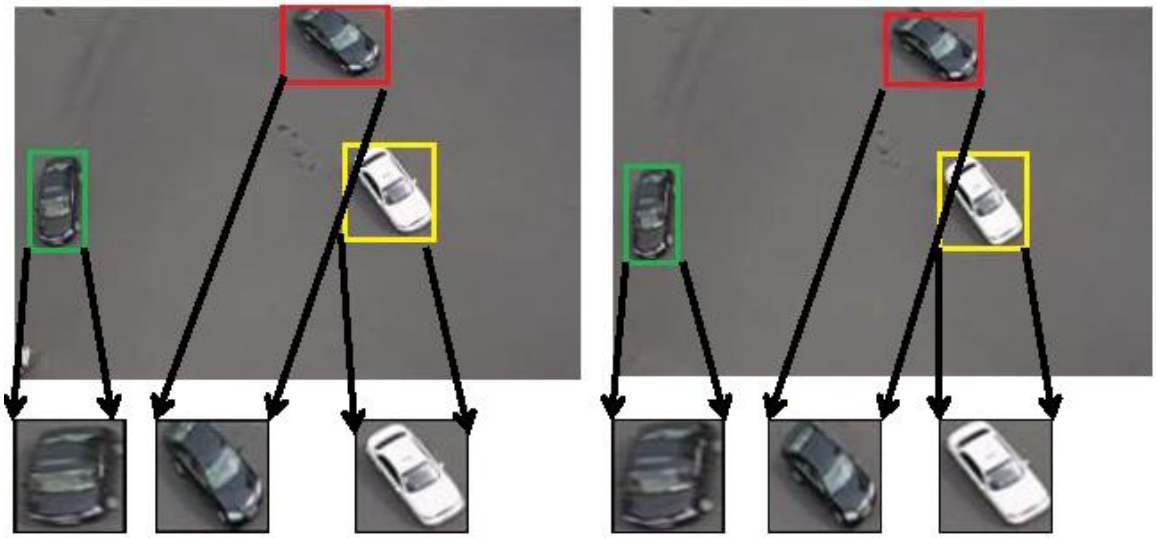
CHAPTER 5

VEHICLE TRACKING AND ACCIDENT DETECTION SYSTEM

5.1. Human Visual System (HVS) Model Analysis

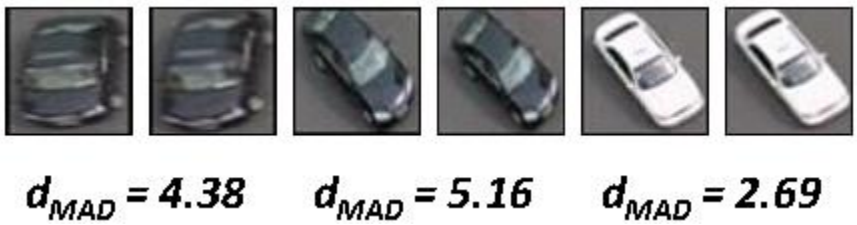
The features described in Chapter 4, section 4.2 can assist in tracking vehicles across multiple frames. However these frames do not explicitly take into account the overall visual appearance of each vehicle as gauged by the human eye. To model this aspect, we employ a visual quality estimator, called MAD (Most Apparent Distortion), recently developed by the authors [113]. Given two images (or image regions), MAD will return an index which is proportional to how dissimilar the two images appear to a human observer. MAD operates by using a combination of visual detection model and a visual appearance model. The detection-based employs models of the human contrast sensitivity function, luminance masking and contrast masking to gauge subtle (near-threshold differences). The appearance-based model employs a log-Gabor transform and local comparisons of log-Gabor coefficient statistics in an attempt to model the visual appearance of clearly visible differences. The MAD index is computed via a weighted geometric mean of these two model outputs.

Here we use MAD to assist in tracking by searching for the vehicle in the next frame that mostly closely matches (i.e., yields the lowest MAD index) for a given vehicle in the current frame. Specifically, after the regions in the frames I_t and I_{t+1} are detected, they are subjected to MAD analysis. In this step each of the regions X_i {i= *1, 2, 3*...} frame I_t are compared one-by-one with each of the regions X_j{j= *1, 2, 3*...} in frame I_{t+1}. Thus the regions in I_t forms the first input to MAD algorithm and the regions in frame I_{t+1} form the second input to MAD algorithm. The



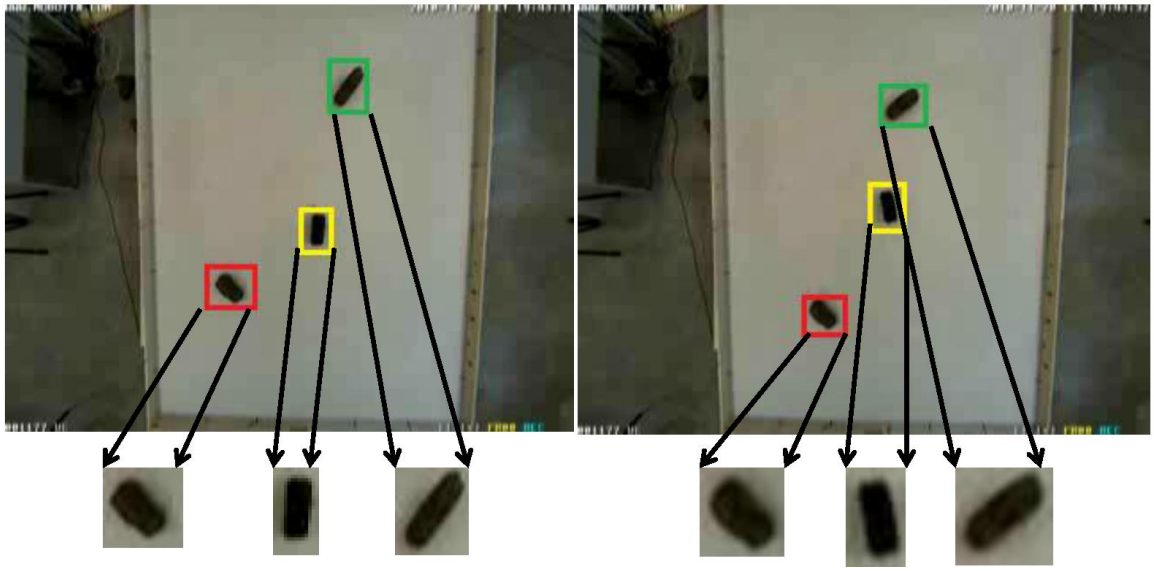
(a) (b)

(c)

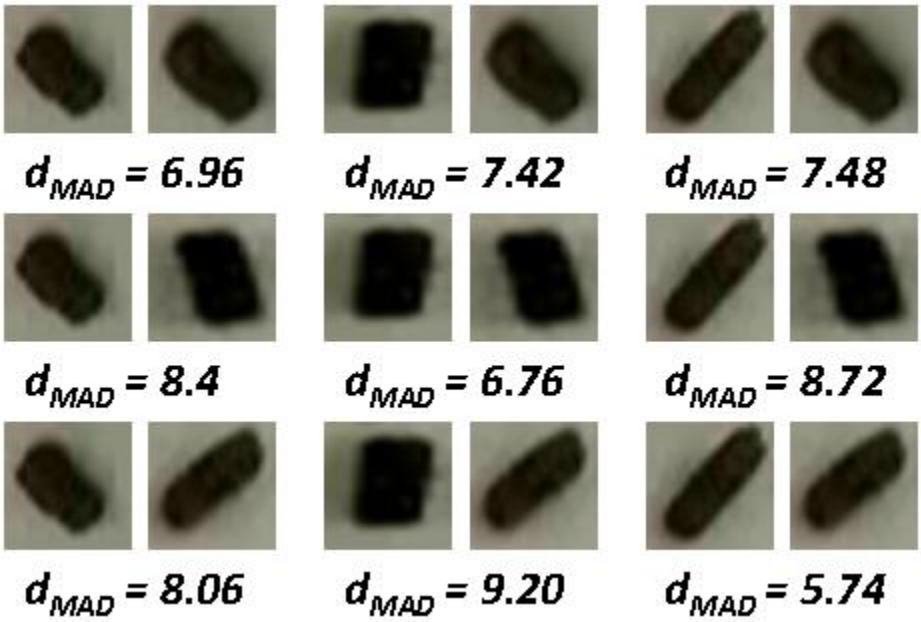


(d)

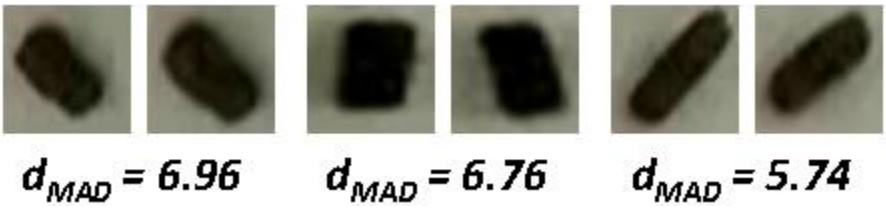
Figure 5.1: Example of HVS model analysis (a) Frame at time *t* (b) Frame at time *t+1* (c) MAD index for different vehicle comparisons (d) MAD index for matching vehicles



(a) (b)



(c)



(d)

Figure 5.2: Example of HVS model analysis (a) Frame at time *t* (b) Frame at time *t+1* (c) MAD index for different vehicle comparisons (d) MAD index for matching vehicles

output for each comparison is denoted by $d_{MAD}(X_i, X_j)$.If a particular region in frame I_{t+1} matches with a region in frame I_t, MAD should yield an index close to zero, meaning that the vehicles detected in frame I_{t+1} and I_t are the same. An example of the MAD analysis is shown in the Figure 5.1 and Figure 5.2. The regions are resized to a common size (of atleast 64x64 pixels) as required by MAD. A lower MAD index denotes a closer visual match between the vehicle regions. From Figure 5.2 it was found out that MAD gave good performance results even when some details of the region such as color, luminance, and texture were not well presented.

5.2. Vehicle Tracking

The tracking is done via corresponding via: (1) Searching for the region in frame I_{t+1} whose features most closely match the features of the given region in frame I_t and (2) searching for the region in frame I_{t+1}, with the lowest MAD index as compared to the given region in frame I_t.

5.2.1. Feature Distance

In this step the feature vector of the regions extracted from frames I_t and I_{t+1} are used. For the purpose of tracking the vehicles accurately across each frame, the Euclidean distance between the feature vector of each region X_i {i= *1, 2, 3*...} in I_t and the feature vector of each region X_j {j= *1, 2, 3*...} in I_{t+1}. Let $\boldsymbol{f}_t(X_i)$ denote the feature vector of the i^{th}_region extracted from frame I_t and $\boldsymbol{f}_{t+1}(X_i)$ denote the feature vector of the j^{th}_region extracted from frame I_{t+1}. Let $d_{features}(X_i, X_j)$ denote the distance between $\boldsymbol{f}_t(X_i)$ and $\boldsymbol{f}_{t+1}(X_i)$ and is given by:

$$d_{features}\left(X_i, X_j\right) = \sqrt{\left(\boldsymbol{f}_t(X_i,) - \boldsymbol{f}_{t+1}\left(X_j\right)\right)^2} \tag{5.1}$$

5.2.2. Weighed Combination of Feature Distance and MAD analysis

Equation (5.1) provides one measure of similarity between vehicle regions in frame I_t and I_{t+1}. This similarity measure can be improved by combining distance index $d_{features}(X_i, X_j)$ and MAD

index $d_{MAD}(X_i, X_j)$ to compute an overall similarity measure between the vehicle regions. For this purpose $d_{features}(X_i, X_j)$ and $d_{MAD}(X_i, X_j)$ are combined using weights. The overall similarity measure $d(X_i, X_j)$ is given by:

$$d(X_i, X_j) = \alpha d_{MAD}(X_i, X_j) + (1 - \alpha) d_{features}(X_i, X_j) \tag{5.2}$$

where α is the weighting factor. The value of α was empirically chosen to be 0.9. Finally the closest matching between vehicle regions in frame I_t and frame I_{t+1} is determined by searching for the minimum value resulting from the weighted combination output $d(X_i, X_j)$ and is given by

$$j^* = \arg min_j \left(d(X_i, X_j) \right) \tag{5.3}$$

Tracking is done typically between two consecutive frames I_t and I_{t+1}. At first instance of the system, vehicle regions in two consecutive frames are detected and their corresponding features are extracted simultaneously. In next step using the vehicle feature vector, feature distance vector $d_{features}$ is computed between vehicle regions in frames I_t and I_{t+1}. Also similarity index d_{MAD} is computed between vehicle regions in frames I_t and I_{t+1} and the tracking is done using a weighted combination of $d_{features}$ and d_{MAD} as explained by the equation (5.2) and equation (5.3).

Since the tracking is done between two consecutive frames at a time, for rest of the instances of the system, the information of the vehicles in frame I_{t+1} are carried over to the next tracking step between frames I_{t+1} and I_{t+2}, and there is no necessity to extract the vehicle information in frame I_{t+1}, since this information is already computed in the previous step, thus saving time and computation. In the following steps the vehicle information in the upcoming frames (I_{t+2}, I_{t+3}, I_{t+4}…) are extracted and are compared with the vehicle information obtained in the previous frames (I_{t+1}, I_{t+2} ...) to track the vehicle as explained earlier. For the purpose of tracking, when the vehicle information computed in the previous frames (e.g., I_{t+1}) are carried

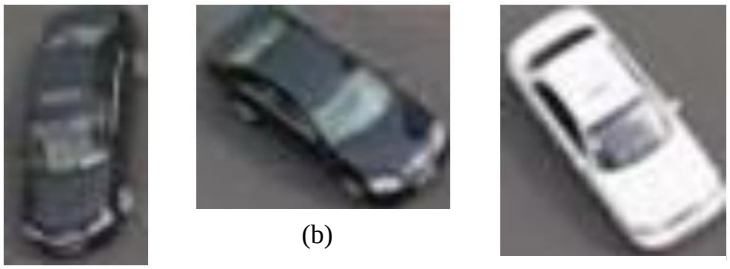
over in the next step, the vehicles in I_{t+1} are labeled in the order they occurred in the frame I_t and then compared with the vehicle information in the frame I_{t+2}, thus the system will be able to know which vehicle it is tracking. This process is repeated for the upcoming frames.

If a match for particular vehicle region in frame I_t cannot be found in the frame I_{t+1}, the vehicle region is assumed to be left out of the scene and the vehicle is no more tracked and its information are not carried over to the next step. Similarly if a new vehicle region is detected in the frame I_{t+1}, the features of the vehicle are extracted and used for tracking in the upcoming frames. These procedures are explained in detail as follows.

While finding the minimum index from the overall similarity measure $d(X_i, X_j)$ to find the matching vehicles between frames I_t and I_{t+1}, suppose a vehicle found in frame I_t has left the scene in frame I_{t+1}, the system still tries to find a closely matched vehicle in frame I_{t+1} corresponding to a vehicle region in frame I_t, which is not a true match of the vehicle in I_t since it has left the scene in I_{t+1} but unknown to the system. Inorder to overcome this situation, when the system finds a false match to a vehicle in I_t, the area and centroid position of the vehicle in search belonging to I_t is compared to the falsely matched vehicle in I_{t+1}. Since they are false matches the Euclidean distance between the area and centroid of these vehicles should be larger than some predefined threshold value. If this condition is satisfied, the system is informed that there was a false match and made to know that the vehicle in search has left the scene. There may be situations where the Euclidean distance may be lesser than the threshold value. In these cases other features such as the orientation and color information of the vehicles were used in addition to area and centroid. Since the comparison between the vehicles are done between consecutive frames, the area and centroid information of the vehicles were effective in finding the false matches in most of the cases. Similarly when a new vehicle region is detected in I_{t+1} not previously detected in I_t, the scenario of false matches cannot be encountered since the system searches for close matches only for the vehicles belonging to I_t in I_{t+1}.

Illustration of vehicle tracking is explained as follows:

Figure 5.3: Example frame at time *t*



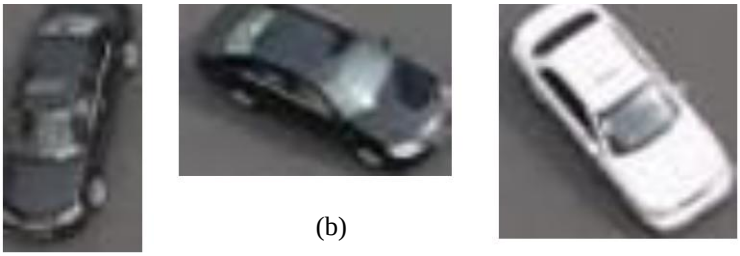
(a) (b) (c)

Figure 5.4: Regions extracted from Figure 5.3

Table 5.1: Features extracted from vehicles in Figure 5.4.

Features		**(a)**	**(b)**	**(c)**
Area(*pixels*)		1301	1271	1502
Centroid(*pixel*)	$\bar{x}$	24	189	219
	$\bar{y}$	119	24	112
Orientation(*degrees*)		82.2	-32.6	-53.1
Luminance		60.8	60.0	84.1
Color	$\overline{a^*}$	0.4	-0.9	0.8
	$\overline{b^*}$	-1.6	-0.9	-0.6

Figure 5.5: Example frame at time *t+1*



(a) (b) (c)

Figure 5.6: Regions extracted from Figure 5.5

Table 5.2: Features extracted from vehicles in Figure 5.5.

Features		**(a)**	**(b)**	**(c)**
Area(*pixels*)		1304	1372	1511
Centroid(*pixel*)	$\bar{x}$	23	193	221
	$\bar{y}$	122	26	114
Orientation(*degrees*)		81.3	-31.6	-52.6
Luminance		60.7	60.2	84.1
Color	$\overline{a^*}$	-0.5	-1.3	0.6
	$\overline{b^*}$	-0.8	-1.7	-0.5

Table 5.3: $d_{features}$ computed between vehicle regions in I_t and I_{t+1}

$d_{features}$		Vehicles in frame I_{t+1}		
		(a)	**(b)**	**(c)**
Vehicles in frame I_t	**(a)**	2.04	308.78	440.23
	(b)	325.91	51.23	228.68
	(c)	444.21	286	7.11

Table 5.4: d_{MAD} computed between vehicle regions in I_t and I_{t+1}

d_{MAD}		Vehicles in frame I_{t+1}		
		(a)	**(b)**	**(c)**
Vehicles in frame I_t	**(a)**	4.38	9.43	12.12
	(b)	9.59	5.16	10.45
	(c)	12.07	10.39	2.69

Table 5.5: Overall similarity measure d computed between vehicle regions in I_t and I_{t+1}

d with α=0.9		Vehicles in frame I_{t+1}		
		(a)	**(b)**	**(c)**
Vehicles in frame I_t	**(a)**	4.14	39.36	54.93
	(b)	41.22	9.77	32.27
	(c)	55.28	37.95	3.13

From the above Table 5.3, the matching vehicle regions between I_t and I_{t+1} are determined based on the equation (5.3) as shown in the Figure 5.7. Example of tracked vehicles is shown in the Figure 5.8. Color rectangle box is drawn around each vehicle and from the Figure 5.8 it can be seen that the color of the box around each vehicle across the frames remains the same indication that the vehicles are detected and tracked correctly.

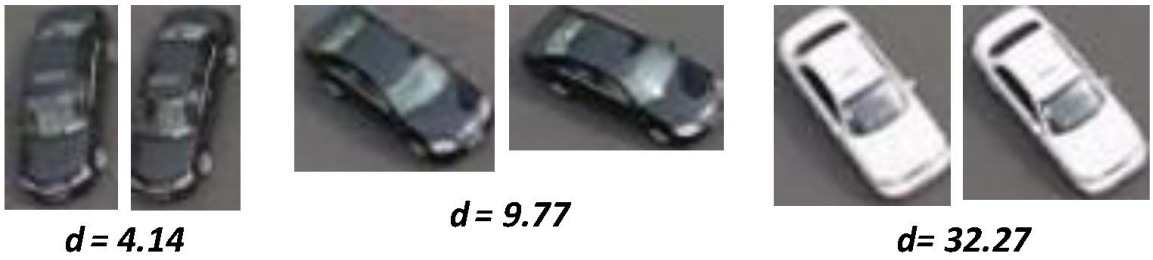


Figure 5.7: Overall similarity measure *d* for matched vehicles in I_t and I_{t+1}

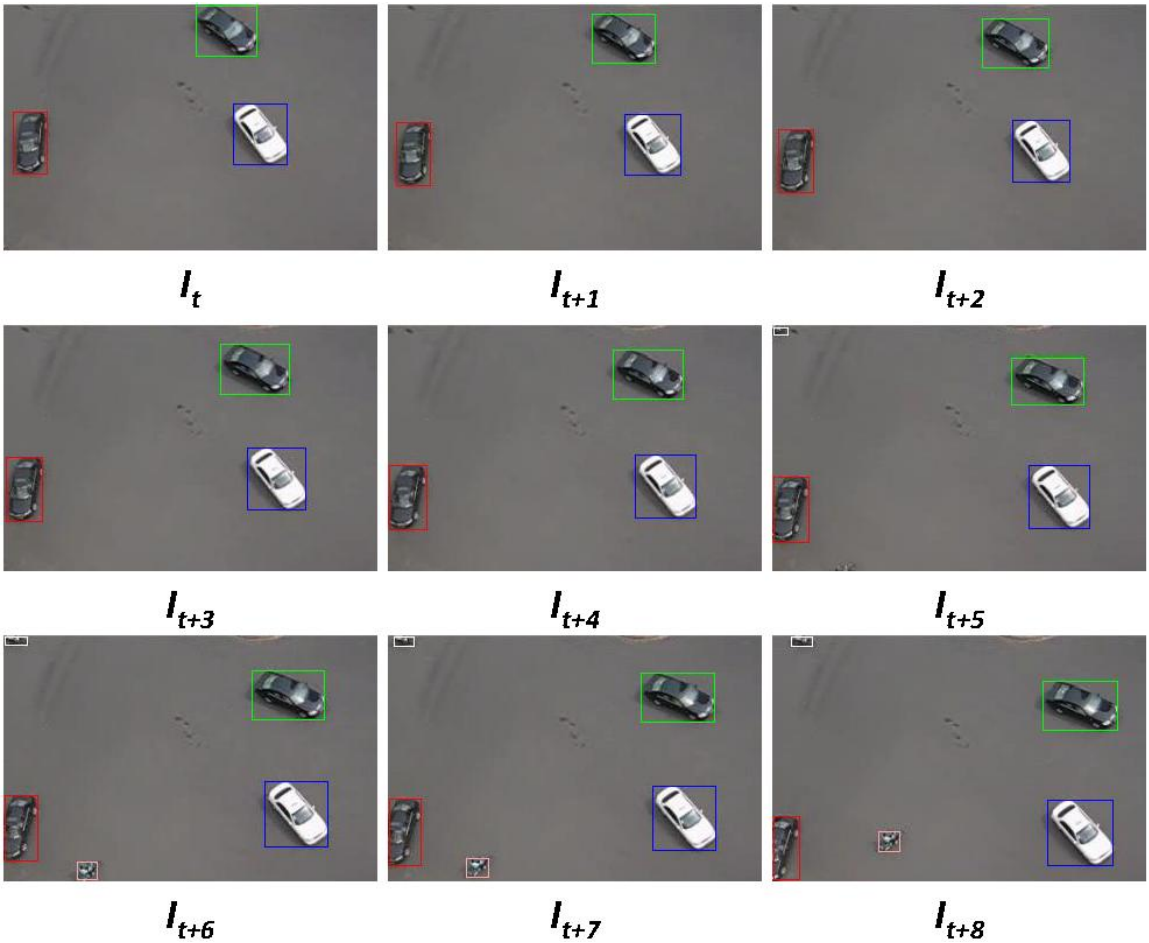


Figure 5.8: Example showing tracked vehicles across different frames

5.3. Computation of Vehicle Parameters

Once the vehicles are detected and tracked correctly, vehicle parameters such as speed and trajectory are computed using the features extracted from the tracked vehicles.

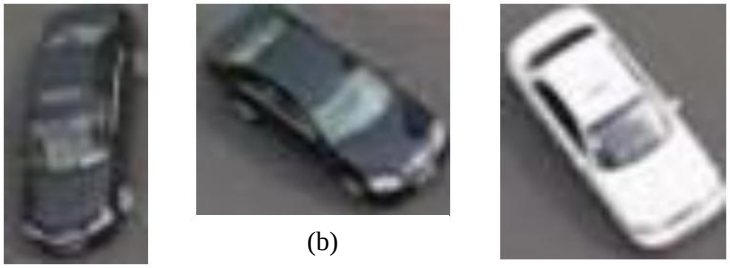
5.3.1. Speed of the Vehicles

Speed of a particular vehicle region X_i in frame I_t is computed using the distance travelled by the vehicle in frame I_{t+1} and the frame rate of the video from which the image sequence are extracted. The distance travelled by the vehicle is computed using the centroid position $(\bar{x}, \bar{y})$ of the vehicle in I_t and I_{t+1}. Let X_i denote a particular vehicle detected in I_t and X_j denote the same vehicle detected in I_{t+1}, assuming the correspondence between the vehicles is determined using the vehicle tracking step. Speed of a particular vehicle region X_i is given by:

$$Speed(X_i) = \frac{\sqrt{\left(\bar{x}(X_i) - \bar{x}(X_j)\right)^2 + \left(\bar{y}(X_i) - \bar{y}(X_j)\right)^2}}{\frac{1}{frame\ \ rate}} \tag{5.4}$$

The above equation (5.4) gives the speed of the vehicle in terms of *pixels/sec*. Inorder to determine the speed of the vehicle in terms of real-world unites (*miles/hr*), camera calibration process is used as explained in Chapter 3, Section 3.4. The relation between the image coordinates and real-world coordinates is given by the equation (3.11). Using the equation (3.11) the centroid position of a particular vehicle in I_t and I_{t+1} is converted from pixel coordinates to real-world coordinates. From this step the speed of the vehicle in terms of (*miles/hr*) is determined. Similar process is repeated for all the vehicle regions detected and tracked. An illustration of determination of vehicle speed is given below:

Figure 5.9: Example frame at time *t*



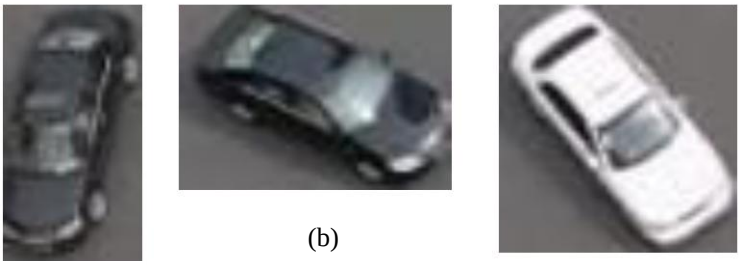
(a) (b) (c)

Figure 5.10: Regions extracted from Figure 5.9

Table 5.6: Centroid of vehicles detected in Figure 5.10

Centroid(pixels)	**(a)**	**(b)**	**(c)**
$\bar{x}$	24	189	219
$\bar{y}$	119	24	112

Figure 5.11: Example frame at time *t+1*



(a) (b) (c)

Figure 5.12: Regions extracted from Figure 5.11

Table 5.7: Centroid of vehicles detected in Figure 5.12

Centroid(pixels)	**(a)**	**(b)**	**(c)**
$\bar{x}$	23	193	221
$\bar{y}$	122	26	114

$$Speed\ of\ vehicle\ (a) = \frac{3.16\ pixels}{1/15} = 47.4\ pixels/second$$

$$Speed\ of\ vehicle\ (b) = \frac{4.48\ pixels}{1/15} = 67.2\ pixels/second$$

$$Speed\ of\ vehicle\ (c) = \frac{2.83\ pixels}{1/15} = 42.45\ pixels/second$$

Since for the *Saigon* video the camera is assumed to be parallel to the ground plane, the scale factor as explained by equation (3.2) was obtained by comparing the known vehicle width and height with the pixel width and height of the vehicle as found in the image sequence. The scale factor that relates pixel distance to real-world distance was approximately found to be ($1pixel \simeq$ 10cm). Therefore scale factor = 0.00001(in terms of *km* units). Therefore the speed of the vehicle is in terms of *miles/hr* is given by:

$$Speed\ of\ vehicle\ (a) = 11miles/hr$$

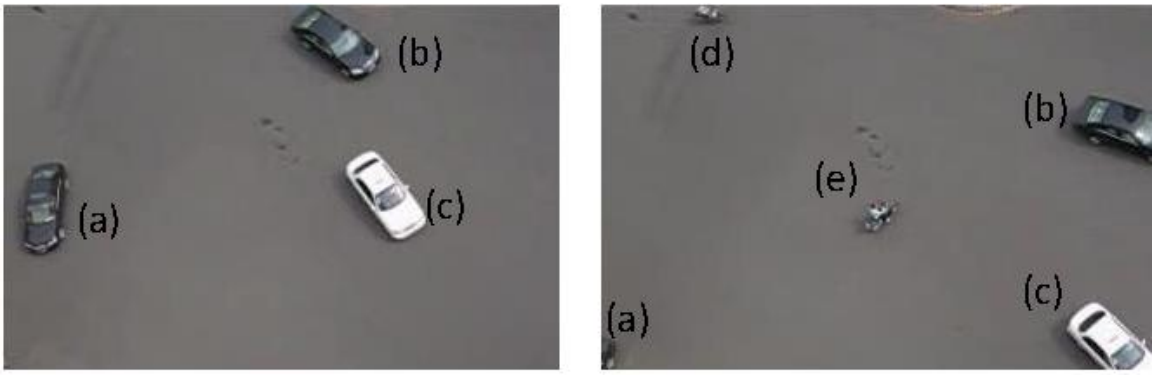
$$Speed\ of\ vehicle\ (b) = 15miles/hr$$

$$Speed\ of\ vehicle\ (c) = 10miles/hr$$

These speeds are typically the speed of vehicles expected at traffic intersection. Therefore the assumption of the camera placed parallel to the ground plane to capture the video holds good in this case.

5.3.2. Trajectory of the Vehicles

Similar to finding the speed of the vehicles, trajectory of the vehicles are also estimated using the centroid information of the vehicles in frame I_t and I_{t+1}. A line fit is made connecting the centroid of a particular vehicle detected and tracked correctly in I_t and I_{t+1} from the instant the vehicle enters the scene until it leaves the scene. Figure 5.14 and 5.16, illustrates the vehicle trajectory obtained for frame I_t to I_{t+n}.



(a) (b)

Figure 5.13: Example frames for showing the vehicle trajectory (a) Frame at time *t* (b) Frame at time *t+n*

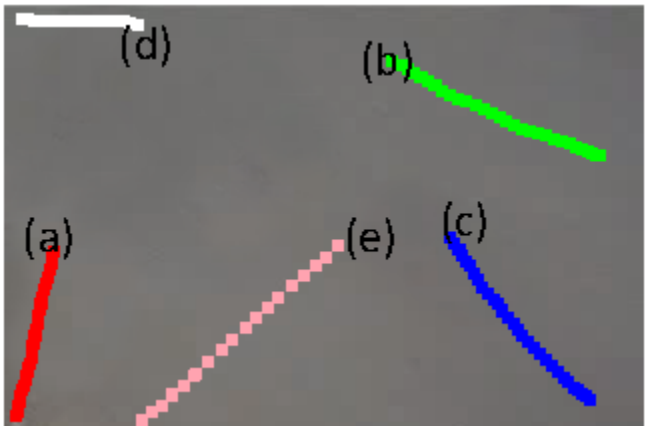
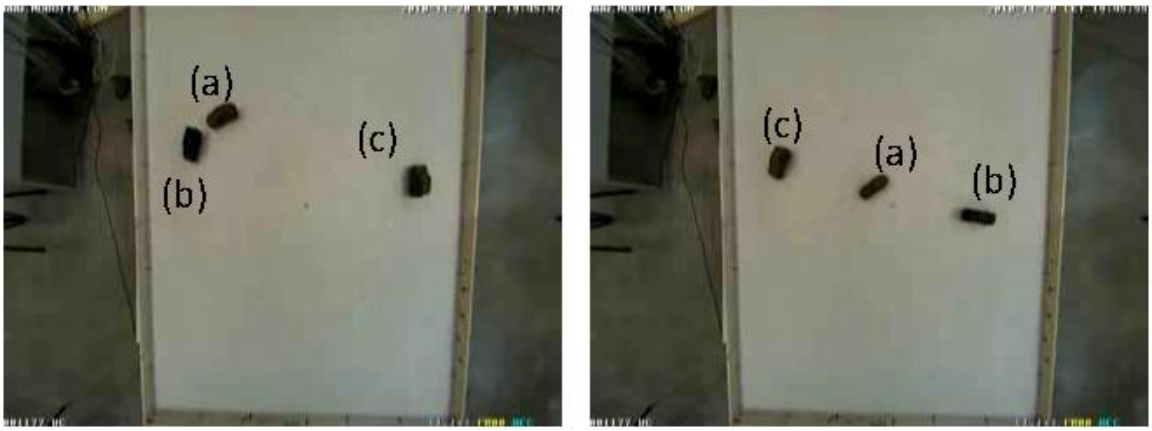


Figure 5.14: Trajectory of the vehicles tracked from frame I_t to I_{t+n}



(a) (b)

Figure 5.15: Example frames for showing the vehicle trajectory (a) Frame at time *t* (b) Frame at time *t+n*

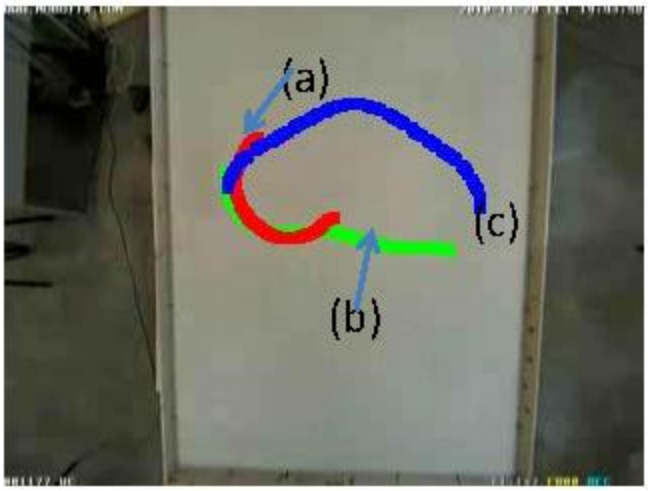


Figure 5.16: Trajectory of the vehicles tracked from frame I_t to I_{t+n}

5.4. Accident Detection System

Once the vehicles are detected and tracked correctly in frames I_t and I_{t+1}, the next step in the process is to determine the occurrence of accident using vehicle parameters such as the speed, trajectory and the features extracted from individual vehicles detected in I_t and I_{t+1}. In this thesis the work done by Ki and Lee [106] was adapted for the implementation of accident detection system. To determine the occurrence of accident at traffic intersection the variation in speed, area, orientation and position of the vehicles tracked are used. The accident detection process is explained as follows.

5.4.1. Variation in Speed of the Vehicles

Speed of the vehicles is an important factor while determining the occurrence of crashes at traffic intersection. Rapid change in the speed of the vehicles is a useful descriptor for a traffic accident. For example if a particular vehicle travels with a velocity *v*, after an occurrence of an accident there is rapid change in the velocity of the vehicle. Therefore variation in velocity of the vehicles across frames is used as a factor for judging the occurrence of crashes by the system. In the accident detection system vehicles are detected and tracked correctly and their velocity

information is extracted at each frame the vehicle occurs. After successful tracking of a vehicle in two consecutive frames I_t and I_{t+1}, the velocity information of the tracked vehicle obtained from I_t and I_{t+1} is compared with that obtained from I_{t-1} and I_t. Since it is assumed that the vehicles moves at approximately constant velocity, if a vehicle crashes with another vehicle in frame I_{t+1}, the velocity of the vehicles is expected to go down drastically. So when the velocity of the vehicle determined in I_{t+1} is compared with that obtained in I_t, there should be a larger difference in the velocity of the vehicle indicating that a crash has occurred. To determine the occurrence of accident the difference in velocity of vehicles obtained between two consecutive frames compared with that of a predefined threshold. $Change\ in\ velocity = \Delta v = v\ at\ I_{t+1} - v\ at\ I_t$. The following expression is used for the traffic accident detection algorithm:

$$VI = \begin{cases} 1, & if\ \Delta v \geq a \\ 0, & otherwise \end{cases} \tag{5.4}$$

where *VI* is the velocity index and a is the speed threshold. Figure 5.17 shows a scenario for accident detection.

5.4.2. Variation in Area of the Vehicles

Rapid change in the area of the vehicles detected and tracked can be used as a descriptor to detect accidents. When an accident occurs, two vehicles come into contact and there is possibility that the bounding box of the vehicles may intersect and in this case there is a rapid change in the area of the vehicles detected. To detect accidents the area of the vehicles detected and tracked in I_t and I_{t+1} are compared and if the change in area of the vehicles exceeds a area threshold, then there may be possibility of accident. $Change\ in\ area = \Delta area = area\ at\ I_{t+1} - area\ at\ I_t$.The following expression is used as a factor for traffic accident detection:

$$DI = \begin{cases} 1, & if\ \Delta area \geq b \\ 0, & otherwise \end{cases} \tag{5.5}$$

where *DI* is the area index and *b* is the speed threshold. Figure 5.18 shows a scenario for accident detection using change in area factor.

5.4.3. Variation in Position of the Vehicles

Change in centroid position of the vehicles in frames I_t and I_{t+1} can be used as a factor to determine the occurrence of accident. Just like the change in area of the vehicles, when an accident occurs the bounding of two vehicles intersect causing a change in overall position of the vehicles. Therefore change in centroid of the vehicles in consecutive frames can be used as a descriptor to determine the occurrence of an accident. Change in centroid is given by:

$$\Delta\bar{x} = \bar{x}\ at\ I_{t+1} - \ \bar{x}\ at\ I_t \tag{5.6}$$

$$\Delta\bar{y} = \bar{y}\ at\ I_{t+1} - \ \bar{y}\ at\ I_t \tag{5.7}$$

The following expression is used as a factor for traffic accident detection:

$$PI = \begin{cases} 1, if\ \Delta\bar{x}\ \geq c, if\ \Delta\bar{y}\ \geq d \\ 0, otherwise \end{cases} \tag{5.8}$$

where *PI* is the position index and *c*,*d* are thresholds. Figure 5.19 shows an illustration of determining accidents using centroid information.

5.4.4. Variation in Orientation of the Vehicles

Variation in orientation of the vehicles can be used as a factor to determine the occurrence of an accident. As in the case of speed, area and centroid of the vehicles, the orientation of the vehicle in frames I_t and I_{t+1} are compared. If there is a significant change in the orientation of the vehicles detected and tracked between two consecutive frames, then a possibility of accident is determined. $Change\ in\ orientation = \ \Delta\theta = \theta\ at\ I_{t+1} - \ \theta\ at\ I_t$. The orientation index *OI* is given by:

$$OI = \begin{cases} 1, \ if \ \Delta\theta \ \geq e \\ 0, otherwise \end{cases} \tag{5.9}$$

where e is the threshold for change in orientation of the vehicles. . Figure 5.19 shows an illustration of determining accidents using change in orientation.

5.4.5. Overall Accident Index

After computing the velocity, area, position and orientation index of the vehicles, the overall accident index is determine by the sum of individual indexes. The overall accident index is then compared with the accident threshold to determine the occurrence of accident. If the accident index exceeds a particular threshold, then an occurrence of accident is signaled, otherwise the system determines that there is no accident and the process is repeated until an accident is detected. The overall Accident Index (AI) is given by:

$$AI = VI + DI + PI + OI \tag{5.10}$$

The occurrence of accident is determined by:

$$Signal\ Accident = \begin{cases} 1, \ if \ AI \geq accident\ threshold \\ 0, \qquad\qquad\qquad otherwise \end{cases} \tag{5.11}$$

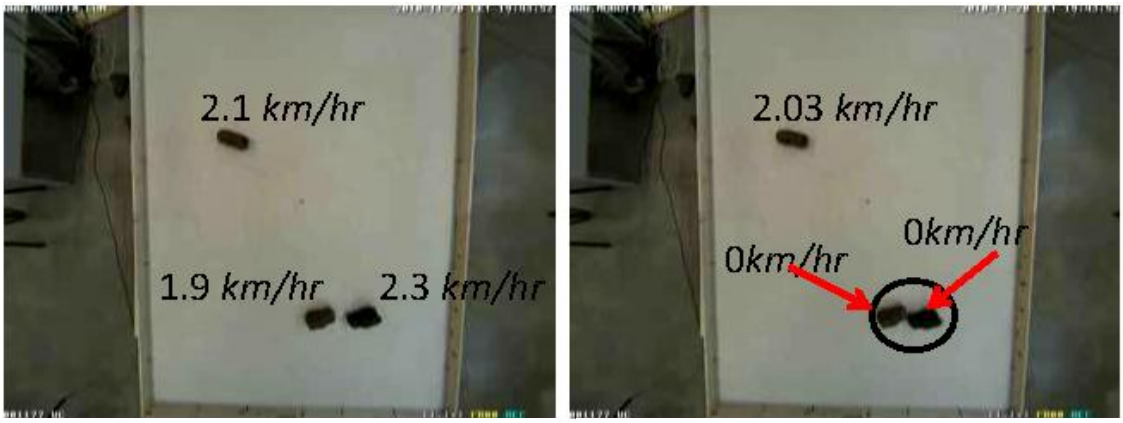
An outline of the accident detection process is shown in the Figure 5.21 and the accident detection algorithm is summarized as follows:

1. Vehicle regions are detected in image frames.
2. Low-level features such as area, orientation, centroid, luminance and color of the detected vehicles are extracted.
3. Vehicles are tracked using the tracking algorithm.
4. Speeds of the tracked vehicles are calculated.
5. Velocity, Area, Position and Orientation Indexes are calculated.

6. Overall Accident Index is calculated using the sum of individual indexes and occurrence of accident is identified.

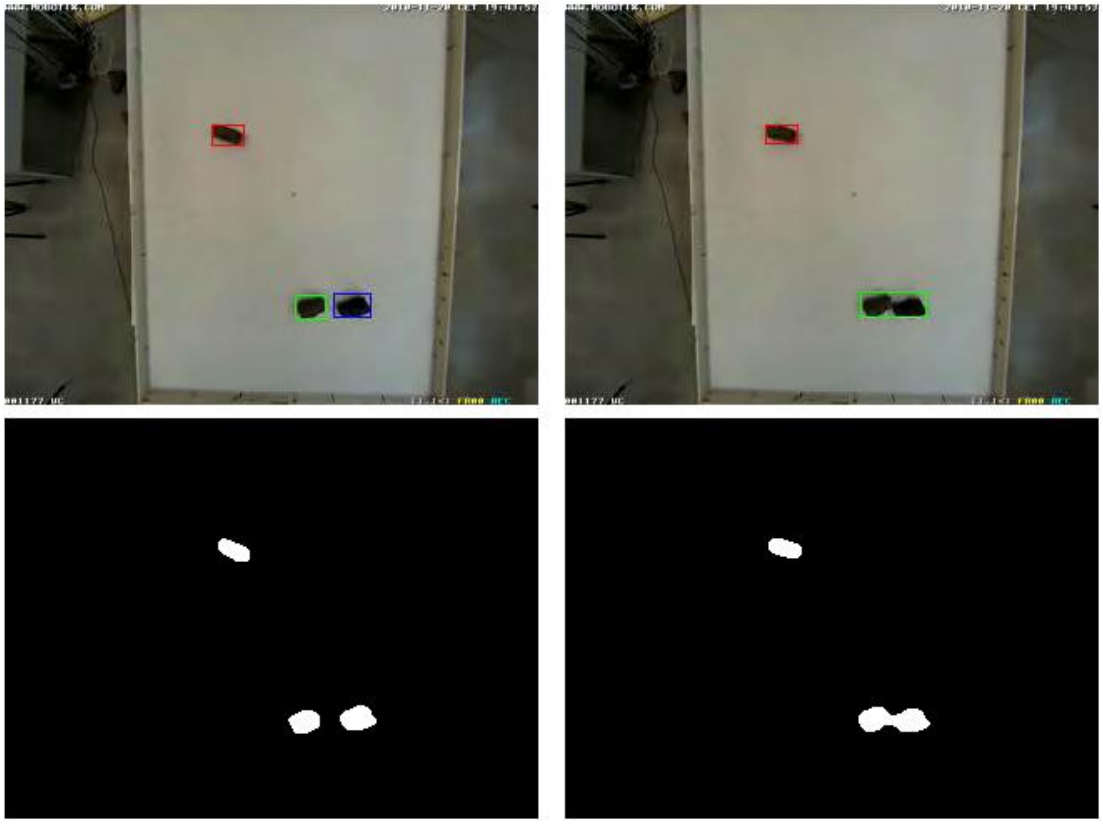
5.4.6. Locating the point of accident

Once an occurrence of accident is determined by the system, the next step is to locate the point where the accident has occurred. This information can be obtained using the position of the vehicles that were involved in the accident at a particular frame. By using this information the user is informed about the occurrence of accident along with the point where the accident has occurred. This information is useful because when accidents has been detected and suppose this information has to be transmitted to an information center or analyzed for future purposes, having the crash clip recorded along with the point of occurrence of accident reduces the redundant image frames need to be transmitted and also the end user can detect the occurrence of accident without having to analyze the video being transmitted. This results in considerable saving of analyzing time. Figure 5.22 shows a frame with an accident detected and the point where the accident is located.



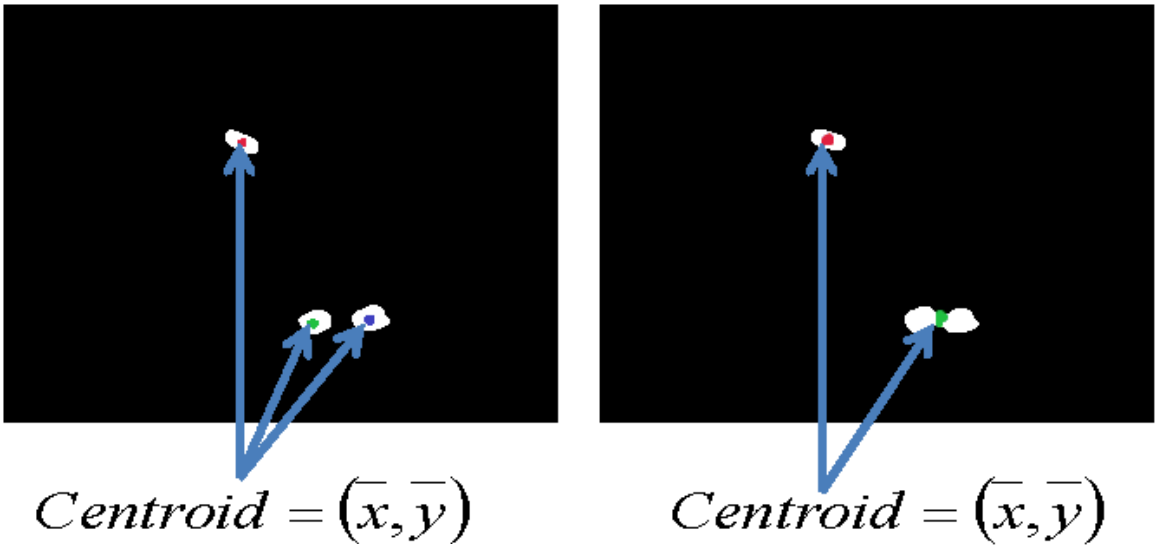
(a) (b)

Figure 5.17: Example of accident scenario (a) Frame before the occurrence of an accident (b) Frame after occurrence of an accident



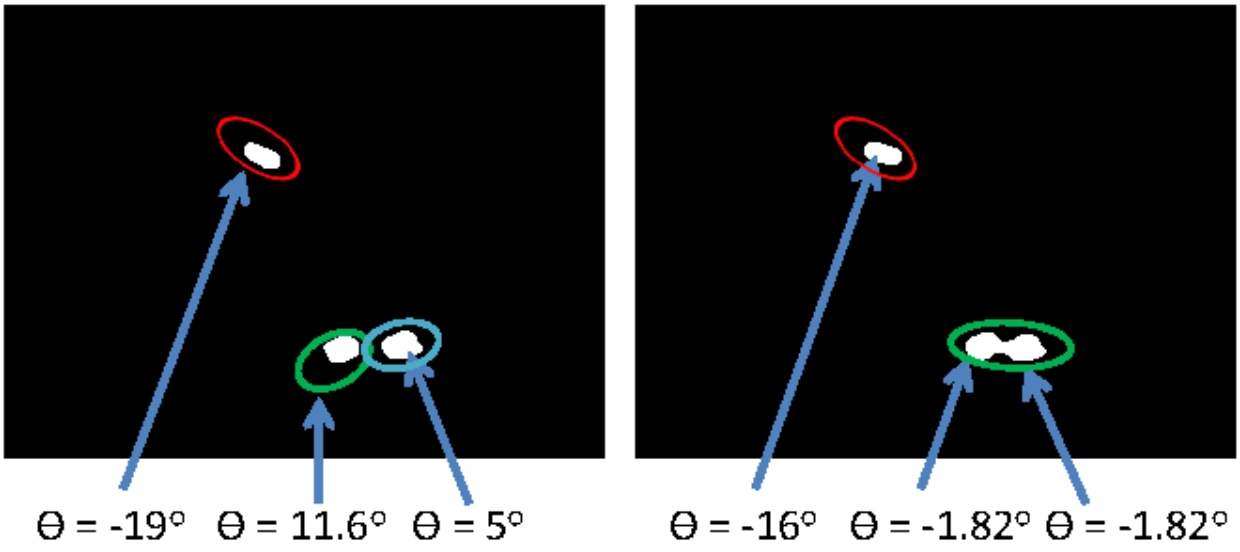
(a) (b)

Figure 5.18: Illustration of accident detection using change in area (a) Frame and its corresponding binary image before occurrence of an accident (b) Frame and its corresponding binary image after occurrence of an accident



(a) (b)

Figure 5.19: Illustration of accident detection using change in centroid (a) Binary image showing centroid position of vehicles before accident (b) Binary image showing centroid position of vehicles after accident



(a) (b)

Figure 5.20: Illustration of accident detection using change in orientation (a) Binary image showing orientation of vehicles before accident (b) Binary image showing orientation of vehicles after accident

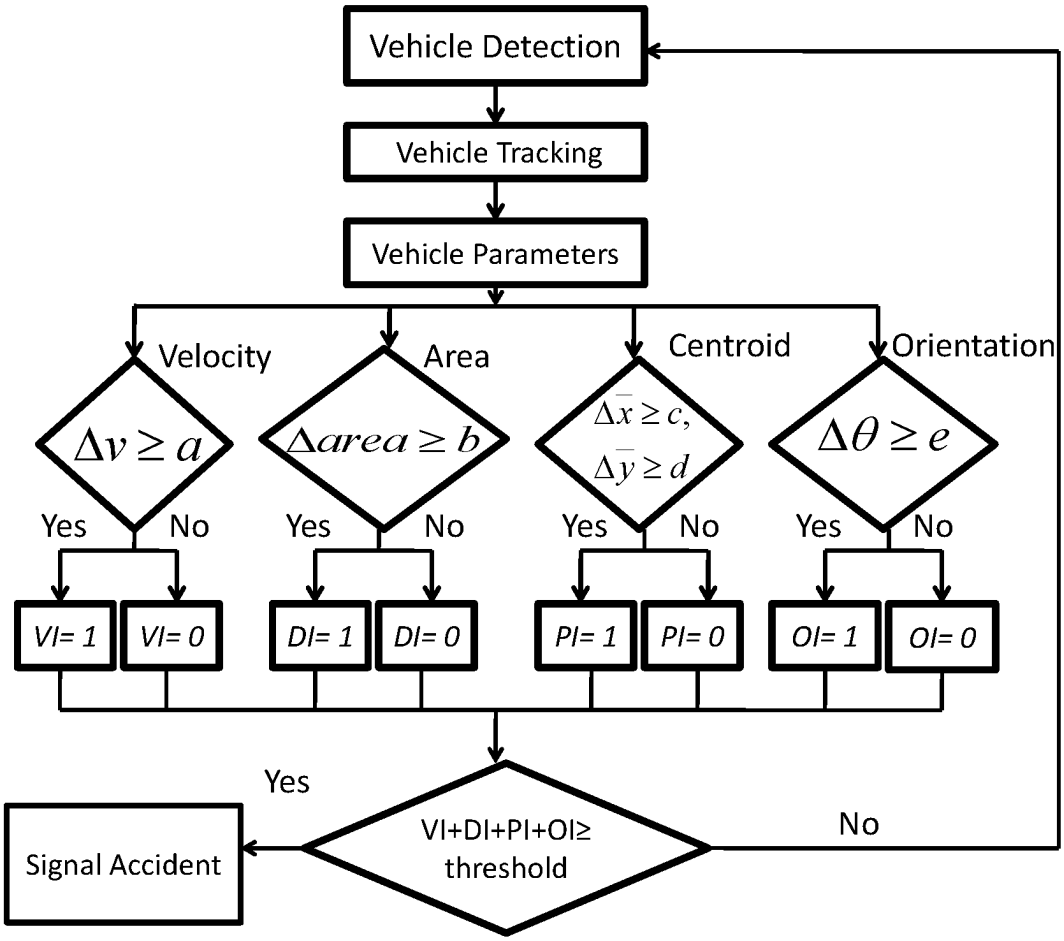


Figure 5.21: Flowchart of the Accident Detection Algorithm [106]

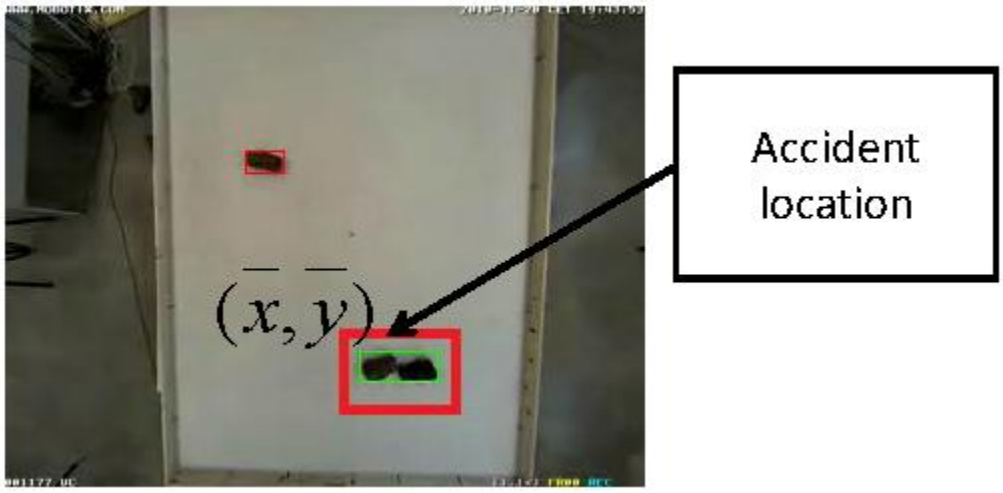


Figure 5.22: Location of occurrence of accident

CHAPTER 6

REAL-TIME IMPLEMENTATION AND RESULTS

6.1. Experimental Results of Detection and Tracking Algorithm

Since the main objective of this thesis is the real-time implementation of accident detection system at traffic intersection, certain aspects of vehicle detection and vehicle tracking are compromised to achieve good real-time performance. Especially, in the vehicle detection step, already pre-processed background frame was used for background subtraction inorder to extract vehicle regions. And also the system does not address the problem of vehicle occlusion and changes in illumination conditions. Also the accident detection algorithm was designed to detect accidents at day-light conditions. Inspite of these limitations, the system was able to perform fairly well at the same time achieve good-real time performance. Initially the algorithm was developed using MATLAB and Image Processing Toolbox, since MATLAB environment is convenient for testing purposes. Using the MATALB results, the algorithm was converted in C++ platform for the real-time implementation of accident detection system.

6.1.1. Vehicle Detection and Tracking Performance of the Algorithm

For the purpose of training and testing the tracking algorithm, two video sequences obtained from traffic intersection at Saigon, Vietnam were used. These videos presented a good representation of busy intersection with different classes of vehicles. The complexity involved with these videos is that the vehicles do not have particular lanes to travel. They travelled at random direction with traffic coming at all directions. So these sequences made the vehicle tracking process bit difficult, since the vehicles cannot be tracked using estimated path as in Kalman filter or search window.

The testing was done offline on the videos named *saigon01.avi* and *saigon02.avi*. These sequences were shot in daylight conditions from a top-down view. The video sequences have resolution of 320x240 *pixels* and a frame rate of 30*Hz*. The lengths of the sequences were 22 and 14 *seconds* respectively containing 684 and 424 *frames* in total. But for the tracking purpose the sequences were resampled at 15*Hz*, since a frame rate of 15*Hz* was able to give enough information about the vehicle speed and trajectory. By this conversion, the algorithm averts processing redundant frames. The choice of binary threshold value *T* for vehicle detection and weighing factor for vehicle tracking are discussed in the following sections.

6.1.1.1. Selection of Binary threshold *T* for Vehicle Detection

For the purpose of selecting the binary threshold *T* for converting the difference image $I_{diff}(x,y)$ to binary image $bw(x,y)$, different threshold values of 0.05,0.1,0.15 and 0.2 were used and the threshold value that yielded best results in terms of vehicle detection were chosen. To determine the best threshold value for test video sequences, a subset of frames were chosen (frame number: 1 to 100) from two test video sequences, *saigon01.avi* and *saigon02*.avi. Number of vehicles detected using different threshold values were compared to the ground truth which was computed manually using a naive human subject for the subset of frames. A total of 200 frames were used from both video sequences to determine the threshold value. Table 6.1 and 6.2 shows the number of vehicles detected for subset of frames chosen from *saigon01.avi* and *saigon02*.avi with different threshold values. From Table 6.1, it is shown that a threshold value of $T = 0.1$, yielded the best results in terms of vehicle detection for *saigon01.avi*. And from Table 6.2 it is shown that a threshold value of $T = 0.15$, yielded the best results in terms of vehicle detection for *saigon02.avi*. Although a threshold value of 0.1 yielded best results for *saigon02.avi*, the numbers of false positives were more when compared to the results produced by a threshold value of $T = 0.1$. Therefore from the results obtained from two videos, the threshold value for test sequences were chosen as 0.1. Figure 6.1 and 6.3 shows example frames used for to determine the threshold

value. Figure 6.2 and 6.4 shows the binary map generated for the example frames using different threshold values. In Figure 6.2 and 6.4, the first row shows the binary maps generated using T = 0.05, second row shows the binary maps generated using T = 0.1, third row shows the binary maps generated using T = 0.15 and the fourth row shows the binary maps generated using T = 0.2.

Table 6.1: Comparison of number of detected vehicles using different *T* values for subset of frames in *saigon01.avi*

Threshold	**0.05**	**0.1**	**0.15**	**0.2**
Ground Truth	752	752	752	752
Algorithm	1165	735	658	805

Table 6.2: Comparison of number of detected vehicles using different *T* values for subset of frames in *saigon02.avi*

Threshold	**0.05**	**0.1**	**0.15**	**0.2**
Ground Truth	806	806	806	806
Algorithm	895	746	759	918

Figure 6.1: Example frames from *saigon01.avi* used to determine threshold *T*

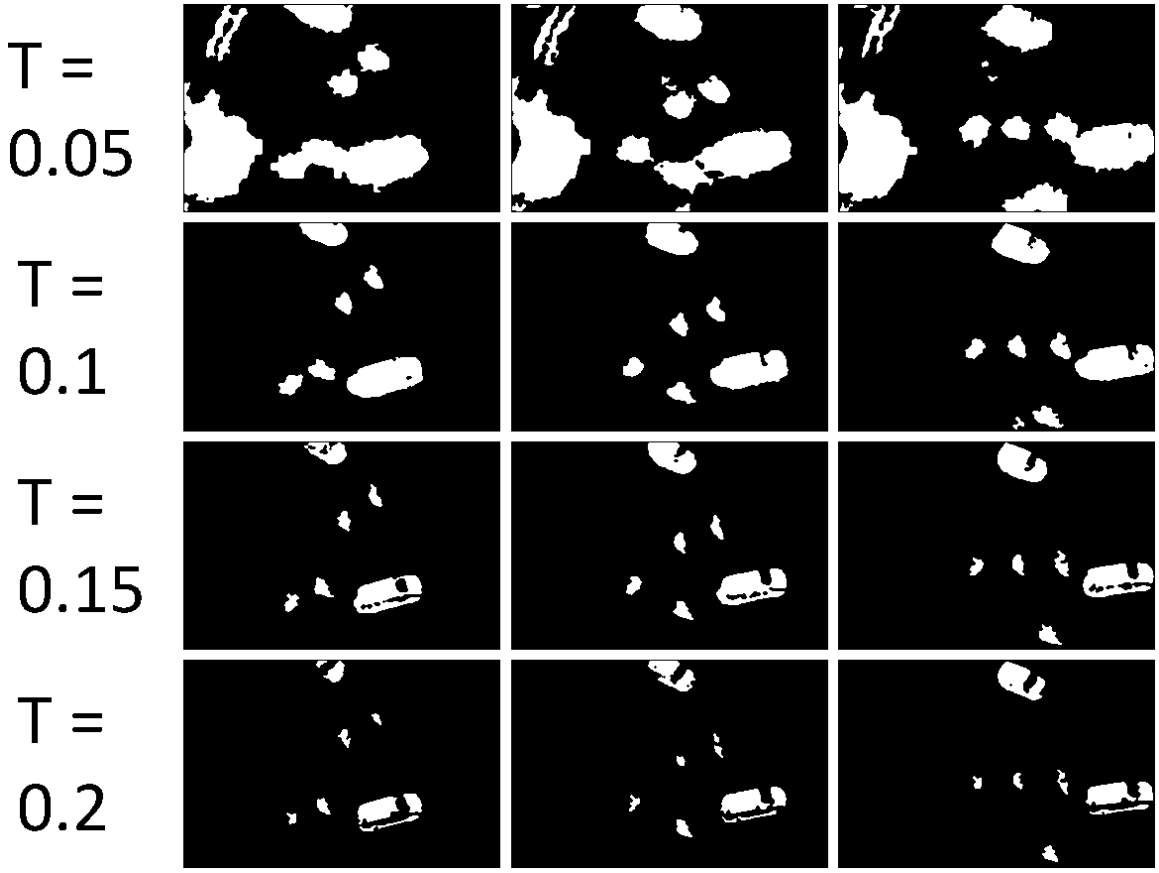


Figure 6.2: Binary maps generated for frames shown in Figure 6.1 using different threshold values

Figure 6.3: Example frames from *saigon02.avi* used to determine threshold *T*

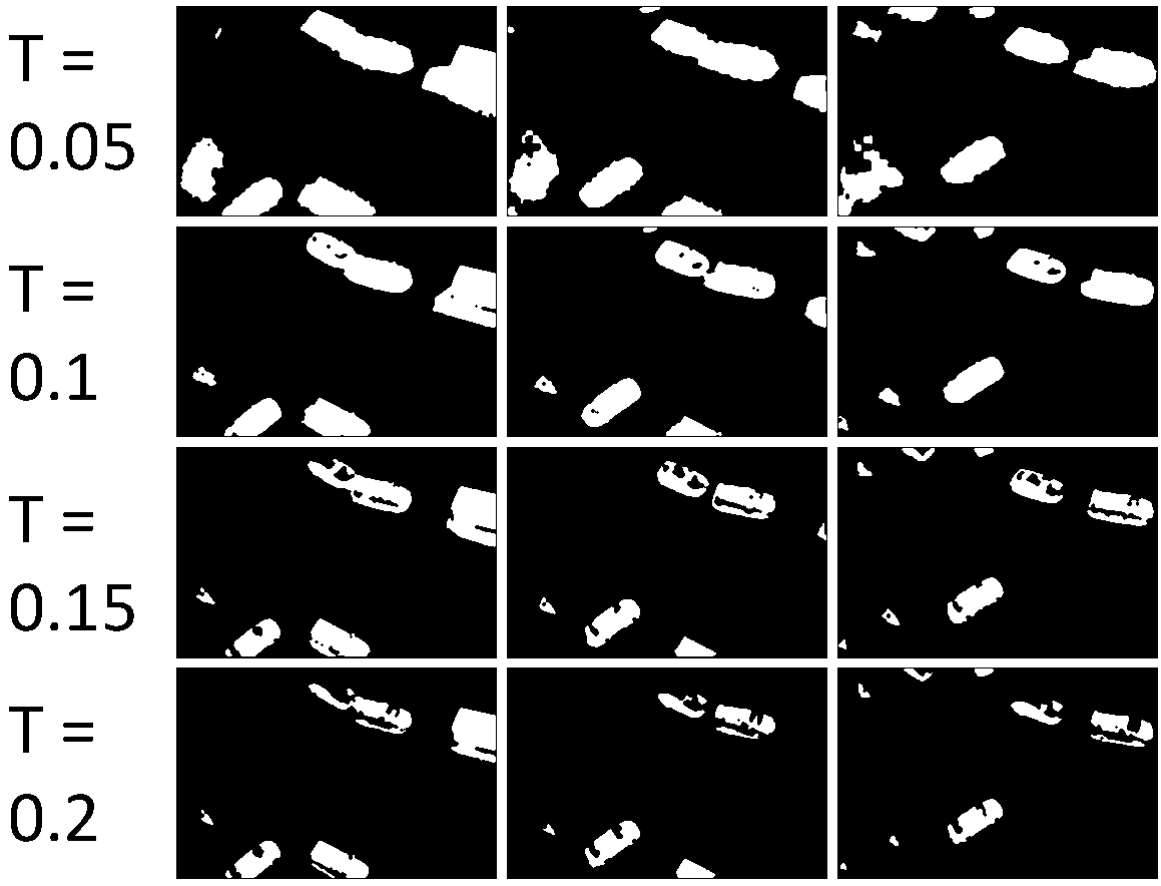


Figure 6.4: Binary maps generated for frames shown in Figure 6.2 using different threshold values

6.1.1.2. Selection of weighing factor α used for Vehicle Tracking

In equation (5.2) α was used as a weighting factor to combine feature distance index $d_{features}$ and MAD index d_{MAD} to obtain the overall similarity measure between vehicle regions detected in frames I_t and I_{t+1}. To determine the best weighing factor α, a subset of frames chosen from two video sequences *saigon01.avi* (Frames 100-200) and *saigon02.avi* (Frames 1-100) were used and α was chosen based on the tracking results obtained. A total of 200 frames were used from both video sequences to determine the weighing factor α. Number of vehicles tracked correctly using different weighing factors were compared to the ground truth computed manually using a naive human subject for the subset of frames. Table 6.3 and 6.4 shows the comparison of number of vehicles detected by the algorithm with the number of correctly tracked vehicles for different weighing factor α for videos *saigon01.avi* and *saigon02.avi* respectively.

Weights chosen for comparison are 0, 1, 0.5, 0.75 and 0.9 where $\alpha = 0$ means the vehicles are tracked purely based on feature distance $d_{features}$ and $\alpha = 1$ means the vehicles are tracked purely based on MAD index d_{MAD}. Although the feature distance $d_{features}$ and MAD index d_{MAD} on their own produced good tracking results, it was found that the performance of tracking can be increased by combining these two factors by giving appropriate weights to them. From Table 6.3 and 6.4 it was found that d_{MAD} on its own is good enough to produce reasonable tracking performance, however when two vehicles of same description (e.g., two cars of same model and color)as shown in Figure 6.5 were detected, MAD index d_{MAD} found difficult to distinguish between them and produce false tracking results. Similarly the feature index $d_{features}$ failed in cases when two vehicles of same area and orientation were found to be close together in terms of their position in consecutive frames as shown in Figure 6.6. Therefore there was a necessity to combine $d_{features}$ and d_{MAD} using suitable weighing factors to compensate for errors in vehicle tracking algorithm. Finally from Table 6.3 and 6.4, it is shown that a weighing factor of $\alpha = 0.9$ yielded best tracking results for both the video sequences.

Table 6.3: Comparison of tracking performance using different weighing factors *α* for subset of frames in *saigon01.avi*

Weighing factor *α*	No. of detected vehicles by algorithm	No. of correctly Tracked vehicles by algorithm
0	12	9
1	12	10
0.5	12	9
0.75	12	11
0.9	12	12

Table 6.4: Comparison of tracking performance using different weighing factors *α* for subset of frames in *saigon02.avi*

Weighing factor *α*	No. of detected vehicles by algorithm	No. of correctly Tracked vehicles by algorithm
0	12	9
1	13	12
0.5	13	12
0.75	13	13
0.9	13	13

The weighing factor α=0.9 suggests that more importance has been given to low-level vision analysis part to help vehicle tracking. From detailed analysis it is found out that weighing factor α acts more like a scaling factor to bring both the feature index $d_{features}$ and MAD index d_{MAD} to equal proportion, since these indexes could not be normalized before combining them. In some cases the value of feature index was found to be high in orders of 100 and the value of MAD index was found to be low in orders of 1. Therefore there is a necessity to scale both the indexes relatively to a common order before combining them.

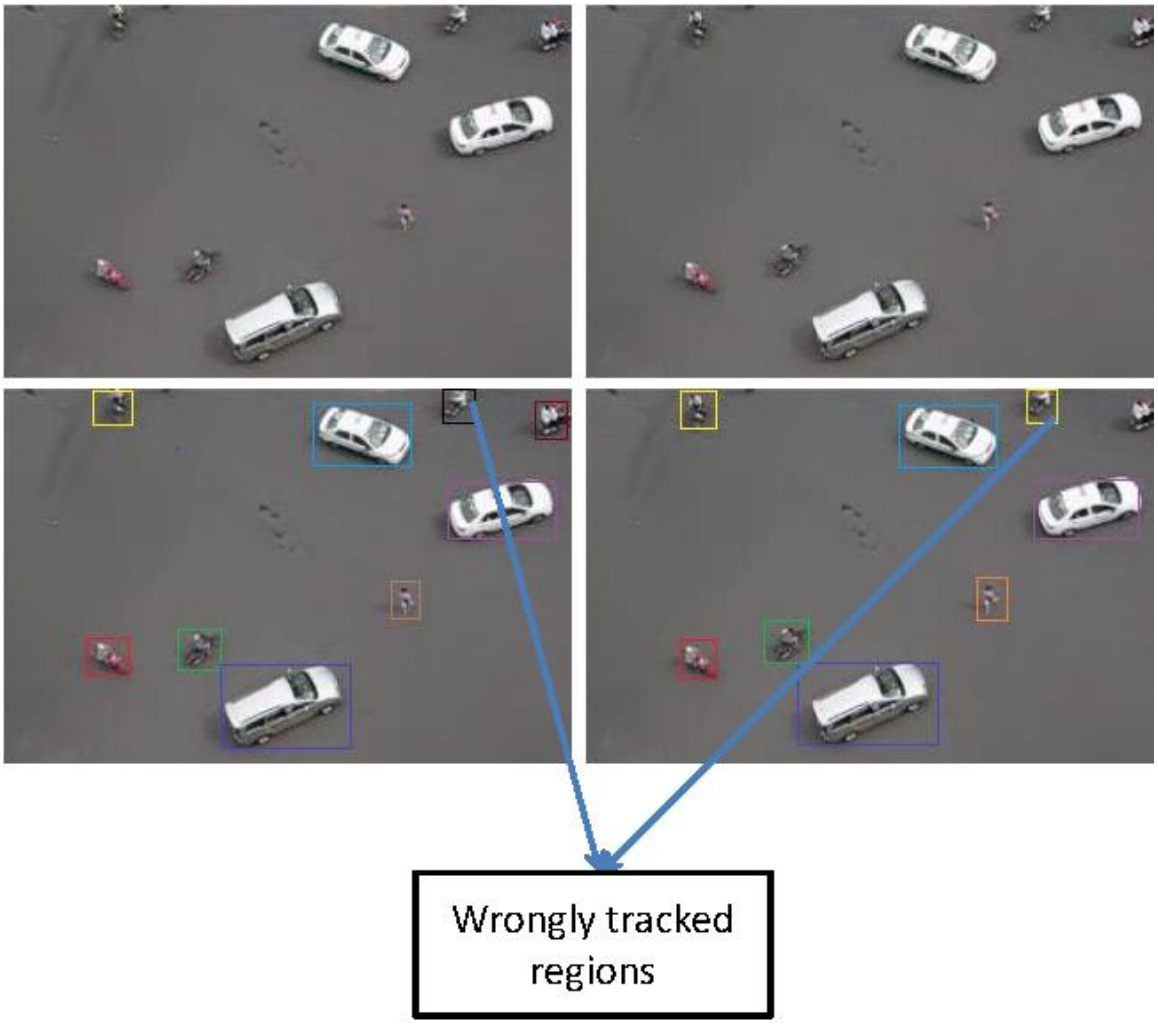


Figure 6.5: Example showing failure in tracking using MAD index d_{MAD} only

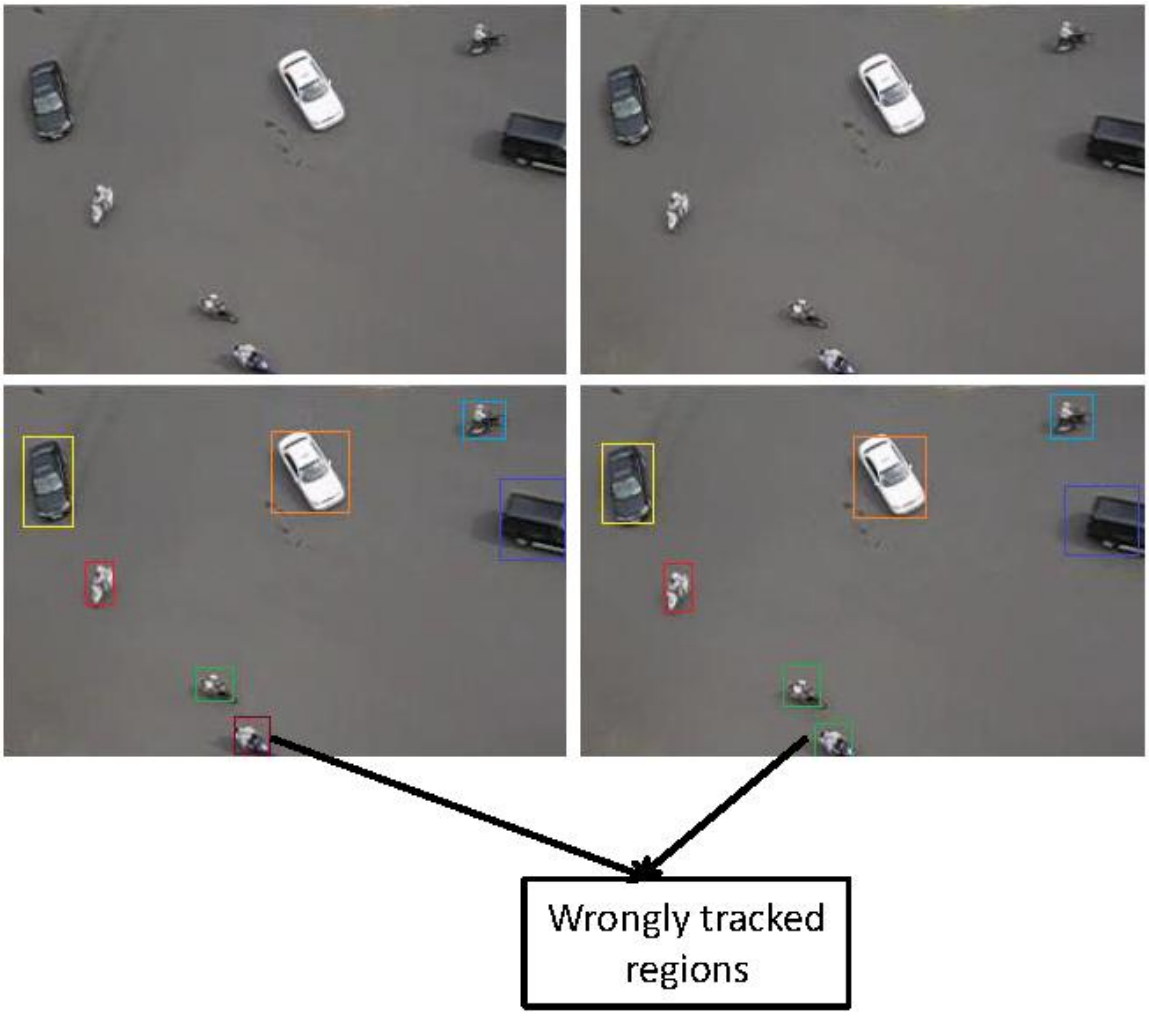
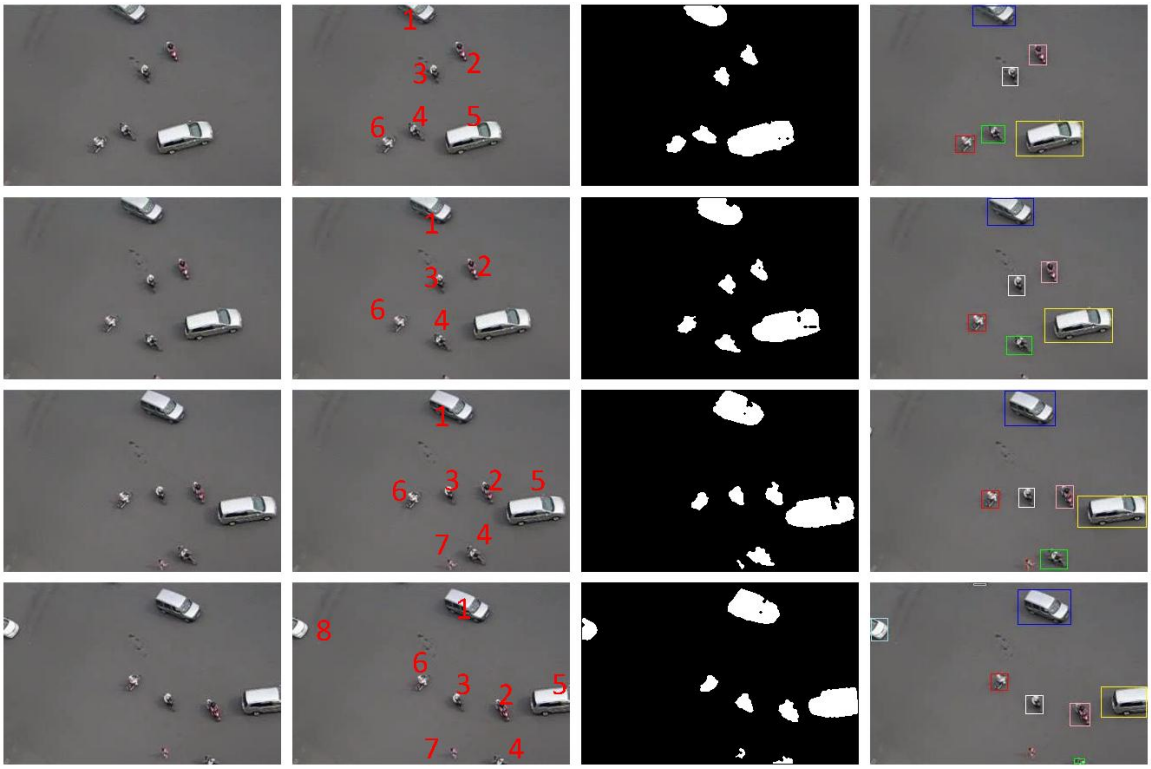


Figure 6.6: Example showing failure in tracking using feature index $d_{features}$ only

6.1.1.3. Tracking Results

Figure 6.7 and 6.8 shows the tracking results on some of the frames in *saigon01.avi* and Figure 6.9 and 6.10 shows the tracking results on some of the frames in *saigon02.avi*. Figures [6.7, 6.9, 6.11, and 6.13] (a) shows the original frame extracted from the video. Figures [6.7, 6.9, 6.11and 6.13] (b) shows the ground truth for the number of vehicles in each frame. Figures [6.7, 6.9, 6.11 and 6.13] (c) shows the segmentation results indicating the detected vehicle regions and Figures [6.7, 6.9, 6.11, and 6.13] (d) shows the tracking results on the vehicles detected. Color rectangle box is drawn around each vehicle as a reference for tracking and from Figures [6.7, 6.9, 6.11 and 6.13] (d) it is shown that the color of the box around each vehicle remains the same indicating that the vehicles are detected and tracked correctly. Also in the Figures 6.8, 6.10, 6.12 and 6.14, the trajectory of the vehicles found in Figures 6.7, 6.9, 6.11 and 6.13 are shown. Figures [6.8, 6.10, 6.12 and 6.14] (a, b) shows the start frame and end frame respectively used for illustrating vehicle tracking and trajectory. Figures [6.8, 6.10, 6.12 and 6.14](c) shows the trajectory of the vehicles from start frame and end frame. Figures [6.8, 6.10, 6.12 and 6.14] (d) shows the temporal smoothness map used for reference as ground truth. From the above illustration it is shown that the algorithm performs fairly well in detecting and tracking vehicles.

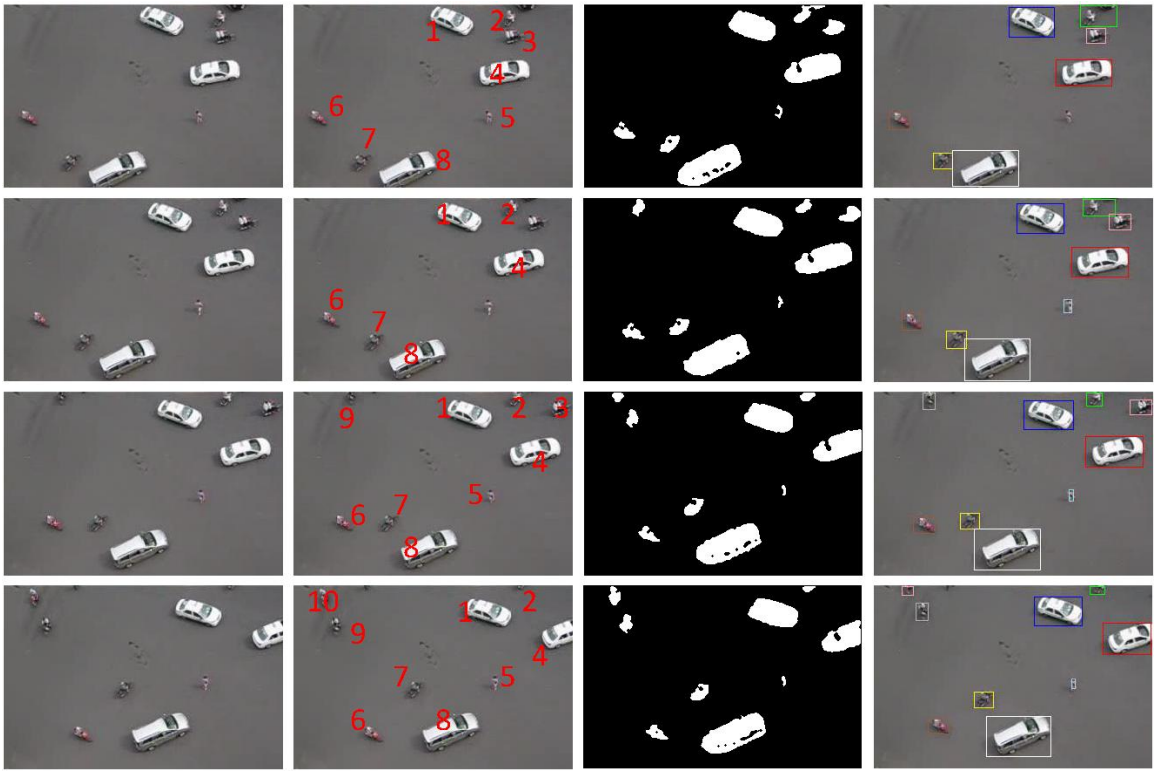


(a) (b) (c) (d)

Figure 6.7: Some tracking results on *saigon01* (a) Original frame (b) Ground Truth (c) Segmentation (d) Tracking results

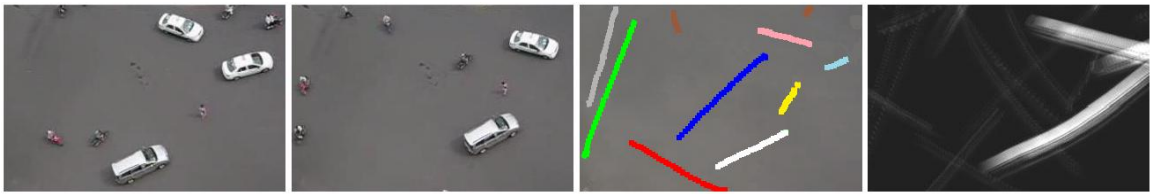
(a) (b) (c) (d)

Figure 6.8: Illustration of Vehicle Trajectory for Figure 6.7 (a) Start Frame (b) End Frame (c) Vehicle trajectory obtained from algorithm (c) Temporal smoothness map



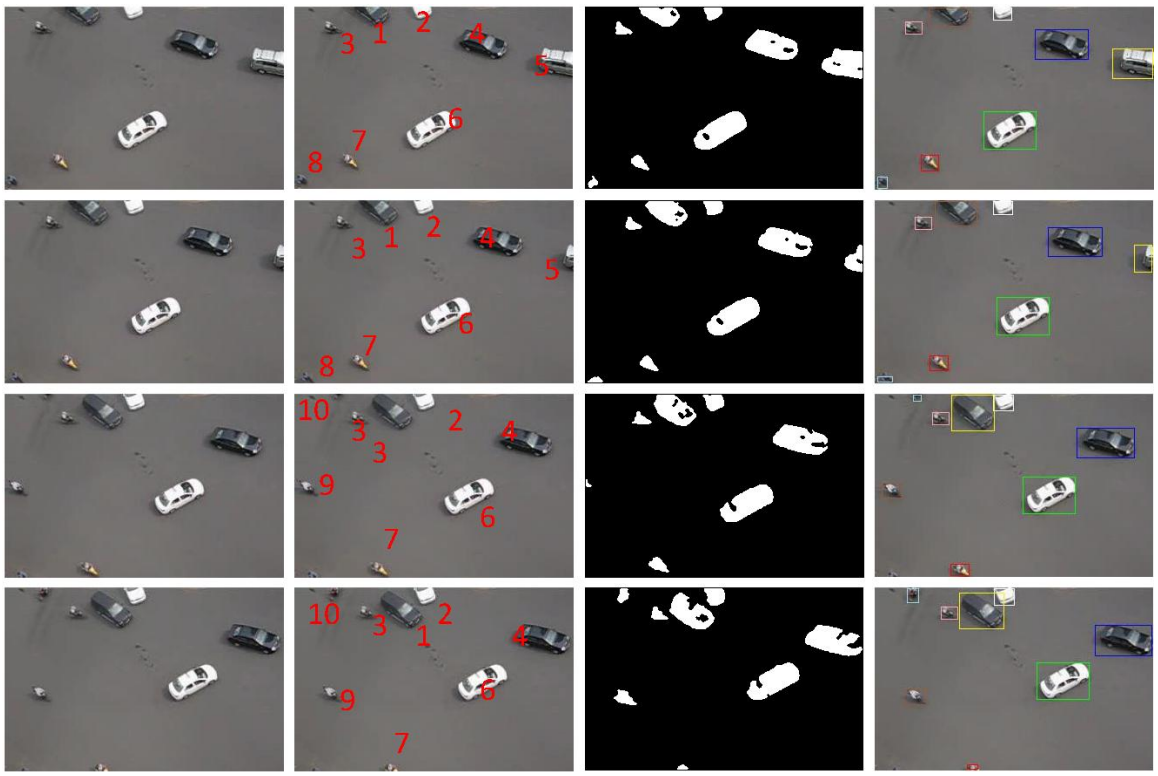
(a) (b) (c) (d)

Figure 6.9: Some tracking results on *saigon01* (a) Original frame (b) Ground Truth (c) Segmentation (d) Tracking results



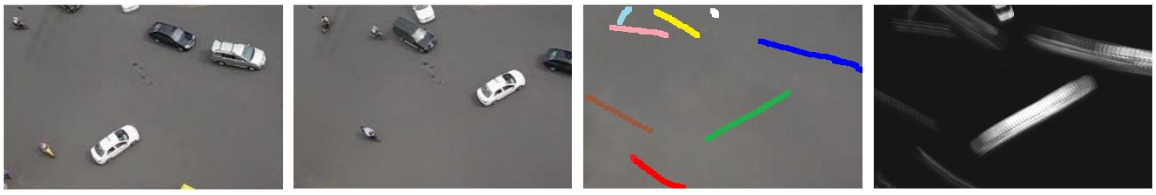
(a) (b) (c) (d)

Figure 6.10: Illustration of Vehicle Trajectory for Figure 6.9(a) Start Frame (b) End Frame (c) Vehicle trajectory obtained from algorithm (c) Temporal smoothness map



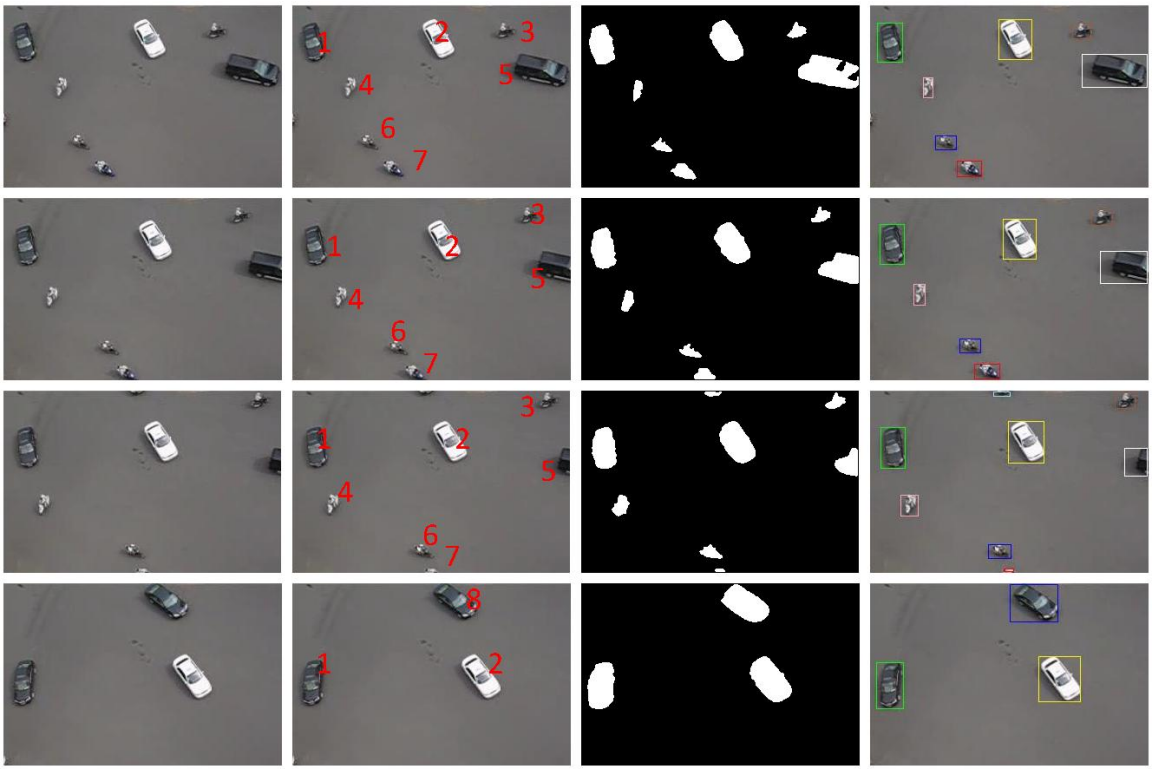
(a) (b) (c) (d)

Figure 6.11: Some tracking results on *saigon02* (a) Original frame (b) Ground Truth (c) Segmentation (d) Tracking results



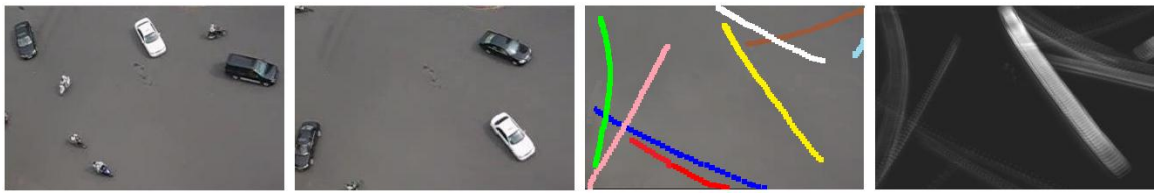
(a) (b) (c) (d)

Figure 6.12: Illustration of Vehicle Trajectory for Figure 6.11 (a) Start Frame (b) End Frame (c) Vehicle trajectory obtained from algorithm (c) Temporal smoothness map



(a) (b) (c) (d)

Figure 6.13: Some tracking results on *saigon02* (a) Original frame (b) Ground Truth (c) Segmentation (d) Tracking results

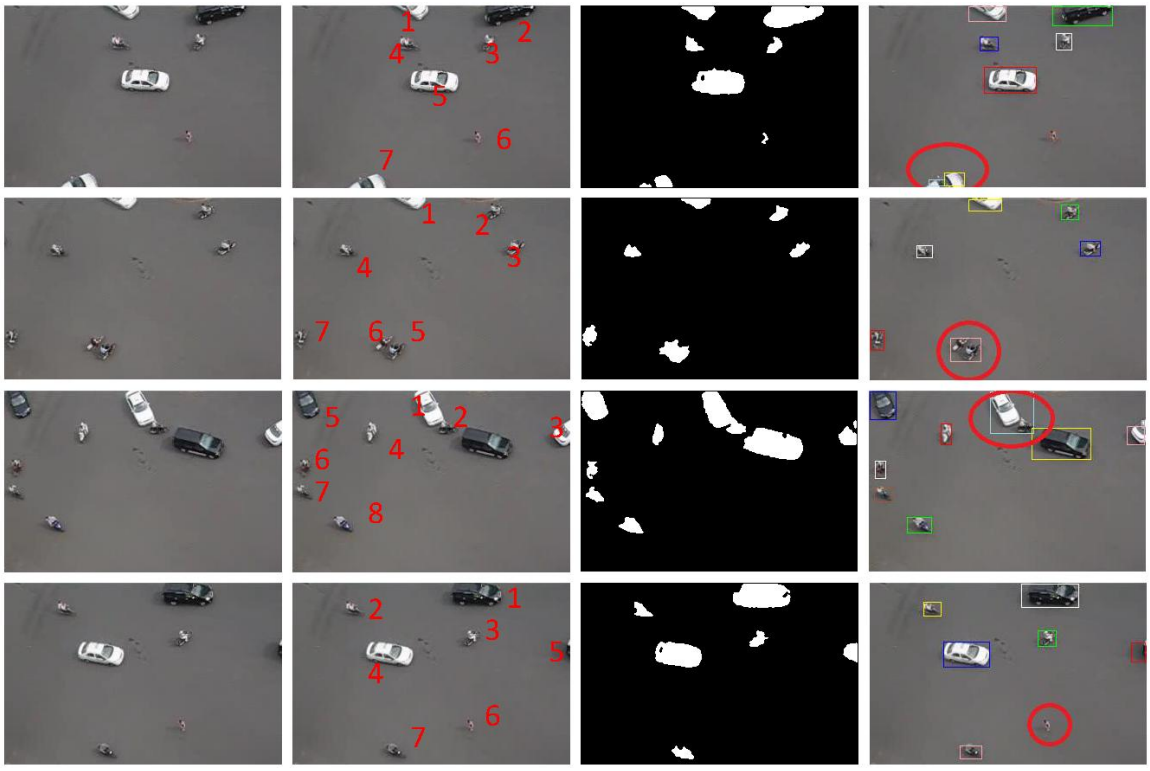


(a) (b) (c) (d)

Figure 6.14: Illustration of Vehicle Trajectory for Figure 6.13 (a) Start Frame (b) End Frame (c) Vehicle trajectory obtained from algorithm (c) Temporal smoothness map

6.1.1.4. Errors in Vehicle Detection and Tracking

Errors in vehicle tracking were primarily caused by wrong vehicle detection. Wrong vehicle detections were caused by occlusion, small size of the vehicles and vehicles have low contrast with the background. Few error examples are shown in the Figure 6.15. Figure 6.15(a) shows the frames used, Figure 6.16(b) shows the ground truth labeled manually, Figure 6.15(c) shows the segmentation map for the frame used and Figure 6.15(d) shows the error results. From the first row of the Figure 6.17 it can be seen that a single vehicle is identified as two objects due to the fact that part of the object has a contrast similar to the background and hence is considered as a part of background giving rise to a disjoint vehicle which is incorrectly interpreted as two separate vehicles. The error is zoomed and shown in the Figure 6.16. From the second and third row of the Figure 6.15 it can be seen that part of an object is merged with other object due to existence of shadow which creates partial occlusion or due to small size of the vehicle and hence there is a problem of two or more objects detected as single object. The errors are zoomed for better display in Figure 6.17. From the fourth row of Figure 6.15 it can seen that an object is not been detected due to its relatively small size and hence not been tracked correctly.



(a) (b) (c) (d)

Figure 6.15: Error Examples (a) Original Frame (b) Ground Truth (c) Segmentation (c) Error Detection

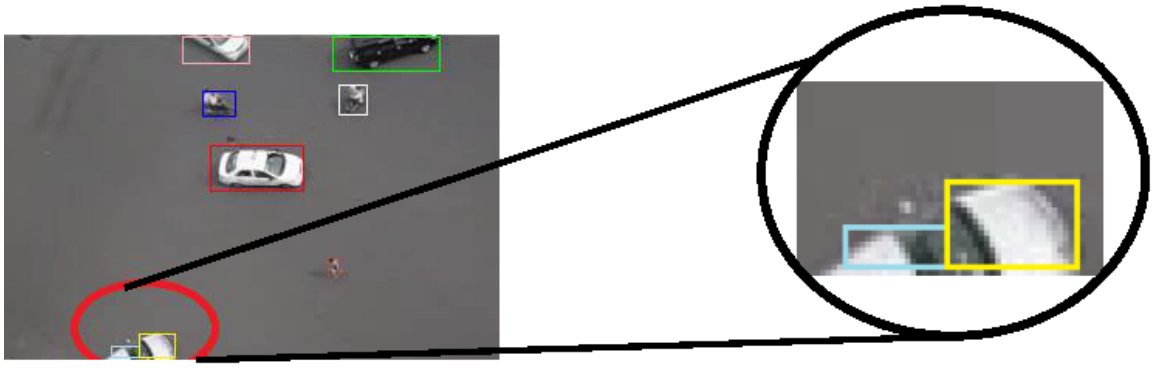


Figure 6.16: Error due to vehicle having low contrast with background

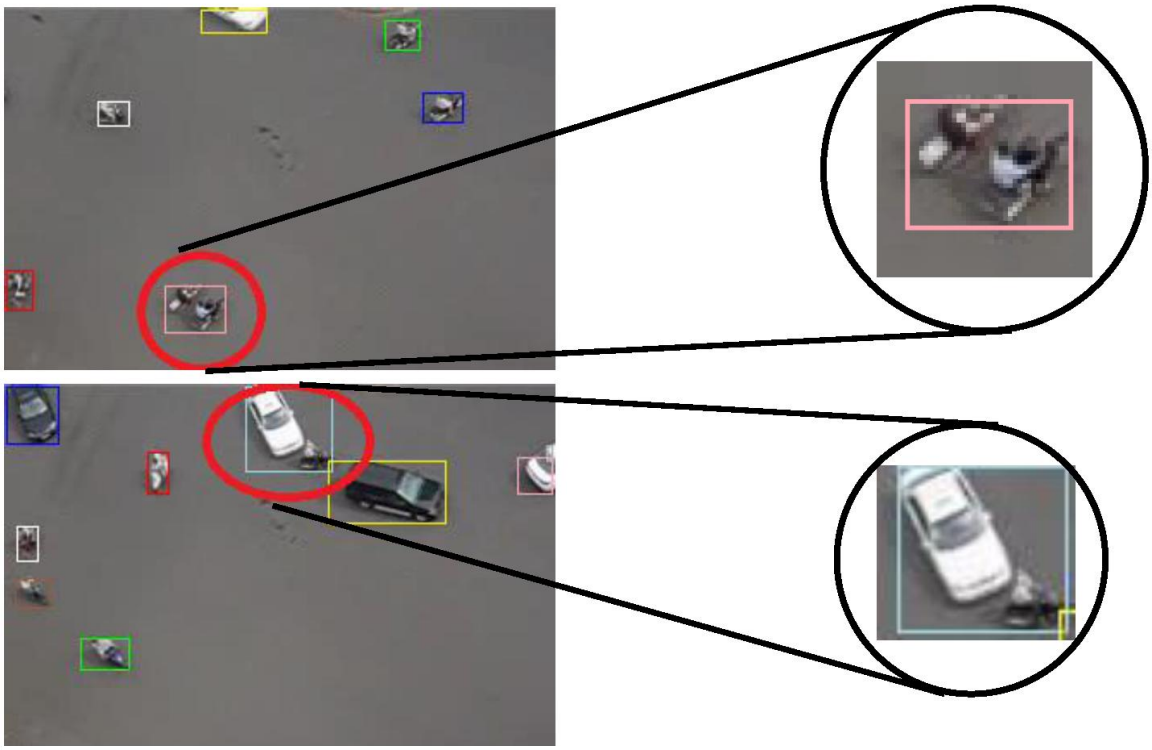


Figure 6.17: Errors due to merging of vehicles

Table 6.5: Evaluation on Detection and Tracking

		saigon01	***saigon02***
Single Frame Detection	Frames	342	212
	Vehicle Instances	2572	1403
	Merged Instances	89	70
	Detections	2395	1274
	Detection Rate	93.1%	90.8%
	False Detection	21	15
Tracking through video	Individual vehicles	39	27
	Complete tracks	36	24
	Incomplete tracks	3	3
	Tracking rate	92.3%	88.8%

6.1.1.5. Overall Performance of Vehicle Detection and Tracking Algorithm

The detection and tracking algorithm were tested on two video sequences *saigon01.avi and saigon02.avi.*The original frame rate of the videos were 30 *Hz* but were resampled to 15 *Hz* to reduce processing redundant frames. The segmentation and tracking performance of the proposed method were evaluated quantitatively. Ground truth labels for these videos were manually determined using a naive human subject. Table 6.5 shows the performance of the algorithm on vehicle detection and tracking. To evaluate vehicle segmentation, each frame was processed independently. Correct detection is defined as detection which has an overlap of more than 80% with a ground truth vehicle. The detection rate of the algorithm was found to be quite high (93.1% for *saigon01.avi* and 90.8% for *saigon02.avi).* This is considering the fact that detection is based only on background subtraction and no occlusion detection process has been added to the detection algorithm. Miss detection is mostly due to two reasons: 1) large part of the objected is not detected as foreground, because of low contrast to the background or scene occlusion 2) large part of the object is occluded by other objects. Among a total of 39 individual vehicles in *saigon01.avi,* 36 are tracked completely. 24 of 27 individual vehicles in *saigon02.avi* have complete tracks. Shorter tracks or broken tracks are mostly caused by persistent miss-detections. Overall the algorithm had a reasonable performance rate and results were promising.

6.1.1.6. Speed of the tracked vehicles

Once the vehicles are detected and tracked correctly, speed of the vehicles can be calculated using the distance travelled by the vehicle between consecutive frames. The computation of speed of the vehicles is explained in Chapter 5, Section 3.1. Table 6.6 shows the average velocity of the vehicles in *mph* in the two video sequences (*saigon01.avi* and *saigon02.avi*). Speed of the vehicles obtained from the algorithm is reasonable, since they closely resemble the speeds of

vehicles at busy intersections. Since there is no ground truth for vehicle velocities in two videos, the exact accuracy and precision of the method cannot be determined.

Table 6.6: Average velocity of the vehicles in video sequences

Vehicle Number	**Average Speed (mph)** ***saigon01.avi***	**Average Speed (mph)** ***saigon02.avi***
1	6.97	9.1
2	12.2	7.2
3	6.4	8.4
4	13.1	12.6
5	7.2	22.2
6	14.1	11.8
7	Incorrectly tracked	10.6
8	18.8	10.1
9	16.1	12.7
10	11.5	Incorrectly tracked
11	14.5	17.3
12	21.3	11.6
13	10.31	10.3
14	Incorrectly tracked	13.7
15	16.25	22.1
16	10.1	14.1
17	17.3	12.4

Continued on next page…..

Vehicle Number	**Average Speed (mph)** ***saigon01.avi***	**Average Speed (mph)** ***saigon02.avi***
18	13.5	10.3
19	18.3	9.2
20	9.1	Incorrectly tracked
21	15.1	7.8
22	15.3	Incorrectly tracked
23	Incorrectly tracked	15.5
24	23.8	16.1
25	12.8	17.1
26	10.17	15.9
27	7.45	14.7
28	17.6	-
29	12.2	-
30	18.1	-
31	15.3	-
32	12.9	-
33	14.5	-
34	17.1	-
35	13.5	-
36	21.7	-
37	16.4	-
38	10.9	-
39	8.9	-

6.1.1.7. MATLAB Performance

Vehicle Detection and Tracking algorithm were developed on MATLAB and the performance of the algorithm was reasonably good as discussed earlier. Table 6.7 presents the timing performance of the detection and tracking algorithm using MATLAB. The algorithm was tested on Intel Core2Quad 2.4 *GHz* CPU with 3 *GB* RAM. The operating system used was Microsoft Windows 7, 64 bit operating system. As one can expect MATLAB is good for testing and simulation but not suitable for real-time implementation. Also from the analysis, it was found that average processing time per frame using MATLAB is approximately 22 seconds. The vehicle detection and feature extraction step took 1000 *ms* on an average and Tracking and Speed detection step took 20 *ms* on an average. But from the table was found that the MAD analysis had high computational rate because of block based processing of the algorithm. On an average MAD analysis took 21 *seconds* for analysis on two consecutive frames. Using features alone based detection and tracking the average processing time was found to be 1020 *ms* per frame, which was quite acceptable. Therefore there was a need to replace Human vision based (MAD) analysis by equivalent metric (e.g., Mean squared error), which can yield acceptable performance and results.

Table 6.7: Timing Performance of Detection and Tracking Algorithm using MATLAB

Algorithm	**Average Timing (milliseconds) per frame**
Vehicle Detection	600
Feature Extraction	400
MAD Analysis	21000 (two consecutive frames)
Tracking and Speed Detection	20 (two consecutive frames)
Total	22020

6.1.1.8. MAD equivalent metric suitable for Vehicle Tracking

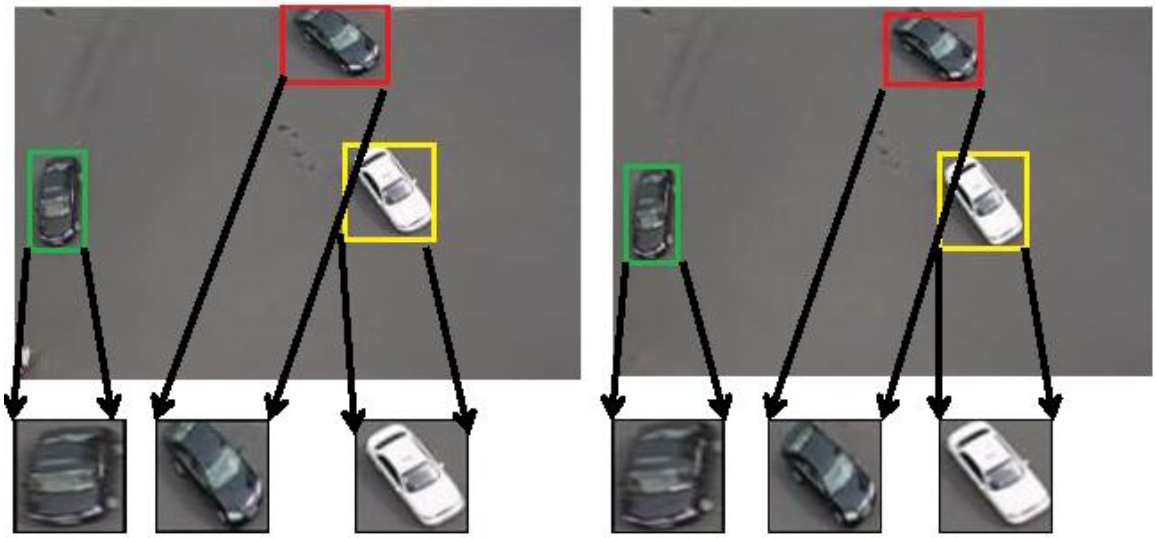
From the Table 6.7 it can be seen that average processing time per frame for detection and tracking was 22 seconds, this was largely due to the high computation rate of MAD analysis. Therefore there was a need to replace MAD by another equivalent metric that closely resembles Human vision based analysis on image similarity. For this purpose Mean squared error (MSE) metric was chosen. Although MSE is not good metric for predicting visual similarity of two images (objects, vehicles), the main advantage of MSE is the computation time. It has fairly low computational complexity. And also when MAD was replaced by MSE in our algorithm, the system produced reasonable results and also the processing time per frame was way faster than before. Just like MAD, MSE compares two vehicle regions extracted from consecutive frames and gives the similarity index between the two regions. Figure 6.18 shows an illustration of how MSE works.

From Table 6.19 it is shown that MAD index obtained for Figure 6.18 and MSE index obtained for Figure 6.20, is equivalent suggesting that MAD index indeed can be replaced by MSE. Similar to MAD the regions have to resized to be common size of atleast (64x64 *pixels*) and lower the MSE index higher is the match between two regions. However it should be noted it works only in case of vehicles here, where the vehicle regions are of low resolution and assumed noise free. However for other image processing applications MAD has high performance when compared to MSE. Therefore in the tracking algorithm MAD index was replaced by equivalent MSE index. However the same weighing factor $\alpha = 0.9$ was used to combined feature index and MSE index. Table 6.9 shows the tracking performance of the algorithm where MAD index is replace by MSE index. From the table it was shown that the performance of tracking using the combination of feature index and MSE index compared to the combination of MAD index and MSE index remains the same. This was largely due to the fact that errors created by MSE index are overcome by feature index and vice-versa. Therefore the combination of feature index and

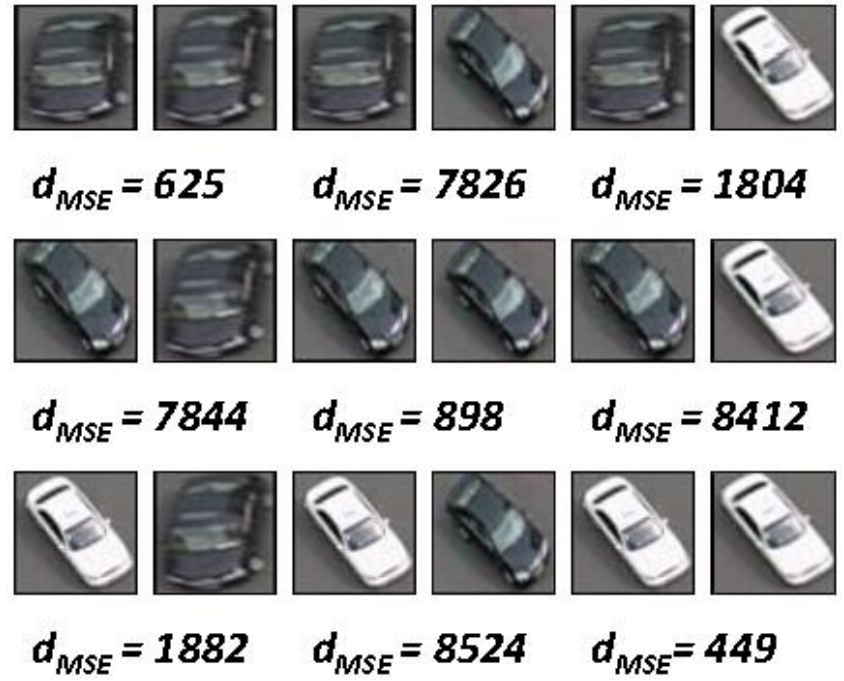
MSE was chosen for vehicle tracking for real-time implementation of the vehicle tracking system. Table 6.10 shows the timing performance of the algorithm using MSE. The average processing time per frame was 1320 *ms* which is much faster than using MAD. MSE analysis on two consecutive frames consumed about 300 *ms* when compared to 22 *seconds*. This may not be reasonable comparison since MAD was originally developed for predicting the visual quality of an image but was used in the tracking algorithm because of its advantage of predicting image (vehicle) similarity.

Table 6.8: Comparison of MAD and MSE index for Figure 6.18

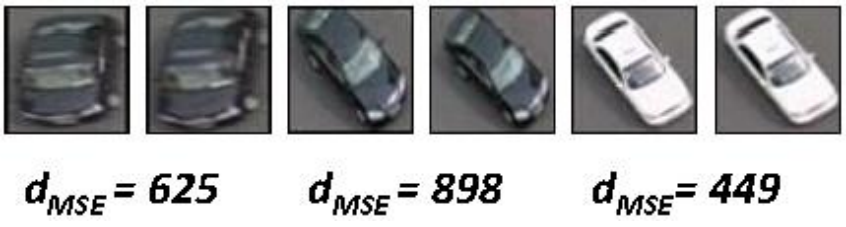
Regions						
	d_{MAD}	d_{MSE}	d_{MAD}	d_{MSE}	d_{MAD}	d_{MSE}
	4.38	**625**	9.43	7826	12.12	1804
	d_{MAD}	d_{MSE}	d_{MAD}	d_{MSE}	d_{MAD}	d_{MSE}
	9.59	7844	**5.16**	**898**	10.45	8412
	d_{MAD}	d_{MSE}	d_{MAD}	d_{MSE}	d_{MAD}	d_{MSE}
	12.07	1822	10.39	8524	**2.69**	**449**



(a) (b)



(c)



(d)

Figure 6.18: Example of MSE (a) Frame at time *t* (b) Frame at time *t+1* (c) MSE index for different vehicle comparisons (d) MSE index for matching vehicles

Table 6.9: Tracking performance of the algorithm with MSE

		saigon01	***saigon02***
Tracking through video	Individual vehicles	39	27
	Complete tracks	36	24
	Incomplete tracks	3	3
	Tracking rate	92.3%	88.8%

Table 6.10: Timing Performance of Detection and Tracking Algorithm using MSE

Algorithm	**Average Timing (milliseconds) per frame**
Vehicle Detection	600
Feature Extraction	400
MSE Analysis	300 (two consecutive frames)
Tracking and Speed Detection	20 (two consecutive frames)
Total	1320

6.2. Experimental Results of Accident Detection System

Since there were no real crash data to work on, an experimental setup was build indoor in the laboratory. The complete description of the setup is explained in detail in Chapter 3, Section 3.3. Different scenarios involving collisions were implemented that are difficult to be simulated in the real world due to constraint of resources and technology. Different collision scenarios such as side-on collision, rear-collision, front-end collision were simulated and tested. The objective was to create as much as scenario as possible and to study the performance of the algorithm on detecting accidents at intersection. Figure 6.19 shows a complete description of different collision scenarios.

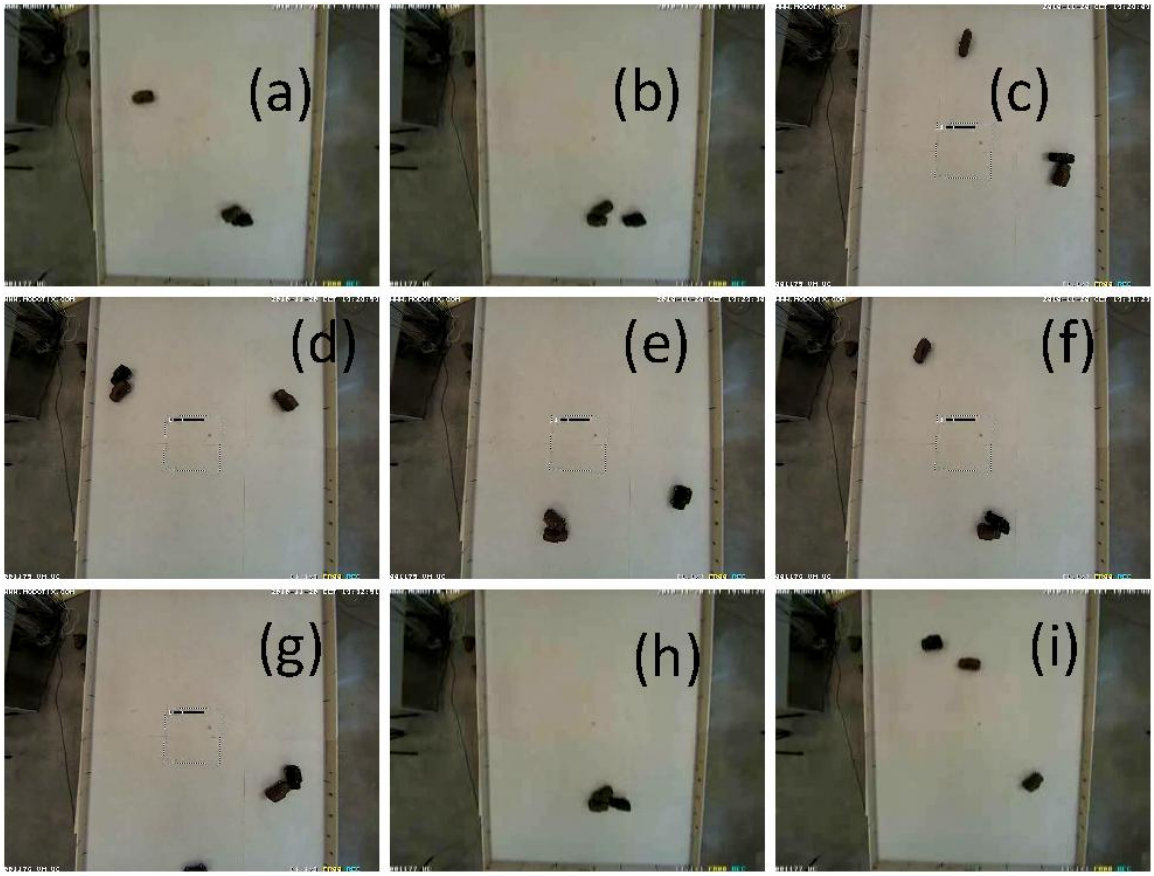


Figure 6.19: Different collision scenarios (a) Vehicles moving closely (b) Rear end collision by the incoming vehicle (c) Side-on collision at intersection (d) Rear end collision by the trailing vehicle (e) Side-on collision at turn (f) Side on collision from left (g) Side-on collision from right (h) Multiple vehicle collision (g) Normal tracking

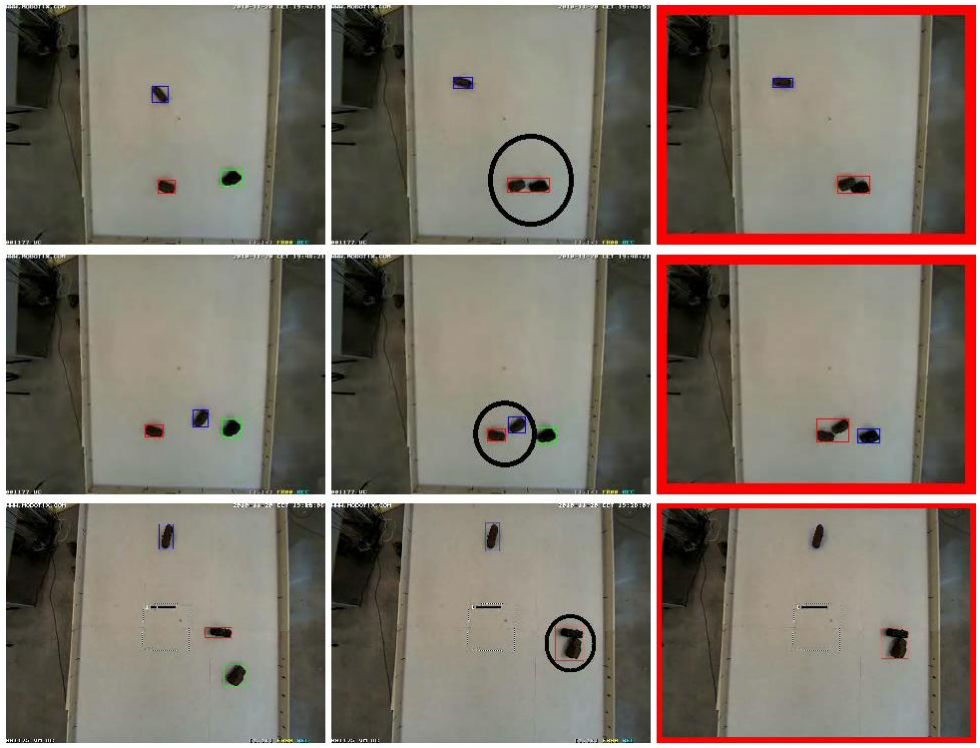
Along with collision data, normal movement of vehicles with occlusion was also added to test the accuracy and precision of the system.

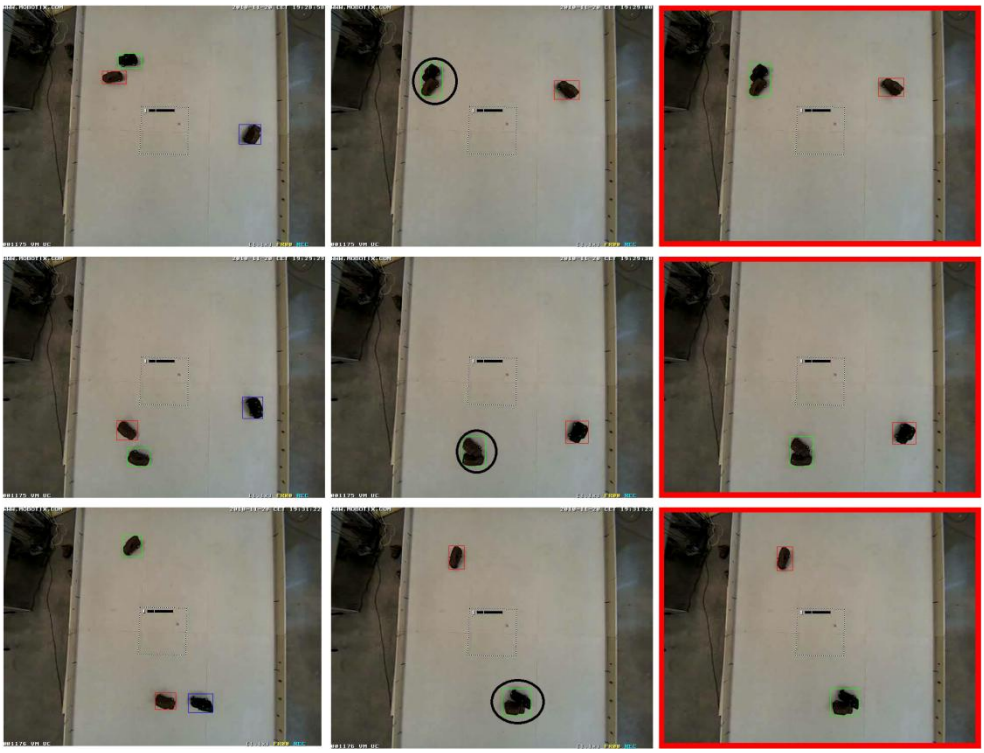
6.2.1. Performance of the algorithm

The performance of the accident detection system as described in Chapter 5, Section 5.4 was tested on different collision scenarios created. The results of the collision detection algorithm on various crash scenarios are presented in Figure 6.20. Figure 6.20(a) shows the scene before the occurrence of accident. Figure 6.20(b) shows the scene at point of accident as denoted by drawing black circle around the point of accident and Figure 6.20(c) shows the scene after the accident. To denote the occurrence of accident a red box is drawn around the scene of accident. As discussed earlier using the Accident Index *AI* the occurrence of accident is determined and the point of accident can be obtained from the centroids of the vehicles involved in the accident. While determining the occurrence of accident the speeds of the vehicles were critical, since the speed was a major factor in determining accident. The algorithm also presented false cases, especially when the vehicles are moving together. Due to the problem of merging of closely moving vehicles, the area, centroid, and orientation of the vehicles being tracked changes significantly, thus presenting a scenario for false detection. This is illustrated in the Figure 6.21. In this case the vehicles follow each other closely and because of the merging, a false alarm was raised.

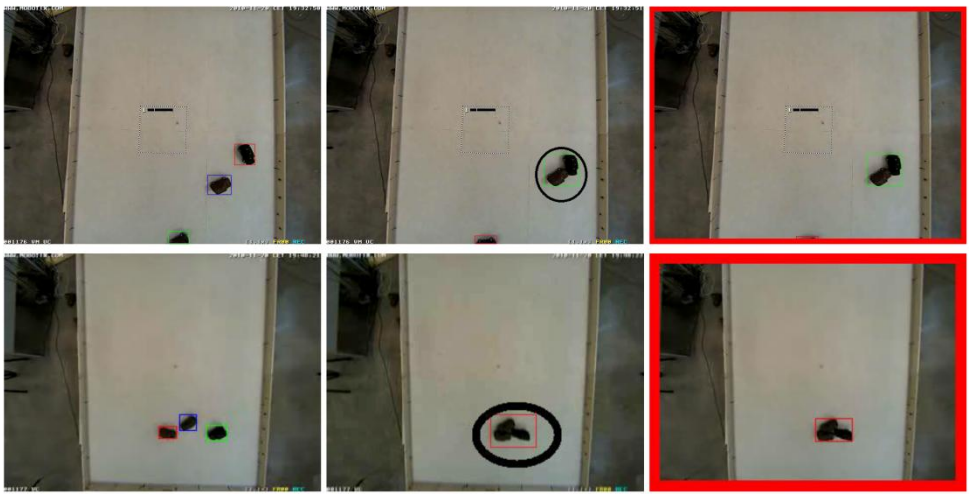
6.2.2. Algorithm Evaluation

Since the main objective of the algorithm was to detect all possible cases of accident, false alarms are acceptable as long as there are not many false alarms. When a collision happens and it is detected correctly, it is true positive. When a collision does not happen and collision detection is determined, it is false positive. When a collision happens and it is not detected by the system, it is false positive. The algorithm was designed to have more true positives and false positives; therefore there were no false negatives for the simulated crash scenarios.



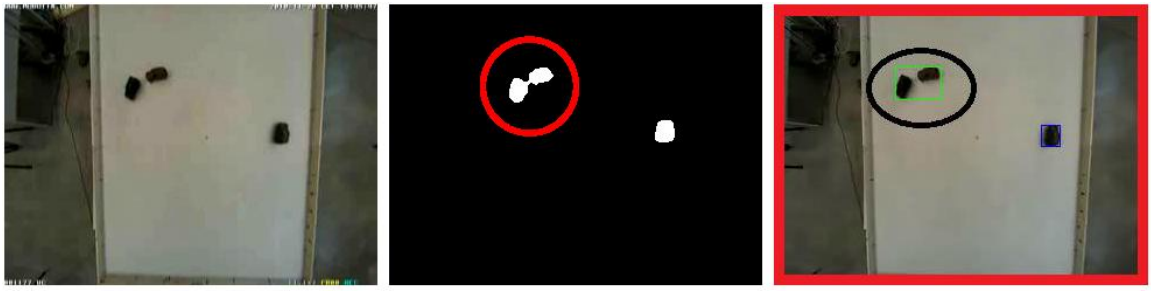


Continued on Next Page…..



(a) (b) (c)

Figure 6.20: Illustration of accident detection system for various crash scenarios (a) Before occurrence of accident (b) At point of accident (c) After accident



(a) (b) (c)

Figure 6.21: Example of False alarm (a) Closely moving vehicles (b) Merged Blob (c) False alarm raised by the system

The performance of the algorithm is evaluated in terms of precision (of all the detections) and coverage (of all the collisions respectively) [114]. The terms are described in Figure 6.22.

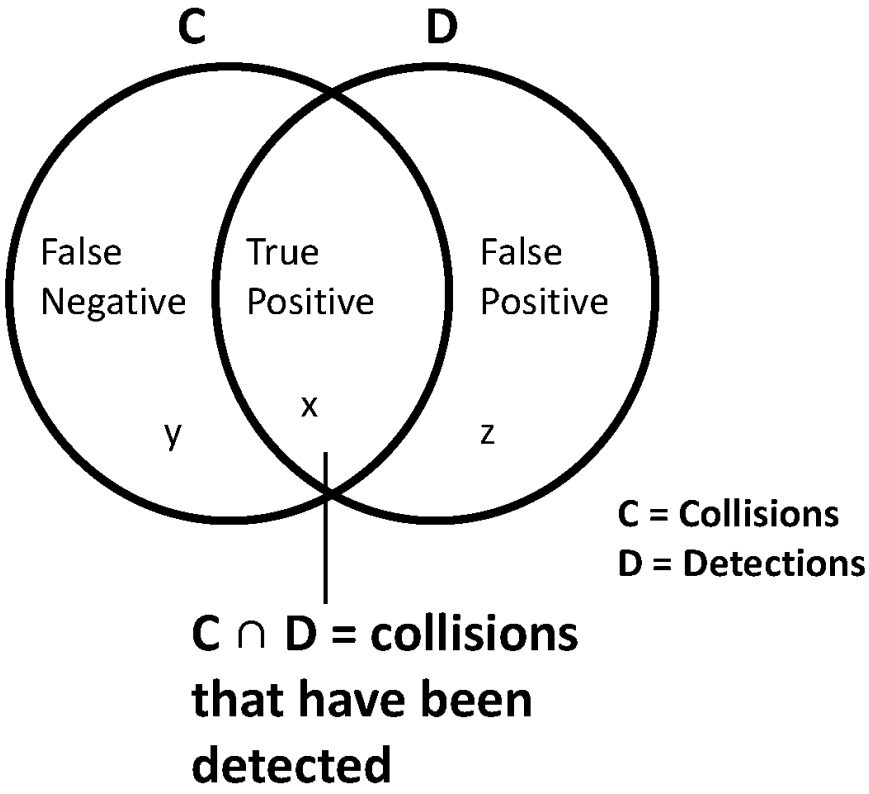


Figure 6.22: Performance Evaluation Terms (114)

$$precision = \frac{no.of\ valid\ Detections}{total\ collision\ Detections}$$

$$= \frac{true\ \ positive}{(true\ \ positive + false\ \ positive\ )} = \frac{x}{x+z} \tag{6.1}$$

$$coverage = \frac{no.of\ valid\ Detections}{total\ Collisions}$$

$$= \frac{true\ \ positive}{(true\ \ positive + false\ \ negative\ )} = \frac{x}{x+z} \tag{6.2}$$

For the nine test scenarios as shown in Figure 6.19 the precision and coverage of the algorithm is given as follows as: Number of true positives = 7, Number of false positive = 1 and Number of false negative = 0. Therefore

$$precision = \frac{7}{8} = 0.875$$

$$coverage = \frac{7}{7} = 1$$

Based on accuracy evaluation of the algorithm, the precision of the algorithm is about 87.5% and coverage is about 100% for the test scenarios. Besides proving the effectiveness of the collision detection by the system, this evaluation method helps to find parts of the collision detection that needs improvement. Overall the performance of the collision detection system was acceptable.

6.3. Real-Time Implementation of the System

For the Real-time implementation of the collision detection system at traffic intersection the MATLAB algorithm was converted to C++. The videos were obtained from Mobotix Q24 camera and images were read by *gbuffer* class in C++ implemented by Dr. Damon Chandler. Since the main objective of this thesis is to develop a collision detection system capable of operating in real-time, some compromises were made in vehicle detection part, especially the incapability of the system to handle occlusion. Also for the system to operate in real-time the MAD index was replaced by equivalently good MSE index. And it was shown earlier that MSE index indeed worked well for vehicle tracking. The detection and tracking part of the algorithm were tested on real-traffic scenarios (*saigon01.avi* and *saigon0.avi*) and average processing time per frame was found to be *210 ms* which is about 5 *frames/second*. But frame rate of the videos were 15 *frames/second*. But it was found that 5 *frames/second* processing speed was good enough, since some of the frames were redundant and the algorithm performed fairly well in terms of detection and tracking which included vehicle speed and trajectory calculation. The above processing time were computed using Intel Core2Quad 2.4 *GHz* CPU with 3 *GB* RAM. The operating system used was Microsoft Windows 7, 64 bit operating system. Table 6.11 shows the overall timing performance of detection and tracking algorithm using C++. These timings were obtained for

videos with resolutions 320x240 *pixels*. Table 6.12 shows the overall timing of the detection and tracking algorithm implemented on different core processors.

Table 6.11: Timing Performance of Detection and Tracking Algorithm using C++

Algorithm	Average Timing (milliseconds) per frame
Vehicle Detection	40
Feature Extraction	130
MSE Analysis	10 (two consecutive frames)
Tracking and Speed Detection	30 (two consecutive frames)
Total	210

Table 6.12: Processing speed of tracking algorithm obtained using different core processors

Processor Specifications	Overall time (milliseconds) per frame
Intel Core2Quad 2.4 *GHz*	210
Intel Core2Duo 2.00 *GHz* (Laptop)	318
Intel Core 2.4 *GHz*	266

For the C++ implementation of collision detection system, the developed algorithm was tested on crash scenarios simulated in the testbed. The original resolution of the videos used was 640x480 *pixels* with a frame rate of 10 *Hz*. Table 6.13 shows the timing performance of the collision detection algorithm for the test cases with an image resolution of 640x480 *pixels* and Table 6.13shows the timing performance of the collision detection algorithm for the test cases with an image resolution of 320x240 *pixels*. From Table 6.13 and Table 6.14 it is shown that the videos having image resolution of 320x240 *pixels* were processed faster compared to videos having image resolution of 640x480 *pixels*. The difference in processing time was primarily due to vehicle detection and feature extraction step, where quality time is consumed by binary processing, connected component labeling and finding the region properties of the extracted vehicle regions.

From Table 6.15 it is shown that the collision detection performance of the algorithm on videos having image resolution of 320x240 *pixels* and resolution of 640x480 *pixels* were the same, indicating that the resolution of images can be scaled down by atleast a factor of two for faster processing of the collision detection algorithm.

Table 6.13: Timing Performance of Collision Detection algorithm on image resolution of 320x240 pixels

Algorithm	**Average Timing (milliseconds) per frame**
Vehicle Detection and Feature Extraction	130
Vehicle Tracking and Speed Detection	40
Collision Detection	10
Total	180

Table 6.14: Timing Performance of Collision Detection algorithm on image resolution of 640x480 pixels

Algorithm	Average Timing (milliseconds) per frame
Vehicle Detection and Feature Extraction	630
Vehicle Tracking and Speed Detection	40
Collision Detection	10
Total	680

Table 6.15: Comparison of Collision Detection Algorithm on different image resolution

Evaluation terms	Image resolution of 640x480 *pixels*	Image resolution of 320x240 *pixels*
True positive	7	7
False positive	1	1
False Negative	0	0

6.4. Comparison with other existing methods

It is a fact that an exact comparison between vision-based works of traffic monitoring is nearly impossible because of difference of difference of input stream as aspect of complexity, light condition, average number of vehicles running on the highway, quality of film, height and angle of camera and also quality of image acquisition [97]. But from the tracking and collision detection results it is shown that the collision detection system performs reasonably well in comparison with other methods.

Some of the algorithms that have focused on real-time vehicle tracking using video processing are that of Rad and Jamzad [97]. Their method was able to process 11*frames/second* with frame sizes (320x320 *pixels*) and had accuracy of 96% on tracking vehicles. Kim and Malik [9] work was able to process 10 *frames/second* with frame sizes (200x200*pixels*) and had accuracy of 85% on tracking vehicles. Gupte *et al*. [76] work was able to process 11 *frames/second* and accuracy of 90% on tracking and detecting vehicles. Kanhere and Birchfield [115] work was able to process 30 frames/second with frame sizes (320x240 *pixels*) and had accuracy of 91-97% on tracking vehicles. Lin *et al.* [116] method was able to process at 15 *ms*.

In comparison with the above methods, the tracking algorithm presented in this thesis was able to process 5 *frames/*second and had an accuracy of 88-92% on vehicle detection and tracking. Along with tracking the speed of the vehicles are computed which is an essential step in collision detection systems. However, using computers with higher CPU speed this frame rate can be improved.

Although there are number of vehicle detection and tracking methods that are capable of operating in real-time, rarely few methods have focused on real-time implementation of collision detection system which is quite complex. Ki and Lee [106] developed a collision detection method that had a detection rate of 60% and false alarm rate of 0.00496%, but their processing

speed was unknown. Salim *et al*. [114] developed a computer simulation for detecting collision and their method had precision and coverage of 100% for their test cases. In comparison with these methods, the proposed method in this thesis is capable of operating in real-time and also has sufficiently good collision detection performance rate for the test cases considered. It is believed that with more training data and analysis the performance of the algorithm can be improved for real-traffic accidents.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1. Conclusion

In this thesis a crash detection system at traffic intersections that is capable of operating in real-time is presented. More emphasis has been given to vehicle detection and tracking stage, since they are essential to extract suitable vehicle features and vehicle parameters that can be used as factors for determining crashes at intersections. In this work, a tracking algorithm that uses a weighted combination of low-level features extracted from moving vehicles and low-level vision analysis on vehicle regions extracted from different frames is presented. The vehicle detection rate of the proposed algorithm is about 90-93% and tracking rate is about 88-92% on two test videos. The average processing speed of the algorithm is about 5 *frames/*second for the two test videos used. The detection and tracking rate of the algorithm was decremented due to the problem of shadows and occlusion, which the algorithm did not address in this work. If the problems of shadows and occlusion are addressed, the performance rate of the proposed algorithm is expected to go high. Overall from the work presented it is shown that proper combination of low-level features that have low computational complexity are sufficient for vehicle tracking compared to complex feature tracking methods.

Using the low-level features and vehicle velocity of the correctly tracked vehicles, crashes were detected by the system using an accident index calculated from speed, area, orientation and position indexes of the tracked vehicles. The proposed crash detection system has

a precision (correct detection rate) of 87.5% and detection rate of 100% for test crashes created using experimental test-bed. Overall the performance of the collision detection system is good particularly considering the fact that the algorithm is capable of operating in real-time. Also the method used a low-level features instead of any learning algorithm such as Hidden Markov Model, Neural Networks, etc. that can consume a lot of time for computation and take decision. It is believed that with more analysis of traffic crashes data and more training for the collision detection algorithm, it can be implemented for monitoring real-time traffic scenarios.

7.2. Future work

To improve the performance of the detection and tracking algorithm, problems created by shadows and occlusion is planned to be addressed by using better background modeling techniques and re-segmentation of the segmented vehicle region using average color and lightness distance of blocks in the segmented vehicle regions. And also it is planned to make the vehicle detection and tracking algorithm operate under night conditions. Currently studies having been done on determining the features to be used to detect and track vehicles in night conditions. Also it is planned to collect more traffic data from different camera angles to make the algorithm robust to various conditions and situations. Also the current focus is on analyzing the parts of the algorithm that can be optimized to increase the processing speed of the detection and tracking algorithm.

To improve the performance of the collision detection algorithm, it is planned to collect more crash cases from real-traffic situations by installing a camera at busy intersection and to analyze the performance of the algorithm on real-traffic situations. Currently the focus is on to determine the factors that can be added with existing low-level features and velocity information of the vehicle that can improve the overall performance and increase the robustness of the system.

REFERENCES

[1] Advanced Transportation Management System, [last accessed December 16, 2010] ; Available from: http://www.fhwa.dot.gov/tfhrc/safety/tms.htm

[2] *Intelligent Transportation Systems Research.* [Last accessed December 16, 2010]; Available from: http://www.tfhrc.gov/its/its.htm

[3] D. Beymer, P. McLauchlan, B. Coifman, J. Malik, A Real-time Computer Vision System for Measuring Traffic Parameters, Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 1997.

[4] H. S. Mahmassani, C. Haas, S. Zhou, and J. Peterman. Evaluation of incident detection methodologies. Technical Report FHWA/TX-00/1795-1, Center of Transportation Research, The University of Texas at Austin, Oct. 1998.

[5] W. H. Organization, "Fact sheet: Top ten causes of death," *Fact Sheet No. 310*, 2007, www.who.int/mediacentre/factsheets/fs310.pdf

[6] Jones, W.D., "Keeping cars from crashing," *Spectrum, IEEE* , vol.38, no.9, pp.40-45, Sep 2001
doi: 10.1109/6.946636

[7] U. D. of Transportation, "Traffic safety facts 2006," *National Highway Traffic Safety Administration*, 2006, http://www-nrd.nhtsa.dot.gov.

[8] W. Evanco, “The potential impact of rural mayday systems on vehicular crash fatalities,” *Accident Analysis and Prevention,* vol. 31, pp. 455–462, September 1999.

[9] Kim, Z.; Malik, J.; , "Fast vehicle detection with probabilistic feature grouping and its application to vehicle tracking," *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on* , vol., no., pp.524-531 vol.1, 13-16 Oct. 2003,doi: 10.1109/ICCV.2003.1238392

[10] X. Jin, D. Srinivasan, and R. L. Cheu, “Comparative appraisal of adaptive ANN-based freeway incident detection models,” in *Proc. IEEE 5th Intell. Transp. Syst. Conf.*, Singapore, 2002, pp. 720–726.

[11] D. Srinivasan, W. H. Loo, and R. L. Cheu, “Traffic incident detection using particle swarm optimization,” in *Proc. IEEE Int. SIS*, 2003, pp. 144–151.

[12] D. Srinivasan, R. L. Cheu, and Y. P. Poh, “Hybrid fuzzy logic-genetic algorithm technique for automated detection of traffic incidents on freeways,” in *Proc. IEEE Intell. Transp. Syst. Conf.*, Oakland, CA, Aug. 2002, pp. 352–357.

[13] H. Xu, C. M. Kwan, L. Haynes, and J. D. Pryor, “Real-time adaptive online traffic incident detection,” in *Proc. IEEE Int. Symp. Intell. Control*, Sep. 1996, pp. 200–205.

[14] T. Shuming, G. Xiaoyan, and W. Feiyue, “Traffic incident detection algorithm based on non-parameter regression,” in *Proc. IEEE 5th Intell. Transp. Syst. Conf.*, Singapore, 2002, pp. 714–719.

[15] S. Bhonsle, M. Trivedi, and A. Gupta, “Database-centered architecture for traffic incident detection, management, and analysis,” in *Proc. IEEE Intell. Transp. Syst. Conf.*, Dearborn, MI, Oct. 2000, pp. 149–154.

[16] I. Ohe, H. Kawashima, M. Kojima, and Y. Kaneko, "A method for automatic detection of traffic incidents using neural networks," in *Proc. IEEE Conf. Vehicle Navigat. and Inf. Syst.*, 1995, pp. 231–235.

[17] H. Dia and G. Rose, "Development and evaluation of neural network freeway incident detection models using field data," *Transp. Res., Part C Emerg. Technol.*, vol. 5, no. 5, pp. 313–331, Oct. 1997.

[18] M. Dougherty, "A review of neural networks applied to transport," *Transp. Res., Part C Emerg. Technol.*, vol. 3, no. 4, pp. 247–260, 1995.

[19] R. Gangisetty, "Advanced traffic management system on I-476 in Pennsylvania," in *Proc. IEEE ITSConf '97*.

[20] S. Lee, R. A. Krammes, and J. Yen, "Fuzzy-logic-based incident detectionfor signalised diamond interchanges," *Transp. Res., Part C Emerg. Technol.*, vol. 6, no. 5, pp. 359–377, Dec. 1998.

[21] E. R. Green, K. R. Agent, and J. G. Pigman, "Evaluation of Auto incident recording system (AIRS)," Kentucky Transp. Center.,Univ. Kentucky, Lexington, KY, Res. Rep. KTC-05-09/SPR277-03-1F,May 2005

[22] C. Harlow and Y. Wang, "Automated accident detection system," *Transp. Res. Rec.: J. Transp. Res. Board*, no. 1746, pp. 90–93, 2001.

[23] J. C. Rojas and J. D. Crisman, "Vehicle detection in color images," in *Proc. IEEE ITS Conf '97*.

[24] N. Zeng and J. D. Crisman, "Vehicle matching using color," in *Proc. IEEE ITS Conf '97*.

[25] Z. Kim, G. Gomes, R. Hranac and A. Skabardonis, "A Machine Vision System for Generating Vehicle Trajectories over Extended Freeway Segments", 12th World Congress on Intelligent Transportation Systems, 2005.

[26] A. H. S. Lai and N. H. C. Yung, "A video-based system methodology for detecting red light runners," in *Proc. IAPR Workshop on MVA '98*, pp. 23–26.

[27] Y. Cho and J. Rice. Estimating velocity fields on a freeway from low-resolution videos. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):463–469, Dec. 2006.

[28] D. Dailey, F.W. Cathy, and S. Pumrin. An algorithm to estimate mean traffic speed using uncalibrated cameras. *IEEE Transactions on Intelligent Transportation Systems*, 1(2):98–107, June 2000.

[29] Y.-K. Ki and D.-Y. Lee. A traffic accident recording and reporting model at intersections. *IEEE Transactions on Intelligent Transportation Systems*, 8(2):188–194, June 2007.

[30] T. N. Schoepflin and D. J. Dailey. Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation. *IEEE Transactions on Intelligent Transportation Systems*, 4(2):90–98, June 2003.

[31] C.Wern, A. Azarbayejani, T. Darrel, and A. Petland, "Pfinder: real-time tracking of human body," *IEEE Transactions on PAMI*, 19(7):780–785, July 1997.

[32] T. E. Boult, R. Micheals, X. Gao, P. Lewis, C. Power, W. Yin, and A. Erkan, "Frame-rate omnidirectional surveillance & tracking of camouflaged and occluded targets," in *Proceedings IEEE Workshop on Visual Surveillance*, 48–55, IEEE Computer Society, 1999.

[33] X. Gao, T. Boult, F. Coetzee, and V. Ramesh, "Error analysis of background adaption," in *Proceedings of IEEE conference Computer Vision and Pattern Recognition*, 503–510, IEEE Computer Society, 2000.

[34] I. Haritaoglu, D. Harwood, and L. Davis, "*W*4: Real-time surveillance of people and their activities," in *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):809–830, August 2000.

[35] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," in *Proceedings of IEEE Int'l Conf. on Computer Vision*, 255–261, IEEE Computer Society, 1999.

[36] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. CVPR 1999*, June 1999, pp. 246–252.

[37] A. Lipton, H. Fujiyoshi, and R. Patil, "Moving target classification and tracking from real-time video," in *Proceedings IEEE Workshop on Application of Computer Vision*, 8–14, IEEE Computer Society, 1998.

[38] A. A. Ambardekar, Masters Thesis titled "Efficient Vehicle Tracking and Classification for an Automated tracking Surveillance System", Department of Computer Science, University of Nevada Reno, December 2007.

[39] A. Elgammal, R. Duraiswami, D. Harwood, and L. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," in *Proceedings of the IEEE,* 90:1151–1163, 2002.

[40] A. Mittal and N. Paragios, "Motion-based background subtraction using adaptive kernel density estimation," in *Proceedings of CVPR*, 2:302–309, July 2004.

[41] K. Kim, D. Harwood, and L. S. Davis, “Background updating for visual surveillance,” in *Proceedings of the International Symposium on Visual Computing*, 1:337–346, Dec. 2005.

[42] A. Tavakkoli, M. Nicolescu, G. Bebis, "Robust Recursive Learning for Foreground Region Detection in Videos with Quasi-Stationary Backgrounds,” *Proceedings of the International Conference on Pattern Recognition*, 315-318, August 2006.

[43] A. Tavakkoli, M. Nicolescu, G. Bebis, "A Novelty Detection Approach for Foreground Region Detection in Videos with Quasi-stationary Backgrounds”, in *Proceedings of the 2nd International Symposium on Visual Computing*, Lake Tahoe, Nevada, 40-49, November 2006.

[44] J. P. Tarel, S.S. Ieng, and P. Charbonnier, “Using robust estimation algorithms for tracking explicit curves,” in *Proceedings of European Conference on Computer Vision*, 492–507, 2002.

[45] R. P. N. Rao, “Robust Kalman filters for prediction, recognition, and learning,” in *Technical Report 645*, University of Rochester, Computer Science, 1996.

[46] Y. Ivanov, A. Bobick, and J. Liu, “Fast lighting independent background subtraction,” in *International Journal of Computer Vision*, 37(2):199–207, 2000.

[47] G. Gordon, T. Darrell, M. Harville, and J.Woodfill, “Background estimation and removal based on range and color,” in *Proceedings of Computer Vision and Pattern Recognition*, 2:459–464, IEEE Computer Society, 1999.

[48] G.C. deSilva, Masters Thesis titled “Automation of Traffic Flow Measurement Using Video Images”, Department of Computer Science and Engineering, University of Moratuwa, February 2001.

[49] D. Koller, K. Danilidis, H. H. Nagel, Model-based Object Tracking in Monocular Image Sequences of Road Traffic Scenes, IJCV 10:3 257-281(1993), ©1993 Kluwer Publications, Netherlands.

[50] F. Dellart, D. Pomerlain, C. Thorpe, Model-based Car Tracking with a Road Follower, International Conference on Robotics and Automation, May, 1998.

[51] S. M. Smith, ASSET-2: Real-Time Motion Segmentation and Shape Tracking, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No. 8, August 1995.

[52] B. K. Hom, B. G. Schrunck, "Detemining optical flow," in *Artificial Intelligence,* 17:185-203, 1981.

[53] A. Bainhridge-Smith, R. G. Lane, "Deremining optical flow using a differential method," in *Image and Vision Computing*, 15:11-22, 1997.

[54] L. Wixson, "Detecting salient motion by accumulating directionally-consistent flow," in *Pattern Analysis and Machine Intelligence*, 22(8):774–780, 2000.

[55] F. Liu and R. Picard, "Finding periodicity in space and time," in *Proceedings of International Conference on Computer Vision*, 376–383, 1998.

[56] J. Batista, J. Dias, H. Araújo, A. Traça de Almeida, "Monoplanar Camera Calibration Iterative Multi-Step Approach," in *Proceedings of British Machine Vision Conference*, 479-488, 1993.

[57] O. Faugeras, "Three-Dimensional Computer Vision: a Geometric Viewpoint," *MIT Press*, 1993.

[58] S. J. Maybank and O. D. Faugeras, "A theory of self-calibration of a moving camera," in *International Journal of Computer Vision*, 8(2):123–152, 1992.

[59] http://gear.kku.ac.th/~nawapak/178353/Appendix1.ppt [last accessed: November 15th, 2010].

[60] D. C. Brown, "Close-range camera calibration," *Photogrammetric Engineering*, 37:855-866, 1971.

[61] W. Faig, "Calibration of close-range photogrammetry systems: Mathematical foundation," *Photogrammetric Engineering in Remote Sensing*, 41:1479-1486, 1975.

[62] D. B. Gennery, "Stereo-camera calibration," in *Proceedings of Image Understanding Workshop*, 101-108, 1979.

[63] E. L. Hall,, M. B. K. Tio, C. A. McPherson, and F. A. Sadjadi, "Curved surface measurement and recognition for robot vision," in *Conference Records of IEEE workshop on Industrial Applications of Machine Vision*, May 1982.

[64] S. Ganapaphy, "Decomposition of Transformation matrices for robot vision," in *Proceedings of International Conference on Robotics and Automation*, 130-139, 1984.

[65] T. M. Strat, "Recovering the camera parameters from a transformation matrix," in *Proceedings of DARPA Image Understanding Workshop*, 264-271, Oct. 1984.

[66] R. I. Hartley, "An algorithm for self calibration from several views," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 908–912, June 1994.

[67] Q.-T. Luong and O. Faugeras, "Self-calibration of a moving camera from point correspondences and fundamental matrices," in *The International Journal of Computer Vision*, 22(3):261–289, 1997.

[68] A. Worrall and G. Sullivan and K. Baker, "A simple intuitive camera calibration tool for natural images," in *Proceedings of British Machine Vision Conference*, 781-790, 1994.

[69] Ling-Ling Wang and Wen-Hsiang Tsai, "Camera Calibration by Vanishing Lines for 3D Computer Vision," in *Transactions on Pattern Analysis and Machine Vision*, 13(4):370-376, 1991.

[70] E. K. Bas and J. D. Crisman. An easy to install camera calibration for traffic monitoring. In *Proceedings of the IEEE Conference on Intelligent Transportation Systems*, pages 362–366, 1997.

[71] H. S. Lai. *Vehicle extraction and modeling, an effective methodology for visual traffic surveillance*. Thesis published at The University of Hong Kong, 2000.

[72] G. S. K. Fung, N. H. C. Yung, and G. K. H. Pang. Camera calibration from road lane markings. *Optical Engineering*, 42:2967–2977, Oct. 2003.

[73] D. Dailey, F.W. Cathy, and S. Pumrin. An algorithm to estimate mean traffic speed using uncalibrated cameras. *IEEE Transactions on Intelligent Transportation Systems*, 1(2):98–107, June 2000.

[74] T. N. Schoepflin and D. J. Dailey. Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation. *IEEE Transactions on Intelligent Transportation Systems*, 4(2):90–98, June 2003.

[75] Z. Zhang, M. Li, K. Huang, and T. Tan. Practical camera auto-calibration based on object appearance and motion for traffic scene visual surveillance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[76] S. Gupte, O. Masoud, R. F. K. Martin, and N. P. Papanikolopoulos, “Detection and Classification of Vehicles,” in *IEEE Transactions on Intelligent Transportation Systems*, 3(1):37-47, 2002.

[77] D. Magee, "Tracking multiple vehicles using foreground, background and motion models," In *Proceedings of ECCV Workshop on Statistical Methods in Video Processing*, 2002.

[78] J. Malik and S. Russell, “Traffic Surveillance and Detection Technology Development: New Traffic Sensor Technology,” *California PATH Research Final Report*, University of California, Berkeley, UCB-ITS-PRR-97-6, 1997.

[79] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell, “Toward robust automatic traffic scene analysis in real-time,” in *Proceedings of International Conference on Pattern Recognition*, 126–131, 1994.

[80] D. Koller, K Dandilis, and H. H. Nagel, "Model based object tracking in monocular image sequences of road traffic scenes," in *International Journal of Computer Vision*, 10(3):257–281, 1993.

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

[82] J. M. Ferryman, A. D. Worrall, and S. J. Maybank, "Learning enhanced 3D models for vehicle tracking," in *British Machine Vision Conference*, 873–882, 1998.

[83] C. Schlosser, J. Reitberger, and S. Hinz, “Automatic car detection in high resolution urban scenes based on an adaptive 3D-model,” In *EEE/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas*, 98–107, 2003.

[84] Weiming Hu; Xuejuan Xiao; Dan Xie; Tieniu Tan; , "Traffic accident prediction using vehicle tracking and trajectory analysis," *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE* , vol.1, no., pp. 220- 225 vol.1, 12-15 Oct. 2003,doi: 10.1109/ITSC.2003.1251952.

[85] T. N. Tan, G. D. Sullivan, and K. D. Baker, “Model-based localization and recognition of road vehicles,” in *International Journal of Computer Vision*, 27(1):5–25, 1998.

[86] T. N. Tan and K. D. Baker, “Efficient image gradient based vehicle localization,” in *IEEE Transactions on Image Processing*, 9:1343–1356, Aug. 2000.

[87] A. E. C. Pece and A. D.Worrall, “Tracking without feature detection,” in *Proceedings of International Workshop on Performance Evaluation of Tracking and Surveillance*, 29-37, 2000.

[88] S. Kamijo, K. Ikeuchi, and M. Sakauchi, "Vehicle tracking in low-angle and front view images based on spatio-temporal markov random fields," in *Proceedings of the 8th World Congress on Intelligent Transportation Systems*, 2001.

[89] N. K. Kanhere, S. T. Birchfield, and W. A. Sarasua, "Vehicle Segmentation and Tracking in the Presence of Occlusions," in *Transportation Research Board Annual Meeting*, 2006.

[90] Lili Huang; Barth, M.; , "Real-time multi-vehicle tracking based on feature detection and color probability model," *Intelligent Vehicles Symposium (IV), 2010 IEEE* , vol., no., pp.981-986, 21-24 June 2010, doi: 10.1109/IVS.2010.5548060

[91] Goo Jun; Aggarwal, J.K.; Gokmen, M.; , "Tracking and Segmentation of Highway Vehicles in Cluttered and Crowded Scenes," *Applications of Computer Vision, 2008. WACV 2008. IEEE Workshop on* , vol., no., pp.1-6, 7-9 Jan. 2008, doi: 10.1109/WACV.2008.4544017

[92] Luo Di; Huang Xiangnian; , "Texture Analysis for Shadow Removing and Tracking of Vehicle in Traffic Monitoring System," *Intelligent Information Technology Application Workshops, 2008. IITAW '08. International Symposium on* , vol., no., pp.863-866, 21-22 Dec. 2008.

[93] Jae-Young Choi; Kyung-Sang Sung; Young-Kyu Yang; , "Multiple Vehicles Detection and Tracking based on Scale-Invariant Feature Transform," *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE* , vol., no., pp.528-533, Sept. 30 2007-Oct. 3 2007.

[94] Chachich, A. A. Pau, A. Barber, K. Kennedy, E. Oleiniczak, J. Hackney, Q. Sun, E. Mireles, "Traffic sensor using a color vision method," in *Proceedings of the International Society for Optical Engineering*, 2902:156-164, 1997.

[95] Z. Sun, G. Bebis, R. Miller, "Improving the Performance of On-Road Vehicle Detection by Combining Gabor and Wavelet Features," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, 2002.

[96] X. Song and R. Nevatia, "Detection and tracking of moving vehicles in crowded scenes," IEEE Workshop on Motion and Video Computing, vol., no., pp. 4-4, February 2007.

[97] Roya Rad and Mansour Jamzad. 2005. Real time classification and tracking of multiple vehicles in highways. *Pattern Recogn. Lett.* 26, 10 (July 2005), 1597-1607. DOI=10.1016/j.patrec.2005.01.010

[98] H. Ikeda, T. Matsuo, Y. Kaneko, and K. Tsuji, "Abnormal incident detection system employing image processing technology," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Tokyo, Japan, Oct. 1999, pp. 748 752.

[99] M. Kimachi, K. Kanayama, and K. Teramoto, "Incident prediction by fuzzy image sequence analysis," in *Proc. IEEE Int. Conf. VNIS*, 1994, pp. 51–57.

[100] M. M. Trivedi, I. Mikic, and G. Kogut, "Distributed video networks for incident detection and management," in *Proc. IEEE Intell. Transp. Syst. Conf.*, 2000, pp. 155–160.

[101] J. Blosseville, J. Morin, and P. Lochegnies, “Video image processing application: Automatic incident detection freeways,” in *Proc. Pacific Rim Transp. Technol. Conf.*, Jul. 25–28, 1993, pp. 69–76.

[102] J. Versavel and B. Boucke, “Sparing lives and saving time: A unified approach to automatic incident detection,” in *Proc. Int. Symp. Automotive Technol. and Autom.*, Dublin, Ireland, Sep. 25–27, 2000, pp. 205–209.

[103] P. Michalopoulos and R. Jacobson, “Field implementation and testing of machine vision based incident detection system,” *Transp. Res. Rec.: J. Transp. Res. Board*, no. 1394, pp. 1–7, 1993.

[104] Atev, S.; Arumugam, H.; Masoud, O.; Janardan, R.; Papanikolopoulos, N.P.; , "A vision-based approach to collision prediction at traffic intersections," *Intelligent Transportation Systems, IEEE Transactions on* , vol.6, no.4, pp. 416- 423, Dec. 2005, doi: 10.1109/TITS.2005.858786

[105] Weiming Hu; Xuejuan Xiao; Dan Xie; Tieniu Tan; , "Traffic accident prediction using vehicle tracking and trajectory analysis," *Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE* , vol.1, no., pp. 220- 225 vol.1, 12-15 Oct. 2003, doi: 10.1109/ITSC.2003.1251952

[106] Yong-Kul Ki; Dong-Young Lee; , "A Traffic Accident Recording and Reporting Model at Intersections," *Intelligent Transportation Systems, IEEE Transactions on* , vol.8, no.2, pp.188-194, June 2007, doi: 10.1109/TITS.2006.890070

[107] Kamijo, S.; Matsushita, Y.; Ikeuchi, K.; Sakauchi, M.; , "Traffic monitoring and accident detection at intersections," *Intelligent Transportation Systems, 1999. Proceedings. 1999*

IEEE/IEEJ/JSAI International Conference on, vol., no., pp.703-708, 1999, doi: 10.1109/ITSC.1999.821147.

[108] N.K. Kanhere, Doctor of Philosophy Dissertation titled "Vision-Based Detection, tracking and Classification of Vehicles using Stable Features with Automatic Camera Calibration", Department of Electrical Engineering, Clemson University, August 2008.

[109] http://www.cse.unr.edu/~bebis/CS791E/Notes/CameraParameters.pdf [last accessed: November 20th, 2010].

[110] R. J. Schalkoff. Digital Image Processing and Computer Vision. John Wiley & Sons, Inc., first edition, 1989.

[111] R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, second edition, 2003.

[112] http://www.vision.caltech.edu/bouguetj/calib_doc/index.html#links [last accessed: November 20th, 2010].

[113] E.C. Larson and D.M. Chandler, "Most apparent distortion: A dual strategy for full reference image quality assessment," *Journal of Electronic Imaging* **19**(1), 011006 (Jan–Mar **2010**).

[114] Salim, F.D.; Seng Wai Loke; Rakotonirainy, A.; Srinivasan, B.; Krishnaswamy, S.; , "Collision Pattern Modeling and Real-Time Collision Detection at Road Intersections," *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE* , vol., no., pp.161-166, Sept. 30 2007-Oct. 3 2007, doi: 10.1109/ITSC.2007.4357693.

[115] Kanhere, N.K.; Birchfield, S.T.; , "Real-Time Incremental Segmentation and Tracking of Vehicles at Low Camera Angles Using Stable Features," *Intelligent Transportation Systems, IEEE Transactions on* , vol.9, no.1, pp.148-160, March 2008, doi: 10.1109/TITS.2007.911357.

[116] Bing-Fei Wu; Shin-Ping Lin; Yuan-Hsin Chen; , "A real-time multiple-vehicle detection and tracking system with prior occlusion detection and resolution," *Signal Processing and Information Technology, 2005. Proceedings of the Fifth IEEE International Symposium on*, vol., no., pp.311-316, 21-21 Dec. 2005.

[117] Althoff, M.; Stursberg, O.; Buss, M.; , "Model-Based Probabilistic Collision Detection in Autonomous Driving," *Intelligent Transportation Systems, IEEE Transactions on* , vol.10, no.2, pp.299-310, June 2009.

[118] Yuexian ZOU, Guangyi SHI, Hang SHI, and Yiyan WANG, “Image sequences based traffic incident detection for signaled intersections using HMM,” IEEE International Conference on Hybrid Intelligent Systems 2009, HIS 2009, August 12-14th 2009, Shenyang, China.

VITA

Logesh Vasu

Candidate for the Degree of

Master of Science

Thesis: AN EFFECTIVE STEP TO REAL-TIME IMPLEMENTATION OF ACCIDENT DETECTION SYSTEM USING IMAGE PROCESSING

Major Field: Electrical Engineering

Biographical:

Education:

Completed the requirements for the Master of Science in Electrical Engineering at Oklahoma State University, Stillwater, Oklahoma in December, 2010.

Experience:

Spring 2009: Teaching Assistant for Network Analysis Laboratory
Fall 2010: Research Assistant at CPIQ

Professional Memberships:

Student Member of IEEE

Name: Logesh Vasu Date of Degree: December, 2010

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: AN EFFECTIVE STEP TO REAL-TIME IMPLEMENTATION OF ACCIDENT DETECTION SYSTEM USING IMAGE PROCESSING

Pages in Study: 130 Candidate for the Degree of Master of Science

Major Field: Electrical Engineering

Scope and Method of Study: Real-Time Implementation

Findings and Conclusions:

Studies in the past have shown that number of traffic related fatalities is highly dependent on the emergency response time after the occurrence of an accident. Also traffic intersections were found to be one of the most vulnerable places for occurrence of an accident. Therefore there is a need to reduce the emergency response time by alerting the emergency response team by an automated accident detection system at traffic intersections, once an accident is detected.

The goal of this project is develop an accident detection system at traffic intersections that is capable of operating in real-time with good performance rate. Therefore an accident detection system was developed which uses the vehicle parameters such as the speed and trajectory and other features such as area, orientation and position of the vehicle. Since one of the key elements in accident detection step is accurate tracking of moving vehicles, more focus was given to vehicle detection and tracking step. In this work, a tracking algorithm that uses a weighted combination of low-level features extracted from moving vehicles and low-level vision analysis on vehicle regions extracted from different frames is implemented.

The speed of the tracked vehicles are calculated and along with the features extracted from the tracked vehicle, an accident detection system is designed which validates the factors cueing the occurrence of an accident. Once an accident is detected, the user is signaled about the occurrence of an accident.

The detection and tracking performance of the algorithm was around 90% for two test videos used and collision detection system produced a correct detection rate of 87.5% for the test crashes simulated in the test bed setup in the laboratory. Overall the algorithm shows promise since it has a processing rate of 5frames/sec with good collision detection performance. With more test crashes and real-crashes data training, the performance of the algorithm is expected to improve.

ADVISER'S APPROVAL: Dr. Damon Chandler