8-2008

Vision-based Detection, Tracking and Classification of Vehicles using Stable Features with Automatic Camera Calibration

Neeraj Kanhere
Clemson University, nkanher@clemson.edu

Vision-based Detection, Tracking and Classification of Vehicles using Stable Features with Automatic Camera Calibration

---

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Electrical Engineering

---

by
Neeraj Krantiveer Kanhere
Aug 2008

Accepted by:
Dr. Stanley Birchfield, Committee Chair
Dr. John Gowdy
Dr. Robert Schalkoff
Dr. Wayne Sarasua

Abstract

A method is presented for segmenting and tracking vehicles on highways using a camera that is relatively low to the ground. At such low angles, 3D perspective effects cause significant appearance changes over time, as well as severe occlusions by vehicles in neighboring lanes. Traditional approaches to occlusion reasoning assume that the vehicles initially appear well-separated in the image, but in our sequences it is not uncommon for vehicles to enter the scene partially occluded and remain so throughout. By utilizing a 3D perspective mapping from the scene to the image, along with a plumb line projection, a subset of features is identified whose 3D coordinates can be accurately estimated. These features are then grouped to yield the number and locations of the vehicles, and standard feature tracking is used to maintain the locations of the vehicles over time. Additional features are then assigned to these groups and used to classify vehicles as cars or trucks. The technique uses a single grayscale camera beside the road, processes image frames incrementally, works in real time, and produces vehicle counts with over 90% accuracy on challenging sequences. Adverse weather conditions are handled by augmenting feature tracking with a boosted cascade vehicle detector (BCVD). To overcome the need of manual camera calibration, an algorithm is presented which uses BCVD to calibrate the camera automatically without relying on any scene-specific image features such as road lane markings.

Dedication

I dedicate this work to my parents who have struggled hard in life to give me the best.

Acknowledgments

I am most grateful to my adviser, Dr. Stanley Birchfield for guiding me at every step of this endeavor. The freedom he gave me to pursue new ideas made this work a truly enjoyable learning experience. I would also like to thank Dr. Robert Schalkoff for his introductory course in image processing. Things I learned in his course have helped me at every step of this research. Thanks to Dr. John Gowdy for valuable suggestions and encouragement. Many thanks to Dr. Wayne Sarasua for providing the practical insights into the traffic engineering aspect of the research. Facilitating equipment, travel, and plenty of data are just a few of the many ways in which his help was invaluable.

I would like to thank the James Clyburn Transportation Center at South Carolina State University and the Dwight David Eisenhower Transportation Fellowship Program for supporting this research.

Special thanks Shrinivas and Nikhil for being such helpful lab mates and cherished friends. Finally, I would like to thank my family and my wife Uma for their immense love and patience.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Traffic data such as vehicle counts, speeds, and classification are important in traffic engineering applications, transportation planning, and Intelligent Transportation Systems (ITS). Collecting traffic data manually by direct observations of human observers has a number of drawbacks [4] including high cost, extreme weather and difficulties imposed by staffing limitations. These data can be acquired automatically using one of the many available sensor technologies summarized in Table 1.

While in-road technologies such as inductive loop detectors offer good accuracy for counts and presence detection, their installation and maintenance causes traffic disruption. Sensors that are placed on the pavements (magnetometers, road tubes) can be damaged by snow removal equipment or street sweepers. As mentioned in [4] at times it is difficult to obtain accurate counts using intrusive technologies due to roadway geometry (e.g., geometry where there are significant lane changes or where vehicles do not follow a set path in making turns). Some of the non-intrusive roadside sensors might be prohibitive due to high cost (e.g., laser) or low precision (e.g., microwave). Infrared sensors have an advantage of day/night operation and perform better than visible wavelength sensors in fog. However, in addition to the problem of unstable detection zones, for reliable operation at least one sen-

sor is required in each traffic lane (a notable exception is the TIRTL sensor [1]). Ultrasonic sensors exhibit difficulty in detecting snow-covered vehicles and are sensitive to changes in ambient temperature and humidity. In addition, the problem of detecting motorcycles remains elusive for the sensors described above.

The output of these sensors is a poor description of the traffic events. This is a serious limitation in case of a critical situation, where a human operator is required to make a decision based on the sensor data. In such cases, video sensors provide the information in the form of live video of the scene. In addition, a single video sensor placed at an appropriate position provides wide area coverage making it possible to detect incidents in multiple lanes simultaneously. The same is the case in calculating queue lengths. Another advantage of video is that it provides sufficient information for vehicle tracking to be feasible, which is useful for detecting events such as sudden lane changes, vehicles moving in the wrong direction, stalled vehicles etc.

1.1 Video detection and vision-based tracking

The use of video image processing for traffic monitoring was initiated in the mid 1970s in the United States and abroad, most notably in Japan, France, Australia, England, and Belgium [50]. The hardware and the algorithms used for estimating traffic parameters have seen a great improvement over the years. All video detection systems used for traffic monitoring can be broadly classified in two categories: 1) Systems which rely on localized incident detections, and 2) Systems which track individual vehicles. The advantage of the first is that the computational requirements are quite low, and algorithms are relatively simple. In the case of vehicle tracking systems, sophisticated algorithms are needed and are usually computationally demanding. Vehicle tracking systems offer more accurate estimation of microscopic traffic parameters like lane changes, erratic motion etc. By the

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
Magnetometer	• 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

Table 1.1: Performance comparison among existing incident detection technologies [47].

late 1980s, video-detection systems for traffic surveillance generated sufficient interest to warrant research to determine their viability as an inductive loop replacement [51]. At present, there are a number of commercial systems being used throughout U.S. for manual as well as automatic traffic monitoring and incident detection. Majority of these systems use localized detection zones for counting vehicles. Once the detection zones are marked on the image, the pixel values in each detection zone is monitored for a change over time. Combining this simple technique with some heuristics gives accurate vehicle counts in favorable conditions (camera placement high above the ground, head-on view, free flowing traffic, clear weather and absence of shadows).

In case of non-ideal camera placement, spillover (due to camera perspective, the image of a tall vehicle spills over into neighboring lanes) results into false detections. Figure 1.1 illustrates an examples of this problem where a large vehicle wrongfully triggers multiple detection zones in a popular commercial system. Another instance where such a simple approach fails is in the case of shadows. As shown in Figure 1.2 shadow of a car triggers the detection zone and is counted as a vehicle in another commercial system (Iteris vantage). In case of a busy intersection, such a false alarms (especially in left-turn lanes) would have an adverse effect on the signal timing coordination.

Such errors can be avoided by expanding the goal of the system to detect and track vehicles over time as opposed to local change-detection methods (simple image processing techniques). In addition vehicle tracking makes it possible to detect traffic events such as near crashes and hazardous driving patterns. With availability of powerful and low-cost computing resources, using computer vision for detection and tracking of vehicles is now feasible for practical applications.

Figure 1.1: An example where large vehicles trigger multiple detection zones resulting in over counting. The output is from a popular commercial system (Autoscope). It should be noted that the commercial system is not designed to handle such a situation and it produces good results when the camera is placed high above the ground (40 feet or higher) with sufficient tilt angle.

Figure 1.2: Shadow of a car incorrectly triggers a detection in neighboring lane.

1.2 Previous work

Tracking vehicles using computer vision has been an interesting topic of research [6, 46, 13, 15, 44, 12, 22, 65, 43, 31, 8, 35, 34, 36, 37]. Number of different approaches have been proposed in the past, each having its own advantages and shortcomings. Approaches which assume that objects to be tracked (vehicles) have already been initialized are not considered in the following discussions, since such systems can not be used in automatic traffic analysis. Techniques used for vehicle detection and tracking can be classified into following popular approaches:

Background subtraction: Background subtraction is a popular technique used by many vehicle-tracking systems to detect and track vehicles when they are well-separated in the image [6, 46, 13, 15, 44, 12]. Many advancements have been made in recent years in adapting the background image to lighting changes [12, 22, 30, 65] and in reducing the effects of shadows [28, 38]. A well-known challenge for background subtraction (as well as with the closely-related approach of frame differencing [17, 58, 39, 48, 14]) occurs when vehicles overlap in the image, causing them to merge into a single foreground blob. Koller et al. [43] use 2D splines to solve this occlusion problem, while other researchers employ graph association or split-and-merge rules to handle partial or complete occlusions [22, 48, 49, 30]. Although these solutions can disambiguate vehicles after an occlusion occurs, they require the vehicle to either enter the scene unoccluded or to become unoccluded at some point during its trajectory in the camera field of view. In congested traffic, such may never be the case.

Active contours: A closely related approach to blob tracking is based on tracking active contours (popularly knows as *snakes*) representing an object's boundary. Vehicle tracking

using active contour models has been reported in [43]. Contour tracked is guided by intensity and motion boundaries. A contour is initialized for a vehicle using a background difference image. Tracking is achieved using two Kalman filters, one for estimating the affine motion parameters, and the other for estimating the shape of the contour. An explicit occlusion detection step is performed by intersecting the depth ordered regions associated to the objects. The intersection is excluded in the shape and motion estimation. Results are shown on real world sequences without shadows or severe occlusions. The algorithm is limited to tracking cars.

Wireframe models: An alternative to using temporal information is to match wireframe models to video images [70, 42, 62, 23]. Ferryman et al. [19] combine a 3D wireframe model with an intensity model of a vehicle to learn the appearance of the vehicle over time. Kim and Malik [41] match vehicle models with line features from mosaic images captured from cameras on top of a 30-story building next to the freeway in order to recover detailed trajectories of the vehicles. Alessandretti et al. [5] employ a simpler model, namely the 2D symmetry of the appearance of a vehicle in an image. One of the major drawbacks to model-based tracking is the large number of models needed due to differing vehicle shapes and camera poses.

Markov random field: An algorithm for segmenting and tracking vehicles in low angle frontal sequences has been proposed in [31]. In their work, the image is divided into 8×8 pixel blocks, and a spatiotemporal Markov random field (ST-MRF) is used to update an object map using the current and previous image. Motion vectors for each block are calculated, and the object map is determined by minimizing a functional combining the number of overlapping pixels, the amount of texture correlation, and the neighborhood proximity. The algorithm does not yield 3D information about vehicle trajectories in the world coor-

dinate system, and to achieve accurate results it is run on the sequence in reverse so that vehicles recede from the camera. The authors found that the low-angle scenario is indeed a challenging problem, although the accuracy of their results increased two folds, when they processed the sequence in reverse.

Color and pattern: Chachich et al. [11] use color signatures in quantized RGB space for tracking vehicles. In this work, vehicle detections are associated with each other by combining color information with driver behavior characteristics and arrival likelihood. In addition to tracking vehicles from a stationary camera, a pattern recognition-based approach to on-road vehicle detection has been studied in [67]. The camera is placed inside a vehicle looking straight ahead, and vehicle detection is treated as a pattern classification problem using support vector machines (SVMs).

Feature points: A third alternative that has been employed is the tracking of point features. Beymer et al. [8] describe a system that tracks features throughout the video sequence, then groups the features according to motion cues in order to segment the vehicles. Because the camera is high above the ground, a single homography is sufficient to map the image coordinates of the features to the road plane, where the distances between pairs of features and their velocities are compared. In another approach, Saunier et al. [55] use feature points to track vehicles through short-term occlusions, such as poles or trees. Like the background subtraction systems mentioned above, their approach has difficulty initializing and tracking partially occluded vehicles. Recently Kim [40] proposed an approach of combining background subtraction with dynamic multi-level feature grouping for tracking vehicles. However, grouping parameters are computed using semi-supervised learning which needs manual intervention.

All of this previous work applies to cameras that are relatively high above the

ground. At such heights, the problems of occlusion and vehicle overlap are mitigated, thus making the problem easier. One exception to this rule is the work of Kamijo et al. [32], in which a spatiotemporal Markov random field is used to update an object map using the current and previous images. Motion vectors for each image region are calculated, and the object map is determined by minimizing a functional combining the number of overlapping pixels, the amount of texture correlation, and the neighborhood proximity. To achieve accurate results, the algorithm is run on the image sequence in reverse so that vehicles recede from the camera. Extending the work of Beymer et al. [8] to the case of low-angle cameras, a simple but effective technique is introduced for estimating the 3D coordinates of features in an incremental fashion. The contribution of this research is an effective combination of background subtraction and feature tracking to handle occlusions, even when vehicles remain occluded during their entire visible trajectory. Unlike their work, the approach presented in this dissertation handles features that cannot be tracked continually throughout the trajectory, which is a common occurrence in dense traffic conditions.

1.3 Calibration of traffic monitoring cameras

Camera calibration is an essential step in such systems to measure speeds, and it often improves the accuracy of tracking techniques for obtaining vehicles counts as well. Typically, calibration is performed by hand, or at least semi-automatically. For example, an algorithm for interactive calibration of a Pan-Tilt-Zoom (PTZ) camera has been proposed in [64]. Bas and Crisman [7] use the known height and the tilt angle of the camera for calibration using a single set of parallel lines (along the road edges) drawn by the user, while Lai [45] removes the restriction of known height and tilt angle by using an additional line of known length perpendicular to the road edges. The technique of Fung et al. [21], which uses the pavement markings and known lane width, is robust against small perturbations in

the markings, but it requires the user to draw a rectangle formed by parallel lane markings in adjacent lanes. The problem of ill-conditioned vanishing points (i.e., parallel lines in the world appearing parallel in the image) has been addressed by He et al. [26] using known length and width of road lane markings. Additional techniques for manual camera calibration are described in [22, 8].

Recently the alternative of automatic camera calibration has gained some attention. Automatic calibration would not only reduce the tediousness of installing fixed cameras, but it would also enable the use of PTZ cameras without manually recalibrating whenever the camera moves. Dailey et al. [17] 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. Sequential image frames are subtracted, and vehicles are tracked by matching the centroids of the resulting blobs. At low camera heights, the resulting spillover and occlusion cause blobs to be merged, which renders such tracking ineffective. In follow-up research, Schoepflin and Dailey [58] dynamically calibrate PTZ cameras using lane activity maps which are computed by frame-differencing. As noted in their paper, spillover is a serious problem for moderate to large pan angles, and this error only increases with low camera heights. During experiments it was found that estimating lanes using activity maps is impossible with pan angles as small as 10° when the camera is placed 20 feet above the ground, due to the large amount of spillover and occlusion that occur due to tall vehicles. In an alternate approach, Song et al. [60] use edge detection to find the lane markings in the static background image, from which the vanishing point is estimated by assuming that the camera height and lane width are known in advance. The method requires the lane markings to be visible, which may not be true under poor lighting or weather conditions. In addition, estimating the static background is not always possible when the traffic is dense, it requires time to acquire a good background image, and background subtraction does not work well at low camera heights due to occlusion and spillover, as noted above. More

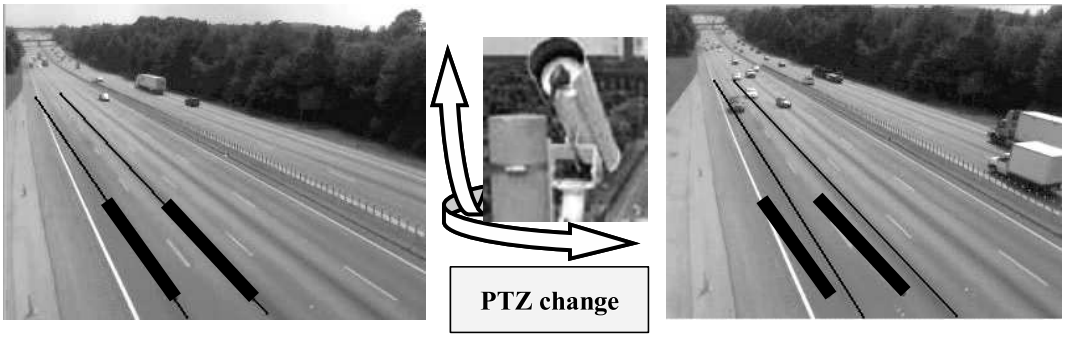


Figure 1.3: Left: Operator sets up detection zones (thick lines) along the lane centers (long thin lines) to count vehicles and measure speeds. Right: Even after a small PTZ movement of the camera, the detection zones are no longer along the lane centers.

recently Zhang et al. [69] presented an approach using three vanishing points to estimate the calibration parameters. However, their approach relies on the presence of sufficient vertical structures or pedestrians in the scene to recover the vanishing point perpendicular to the road plane.

A system for automatic calibration of road-side traffic monitoring cameras will be presented that overcomes several of the limitations mentioned above. The approach does not require pavement markings or prior knowledge of the camera height or lane width; it is unaffected by spillover, occlusion, and shadows; and it works in dense traffic and different lighting and weather conditions.

1.4 Outline

The outline for the rest of the dissertation is as follows. A vehicle detection, tracking and classification system based on feature tracking is discussed in Chapter 2. This work has been published in the IEEE Transactions on Intelligent Transportation Systems [35]. Chapter 3 focuses on the recent efforts to augment the feature tracking based vehicle detection with pattern recognition. Camera calibration is an essential step for tracking and

measuring speeds of vehicles. Different techniques of calibrating a camera from the video are presented in Chapter 4 and finally an algorithm to calibrate the camera automatically using a pattern detector is presented in Chapter 5. The work on automatic calibration has been presented at the 87th annual meeting of the Transportation Research Board (TRB) [33].

Chapter 2

Detection and tracking of vehicles using feature points

A system for detection, tracking and classification of vehicles based on feature point tracking is presented in this chapter. An overview of the system is shown in Figure 2.1. Feature points are automatically detected and tracked through the video sequence, and features lying on the background or on shadows are removed by background subtraction, leaving only features on the moving vehicles. These features are then separated into two categories: stable and unstable. Using a plumb line projection (PLP), the 3D coordinates of the stable features are computed, these stable features are grouped together to provide a segmentation of the vehicles, and the unstable features are then assigned to these groups. The final step involves eliminating groups that do not appear to be vehicles, establishing correspondence between groups detected in different image frames to achieve long-term tracking, and classifying vehicles based upon the number of unstable features in the group. The details of these steps are described in the following subsections.

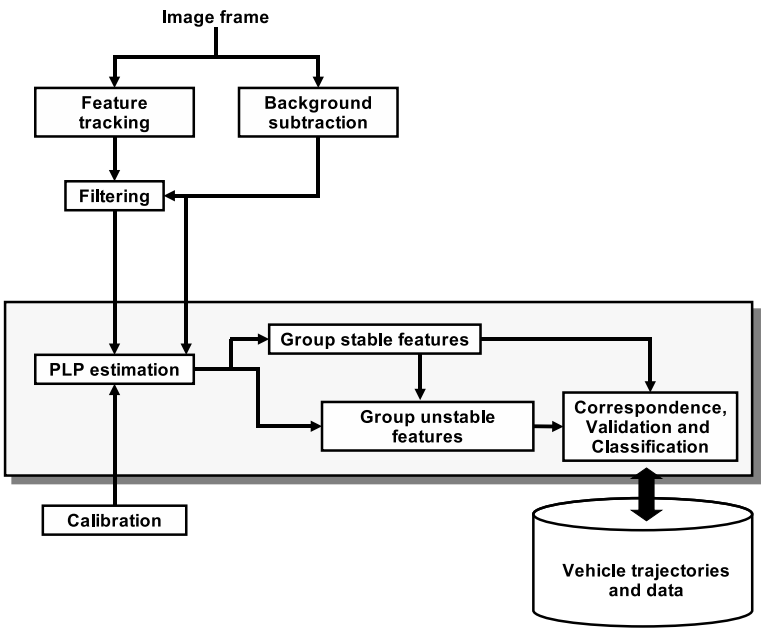


Figure 2.1: Overview of the system for detection and tracking of vehicles using stable features.

2.1 Algorithm description

2.1.1 Calibration

According to a pinhole camera model, a world point $\mathbf{p} = [\,x \quad y \quad z\,]^T$ projects onto a point $\mathbf{u} = [\,u \quad v\,]^T$ on an image plane through the equation

$$\breve{\mathbf{u}} = C\breve{\mathbf{p}}, \tag{2.1}$$

where C is a 3×4 camera calibration matrix, and $\breve{\mathbf{u}} = [\,uw \quad vw \quad w\,]^T$ and $\breve{\mathbf{p}} = [\,x \quad y \quad z \quad 1\,]^T$ are homogeneous coordinates of the image and world points, respectively [24]. Since w is an arbitrary nonzero scale factor, C has 11 unique parameters. Thus, the correspondence of at least six points in a non-degenerate configuration leads to an overdetermined system that can be solved for these parameters.

To calibrate the system, the user manually draws two lines along the edges of the

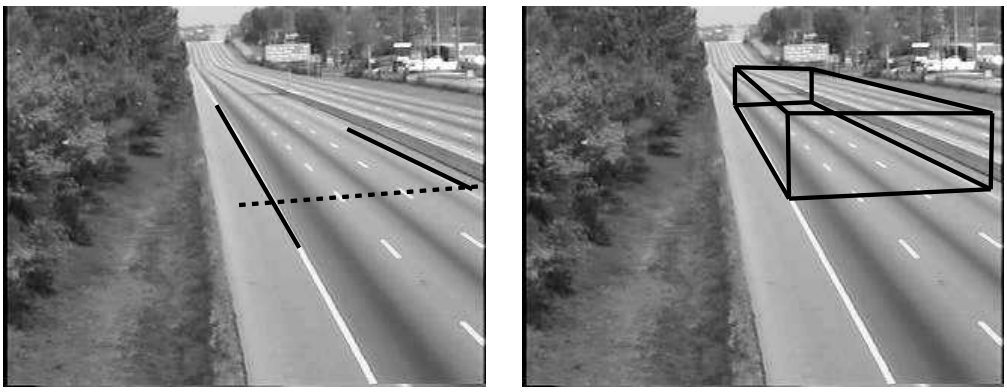


Figure 2.2: Manual camera calibration. LEFT: The user draws three lines, two along the edges of the road (solid) and one perpendicular to the direction of travel (dashed). The lines can be of arbitrary length. RIGHT: The 3D tracking zone is automatically computed.

road and one line perpendicular to the direction of travel, as shown in Figure 2.2. The latter line is estimated by sequencing through the video and finding the intensity edge between the windshield and hood of a light-colored vehicle. These three lines yield two vanishing points, from which the internal and external camera parameters are computed automatically using the mathematical formulation described in chapter 4. The remaining six vertices of the cuboid defining the 3D tracking zone are then computed from the user-specified lane width, number of lanes, and desired length and height of the cuboid. For the world coordinate system, y-axis points along the direction of travel along the road, the z-axis is perpendicular to the road plane with the positive axis pointing upward and $z = 0$ on the road surface, and the x-axis is chosen to form a right-hand coordinate system.

Because the overall system is insensitive to small inaccuracies in the calibration (quantified in Section 2.2), this process is widely applicable to prerecorded sequences captured from unknown cameras. Note that the calibration procedure recovers a full 3D to 2D perspective mapping, which is necessary to handle the perspective effects encountered at low camera angles, unlike previous 2D to 2D calibration tools that recover only a planar mapping between the road surface and image plane [8]. Also note that perspective projec-

tion leads to more robust results than the multi-layer homography used in [37], due to the reduced number of free parameters.

2.1.2 Background subtraction

The background of the scene is learned by storing the average gray level of each pixel over a fixed period of time. For the experimental sequences, 20 seconds of video was found to be sufficient for this task, but a higher traffic density would require proportionally more time to adequately remove the effects of the dynamic foreground objects. Since this learning is performed only once, it is applicable to any stretch of road for which the traffic is moderately dense for some period of time.

Once the background is learned off-line, the technique of background subtraction, including morphological operations and thresholding, is applied to each image of the sequence to yield a binary foreground mask that indicates whether each pixel is foreground or background. To cope with lighting and environmental changes, the background is adaptively updated as the sequence is processed, using this mask to preclude inadvertently adapting to foreground intensities [22]. One of the serious problems in using background subtraction for object tracking is the distraction caused by moving shadows, which mistakenly appear as foreground pixels. It is not uncommon for shadows to cause multiple nearby vehicles to merge into a single blob, or for the shadows to be detected as separate vehicles themselves. Although the problem of shadow detection has been addressed by many researchers, a general solution remains elusive [53, 27, 54, 16, 29, 52, 61, 63].

Background subtraction is used to perform a simple filtering operation on the features, as shown in Figure 2.3. Any feature that lies in the background region is immediately discarded from further processing, leaving only the features that lie on foreground objects. To reduce the effects of shadows, any feature that lies within a small distance τ_s from a

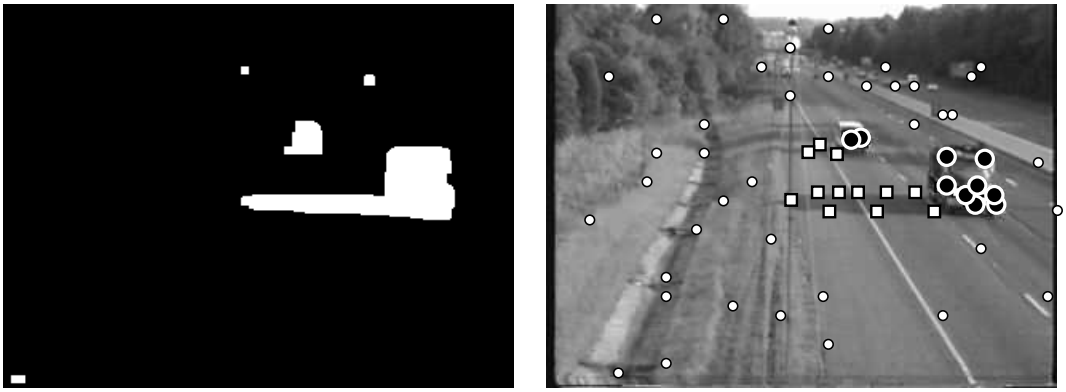


Figure 2.3: LEFT: The foreground mask resulting from background subtraction. RIGHT: The features being tracked in this frame of video, divided into three kinds: (1) those that lie on the background (shown as small dots), (2) those that lie within τ_s pixels of the background (shown as small squares), and (3) those on moving vehicles (shown as large circles). Only the latter features are considered in further processing, thus reducing the potential distraction from the background or shadows.

background pixel is ignored. ($\tau_s = 2$ pixels in all experiments.) This simple procedure removes many of the features due shadow edges alone, since the road surface tends to be fairly untextured, while removing only a small fraction of legitimate foreground features.

2.1.3 Plumb line projections

Feature points are automatically selected and tracked using the Lucas-Kanade feature tracker [59]. The OpenCV implementation of the feature tracker which uses the Sharr gradient operator [10] was used for all the experiments. A coarse-to-fine pyramidal strategy allow for large image motions, and features are automatically selected, tracked, and replaced.

Because of the dimension loss in projecting the 3D world to a 2D image, it is impossible to uniquely determine the coordinates of the corresponding world point from the image coordinates of a feature point. However, if one of the world coordinates is known from some additional source of information, then the other two coordinates can be com-

puted. In this section a method is presented for exploiting this capability.

Suppose we have a feature point $\mathbf{u}$ and a binary foreground mask F from background subtraction, as shown in Figure 2.4. Projecting $\mathbf{u}$ downward in the image plane to the first encountered background pixel yields the point $\mathbf{v}$ that we call the *plumb line projection (PLP)* of $\mathbf{u}$. Let $\mathbf{v} = \psi_F(\mathbf{u})$ denote this transformation. In addition, let $\mathbf{p} = \Phi(\mathbf{u})$ denote the preimage of $\mathbf{u}$ (i.e., the world point whose projection onto the image is $\mathbf{u}$), and let $\mathbf{q} = \Phi(\mathbf{v})$ be the preimage of $\mathbf{v}$. Under certain assumptions whose validity we shall examine in a moment, $\mathbf{p}$ and $\mathbf{q}$ have the same x and y coordinates as each other, and $\mathbf{q}$ lies on the road surface, thus providing us with the constraints that we need to compute the world coordinates of $\mathbf{p}$.

Let $\varphi_z : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ be the mapping from a 2D image point to its corresponding world point at height z. In other words, an image point $\mathbf{u}$ could arise from any world point along the projection ray passing through $\mathbf{u}$ and the camera focal point, and $\mathbf{p} = \varphi_z(\mathbf{u})$ is the one whose third coordinate is z. Expanding and rearranging (2.1) yields the inhomogeneous equation:

$$\varphi_z(\mathbf{u}) = K^{-1}(\mathbf{u})\mathbf{t}_z(\mathbf{u}), \tag{2.2}$$

where

$$K(\mathbf{u}) = \begin{bmatrix} c_{31}u - c_{11} & c_{32}u - c_{12} & 0 \\ c_{31}v - c_{21} & c_{32}v - c_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$\mathbf{t}_z(\mathbf{u}) = \begin{bmatrix} c_{14} - u + z(c_{13} - c_{33}u) \\ c_{24} - v + z(c_{23} - c_{33}v) \\ z \end{bmatrix}, \tag{2.4}$$

$\mathbf{u} = [\,u \quad v\,]^T$ is the projection of $\mathbf{p}$, and c_{ij} is the ij th element of C. (See Appendix A for the derivation.)

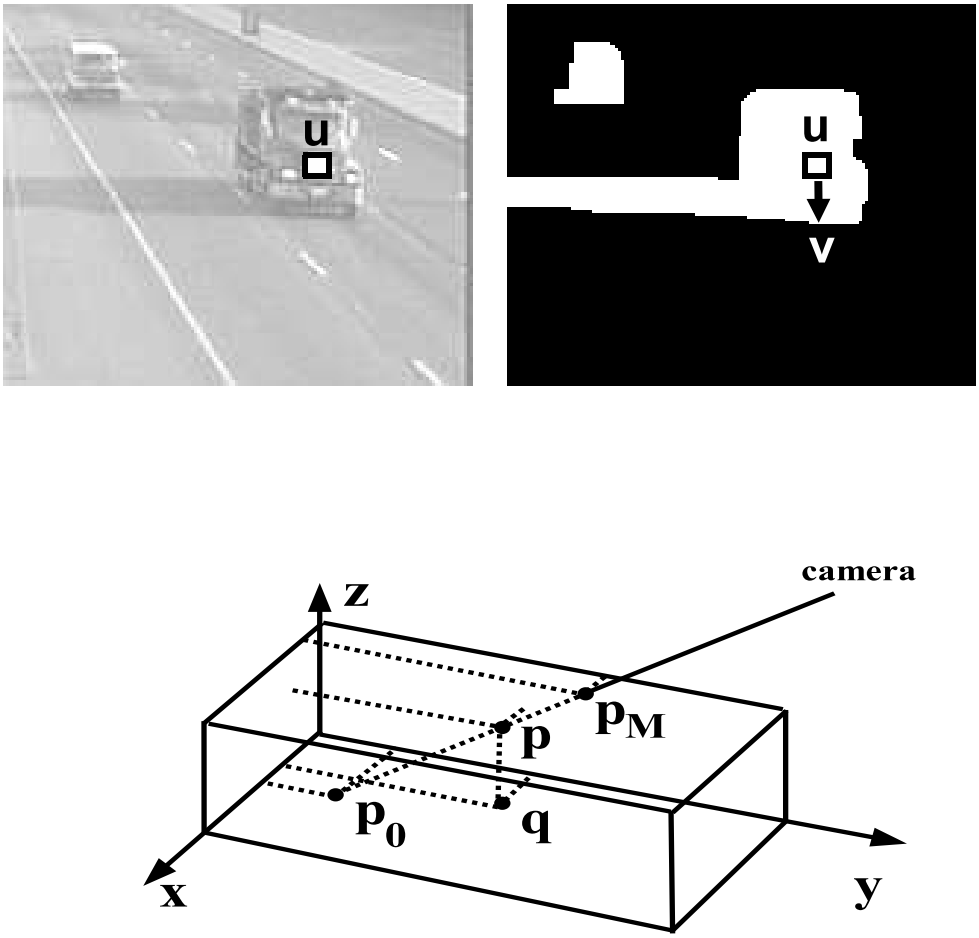


Figure 2.4: TOP: An image (left) and foreground mask F (right) with a feature point $\mathbf{u}$ and its PLP $\mathbf{v} = \psi_F(\mathbf{u})$. BOTTOM: The 3D coordinates of the preimage $\mathbf{p} = \Phi(\mathbf{u})$ of the feature can be computed under the assumption that $\mathbf{q} = \Phi(\mathbf{v})$ lies directly below $\mathbf{p}$ on the surface of the road. The points $\mathbf{p}_0$ and $\mathbf{p}_M$ are the intersections of the projection ray with the top and bottom of the calibration box.

Since the world coordinate system is oriented so that $z = 0$ is the road plane, we can compute the world coordinates of $\mathbf{q}$ as $\varphi_0(\mathbf{v})$, which also yields the x and y coordinates of $\mathbf{p}$. To compute the 3D coordinates of $\mathbf{p}$, then, all we need is to compute its z coordinate, which is done by solving (2.1) in a least squares manner:

$$\tilde{z} = \frac{\mathbf{h}_p^T \mathbf{h}_c}{\mathbf{h}_p^T \mathbf{h}_p}, \tag{2.5}$$

where

$$\mathbf{h}_p = \begin{bmatrix} u\,c_{33} - c_{13} \\ v\,c_{33} - c_{23} \end{bmatrix}$$

$$\mathbf{h}_c = \begin{bmatrix} c_{14} - u\,c_{34} + (c_{11} - u\,c_{31})\,x + (c_{12} - u\,c_{32})\,y \\ c_{24} - v\,c_{34} + (c_{21} - v\,c_{31})\,x + (c_{22} - v\,c_{32})\,y \end{bmatrix},$$

and x and y are the first two coordinates of $\mathbf{p}$ and $\mathbf{q}$. $\tilde{z}$ denotes the estimated height of $\mathbf{p}$.

2.1.4 Identifying and grouping stable features

The technique just presented for computing the 3D coordinates of the preimage of a feature point $\mathbf{u}$ from its plumb line projection relies upon three assumptions: (1) the world points $\mathbf{p} = \Phi(\mathbf{u})$ and $\mathbf{q} = \Phi(\mathbf{v})$ lie on the same vertical axis, (2) the zth coordinate of $\mathbf{q}$ is zero, and (3) the foreground mask F perfectly labels the pixels directly under $\mathbf{u}$ (in the image). In other words, the method assumes that the vehicle is shaped like a box, that the features lie on one of the four surfaces of the box orthogonal to the road plane, and that there are no occluding vehicles or shadows in the vicinity. Let us now examine the validity of these assumptions.

Figure 2.5 shows the side view of a vehicle with three feature points $\mathbf{s}$, $\mathbf{t}$, and $\mathbf{u}$ having preimages $\mathbf{S}$, $\mathbf{T}$, and $\mathbf{U}$, respectively, on the surface of the vehicle. Suppose the third

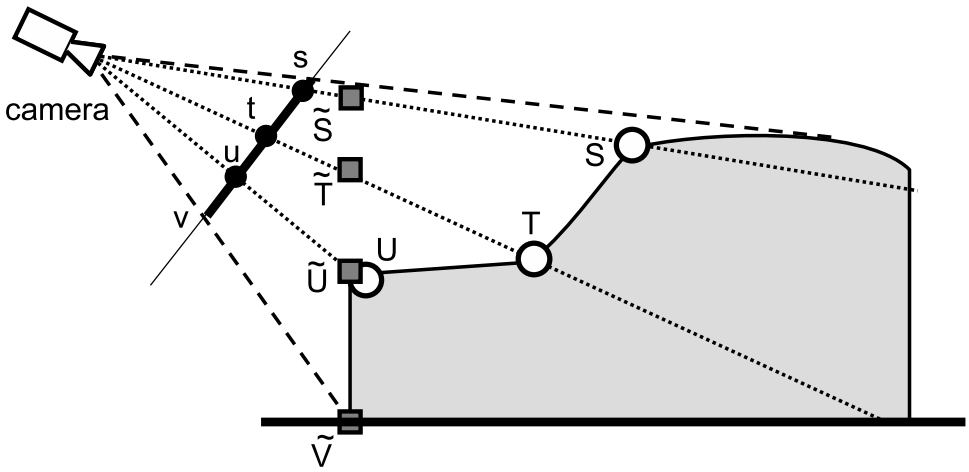


Figure 2.5: Three points on the surface of a vehicle viewed by a camera, with their estimated coordinates using PLP. The points lower to the ground yield less error.

assumption is satisfied, so that $\mathbf{v} = \psi_F(\mathbf{s}) = \psi_F(\mathbf{t}) = \psi_F(\mathbf{u})$, i.e., all three points share the same PLP, and the estimated point $\tilde{\mathbf{V}} = \varphi_0(\mathbf{v})$ is the actual point $\mathbf{V}$. Using the coordinates $\tilde{\mathbf{V}}$, the technique previously described can be used to estimate the world coordinates $\tilde{\mathbf{S}}$, $\tilde{\mathbf{T}}$, and $\tilde{\mathbf{U}}$. From the figure, it is evident that the error in prediction of world coordinates is generally greater for points that are higher above the road plane. More precisely, let us define Ω as the set of vehicle shapes such that the slope of the contour at any point never exceeds the bound $\mu_{\max}(x, z)$ (see Appendix for the derivation). Then we have the following observation:

Observation 1 *For any two points $\boldsymbol{S} = (x_S, y_S, z_S)$ and $\boldsymbol{U} = (x_U, y_U, z_U)$ on the surface of a vehicle such that $z_S > z_U$, the Euclidean error in the estimate $\tilde{\boldsymbol{S}}$ will not be less than that of $\tilde{\boldsymbol{U}}$, i.e., $\| \tilde{\boldsymbol{S}} - \boldsymbol{S} \| \geq \| \tilde{\boldsymbol{U}} - \boldsymbol{U} \|$, as long as the vehicle shape is in Ω.*

Thus, the Euclidean error in estimating the world coordinates of a point on the vehicle is a monotonically non-decreasing function of the height of the point. Keep in mind that the set Ω encompasses nearly all actual vehicle shapes, so that this observation is widely applicable. Only a vehicle with a severe concavity would be outside the set Ω.

Another important observation regards the effect of the height of the estimates on the maximum possible error:

Observation 2 *For any two estimated points* $\tilde{\boldsymbol{S}} = (\tilde{x}_S, \tilde{y}_S, \tilde{z}_S)$ *and* $\tilde{\boldsymbol{U}} = (\tilde{x}_U, \tilde{y}_U, \tilde{z}_U)$ *such that* $z_S > z_U$*, the maximum possible Euclidean error in the estimate* $\tilde{\boldsymbol{S}}$ *is greater than that of* $\tilde{\boldsymbol{U}}$*, i.e.,* $\max \| \tilde{\boldsymbol{S}} - \boldsymbol{S} \| > \max \| \tilde{\boldsymbol{U}} - \boldsymbol{U} \|$.

To see the validity of this observation, notice from Figure 2.5 that the estimated height $\tilde{z}$ of a point will always be greater than or equal to its actual height (as long as the point does not extend past the front of the vehicle). Now consider two vehicles traveling side by side as shown in Figure 2.6, where the camera in 3D is aimed toward the front of the vehicles at an oblique angle. Let $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{U}}$ be the 3D estimates of two preimages using the PLP procedure, with $\tilde{\mathbf{S}}$ higher above the road than $\tilde{\mathbf{U}}$. Using the upper bound $z_{true} \leq \tilde{z}$, the range of possible locations for the actual preimage is much less for the point lower to the ground, i.e., the maximum possible error e_u is less than the maximum possible error e_s. In the example shown, even the maximum error would not cause the estimate point $\tilde{\mathbf{U}}$ to leave the vehicle, whereas with $\tilde{\mathbf{S}}$ the point could be assigned to the wrong vehicle. Both observations lead to the conclusion that points close to the road plane generally exhibit less error.

In addition to the height of a feature, it is also important to consider the side of the vehicle on which the feature lies. For each feature $\mathbf{u} = [u \quad v]^T$, the PLP of the two points obtained by perturbing the feature horizontally in the image plane is computed (See Figure 2.7): $\mathbf{u}^+ = \psi_F([u+\delta \quad v]^T)$ and $\mathbf{u}^- = \psi_F([u-\delta \quad v]^T)$. The 3D coordinates of the preimages are given by $\mathbf{p}_u^+ = [x^+, y^+, z^+] = \varphi_0(\mathbf{u}^+)$ and $\mathbf{p}_u^- = [x^-, y^-, z^-] = \varphi_0(\mathbf{u}^-)$. If the absolute value of the slope in the road plane $\xi = | (y^+ - y^-)/(x^+ - x^-) |$ is small, then the point is more likely to be on the front of the vehicle rather than the side. Since the shadows on the side tend to be more severe than those on the front, the points on the front

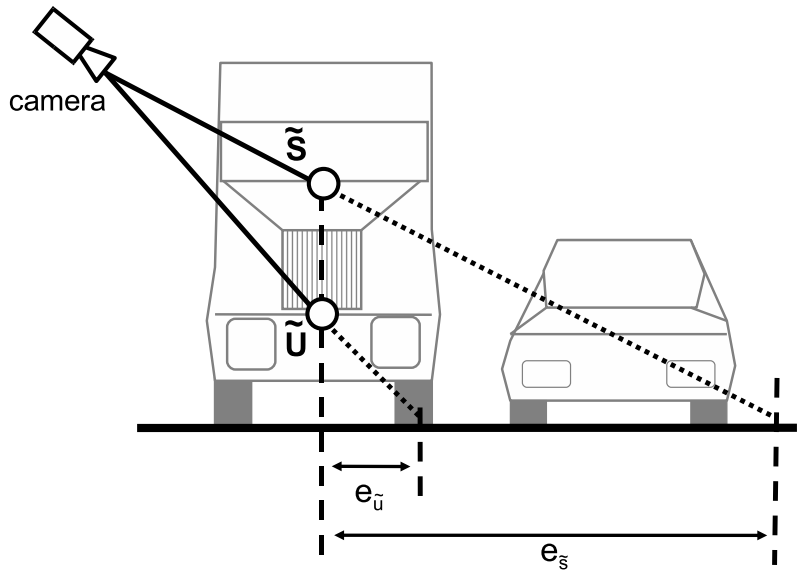


Figure 2.6: Estimated coordinates of two points using PLP. Because the estimated height is nearly always greater than the true height, the higher feature is more likely to be assigned to the wrong vehicle.

are less likely to violate the third assumption and hence are more reliable.

Putting this analysis together, two kinds of features are obtained, namely, *stable* and *unstable*. A feature point **u** is classified as stable if it satisfies the following two conditions:

$$\tilde{z} < \epsilon_z \quad \text{and} \quad \xi < \epsilon_{slope},$$

where ϵ_z and ϵ_{slope} are positive, constant parameters of the system. In other words, features are stable if they lie on the frontal face of the vehicle close to the road plane. Note that these criteria require only a single image frame, are robust with respect to shadows on the side of the vehicle, and are not affected by errors in feature tracking, unlike the criteria used in [37].

Once the stable features have been identified, they are grouped in the road plane (*xy*-plane) as shown in Figure 2.8. Because of the criteria used in selecting stable features, points belonging to the same vehicle generally have a small deviation in their world coor-

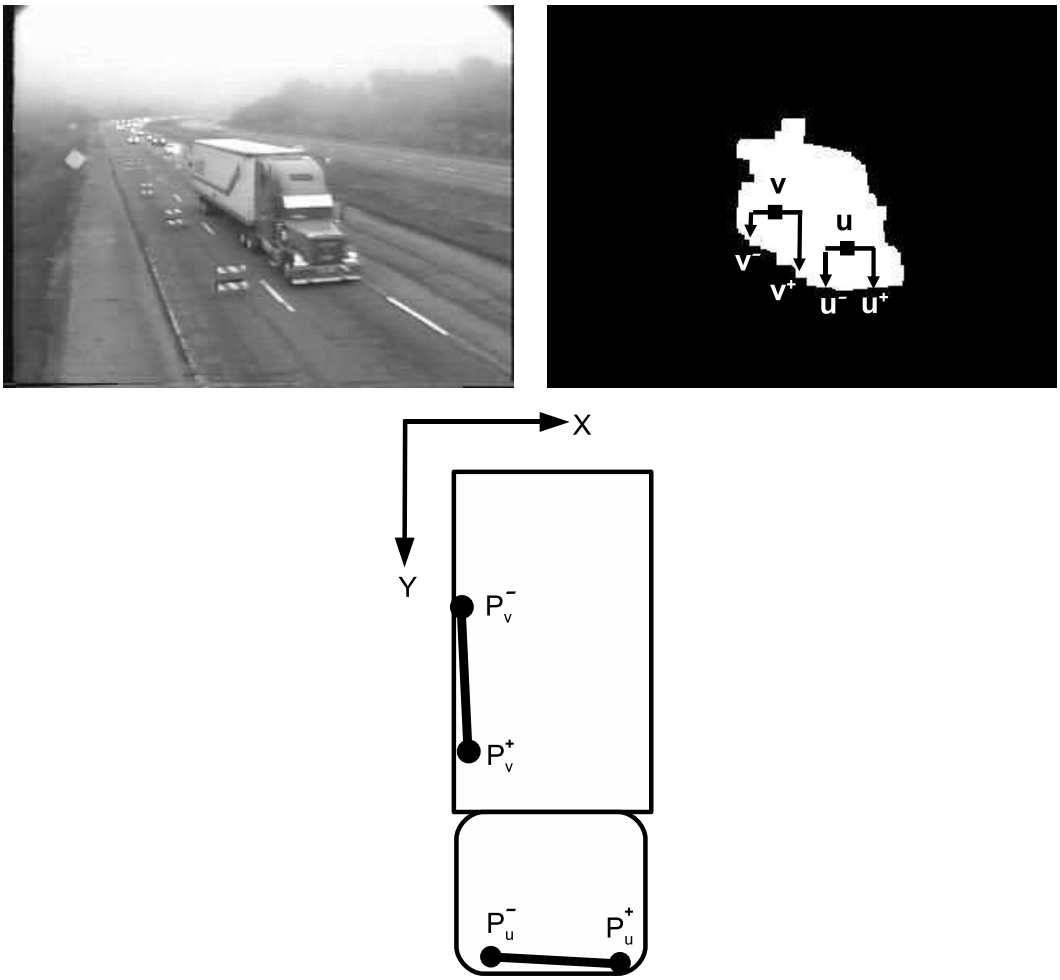
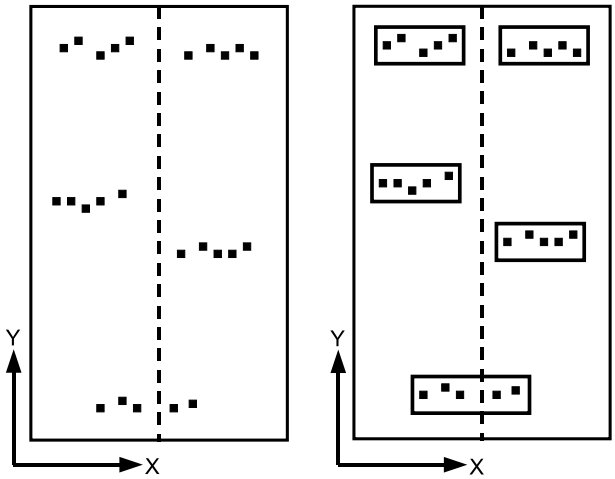


Figure 2.7: TOP: An image (left) and foreground mask (right), with two unrelated feature points (**u** and **v**) and the PLPs ($\mathbf{u}^+$, $\mathbf{u}^-$, $\mathbf{v}^+$, and $\mathbf{v}^-$) of their perturbations. BOTTOM: Points on the front of the vehicle yield a smaller slope in the road plane than points on the side of the vehicle.



(a) (b)

Figure 2.8: Stable features are grouped in the road plane using a region growing algorithm that compares their *y* coordinates.

dinates along the *y*-axis (axis along the length of the road). As a result, a simple region growing algorithm is sufficient to correctly segment the stable features.

The procedure iterates through the points, adding each point to an existing group in the same lane if its predicted *y*-coordinate is within ϵ_y of the mean of the *y*-coordinates of all the features in the group. If no such group is found, then a new group is created. To handle vehicles that straddle two lanes (such as vehicles that are changing lanes), two groups whose means in *y* differ by no more than ϵ_y are combined into a single group if their combined width (along the *x*-axis) is no more than the lane width w_{lane}.

This approach is much more computationally efficient and less sensitive to tracking errors than the technique used in [37], and it operates on a single image frame which facilitates incremental processing of the video. It should be noted that only one stable feature per vehicle is needed in order for the vehicle to be correctly detected, although in practice groups with fewer than three features are discarded to reduce the number of spurious false detections. $\epsilon_y = \epsilon_z = 0.4w_{lane}$, $\epsilon_{slope} = 1.5$, and $\delta = 3$ pixels for all experiments, where w_{lane} is the width of a lane computed during the calibration step.

2.1.5 Grouping unstable features

After grouping the stable features, the unstable features are assigned to these groups using a combination of PLP and motion coherence. Suppose we have two features that are tracked from locations $\mathbf{u}$ and $\mathbf{s}$ in one image frame to $\mathbf{u}'$ and $\mathbf{s}'$ in another (not necessarily consecutive) image frame. Let $\mathbf{p}_z = \varphi_z(\mathbf{u})$ and $\mathbf{q}_z = \varphi_z(\mathbf{s})$ denote their possible preimages in the first frame at height z, and let $\mathbf{p}'_z = \varphi_z(\mathbf{u}')$ and $\mathbf{q}'_z = \varphi_z(\mathbf{s}')$ denote their possible preimages in the other frame. If $\mathbf{s}$ is a stable feature, then we know the coordinates of the preimages $\mathbf{q} = \Phi(\mathbf{s})$ and $\mathbf{q}' = \Phi(\mathbf{s}')$, which can then be used to estimate the preimages $\mathbf{p} = \Phi(\mathbf{u})$ and $\mathbf{p}' = \Phi(\mathbf{u}')$ in the following manner.

The scenario is shown in Figure 2.9, with $z = 0$ the road plane and $z = M$ the top of the calibration box. If we assume that $\mathbf{p}$ and $\mathbf{q}$ are points on the same rigid vehicle that is only translating, then the motion vectors of the two points are the same: $\mathbf{p}'-\mathbf{p} = \mathbf{q}'-\mathbf{q}$. This is the motion coherence assumption. Now each point can be represented parametrically as follows:

$$\begin{aligned} \mathbf{p} &= \mathbf{p}_0 + \alpha(\mathbf{p}_M - \mathbf{p}_0) \\ \mathbf{p}' &= \mathbf{p}'_0 + \alpha'(\mathbf{p}'_M - \mathbf{p}'_0), \end{aligned} \tag{2.6}$$

where $\alpha, \alpha' \in \mathbb{R}$ are the fractional distances along the ray. If we further assume that the road is horizontally flat, then the z component of $\mathbf{p}$ and $\mathbf{p}'$ are equal, from which it can easily be shown that $\alpha = \alpha'$. Substituting these parametric equations into $\mathbf{p}' - \mathbf{p} = \mathbf{q}' - \mathbf{q}$ and solving for α in a least squares manner yields

$$\alpha = \frac{(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)^T(\Delta\mathbf{q} - \Delta\mathbf{p}_0)}{(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)^T(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)}, \tag{2.7}$$

where $\Delta\mathbf{p}_M = \mathbf{p}'_M - \mathbf{p}_M$, $\Delta\mathbf{p}_0 = \mathbf{p}'_0 - \mathbf{p}_0$, and $\Delta\mathbf{q} = \mathbf{q}' - \mathbf{q}$. As a result, the estimated point

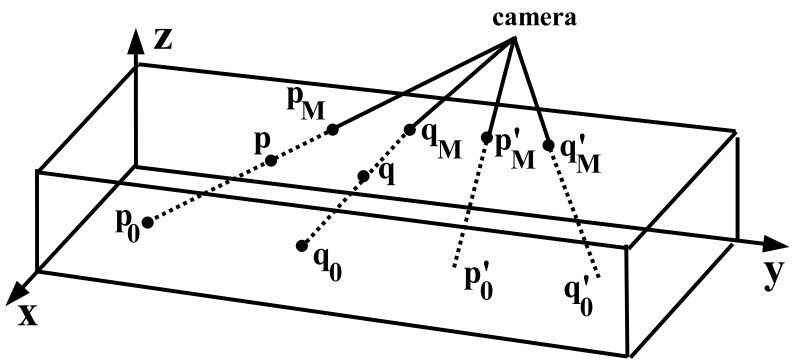


Figure 2.9: Features $\mathbf{p}$ and $\mathbf{q}$ on a vehicle travel to $\mathbf{p}'$ and $\mathbf{q}'$ at a different time. If the vehicle travels parallel to the road plane, then the coordinates of the unstable feature $\mathbf{p}$ can be computed from the coordinates of the stable feature $\mathbf{q}$.

is given by

$$\hat{\mathbf{p}} = \mathbf{p}_0 + \frac{(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)^T(\Delta\mathbf{q} - \Delta\mathbf{p}_0)}{(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)^T(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)}(\mathbf{p}_M - \mathbf{p}_0) \tag{2.8}$$

and similarly for $\mathbf{p}'$. All of the quantities on the right hand side are known, since $\mathbf{p}_0 = \varphi_0(\mathbf{u})$ and $\mathbf{p}_M = \varphi_M(\mathbf{u})$.

Let $\mathbf{q}^i = [\,x_q^i \quad y_q^i \quad z_q^i\,]^T$ be the coordinates of the centroid of the stable features in group i. For each unstable feature $\mathbf{p}$ the above procedure is used to estimate the world coordinates of its preimage with respect to group i by assuming motion coherence with $\mathbf{q}^i$ to yield $\hat{\mathbf{p}}^i = [\,\hat{x}_p^i \quad \hat{y}_p^i \quad \hat{z}_p^i\,]^T$. In addition, the world coordinates are estimated using the PLP procedure described in Section 2.1.4 to yield $\tilde{\mathbf{p}} = [\,\tilde{x}_p \quad \tilde{y}_p \quad \tilde{z}_p\,]^T$. Using these estimates, and assuming conditional independence along the different dimensions, a score is then computed indicating whether $\mathbf{p}$ belongs to group i:

$$\mathcal{L}_{\mathbf{p}}^i = \mathcal{L}_x^i \mathcal{L}_y^i \mathcal{L}_z^i \mathcal{L}_\ell^i \mathcal{L}_h^i, \tag{2.9}$$

where

$$\mathcal{L}_x^i = \exp\left[-(x_q^i - \hat{x}_p^i)^2/\sigma_x^2\right] \tag{2.10}$$

$$\mathcal{L}_y^i = \begin{cases} \exp\left[-(y_q^i - \hat{y}_p^i)^2/\sigma_y^2\right] & \text{if} \quad \hat{y}_p^i > y_q^i \\ \exp\left[-(\hat{y}_p^i - y_q^i + \lambda_\ell)^2/\sigma_y^2\right] & \text{if} \quad \hat{y}_p^i < (y_q^i - \lambda_\ell) \\ 1 & \text{otherwise} \end{cases} \tag{2.11}$$

$$\mathcal{L}_z^i = \begin{cases} \exp\left[-(\hat{z}_p^i)^2/\sigma_z^2\right] & \text{if} \quad \hat{z}_p^i < 0 \\ \exp\left[-(\tilde{z}_p - \hat{z}_p^i)^2/\sigma_z^2\right] & \text{if} \quad \hat{z}_p^i > \tilde{z}_p \\ 1 & \text{otherwise} \end{cases} \tag{2.12}$$

$$\mathcal{L}_\ell^i = \exp\left[-(1 - \ell^i)^2/\sigma_\ell^2\right] \tag{2.13}$$

$$\mathcal{L}_h^i = \exp\left[-(1 - h^i)^2/\sigma_h^2\right] \tag{2.14}$$

The first three factors compute a modified Mahalanobis distance from the estimated coordinates to the centroid of the ith vehicle. $\mathcal{L}_x^i$ favors features which lie close to the centroid along the x-axis. Since the stable features generally lie on the front of the vehicle, $\mathcal{L}_y^i$ assumes that the vehicle occupies a portion of the road between $y = y_q^i$ and $y = y_q^i - \lambda_\ell$, where λ_ℓ is the minimum truck length and the positive y axis points in the direction of traffic flow. Points outside this region are compared with the nearest edge. In the vertical direction, the vehicle is assumed to occupy the space between $z = 0$ and $z = \tilde{z}_p$, based upon the upper bound of z_{true} mentioned in Section 2.1.4.

The last two factors increase the score of larger vehicles, ignoring the actual point $\mathbf{p}$. Three points are considered: the centroid $\mathbf{q}^i = [\,x_q^i \quad y_q^i \quad z_q^i\,]^T$ of the stable features in the group, and two points shifted from the centroid along the y and z axes, $\mathbf{q}_\ell^i = [\,x_q^i \quad y_q^i - \lambda_\ell \quad z_q^i\,]^T$ and $\mathbf{q}_h^i = [\,x_q^i \quad y_q^i \quad z_q^i + \lambda_h\,]^T$. The values λ_ℓ and λ_h are the minimum length and height for a vehicle to be considered a truck. Let the projections of these points onto the image be denoted by $\mathbf{u}^i$, $\mathbf{u}_\ell^i$, and $\mathbf{u}_h^i$, respectively. Let the fraction of pixels

along a straight line between $\mathbf{u}^i$ and $\mathbf{u}^i_\ell$ that are foreground pixels (in the foreground mask) be ℓ^i, and let the same fraction along the line between $\mathbf{u}^i$ and $\mathbf{u}^i_h$ be h^i, so that $0 \leq \ell^i, h^i \leq 1$. In other words, ℓ^i and h^i indicate the fractional length and height of the vehicle compared with the minimum truck length and height, respectively. As a result, the factors $\mathcal{L}^i_\ell$ and $\mathcal{L}^i_h$ encourage features that are high off the ground (i.e., unstable features) to be grouped with larger vehicles (i.e., those with large values of ℓ^i and h^i).

Let a and b be the groups that yield the highest and second highest values, respectively, for the score of this feature. Then the feature is assigned to group a if $\mathcal{L}^a > \mathcal{L}_{min}$ and $\mathcal{L}^a/\mathcal{L}^b > \mathcal{L}_{ratio}$. In other words, these conditions assign an unstable feature to a stable group if the feature is likely to belong to that group (controlled by $\mathcal{L}_{min}$) and at the same time unlikely to belong to other groups (controlled by $\mathcal{L}_{ratio}$). $\sigma_x = \sigma_y = \sigma_z = 5$ feet, $\sigma_\ell = \sigma_h = 0.1$ pixels, $\lambda_\ell = 1.2w_{lane}$, $\lambda_h = 0.8w_{lane}$, $\mathcal{L}_{min} = 0.8$, and $\mathcal{L}_{ratio} = 2$ for all experiments.

2.1.6 Correspondence, validation and classification

The correspondence between the feature groups segmented in the current frame and the vehicles (i.e., feature groups) already being tracked is established by computing the number of stable features shared between the groups. Each vehicle is matched with the segmented feature groups in the current frame and is associated with the group having the maximum number of stable features in common. If a vehicle has no features in common with any of the groups, then its status is updated as “missing”, and its location in subsequent frames is updated using its current velocity. For each vehicle a count is kept of the total number of frames that it was tracked successfully (η_t) and the number of recent consecutive frames that it has been missing (η_m).

After finding a match for all non-missing vehicles, the remaining unassociated fea-

ture groups in the current frame are matched with the missing vehicles based on the closest Euclidean distance between the centroids of the groups in world coordinates. Each missing vehicle is associated, one at a time, with the closest feature group if that group is within a distance of τ_x and τ_y in the x and y axes, respectively. Then the remaining unassociated feature groups in the current frame are initialized as new vehicles.

When a vehicle exits the tracking zone, it is discarded if it has not been tracked for a sufficient number of frames, i.e., $\eta_t < \tau_\eta$. This can be viewed as a simplified temporal filtering to remove spurious and fragmented vehicle detections. In addition, a vehicle is discarded if $\eta_m > \kappa\eta_t$, where $\kappa \geq 0$, which is important to prevent momentary false detections from being retained.

To classify a vehicle as a car or truck, (for the experiments, a car is defined as a vehicle with two axles, and a truck as a vehicle with more than two axles). the number of unstable features associated with that vehicle over all the frames that the vehicle is tracked is summed. Vehicles with more than n_{truck} unstable features are classified as trucks, while the rest are considered cars. Only unstable features are used because they are rarely associated with cars due to their low height, whereas the number of stable features for cars and trucks tends to be about the same. The number of unstable features associated with trucks is usually much greater than that of cars (typically five to ten times higher). $\tau_x = 0.3w_{lane}$, $\tau_y = 0.5w_{lane}$, $\tau_\eta = 4$, $\kappa = 2$, and $n_{truck} = 20$ for all experiments.

2.2 Experimental Results

The system presented in this chapter was tested on eleven grayscale video sequences captured by a 30 Hz camera placed on an approximately nine meter pole on the side of the road and digitized at 320×240 resolution. No additional preprocessing was performed to suppress shadows or to stabilize the occasional camera jitter. For each sequence, an

initial calibration step was used to provide an approximate mapping between 2D image coordinates and 3D world coordinates, as described in Section 2.1.1. After the calibration, the system was fully automatic, outputting the lane counts, vehicle trajectories, and vehicle classification (car/truck) in real time.

To convey the variety of conditions in the processed videos, sample image frames from the sequences are shown in Figure 2.10. As can be seen, these sequences differ by the camera placement, field of view, direction of traffic flow, variations in lighting conditions (including long shadows), curved roads, scale and angle changes, and number of lanes. The "long" sequences L1-L7 are 10 minutes each (18,000 image frames), while the "short" sequences S8 and S9 are approximately 30 seconds each (900 image frames). Sequences S1 and S4 were extracted from the same video from which L1 and L4, respectively, were extracted, with no overlap in image frames between the short and long versions. Due to lack of space, S9 is not shown in the figure but closely resembles S8 in terms of road shape, number of lanes, and camera angle. As mentioned earlier, the same parameter values were used in processing all the sequences.

A quantitative assessment of the algorithm's performance on these sequences is presented in Table 2.1. The segmentation and tracking performance exceeded 90% on all the sequences, and the classification accuracy was more than 95%. The false positive rate exhibited variation, ranging from 1% to 7% of the total vehicles in all the sequences except S9, where long shadows caused the rate to reach 12%. The lower detection rate in the L3 sequence is due to the vehicles receding from the camera, which reduces the number of features successfully detected and tracked because of the relatively low texture on the rear of the vehicles.

Figures 2.11 through 2.13 show the results of the algorithm on some example image frames from the sequences, with the images slightly brightened to increase the contrast of the annotations. Overlaid on each image are all the features (stable and unstable) of that

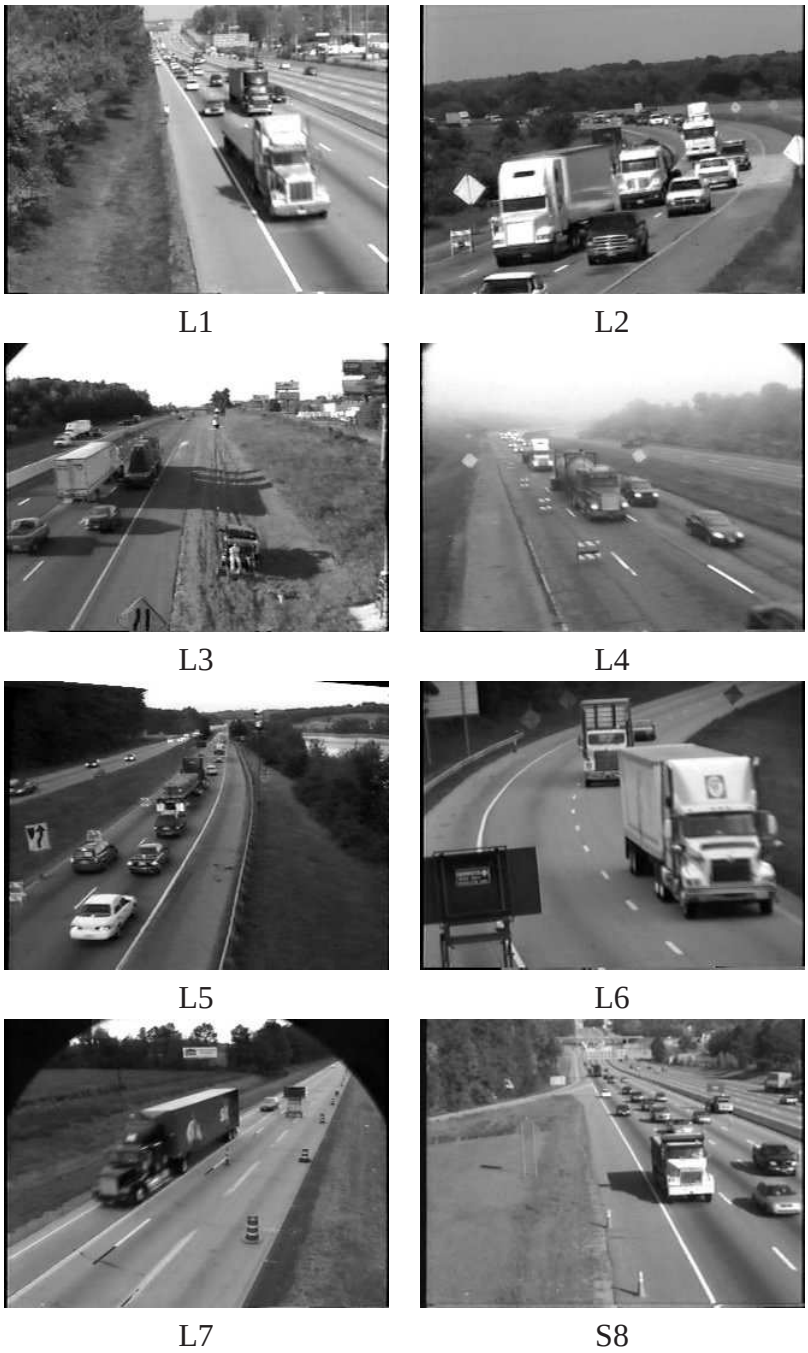


Figure 2.10: Sample image frames from the eleven sequences used in evaluating the algorithm, showing the variety of scenarios considered. S1 and S4 exhibit the same conditions as L1 and L4, respectively; and S9, which is omitted due to lack of space, closely resembles S8.

Seq.	Vehicles (Trucks)	Segmented & Tracked	FP	Classified
L1	627 (50)	610 (97%)	3	99.2% (4/1)
L2	492 (56)	481 (98%)	18	97.3% (2/11)
L3	325 (38)	298 (92%)	6	97.2% (5/4)
L4	478 (57)	456 (95%)	8	98.5% (3/4)
L5	217 (14)	209 (96%)	7	98.1% (1/3)
L6	102 (20)	97 (95%)	1	98.0% (2/0)
L7	157 (29)	146 (93%)	6	96.8% (3/2)
S1	104 (7)	98 (94%)	5	97.1% (2/1)
S4	43 (3)	39 (91%)	3	97.6% (1/0)
S8	113 (8)	107 (95%)	4	98.2% (1/1)
S9	51 (5)	47 (92%)	6	94.1% (1/2)

Table 2.1: Quantitative results for all the test sequences. From left to right the columns indicate the sequence name, the total number of vehicles in the sequence (the number of trucks in parentheses), the number of vehicles correctly segmented and tracked, the number of false positives, and the classification rate. In the last column the numbers in parentheses indicate the number of cars misclassified as trucks, followed by the number of trucks misclassified as cars.

frame, with the convex hull of each group indicated by a thin black line. The number next to each group indicates the number of that vehicle, and the letter T is placed next to each vehicle classified as a truck. The vehicles that are labeled but have no features have already been successfully detected and classified but have already left the tracking zone though they have not yet left the image.

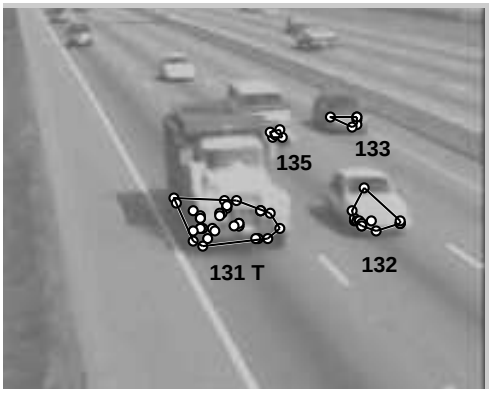
Figure 2.11 demonstrates the ability of the system to segment vehicles which are severely occluded, often by larger vehicles traveling in adjacent lanes. In (a) the van ($\#135$) traveling in the middle lane is detected and tracked by the algorithm despite the fact that it is largely occluded by the truck ($\#131$) throughout the tracking zone. In (c) the car ($\#542$) is detected in the frame shown as it is coming out from being occluded by the truck, just as ($\#541$) was detected in a previous frame while it was still partially occluded by the truck. Similarly, in (d) the vehicle ($\#5$) is detected as it is being disoccluded by the truck in front. In (e) all the vehicles ($\#25$ - $\#28$) appear as a single blob in the foreground mask and yet

the algorithm correctly segments them. Traditionally, separating vehicles in such scenarios has been impossible for background subtraction approaches.

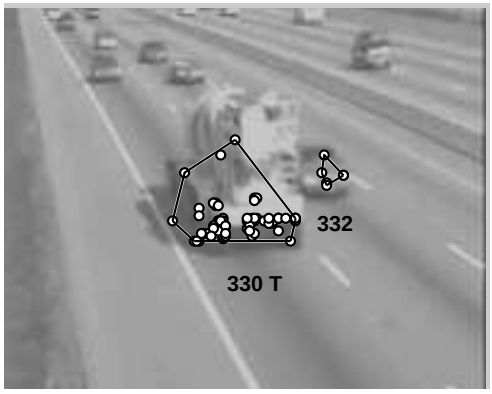
Figure 2.12 shows sample results for vehicles traveling away from the camera in (a) through (d), and for a curved road in (e) and (f). In (a) and (b), the algorithm successfully detects and tracks the vehicles traveling close to each other despite the presence of long shadows. For (c) and (d), vehicles are moving at a low speed and close to each other due to the lane closure but are nevertheless tracked correctly. Notice in (e) that the car ($\#14$) is detected as it is coming out of occlusion from the truck in front. In (f) the cars that were not yet segmented in (e) (i.e., those behind $\#13$) are successfully detected even though they are partially occluded.

Some examples involving large tractor-trailers are shown in Figure 2.13. In (a) both the vehicles ($\#103$ and $\#105$) that are occluded by the white van ($\#101$) are correctly detected and tracked. Similarly, the dark colored SUV ($\#107$) traveling adjacent to the truck ($\#106$) in (b) is detected after a few frames, once a sufficient number of stable features is found. In (c), (d), and (f), the ability of the algorithm to correctly segment and track vehicles that enter the field of view partially occluded and remain occluded throughout the tracking zone is again demonstrated. In (e), the features of a large tractor-trailer are all correctly grouped into one vehicle despite the large extent that they cover in the image. Note that it is the algorithm's identification of large vehicles (trucks) that enables it to prevent declaring false positives in such cases, when the spillover of vehicles into neighboring lanes would confuse traditional 2D algorithms. The algorithm also works when the camera is placed in the center of the road as shown in Figure 2.16.

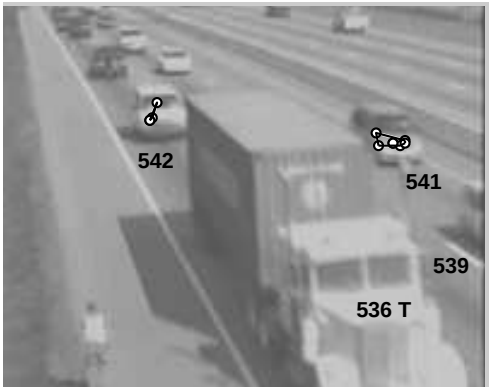
To convey a sense of the limitations of the algorithm, some mistakes are shown in Figure 2.14. In (a) the algorithm fails to detect the car traveling in the first lane (indicated with the letter `M`, for "missing"). Due to the heavy traffic and its being in the far lane, the base of the car remain partially occluded by the vehicle in front ($\#465$) throughout the



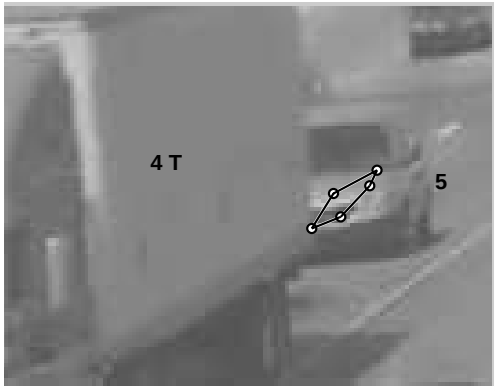
(a) L1: 03516



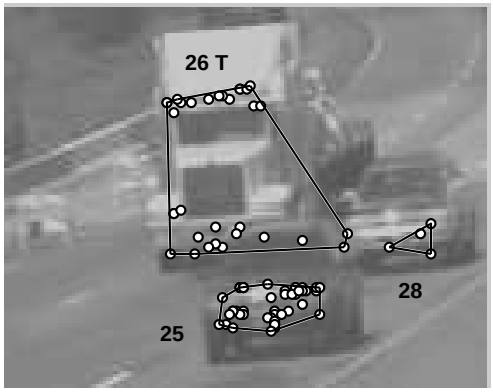
(b) L1: 10302



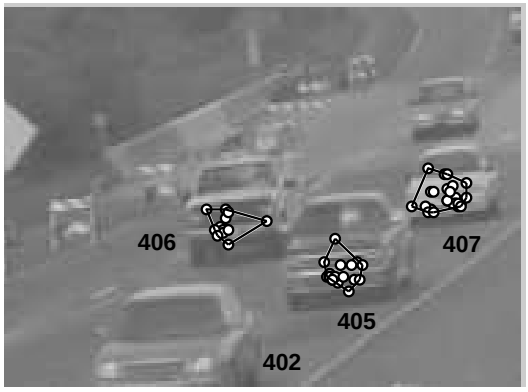
(c) L1: 15440



(d) L2: 00134

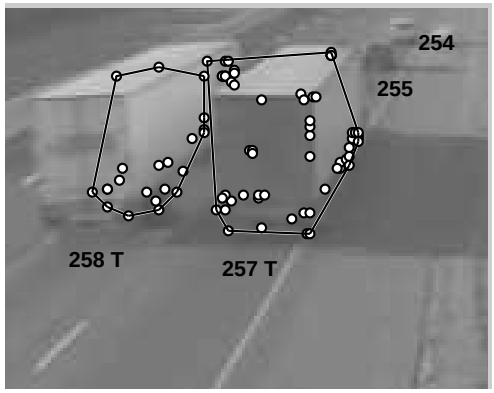


(e) L2: 00913

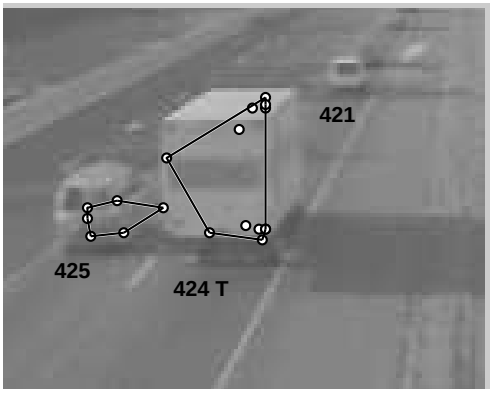


(f) L2: 11960

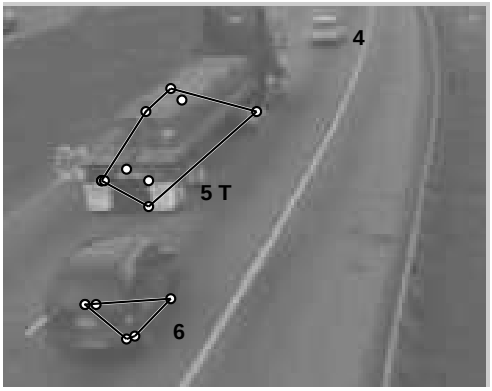
Figure 2.11: Results of the algorithm on some image frames, showing the ability of the algorithm to handle severe occlusions. Below each image is the sequence name and frame number.



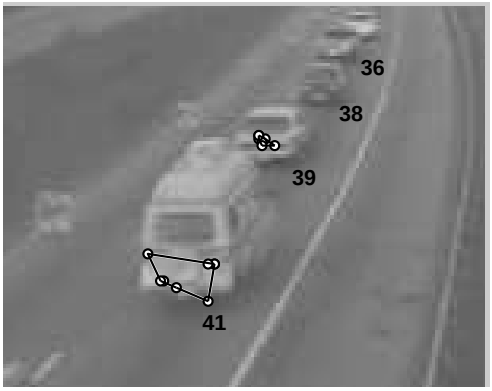
(a) L3: 08860



(b) L3: 17522

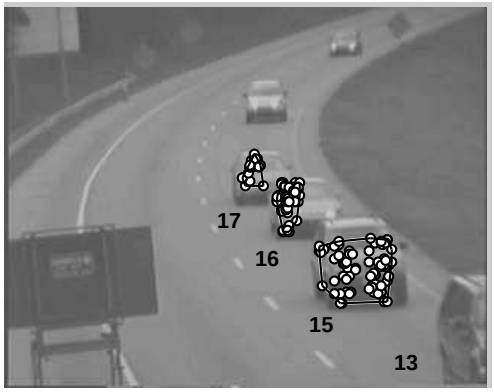


(c) L5: 00439



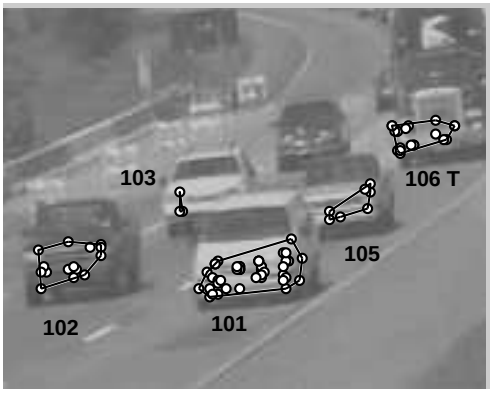
(d) L5: 02618

(e) L6: 01098

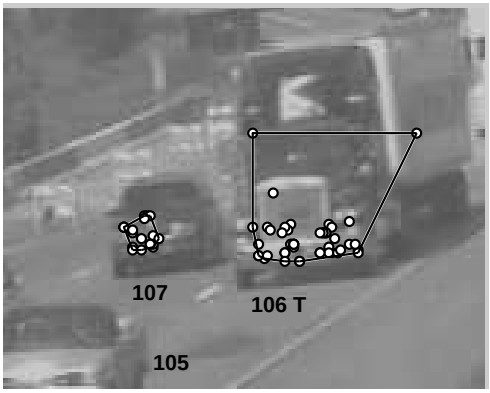


(f) L6: 01190

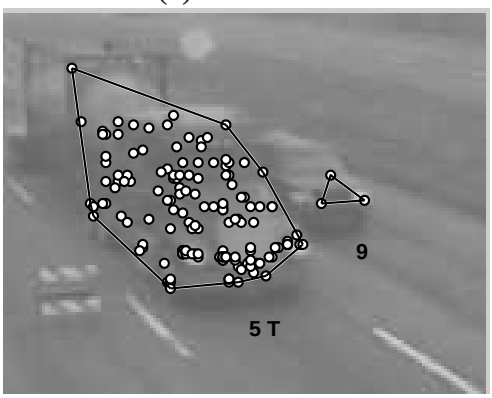
Figure 2.12: Additional experimental results on sequences in which the vehicles are moving away from the camera or the road is curved.



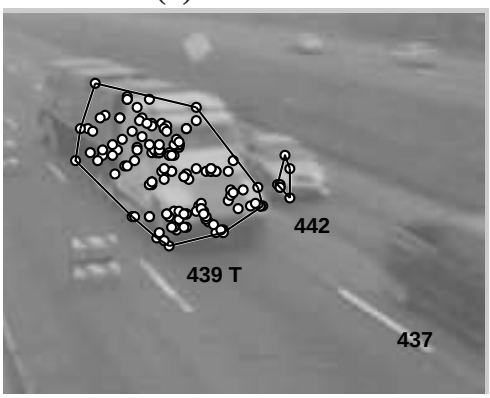
(a) L2: 03204



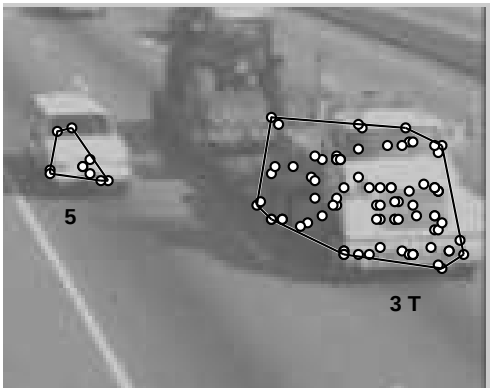
(b) L2: 03260



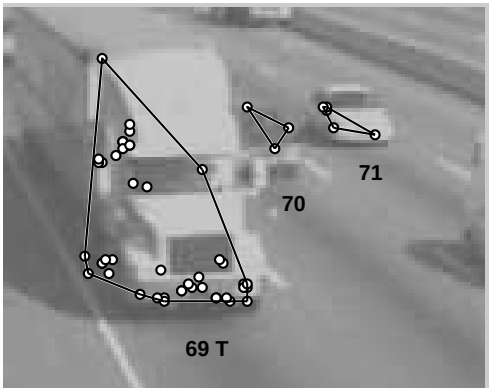
(c) L4: 00566



(d) L4: 14654

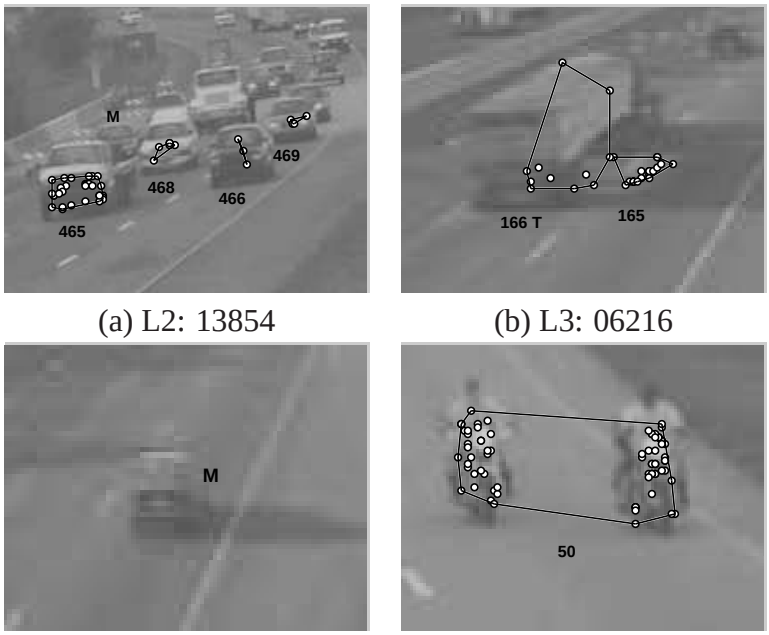


(e) S8: 00078



(f) S8: 00506

Figure 2.13: More experimental results demonstrating the performance of the algorithm when large tractor-trailers occlude other vehicles.



(a) L2: 13854 (b) L3: 06216

(c) L3: 10940 (d) L6: 07240

Figure 2.14: Some instances in which the algorithm makes a mistake.

tracking zone, so that none of the features on the vehicle qualify as stable features. In (b) the shadow of the tractor-trailer is mistakenly detected as a car ($\#165$), thus yielding a false positive. In (c) the algorithm fails to detect a car traveling in isolation because of the lack of a sufficient number of feature points on the vehicle arising from the poor contrast. In (d) the algorithm misinterprets two motorcycles traveling side by side as a single car, an error that could be avoided by including a model for motorcycles and measuring the foreground evidence to validate each vehicle.

In Figure 2.15, the number of vehicles detected by the algorithm is compared with ground truth obtained manually for the S2 sequence. Note that accuracy in the two nearby lanes is quite good, with accuracy in the farthest lane significantly lower due to the increased amount of partial and complete occlusion in that lane. The plot in the middle of the figure shows the trajectories of some vehicles displayed in the road plane. In addition, the mean speed of the vehicles in each lane (computed over one-minute intervals) is plotted

versus time, which corresponds with the general trend evidence in the video sequence.

The detection accuracy was found to be fairly insensitive to the calibration parameters. To quantify this conclusion, each of the end points of the lines corresponding to lane markings was perturbed with additive Gaussian noise with a standard deviation of two pixels in a random direction. Additive Gaussian noise having standard deviation of three pixels was added to the end points of the line perpendicular to the direction of traffic flow. For five different trials on each of the L1 and L4 sequences, the maximum drop in the detection rate was less than 6% of the total number of vehicles (e.g., 97% detection rate became 91%), and the maximum increase in false positives (for L4) was found to be 4 vehicles. (Note that an average user, with a little practice, is able to consistently click within one pixel of the desired location.)

The algorithm was implemented in C++ using the Blepo computer vision library (http://www.ces.clemson.edu/~stb/blepo) and the OpenCV Lucas-Kanade tracker [9]. On a 2.8 GHz P4 laptop computer with 512 MB of memory, the average processing time for a single image frame was 32 ms, which is slightly faster than frame rate. To achieve this speed, the background was updated every 60 frames (two seconds), new features were detected every five frames, and binary morphological operations (dilation and erosion) were performed on subsampled images (by a factor of two in each direction).

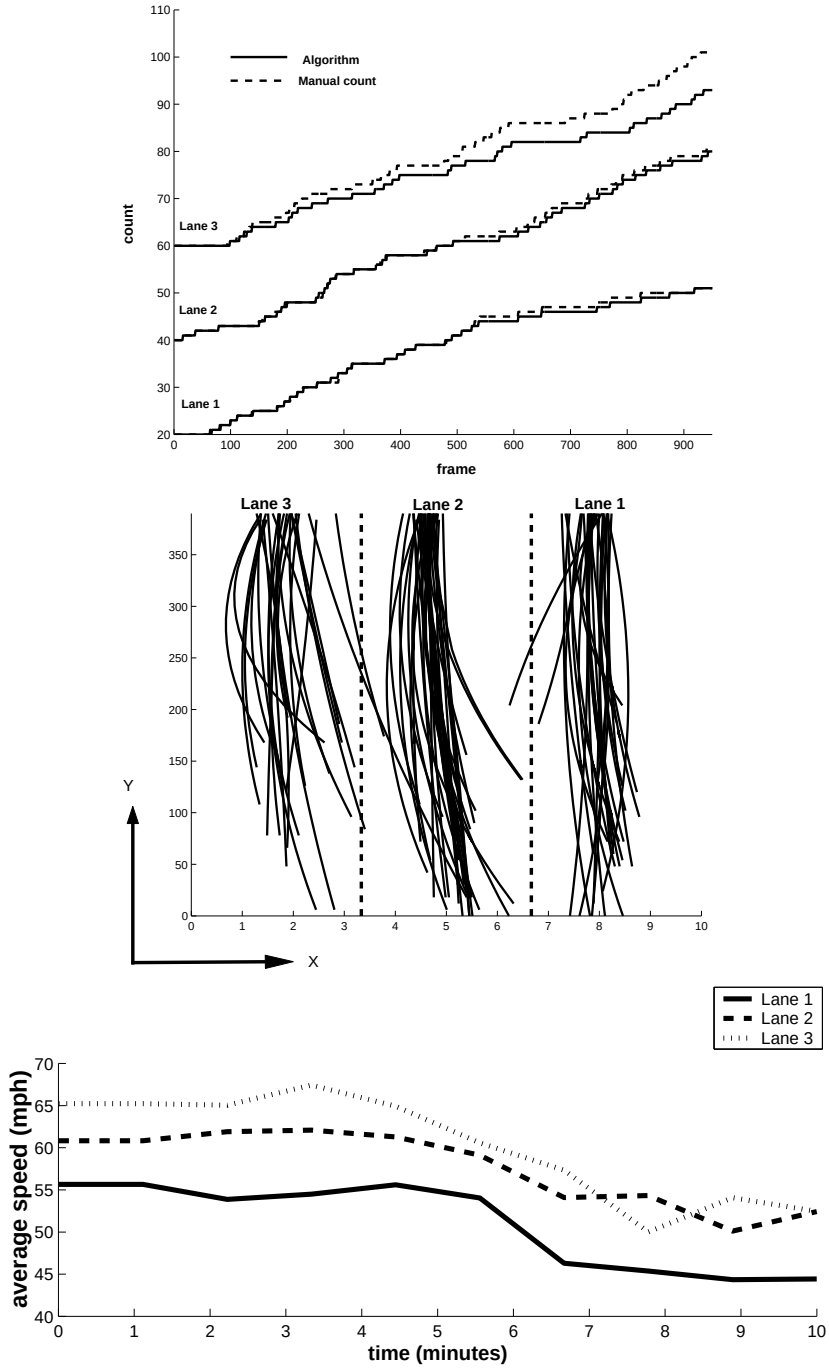


Figure 2.15: Plots displaying the results of the algorithm. TOP: Total vehicles detected in each lane versus time in the S2 sequence, with Lanes 2 and 3 offset by 40 and 60 for viewing clarity. MIDDLE: Some of the vehicle trajectories for L1 as seen in a top-down view, with vehicles that are changing lanes clearly visible. BOTTOM: Mean speed (in miles per hour) for the vehicles in L1 computed over one-minute intervals.

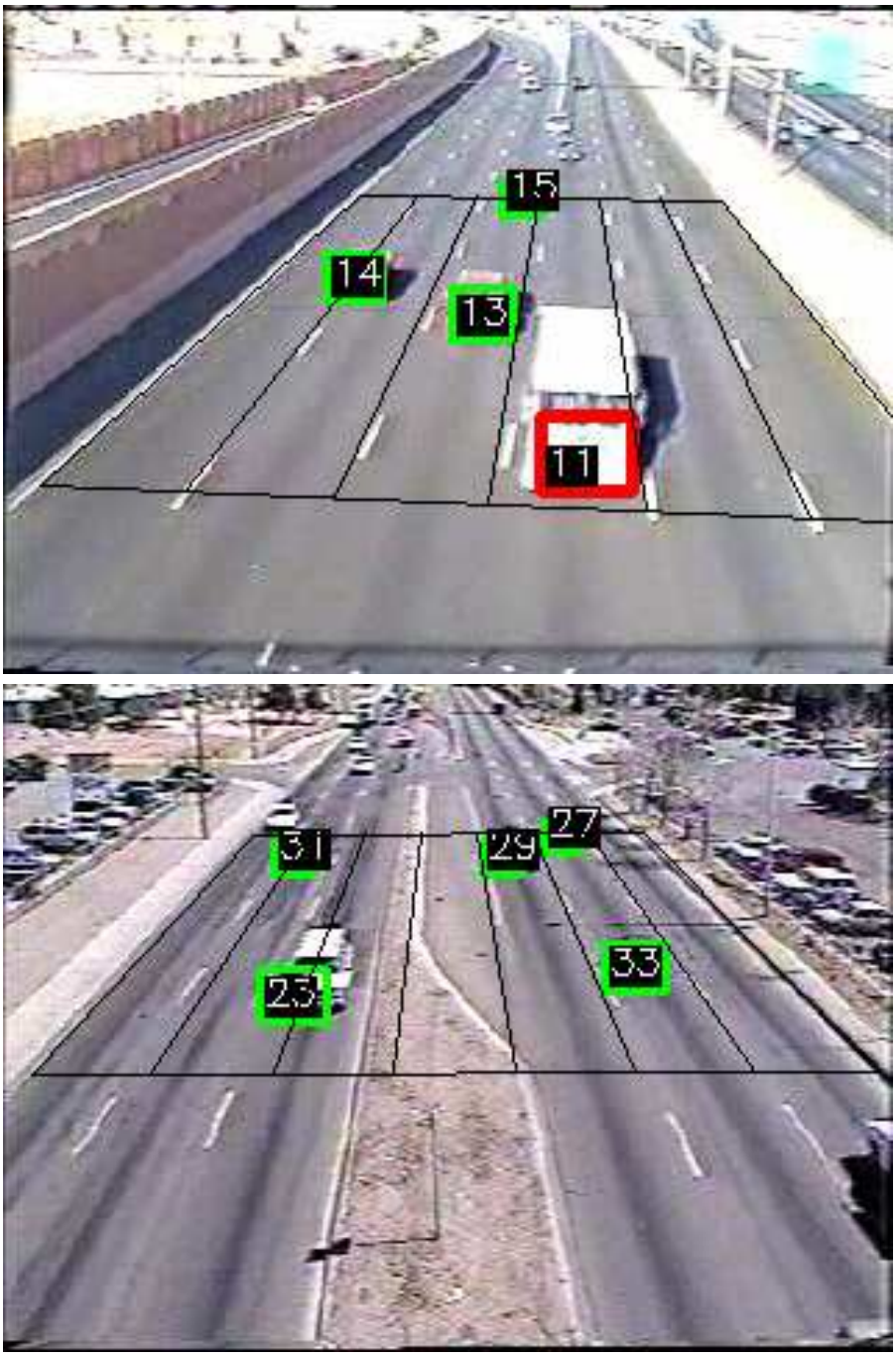


Figure 2.16: Vehicles can be detected and tracked when the camera is mounted in the middle of the road as opposed to the situation in previous experimental results where the camera is on the side of the road.

Chapter 3

Pattern recognition-based detection of vehicles

Pattern recognition is a classification (or labeling) problem where the input data (a pattern) is analyzed to find a suitable class (label) for it based on statistical information extracted from the data or a priori knowledge about the data. Some of the challenges, training methodologies, algorithms and applications in pattern recognition are discussed in [57, 18]. Most supervised pattern recognition systems have at least three stages as shown in Figure 3.1. In the first stage the input data (pattern) is acquired from a sensor (e.g., a camera) and may be pre-processed (contrast stretching, extraction of foreground objects etc.). The raw data acquired from the sensor is usually of high dimension and thus using this data directly as the input of a classifier can result into a significant degradation of performance. A feature extraction stage transforms the raw sensor data into a low-dimensional representation (2D in our example).

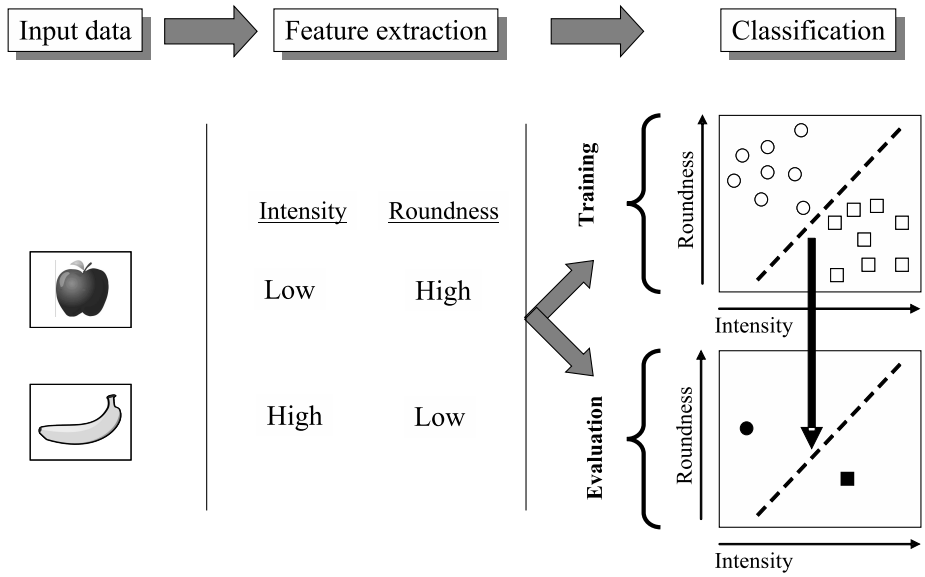


Figure 3.1: A typical pattern recognition system consists of sensor input, feature extraction and a classifier. In this example an image captured by a camera is the raw input. Two features (average pixel intensity, and roundness) are extracted in the feature extraction stage. The classifier finds a decision surface (a dashed line in this example) using the training data (white circles represent training images for apples and white rectangles represent training images for bananas). Black circle and rectangle is the decision of the classifier on a new (previously not seen in the training data) input image.

3.1 Boosted cascade vehicle detector (BCVD)

The problem of pattern recognition has been studied extensively for many years, giving rise to a variety of approaches such as neural networks, support vector machines (SVMs), and Bayesian classifiers. A relatively new approach using a cascade of simple features to detect patterns in images was developed by Viola and Jones [66]. In their approach each image sub-window is passed through a series of tests of increasing difficulty, known as a cascade. The goal of each stage in the cascade is to evaluate the sub-window using a set of image features to decide whether to reject the sub-window as containing the object of interest. Subsequent stages perform more detailed analyses using larger and more discriminating sets of features, with each stage trained to achieve a high detection rate (e.g., 99%) and a liberal false alarm rate (e.g., 50%). Sub-windows in the image which are easily distinguishable as non-vehicles (e.g., an image patch with little or no texture) are discarded in the initial stages of the cascade, resulting in faster processing, so that the complete set of features needs to be evaluated for only the small fraction of sub-windows that reach the final stage of the cascade. The training process ensures that the classification errors in each stage are independent of each other.

3.1.1 Training with integral images and Haar like features

The Viola-Jones algorithm achieves real-time processing not only with the cascade architecture, but also because it uses simple image difference features that are quickly computed using an integral image. The features used in [66] are simply arithmetic additions and subtractions of pixel intensities in a detection window. An example of such a feature is shown in Figure 3.3 where the value of a feature is computed by subtracting the sum of pixel intensities inside black rectangles from the sum of pixel intensities inside white rectangles. Given a set of labeled training images (vehicles and non-vehicles), the training

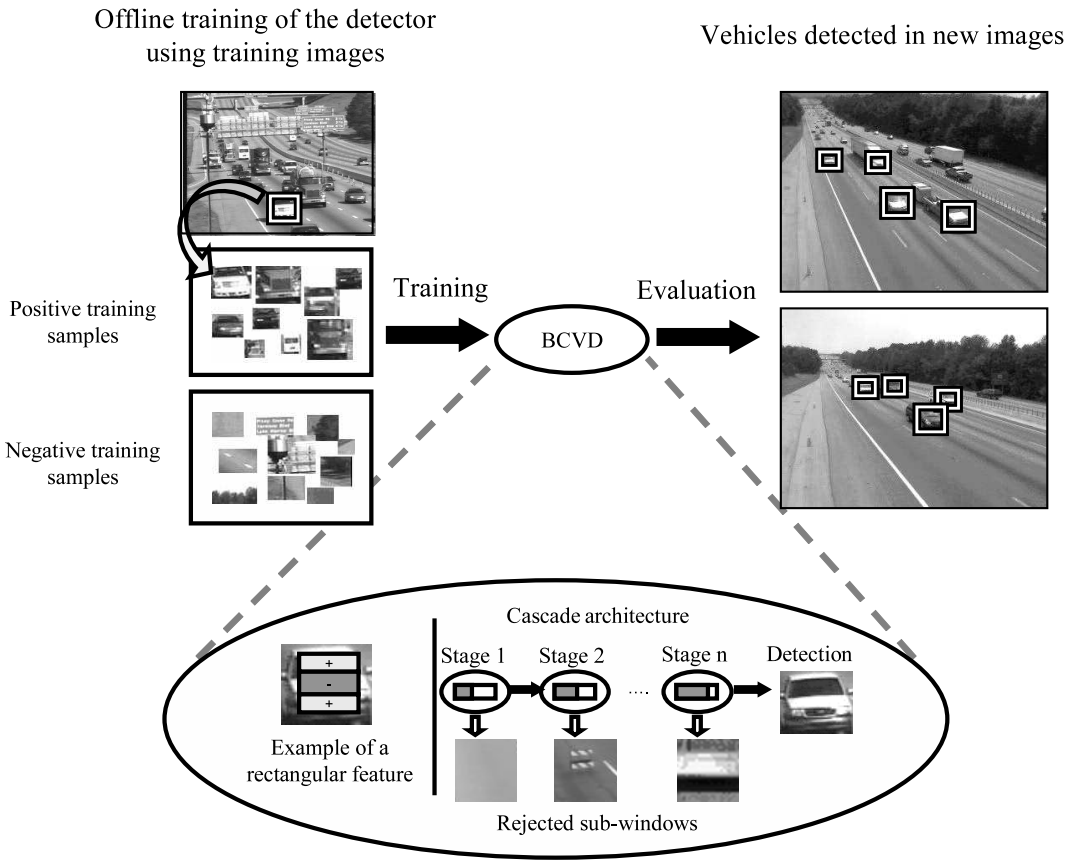


Figure 3.2: Training of boosted cascade vehicle detector (BCVD).

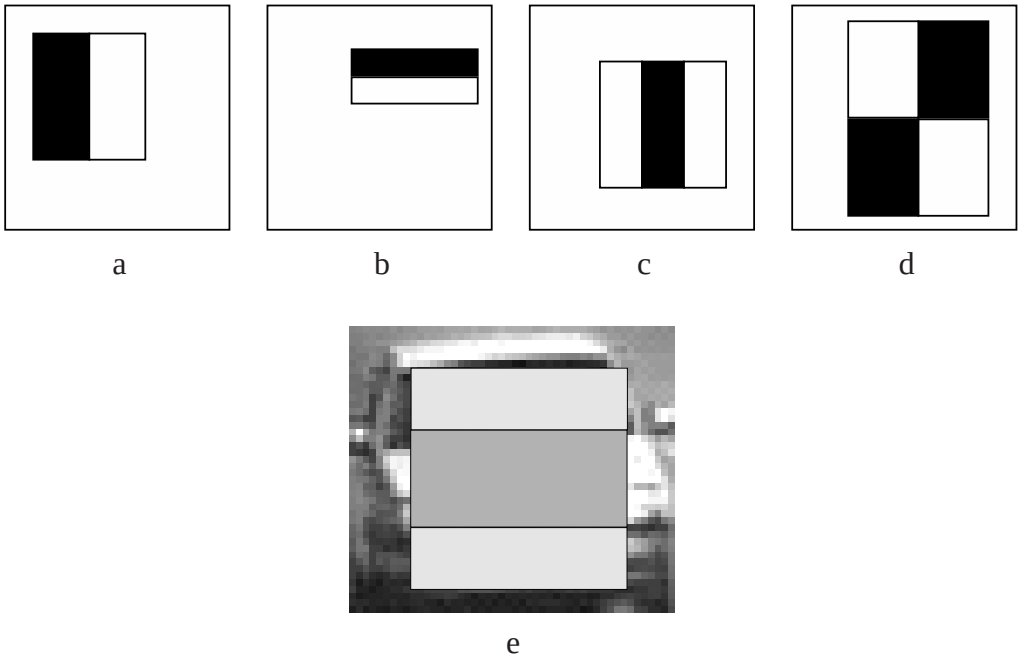


Figure 3.3: Example of rectangular features used for training the pattern detector. A (scalar) value of a single feature is computed by subtracting the sum of pixel intensities in black rectangles from the sum of pixel intensities in the white rectangles. (a) Vertical two-rectangle feature (b) horizontal two-rectangle feature (c) vertical three-rectangle feature (d) a four-rectangle feature. (e) A horizontal three-rectangle feature is overlaid on an image-window of a car for illustration.

process first finds a feature (from a large pool of rectangular features) and a corresponding threshold on the value of the feature that performs best on the training data. A single feature in essence acts as a weak classifier whose decision is at least slightly better than random chance. The idea behind boosting is to combine several such weak classifiers in a way such that the final strong classifier meets the performance requirements. After training, vehicles are detected by sliding the strong classifier over the input image and computing the decision (vehicle or non-vehicle) at each sub-window in the image. To detect vehicles at different scales, the feature set (and in effect the detection window) is scaled (rather than the more traditional approach of resampling of the input image), which further reduces the computational load.

Viola and Jones [66] introduce the idea of integral images which enables computing

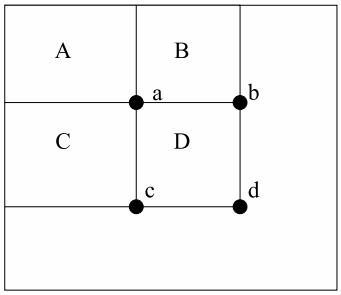


Figure 3.4: Computing a value of feature using an integral image. The sum of pixels within rectangle D can be computed with four array references. The value of integral image at location *a* (*ii[a]*) is the sum of pixels in rectangle A, *ii[b]* is A + B, *ii[c]* is A + C, and *ii[d]* is A + B + C + D. The sum within D can be computed as *ii[d]-ii[b]-ii[c]+ii[a]*. Image adapted from [66].

the values of features described above in efficient manner. The integral image at location x, y is the sum of pixels above and to the left of x, y including the value of x, y itself.

3.1.2 Detection, filtering and tracking using BCVD

Each image of the video sequence is scanned exhaustively at multiple scales by the BCVD to detect vehicles. The output of the BCVD is a rectangle for each detected vehicle, and the midpoint along the bottom edge of the rectangle is retained as the location of the vehicle for the purpose of computing proximity to other vehicles. Vehicles from the previous image frame are tracked by searching among nearby detections in the current image frame. In case a match is not found, the vehicle is flagged as missing and its location is updated by means of a standard template matching mechanism using normalized cross-correlation. If a vehicle is missing for several consecutive frames, it is discarded for the lack of sufficient evidence. Meanwhile, new vehicles are initialized for all the detections that did not yield a match. This straightforward tracking procedure augments the position information of the vehicles with their image trajectories.

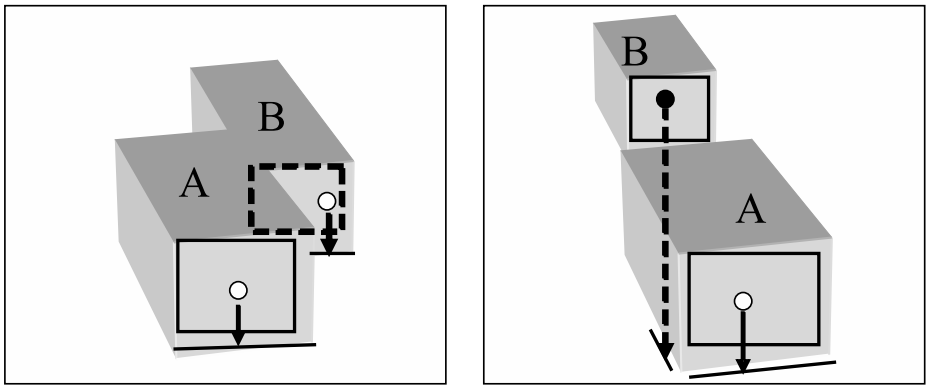
To reduce the amount of false positives a foreground mask is used (obtained by

background subtraction as described in Section 2.1.2) to eliminate detections that belong to the stationary background. In addition we use calibration information to estimate the expected size of the detection rectangle based on the location of the rectangle in the image. A detection is ignored if the size of the detection (corresponding rectangular bounding box) varies significantly from the estimated size at that location.

3.2 Combining BCVD with feature tracking

As seen in Chapter 2, vehicle detection is based on segmenting stable feature points. In a situation as shown in Figure 3.5(b), when the base (corresponding to the side facing the camera) of a vehicle (vehicle B in our example) is occluded by another vehicle in back-to-back manner, the feature points close to the base of vehicle A would be incorrectly projected at a height greater than their true height from the road. However, in such scenarios the BCVD is likely to detected the vehicle since most of the symmetric features on the vehicle still remain visible in the image. On the other hand in a situation of lateral occlusion as illustrated in Figure 3.5 (a), BCVD will fail to detect the vehicle (lateral occlusion hides symmetric features) but stable features can be found on occluded vehicle (using plumb line projection) as long as the vehicle is not occluded completely.
BCVD was combined with feature tracking in following manner:

1. Two sets of vehicles are independently detected using stable features and BCVD.
2. Vehicles that are currently being tracked are matched with detections in the current frame.
3. Unmatched vehicles in the current frame which were detected using stable features are initialized as new detections.



a b

Figure 3.5: In (a) vehicle B undergoes a partial lateral occlusion by vehicle A. In this case points on both vehicles (white circles) will be detected as stable features even though BCVD fails to detect B (dashed rectangle). In another situation shown in (b), vehicle B is traveling behind A. As a result point on B (black circle) will not be detected as a stable feature due to its wrong plumb line projection (dashed arrow) on the base of vehicle A. As such, feature tracking based approach misses vehicle B, however BCVD successfully detects it (solid rectangle).

4. A new vehicle is initialized from each unmatched vehicle detected using BCVD only if there are no other vehicles (either an existing vehicle being tracked or a new detection in current frame) in its vicinity.

3.3 Experimental results

Performance of two BCVD detectors was evaluated, one for detecting both cars and trucks and the other for detecting motorcycles. Figure 3.6 shows the four sequences used to extract the positive samples for training the car/truck detector. Total of 800 samples were manually extracted from the training sequences which were then randomly distorted (rotation on either side within 2 degrees, brightness change within 10%, change in dimensions within 5%) to generate a total of $6,400$ positive training samples. The detector was trained using the Haar-training module of the OpenCV [2] library with 16×12 detector size and 14

Figure 3.6: Four training sequences for BCVD to detect cars and trucks.

stages in the cascade. A large number ($5,000$) of randomly selected high-resolution images were used as a negative training set.

Some examples of training images are shown in Figure 3.7. Figure 3.8 shows a sample output frame from each of the test sequences. Quantitative analysis is presented in Table 3.1. For each test sequence the second column indicates the ground truth, i.e. the actual number of total vehicles in the sequence. Three sets of results are shown in the table for each of the sequence. In the first case a sequence was processed using stable features as described in Chapter 2. Next, the sequence was processed using only the BCVD. Finally a combination of BCVD and stable features (as described in the previous section) was used to process the same sequence. In each case, TP indicates the number of correct vehicle detections (true positives) and FP indicates the number of spurious detections (false

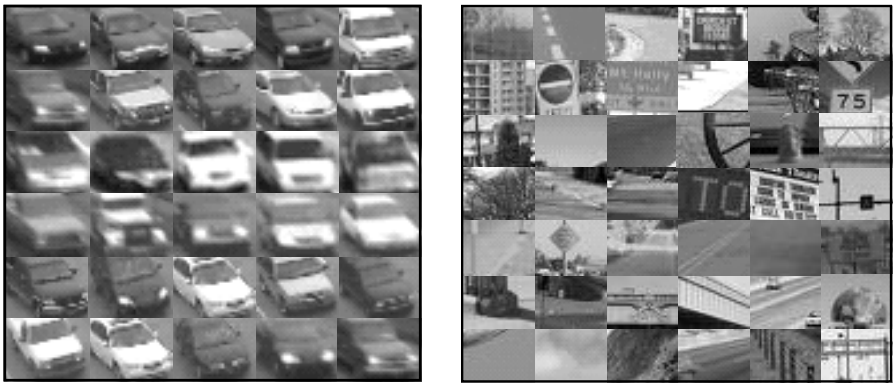


Figure 3.7: Sample positive and negative training images for car/truck BCVD. The original images are 16×12 pixels in size.

C1 C2

C3 C4

Figure 3.8: Four test sequences to evaluate accuracy of BCVD. C4 was captured on a rainy day.

Seq.	Vehicles	Stable features		BCVD		Stable features + BCVD	
		TP	FP	TP	FP	TP	FP
C1	260	210(81%)	7	213(82%)	13	224(86%)	18
C2	312	275(88%)	2	278(89%)	9	287(92%)	8
C3	146	134(92%)	5	114(78%)	7	137(94%)	12
C4	187	124(66%)	5	143(76%)	24	153(81%)	23

Table 3.1: Results comparing performance of stable features, BCVD and combined system. TP is the number of correct detections (true positives). FP is the number of spurious detections (false positives)

positives). A detection is considered a TP only if the vehicle is detected and tracked till it exits the detection zone. Similarly a detection is considered a FP only if it leads to a vehicle exiting the detection zone.

Note that in some cases the number of false positives for combined detection is less than the sum of false detections in the other two. As mentioned in the previous section, in the combined detection mode two sets of vehicles are independently detected using stable features and BCVD, so intuitively the false positives should add up. However, if a BCVD detection (a false detection for example) is in the vicinity of another detection (a detection by stable features for example) then the detection is discarded. The same BCVD detection would have resulted into a false positive if the sequence was being processed using only the BCVD. So for the combined detection, the number of false positives is between the false positives of stable features and the sum of false positives of stable features and BCVD.

3.4 Detecting motorcycles

BCVD can be trained to detect other types of vehicles apart from cars and trucks. A motorcycle detector was trained using a very limited amount of existing data and tested it at two different locations. Since the number of motorcycles in a typical traffic scene

Figure 3.9: Sequences used to train a motorcycle detector.

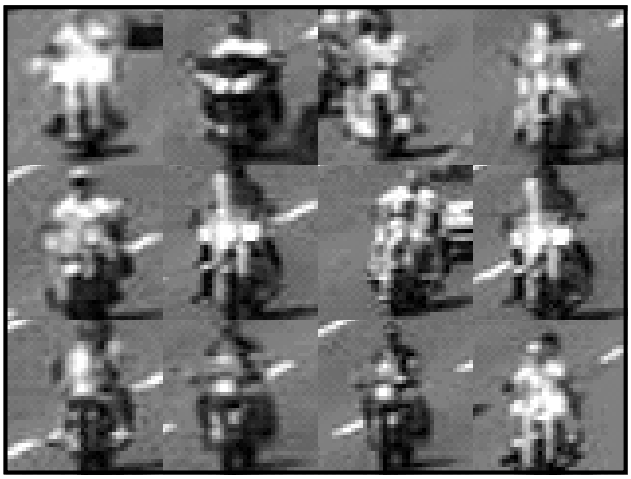


Figure 3.10: Examples of positive training images for motorcycle detector. Original images are 20×20 in size.

is very small (less than 1%), gathering sufficient training data was time consuming. The training sequences shown in Figure 3.9 have 32 motorcycles in total. A total 300 instances of those motorcycles were extracted from the sequence to generate a total 2400 positive training samples using small distortions (similar to the case of car/truck detector). A 14 stage, 20×20 size detector was trained using the OpenCV library.

Figure 3.11 shows a sample frame from each test sequence. The test sequences were captured at special events organized for motorcyclists. From Table 3.2 it appears that the performance of the motorcycle detector is less than that for a car/truck detector (when

M1

M2

M3

Figure 3.11: Test sequences for motorcycle detector.

compared to the results of 3.1). It is plausible that more training data will improve the accuracy of the detector.

Seq.	Motorcycles	TP	FP
M1	80	65 (81%)	11
M2	40	31 (77%)	5
M3	70	59 (84%)	8

Table 3.2: Quantitative results of motorcycle detection.

Chapter 4

Calibration of traffic monitoring cameras

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 is useful to handle scale changes (as vehicles approach or recede from the camera) and to measure speeds. It is important to note that the calibration methods described below do not require knowledge about the camera specifications (if the information is available, it can be easily incorporated to improve the calibration accuracy).

A method for directly estimating the projective matrix is described in the first section of this chapter using point correspondences between points in the image plane and respective points in the world coordinate system. In a situation where obtaining such point-correspondences is difficult, camera model can be simplified under reasonable assumptions to estimate parameters of the assumed camera model and then the projective matrix can be computed from the estimated parameters. In this approach some information about the

scene such as a known measurement along the road surface or prior information about height of the camera is required. Different scenarios for the type of information that is available are discussed in the second section of this chapter.

4.1 Direct estimation of projective matrix

A perspective-projective pinhole camera model is assumed. The general relationship between an object point measured with respect to a user-selected world coordinate system and its image plane point is denoted by a 3×4 homogeneous transformation matrix [56, 24]. This matrix will be referred as the camera calibration matrix **C**.

$$\widehat{\boldsymbol{p}} = \mathbf{C}\,\widehat{\mathbf{P}}\,, \tag{4.1}$$

where $\widehat{\boldsymbol{p}} = [\,uw \quad vw \quad w\,]^T$ and $\widehat{\mathbf{P}} = [\,x \quad y \quad z \quad 1\,]^T$ are vectors containing homogeneous coordinates of image point, $\boldsymbol{p} = [\,u \quad v\,]^T$ and world point $\mathbf{P} = [\,x \quad y \quad z\,]^T$ respectively. Representing the matrix with corresponding entries, we get

$$[\,uw \quad vw \quad w\,] = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \end{bmatrix} [\,x \quad y \quad z \quad 1\,]\,. \tag{4.2}$$

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

Expanding the above equation, yields

$$u = \frac{c_{11}x + c_{12}y + c_{13}z + c_{14}}{w} \tag{4.3}$$

$$v = \frac{c_{21}x + c_{22}y + c_{23}z + c_{24}}{w} \tag{4.4}$$

$$w = c_{31}x + c_{32}y + c_{33}z + 1\,. \tag{4.5}$$

Substituting w into first two equations and rearranging leads to,

$$u = x\,c_{11} + y\,c_{12} + z\,c_{13} + c_{14} - u\,x\,c_{31} - u\,y\,c_{32} - u\,z\,c_{33} \tag{4.6}$$

$$v = x\,c_{21} + y\,c_{22} + z\,c_{23} + c_{24} - v\,x\,c_{31} - v\,y\,c_{32} - v\,z\,c_{33}\,. \tag{4.7}$$

The two equations above 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 two equations of the form (4.6) & (4.7). Six or more points in a non-degenerate configuration lead to an over-determined system:

$$\begin{bmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1\,x_1 & -u_1\,y_1 & -u_1\,z_1 \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1\,x_1 & -v_1\,y_1 & -v_1\,z_1 \\ x_2 & y_2 & z_2 & 1 & 0 & 0 & 0 & 0 & -u_2\,x_2 & -u_2\,y_2 & -u_2\,z_2 \\ 0 & 0 & 0 & 0 & x_2 & y_2 & z_2 & 1 & -v_2\,x_2 & -v_2\,y_2 & -v_2\,z_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & z_n & 1 & 0 & 0 & 0 & 0 & -u_n\,x_n & -u_n\,y_n & -u_n\,z_n \\ 0 & 0 & 0 & 0 & x_n & y_n & z_n & 1 & -v_n\,x_n & -v_n\,y_n & -v_n\,z_n \end{bmatrix} \begin{bmatrix} c_{11} \\ c_{12} \\ c_{13} \\ c_{14} \\ c_{21} \\ \vdots \\ c_{33} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \\ u_n \\ v_n \end{bmatrix} \tag{4.8}$$

which can be solved using a standard least squares technique.

The offline calibration process depends upon the user-specified point correspon-

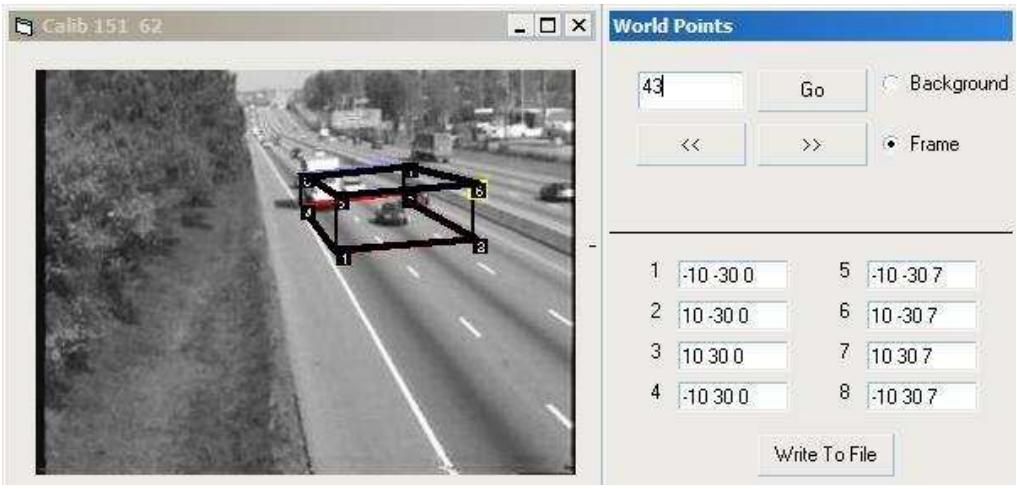


Figure 4.1: Camera calibration tool.

dences 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 distances. For cases where this information is not available (e.g. pre-recorded data), an approximation can be done using standard specifications such the width of a lane, length of a truck etc.

An example of the calibration process is shown in Figure 4.2. First, a marker is placed across the width of the road and perpendicular to the lane markings as shown in (a). With the marker position unchanged, sequence is advanced till the rear end of the truck appears to align with the marker position on the ground. A new marker is placed to align with the height of the truck (b). In the same frame a marker is placed on the ground to align with the front end of the truck (c). Once again, the sequence is advanced till the marker placed on the ground in (c) appears to align with the read end of the truck. This is shown in (d). For the same frame, the marker is realigned with the front end of the truck as shown in (e). A new marker is placed across the width of the road (f). One more time, the sequence is advanced for the new marker to appear aligning with the truck's rear end. An additional marker is placed as shown in (g) in such a way that it appears to be aligned with the height

of the truck. The result looks as shown in (h). Using the dimensions of a known type of vehicle is an approximate method for estimating world coordinates of control points. The table below lists lengths of some of the common vehicle types found on the road. In addition to this, the information about lane width (e.g., 12 feet on an interstate) and number of lanes is used.

The imaging process maps a point in three dimensional space into a two dimensional image plane. The loss of dimension results into an 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 the unknown point in the world.

To measure distances in the road plane, we can substitute $z = 0$ in above equations to get the mapping of points from the image plane (u, v) to corresponding points in the road plane (x, y):

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} c_{11} - uc_{31} & c_{12} - uc_{32} \\ c_{21} - vc_{31} & c_{22} - vc_{32} \end{bmatrix}^{-1} \begin{bmatrix} u \\ v \end{bmatrix}. \tag{4.9}$$

4.2 Parameter-based estimation of projective matrix

As in [45, 58, 60], a pinhole camera model is adopted assuming flat road surface, zero roll angle, and square pixels. In addition, image center is assumed to be the principal point. These are the same assumptions made in [58]. The roll angle of the camera (which does not change with pan-tilt movements) can be easily compensated by rotating the image about its center. The user can manually specify roll angle by drawing a lines in the image along a structure known to be perpendicular to the road plane in real world (e.g., vertical edges of a container behind a tractor trailer). With these assumptions, four parameters are needed to map between pixel distances (measured in the image) and corresponding

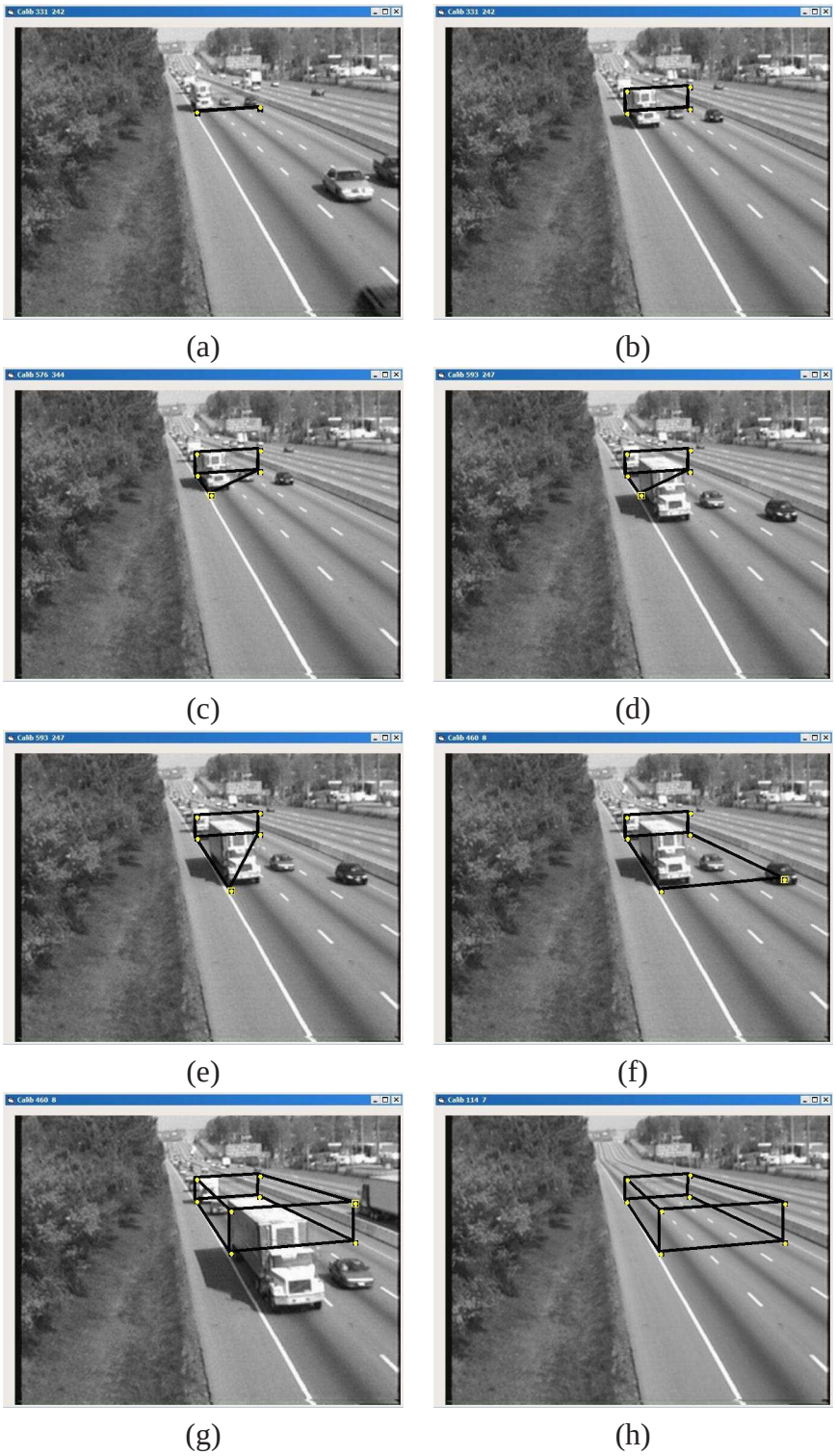
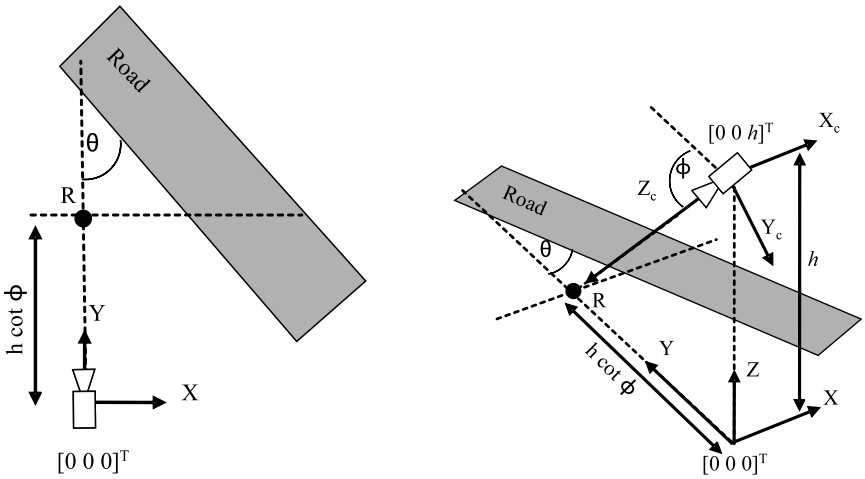


Figure 4.2: Camera calibration process for direct estimation of projective matrix.



(a) Top view of the scene (b)Same scene viewed from the left side

Figure 4.3: Camera is placed at height h feet above the road with down/tilt angle ϕ and pan angle θ. X, Y, Z is the world coordinate system while X_c, Y_c, Z_c is the camera coordinate system. The optical axis of the camera intersects the Y axis of the world coordinate system at $h \cot \phi$. The optical axis of the camera intersects the road plane at R.

distances on the road (measured in Euclidean world units): Focal length (f), tilt angle (ϕ), pan angle (θ), and height of the camera measured from the road surface (h).

A point $\mathbf{X} = [\,x \quad y \quad z \quad 1\,]^T$ in world coordinate frame is related to its image coordinates $\mathbf{x} = [\,wu \quad wv \quad w\,]^T$ as follows:

$$\mathbf{x} \;=\; P\mathbf{X}$$

$$\begin{bmatrix} wu \\ wv \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & -f\sin\phi & -f\cos\phi & fh\cos\phi \\ 0 & \cos\phi & -\sin\phi & h\sin\phi \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \tag{4.10}$$

where

$$P \;=\; KR\,[I_{3\times3} \mid -T]$$

$$
\begin{aligned}
K &= \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
R &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & -\sin\phi & -\cos\phi \\ 0 & \cos\phi & -\sin\phi \end{bmatrix} .
\end{aligned}
$$

K is the camera calibration matrix, R is the rotation matrix corresponding to a rotation of $(90^\circ + \phi^\circ)$ around the X-axis, I is the 3×3 identity matrix and $T = \begin{bmatrix} 0 & 0 & h \end{bmatrix}^T$ is the translation of the camera from the origin of the world coordinate system. $[I_{3\times3} \mid -T]$ is concatenation of I and T. Notice that assuming square pixels, zero skew and principal point as the image center results into a single internal calibration parameter f. Using (4.10) we can express the relationship between the world coordinates (x, y) of a point on the road ($z = 0$) to its image coordinates (u, v) as follows:

$$
u = \frac{wu}{w} = \frac{fx}{y\cos\phi + h\sin\phi} \tag{4.11}
$$

$$
v = \frac{wv}{w} = \frac{fh\cos\phi - fy\sin\phi}{y\cos\phi + h\sin\phi} . \tag{4.12}
$$

Rearranging above equations, we get:

$$
x = \frac{hu\cos\phi(1 + \tan^2\phi)}{v + f\tan\phi} \tag{4.13}
$$

$$
y = \frac{h(f - v\tan\phi)}{v + f\tan\phi} . \tag{4.14}
$$

For any two points in the road plane having the same coordinates along the y-axis,

we can relate the pixel difference in their u-coordinates with the distance between the points along the x-axis in world coordinates using (4.11)

$$\Delta u = \frac{f \Delta x}{y \cos\phi + h \sin\phi} \,. \tag{4.15}$$

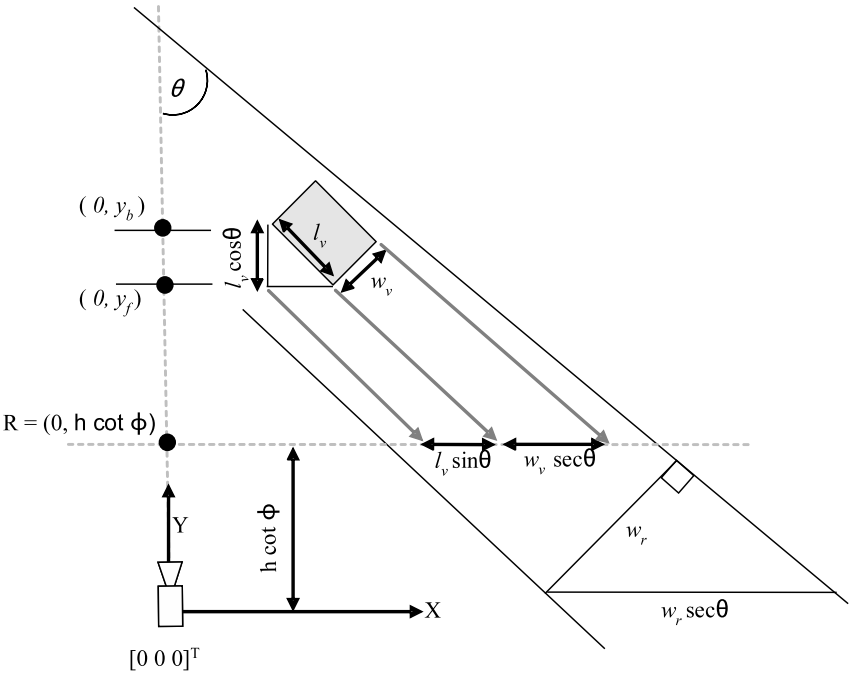
It is clear that the y-coordinate of any point in the world corresponding to a point on the u-axis in the image can be obtained by substituting $v = 0$ in (4.12):

$$y_{v=0} = h \cot\phi \,. \tag{4.16}$$

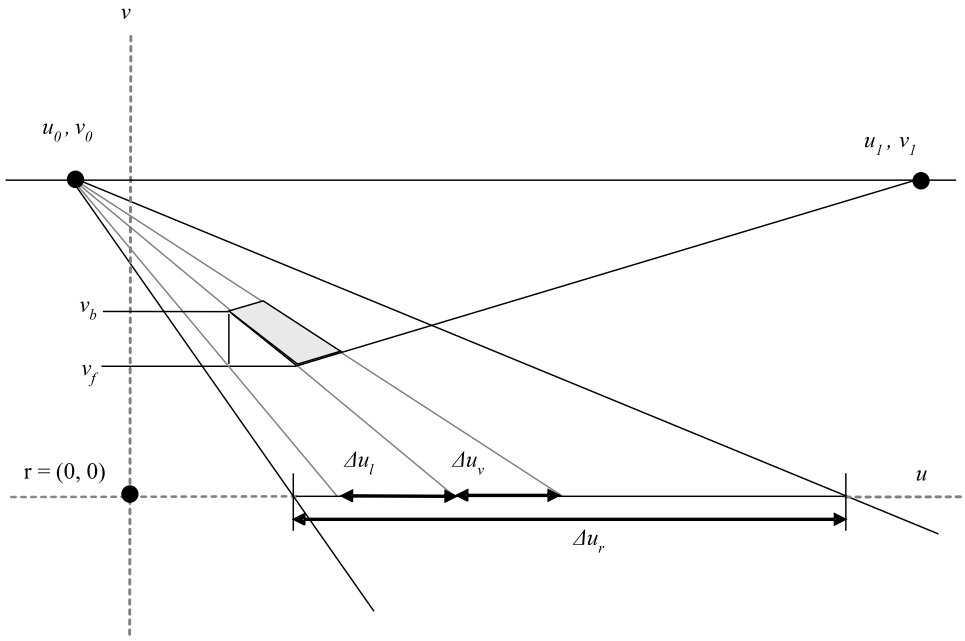
4.2.1 Two vanishing points and known camera height (VVH)

Vanishing points are independent of camera's location and depend on the internal parameters of the camera and its pose [25]. Two vanishing points (one along the direction of flow of traffic and another in a direction orthogonal to it) yield three equations (assumption of zero camera roll leads to identical coordinates along the v-axis) in f, ϕ, and θ. In homogeneous coordinates the vanishing point $p_0 = [\, su_0 \quad sv_0 \quad s \,]^T$ corresponding to the vanishing line $l_0 = [\, -\tan\theta \quad 1 \quad 0 \quad 0 \,]^T$ is obtained as $p_0 = Pl_0$. Similarly, the vanishing point $p_1 = [\, su_1 \quad sv_1 \quad s \,]^T$ corresponding to the vanishing line $l_1 = [\, -1 \quad -\tan\theta \quad 0 \quad 0 \,]^T$ is obtained as $p_1 = Pl_1$.

$$\begin{bmatrix} su_0 \\ sv_0 \\ s \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & -f\sin\phi & -f\cos\phi & fh\cos\phi \\ 0 & \cos\phi & -\sin\phi & h\sin\phi \end{bmatrix} \begin{bmatrix} -\tan\theta \\ 1 \\ 0 \\ 0 \end{bmatrix}$$



(a) Points of interest and measurements in the world coordinate system



(b) Corresponding measurements in the image plane

Figure 4.4: (a) Measurements in the road plane. (b) Measurements in the image plane.

$$\begin{bmatrix} su_1 \\ sv_1 \\ s \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & -f\sin\phi & -f\cos\phi & fh\cos\phi \\ 0 & \cos\phi & -\sin\phi & h\sin\phi \end{bmatrix} \begin{bmatrix} -1 \\ -\tan\theta \\ 0 \\ 0 \end{bmatrix}.$$

Note that l_0 and l_1 correspond to the direction along the length of the road and perpendicular to the length of the road respectively.

By expanding the above equations, we obtain:

$$u_0 = \frac{-f\ \tan\theta}{\cos\phi} \tag{4.17}$$

$$v_0 = v_1 = -f\ \tan\phi \tag{4.18}$$

$$u_1 = \frac{f}{\cos\phi\tan\theta}. \tag{4.19}$$

Solving these three equations gives us

$$f = \sqrt{[-(v_0^2 + u_0u_1)]} \tag{4.20}$$

$$\phi = \tan^{-1}\left(\frac{-v_0}{f}\right) \tag{4.21}$$

$$\theta = \tan^{-1}\left(\frac{-u_0\cos\phi}{f}\right). \tag{4.22}$$

4.2.2 Two vanishing points and known width (VVW)

As seen in the previous subsection, two vanishing points lead to three equations which can be solved to find f, ϕ and θ. At least one measurement in the road plane to solve for the unknown camera height h.

A known distance Δx along the $y = h\cot\phi$ axis corresponding to the pixel distance

Δu along the u-axis can be used to solve for h. From (4.15)

$$h = \left(f\,\frac{\Delta x}{\Delta u} - y\cos\phi\right)\frac{1}{\sin\phi}\,.$$

Using (4.16) we substitute for y:

$$h = \frac{f\,\Delta x\,\sin\phi}{\Delta u}\,. \tag{4.23}$$

Either the lane width (w_r) or the average vehicle width (w_v) can be used to solve for h. As shown in Figure 4.4, the length of a segment connecting the intersections of two adjacent lanes with any axis parallel to the X-axis is $w_r \sec\theta$. Similarly the projection of vehicle's width on the X-axis is $w_v \cos\theta$. Substituting $\Delta x = w_r \sec\theta$ and $\Delta u = \Delta u_r$ in (4.23) we obtain an expression for the height h of the camera using two vanishing points (which yield f,ϕ, and θ) and known lane width.

$$h = \frac{f\,w_r\cos\theta\sin\phi}{\Delta u_r}\,. \tag{4.24}$$

Similarly substituting $\Delta x = w_v \cos\theta$ and $\Delta u = \Delta u_v$ in (4.23) we obtain an expression for the height h using average vehicle width:

$$h = \frac{f\,w_v\cos\theta\sin\phi}{\Delta u_v}\,. \tag{4.25}$$

4.2.3 Two vanishing points and known length (VVL)

As shown in Figure 4.4, length information (l_v) can be easily incorporated using the equations derived in the previous subsection by observing that $\Delta x = l_v \sin\theta$ and $\Delta u =$

Δu_L.

$$h = \frac{f\, l_v \sin\theta \sin\phi}{\Delta u_L}\,. \tag{4.26}$$

However, as θ approaches 0, Δu_L reduces to a point. Another way to incorporate length information is by using (4.14). Assuming that y coordinate of the point on the road corresponding to the point $p_f = [\,u_f \quad v_f\,]^T$ in the image is y_f, then it can be seen from Figure 4.4 that the y coordinate of the point corresponding to the image point $p_b = [\,u_b \quad v_b\,]^T$ is $y_f + l_v \cos\theta$. Substituting in 4.14 we get:

$$\begin{aligned} y_f &= \frac{h(f + v_f \tan\phi)}{v_f + f\tan\phi} \\ y_f &= \frac{h(f + v_b \tan\phi)}{v_b + f\tan\phi} - l_v \cos\theta\,. \end{aligned}$$

Equating the two equations above and solving for h yields

$$h = \frac{f\, l_v \cos\theta (v_f - v_0)(v_b - v_0)}{(v_f - v_b)(f^2 + v_0^2)}\,. \tag{4.27}$$

4.2.4 One vanishing point, known width and length (VWL)

Estimating the vanishing point in the direction orthogonal to the direction of traffic flow is much harder compared to estimating the vanishing point in the direction of traffic flow. Let us now derive the equations for calibrating the camera using a single vanishing point (u_0, v_0), a known length measurement (measurement along the direction of traffic flow) and a known width measurement (along the direction orthogonal to traffic flow).

Equations (4.17) and (4.18) can be used to derive following relationships:

$$\sin^2\phi = v_0^2/(f^2 + v_0^2) \tag{4.28}$$

$$\cos^2\phi = f^2/(f^2+v_0^2) \tag{4.29}$$

$$\sin^2\theta = u_0^2/(u_0^2+f^2+v_0^2) \tag{4.30}$$

$$\cos^2\theta = (f^2+v_0^2)/(u_0^2+f^2+v_0^2)\,. \tag{4.31}$$

Now by equating (4.24) and (4.27) we derive a fourth order equation in f as follows:

$$f^4 + f^2\left[2(u_0^2+v_0^2) - \frac{k_1^2}{v_0^2}\right] + \left[u_0^4+v_0^4+2u_0^2v_0^2-k_1^2\right] = 0 \tag{4.32}$$

where

$$k_1 = \frac{\Delta u_r(v_f-v_0)(v_b-v_0)l_v}{w_r(v_f-v_b)}\,.$$

The above equation is quadratic in f^2 and can be solved to estimate the focal length. It is straight forward to compute ϕ and θ from (4.17) and (4.18). Finally, height h of the camera can be found by using either (4.24), (4.25), or (4.27). It should be noted that (4.32) which was derived for lane width w_r also holds true for vehicle width by substituting w_v in place of w_r and Δu_v in place of Δu_r.

4.2.5 One vanishing point, known width and camera height(VWH)

A camera placed at a known height above the road plane can be calibrated using a single vanishing point p_0 (in the direction of traffic flow) and a measurement along the width of the road i.e. (w_r, Δu_r) or (w_v, Δu_v).

Squaring both sides of (4.24) and rearranging using (4.28)-(4.31) we get

$$(1-k_2^2)f^4 + \left[2v_0^2 - k_2^2(u_0^2+v_0^2)\right]f^2 + v_0^4 = 0 \tag{4.33}$$

Algorithm	Known quantities	Vanishing points	Image measurements	Comments
VVH	camera height	p_0, p_1	none	works even in dense traffic conditions
VVW	lane/vehicle width	p_0, p_1	Δu_r or Δu_v	
VVL	length measurement	p_0, p_1	p_f, p_b	works in moderate traffic
VWL	lane/vehicle width and length measurement	p_0	p_f, p_b and Δu_r or Δu_v	
VWH	lane/vehicle width and camera height	p_0	Δu_r or Δu_v	works even for head-on view and also in dense traffic conditions
VLH	length measurement and camera height	p_0	p_f, p_b	works in moderate traffic

Table 4.1: Comparison between different method of calibrating a traffic monitoring camera.

where

$$k_2 = \frac{w_r v_0}{h \Delta u_r}.$$

4.2.6 One vanishing point, known length and camera height(VLH)

The last scenario that is considered here estimates f using a single length measurement (along the length of the road) when the height of the camera and vanishing point p_0 is known.

Squaring both sides of (4.27) and rearranging using (4.28)-(4.31) we get

$$f^4 + f^2 \left[u_0^2 + 2v_0^2 - \frac{l_v^2 k_3^2}{h^2} \right] + \left[v_0^2 (u_0^2 + v_0^2) \right] = 0 \tag{4.34}$$

where

$$k_3 = \frac{(v_f - v_0)(v_b - v_0)}{(v_f - v_b)}.$$

Chapter 5

Automatic calibration of traffic monitoring cameras

In this chapter, an algorithm to automatically calibrate a road-side traffic monitoring camera is presented that overcomes several of the limitations of previous approaches [17, 60, 58]. The algorithm does not require pavement markings or prior knowledge of the camera height or lane width; it is unaffected by spillover, occlusion, and shadows; and it works in dense traffic and different lighting and weather conditions. The key to the success of the system is a BCVD described in Chapter 3. Since vehicles are detected and tracked using their intensity patterns in the image, the algorithmdoes not suffer from the well-known drawbacks of background subtraction or frame differencing. The technique uses the vehicle trajectories in the image and the intensity gradient along the vehicle windshield to compute the two vanishing points in the image, from which the camera parameters (height, focal length, and pan and tilt angles) are estimated.

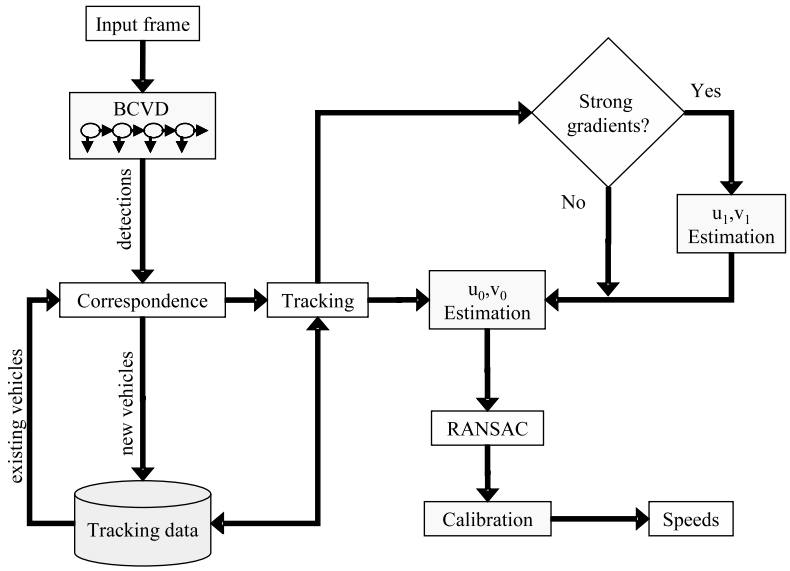


Figure 5.1: Overview of the algorithm for automatic camera calibration.

5.1 Proposed approach

Figure 1 presents an overview of the implemented system. The bulk of the processing is performed by the BCVD (as described in Chapter 3), which is used to detect and track vehicles. The resulting vehicle tracks are then used to estimate the first vanishing point in the direction of travel, while strong gradients near vehicle windshields (in daytime) or the lines joining the two headlights (at night) are used to compute the second vanishing point in the direction perpendicular to the direction of travel. The Random Sample Consensus (RANSAC) algorithm [20] is used to eliminate outliers resulting from noise and/or image compression artifacts. From the vanishing points, the camera is calibrated, which then enables the speed of vehicles to be computed by mapping pixel coordinates to world distances. The only parameter of the system is the mean vehicle width, which is assumed to be 7 feet [3].

One useful characteristic of the approach based on two vanishing points and vehicle-width measurement is that the system is calibrated incrementally. In other words, only two

images of a single vehicle are needed in principle to calibrate the system, thus providing a nearly instantaneous solution to the problem. This unique behavior eliminates the delay inherent in background subtraction techniques, which makes the system amenable for use by PTZ cameras whose parameters are continually changing. In practice, although the first vehicle is used to obtain initial calibration parameters, those parameters are refined over time as more vehicles are detected and tracked in order to obtain more accurate estimates. Additional advantages of the approach include its immunity to shadows (Note that Dailey et al. [17] observed more than 10% error in mean speed estimates due to shadows), as well as its insensitivity to spillover and/or dense traffic, since vehicles are detected using a discriminative set of features as opposed to simple foreground blobs.

5.1.1 Estimating the vanishing point in the direction of traffic flow

Lines which are parallel to each other in the real world generally do not appear parallel in the image (except when they are parallel to the image plane). As an example, consider an aerial photograph of rail-road tracks with the camera looking straight down. The tracks will appear parallel to each other in the image. If another image is taken standing in the middle of the tracks and pointing the camera straight ahead (camera looking towards horizon), the tracks will appear to meet at a finite point in the image plane. This point of intersection is called a vanishing point. A vanishing point is defined only by the direction of lines, in other words, all parallel lines in a particular direction will appear to converge at a single unique location in the image. The vanishing point $p_0 = [\, u_0 \quad v_0 \,]^T$ in the direction of travel is estimated using vehicle tracks. A line is fitted passing through bottom-left and bottom-right image coordinates of all the detection windows for a vehicle. Estimating the vanishing point directly from the vehicle tracks avoids using computationally expensive Hough transform. Figure 3 (a) illustrates a scenario where a vehicle changing lanes (rep-

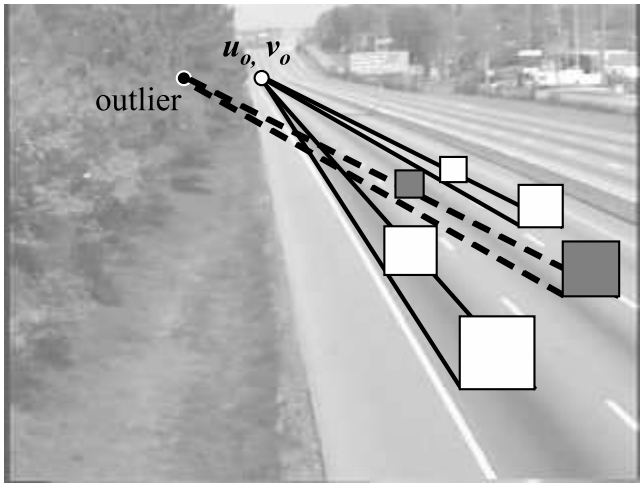
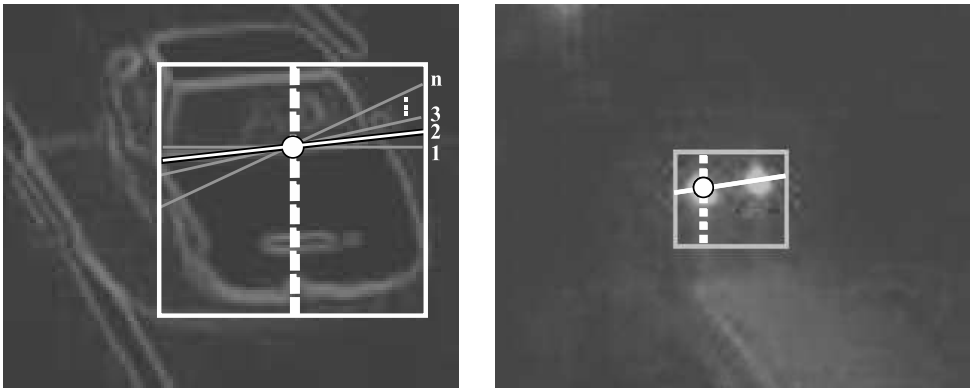


Figure 5.2: Estimation of the vanishing point in the direction of traffic flow.

resented by darker rectangle) results into an outlier. In addition, tracking and localization errors can lead to outliers. RANSAC was used for removing the bias in the estimation of vanishing points resulting from outliers.

5.1.2 Estimating the vanishing point orthogonal to the direction of traffic flow

To estimate the vanishing point $p_1 = [\, u_1 \quad v_1 \,]^T$ in the direction perpendicular to traffic-flow, strong image gradients found on light colored vehicles are employed. Apparent slope of a line in an image (corresponding to a line in real world along the direction perpendicular to traffic-flow) is inversely proportional to its distance from the camera. Estimating p_1 as the intersection of two lines in its direction is very sensitive to measurement errors. With the assumption that the camera has zero roll, p_1 can be found as the intersection of $v = v_0$ and a line corresponding to the perpendicular direction. The detection window that is closest to the camera (close to the bottom edge of an image) is used to search for a hinge point, which is a point of maximum gradient magnitude and lies along the vertical axis passing through the center of the window (along the dashed line). Next, a line is searched



(a) (b)

Figure 5.3: Estimation of the vanishing point in the direction orthogonal to the direction of traffic flow. (a) Strong gradients near windshield are used for daytime (b) Estimated centers of headlights are used for nighttime.

passing through the hinge point and having a slope that maximizes the sum of gradients along that line. In Figure 5.3(a), the white circle indicates the location of the hinge point. Among all the candidates, the line that coincides with the edge of the windshield of the vehicle (line #2) is used to compute p_1. In case of absence of any ambient light, headlights are used to estimate p_1. The hinge point is found along a vertical axis shifted to left by quarter of detection window width as shown in Figure 5.3(b). Note that raw pixel intensities are used in this case as opposed to gradient magnitude image used earlier.

5.1.3 Computing calibration parameters

Once p_0 and p_1 are estimated, for each vehicle detection points $p_2 = [\, u_2 \quad 0 \,]^T$ and $p_3 = [\, u_3 \quad 0 \,]^T$ can be found as intersection of u-axis with the lines connecting p_0 to the two bottom vertices of the detection rectangle. Now using the equations derived in Section 4.2.2, the focal length f in pixels, the tilt angle ϕ, the pan angle θ and the height of the camera h in feet can be computed using following equations:

$$f = \sqrt{[-(v_0^2 + u_0 u_1)]}$$

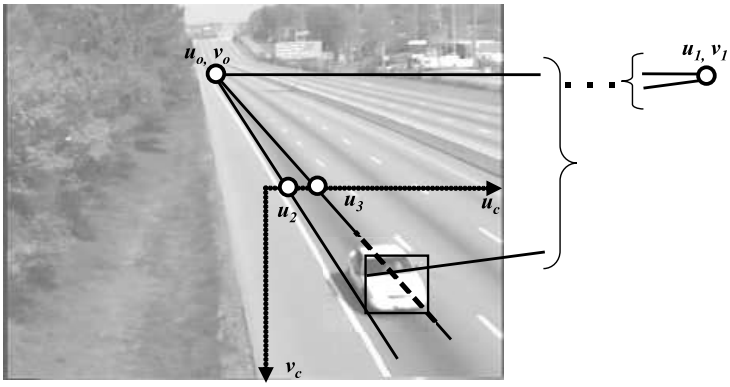


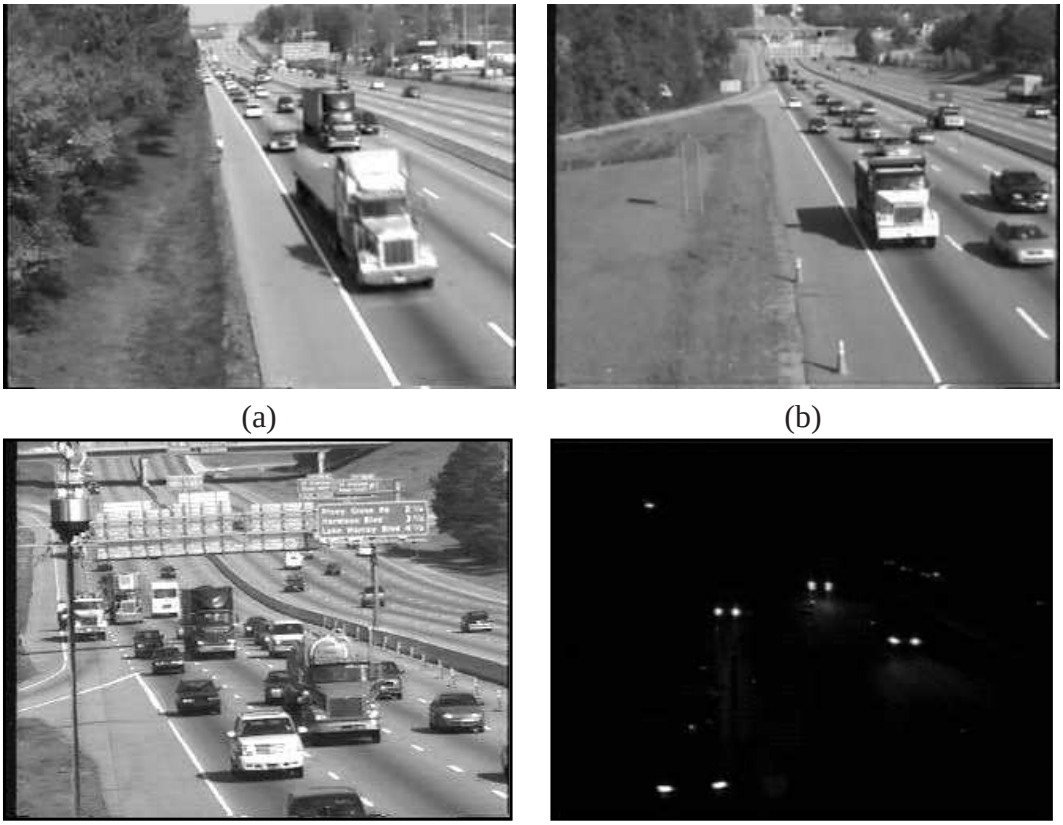
Figure 5.4: Calibration parameters are computed using the four points shown above and from assumed mean width of a vehicle.

$$
\begin{aligned}
\phi &= \tan^{-1}\left(\frac{-v_0}{f}\right) \\
\theta &= \tan^{-1}\left(\frac{-u_0 \cos\phi}{f}\right) \\
h &= \frac{f w_r \cos\theta \sin\phi}{\mid u_3 - u_2 \mid}
\end{aligned}
$$

As more vehicles are detected, estimates of p_0 and p_1 are recomputed from all previous detections using RANSAC and estimate of $\mid u_3 - u_2 \mid$ is recomputed as mean of all previous $\mid u_3 - u_2 \mid$ measurements.

Once the camera has been calibrated, the pixel location of a vehicle in the image (u, v) can be mapped into a location on the road (x, y) using following equations:

$$
\begin{aligned}
x &= \frac{hu \cos\phi(1 + \tan^2\phi)}{v + f\tan\phi} \\
y &= \frac{h(f + v\tan\phi)}{v + f\tan\phi}
\end{aligned}
$$



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

Figure 5.5: Training sequences for the BCVD, (a)-(c) daytime (d) nighttime.

5.2 Experimental results

Two BCVDs were trained (one for the daytime, and one for the nighttime) using the training sequences shown in Figure 5.5. At run time, the system automatically selects the proper detector (day or night) based on the average pixel intensity in the images. To test the system, four image sequences were captured, three during daylight conditions and one at night, using an inexpensive off-the-shelf web camera (Logitech Orbitz) mounted at the top of an adjustable pole. An image from each sequence is shown in Figure 5.6. The images were captured at 15 frames per second at 320x240 pixel resolution. Note that different cameras were used for capturing the training and test sequences, and that the cameras were not placed in the same location, thus demonstrating the robustness of the system.

Figure 5.6 also shows the results overlaid on the images. The rectangles outline the detected vehicles; the false negatives are not a problem since the goal here is mean speed rather than vehicle counts. The white circle indicates the first vanishing point, which is only visible in two of the four test sequences. The second vanishing point is very far from the image and is given by the intersection of the horizon line and the other line drawn. It should be noted that the slope of the line corresponding to the second vanishing point is determined by the image gradients computed near windshields of vehicles and does not depend on the road lane markings.

The sequences were approximately 10 minutes long each. A radar was used to compare the mean speed over the entire sequence for three of the sequences, with the results displayed in the table below. Treating the radar as ground truth, the error of the system ranged from 3 to 6 mph, with a slightly greater standard deviation than the radar. Figure 5.7 shows the error in the distance estimate (displayed as a percentage) versus the amount of data that the algorithm was allowed to use. As mentioned previously, the algorithm instantaneously yields initial estimate, which improves over time as more information is gathered. In two of the sequences the estimate stabilized after only ten vehicles, while the poor weather conditions of the third sequence caused the estimate to require more data.

Table 5.1 shows the accuracy of the estimation of the camera parameters for the four sequences. Accuracy was computed by comparing with camera parameters obtained using the same equations but with hand-labeled vanishing points.

Table 5.2 displays the speed error for twenty individual vehicles in each of the four sequences. The average error ranges from 3 to 6 mph. For the three daytime sequences, speed of every 20th vehicle which was tracked for at least 50 feet was compared with the ground truth speed. For T4 (which is a night time sequence) speed of every 10th vehicle was compared since the sequence contained fewer vehicles. Ground truth speed was measured by advancing the sequence frame by frame to measure time and using markers placed at

(T1) (T2)

(T3) (T4)

Figure 5.6: (T1)-(T4) Four test sequences. (T1) Sequence 1, h = 15 feet, clear day. (T2) Sequence 2, h = 30 feet, clear day. (T3) Sequence 3, h = 30 feet, rain with headlight reflections. (T4) Sequence 4, h = 20 feet, night time, no ambient lighting. The white circle shows the estimated location of p_0 vanishing point. The vanishing point p_1 lies outside the image (intersection of the two lines) and hence could not be shown in the above results.

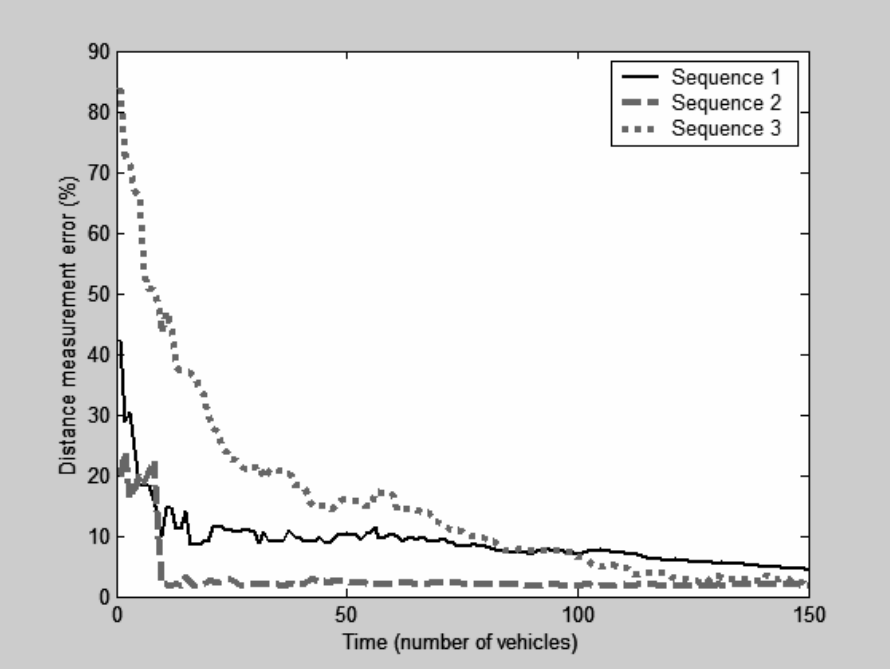


Figure 5.7: Calibration error decreases with increasing number of vehicle detections.

known distances in the scene to measure the distance traveled by the vehicle. Instances where there is a large discrepancy between the speed estimated by the algorithm and the ground truth speed are due to tracking errors (e.g., vehicle 184 and 387 in Sequence T3). Note that vehicle numbers (ID) do not increase by a fixed amount since some of the spurious detections are discarded during tracking and only vehicles which are tracked for more than 50 feet are retained for speed comparison.

To judge the feasibility of the assumptions made about the camera (square pixels, principal point at image center, and zero skew) we calibrated two cameras (Logitech Orbit MP webcam and a PTZ270 high speed dome camera) in the lab using a calibration target (chess-board pattern). The algorithm for calibrating a camera using a planar target was proposed by Zhang [68]. The implementation of Zhang's algorithm by Jean-Yves Bouguet was used to compute the intrinsic camera parameters. The signal from the dome camera was digitized at 320×240 pixel resolution using VideoHome GrabBeeX-light USB video

	Sequence T1		**Sequence T2**		**Sequence T3**		**Sequence T4**	
	Manual	**Algorithm**	**Manual**	**Algorithm**	**Manual**	**Algorithm**	**Manual**	**Algorithm**
f (pixels)	367.28	327.52	342.71	368.92	312.56	348.21	386.21	360.32
φ (degrees)	8.14°	7.21°	16.71°	14.26°	13.71°	12.68°	7.52°	8.17°
θ (degrees)	13.77°	14.19°	20.42°	18.61°	22.38°	19.74°	17.26°	18.93°
h (feet)	13.70	14.2	31.86	29.69	31.17	28.83	20.56	18.62
	Sequence T1		**Sequence T2**		**Sequence T3**			
	Radar	**Algorithm**	**Radar**	**Algorithm**	**Radar**	**Algorithm**		
μ	61.81	63.92	62.22	61.62	54.3	51.66		
σ	4.42	5.97	3.77	4.78	3.7	5.12		
N	187	520	235	491	196	416		

Table 5.1: Accuracy of the estimated parameters compared with parameters computed manually. f is the focal length, ϕ is the tilt angle, θ is the pan angle, h is the camera height. μ, σ and N are mean speed for the entire sequence, standard deviation of speeds and number of observations used for computation.

capture device. Images obtained from the Logitech camera have the same resolution (320×240). As shown in Table 5.3 both the square pixel assumption and the zero skew assumption cause negligible errors in both the cameras. The principal point is off center by about 7, and 5 pixels in u and v directions, respectively, for the Logitech camera. For the dome camera the principal point is off center by about 10 and 24 pixels in the u and v directions, respectively.

Sequence T1					**Sequence T2**			
Vehicle ID	**Lane**	**Measured Speed**	**Algorithm Speed**		**Vehicle ID**	**Lane**	**Measured Speed**	**Algorithm Speed**
1	1	53	51		2	2	58	55
27	2	54	58		24	3	62	60
52	1	55	51		47	2	55	56
78	2	58	63		69	1	54	54
101	1	62	59		92	1	57	56
127	1	57	53		115	2	62	64
149	1	59	55		137	3	64	61
172	2	64	68		160	2	53	51
195	2	63	68		183	3	61	56
207	2	58	63		205	1	53	51
229	1	56	53		229	1	57	55
252	1	52	51		254	2	58	57
273	1	55	58		275	3	63	58
298	2	62	57		298	4	68	61
320	2	59	55		321	4	62	57
344	2	61	58		346	3	57	52
367	1	53	50		368	3	61	59
392	2	61	57		392	4	66	62
415	1	62	58		413	3	58	55
439	1	56	52		436	4	62	58
	Mean absolute error (mph)		**3.7**			**Mean absolute error (mph)**		**3.0**
Sequence T3					**Sequence T4**			
1	2	56	51		1	2	53	55
22	3	62	56		7	2	55	58
45	1	54	51		13	1	48	47
69	2	58	53		20	2	53	57
93	3	63	59		26	1	47	44
116	1	53	50		32	1	46	45
138	1	58	53		39	2	58	59
161	2	61	57		46	1	51	51
184	3	64	49		51	2	56	58
214	2	60	55		58	2	53	56
236	1	56	53		64	1	50	48
263	3	65	61		71	1	52	51
288	1	59	56		77	2	64	68
312	4	67	60		82	1	54	52
335	3	62	59		87	1	49	44
364	1	54	50		93	1	50	51
387	4	63	38		100	2	63	65
411	1	51	48		106	2	67	70
436	2	53	46		112	2	58	62
463	2	56	52		117	1	48	46
	Mean absolute error (mph)		**5.9**			**Mean absolute error (mph)**		**2.3**

Table 5.2: Ground-truth speeds were measured manually by observing the video with the help of markers placed in the scene. Vehicles were chosen at fixed intervals to compare accuracy of speed estimation.

Camera	f_x	f_y	Aspect ratio	Skew	Principal point
Logitech Orbit MP	295.31	287.71	1.03	0.00	[153.42, 115.58]
PTZ270 Dome camera	437.08	434.05	1.01	0.00	[170.20, 144.55]

Table 5.3: Intrinsic parameters for the two cameras.

Chapter 6

Conclusion

Previous approaches to segmenting and tracking vehicles using video generally require the camera to be placed high above the ground in order to minimize the effects of occlusion and spillover. A technique was presented that overcomes this limitation, working when the camera is relatively low to the ground and beside the road. The approach is based upon identifying and grouping feature points in each image frame whose 3D coordinates can be computed in a manner that is relatively immune to the effects of perspective projection. The novelty of the work includes an incremental, on-line, real-time algorithm to estimate the heights of features using a combination of background subtraction, perturbed plumb line projections, projective transformation, and a region-based grouping procedure. Experimental results on a variety of image sequences demonstrate the ability of the algorithm to automatically segment, track, and classify vehicles in low-angle sequences. These results include situations involving severe occlusions in which the vehicle remains partially occluded throughout the sequence, which has proved to be a particularly challenging scenario for previous approaches.

The ability to track vehicles using low-angle cameras opens several possibilities for highway monitoring, such as supporting automated transient traffic studies in locations

unable to afford the infrastructure necessary for mounting cameras high above the ground. In addition, by addressing the important problem of occlusion, many of the concepts contained in this work are directly applicable to existing high-angle scenarios with a large number of traffic lanes, in which large trucks often occlude neighboring vehicles.

To alleviate the requirement of calibrating the camera manually, a method for automatic calibration of roadside traffic monitoring cameras was presented using a Boosted Cascade Vehicle Detector (BCVD). The BCVD detects vehicles in images by comparing the 2D intensity patterns with a model acquired during an off-line, one-time training phase. The training does not have to be performed on images captured at the same location or by the same camera as those used at run-time. The technique overcomes many of the limitations of the common approaches of background subtraction or frame differencing. For example, an estimate is available immediately upon detecting and tracking a single vehicle between two image frames, thus supporting applications such as Pan-Tilt-Zoom (PTZ) cameras in which it may not be feasible to allow the algorithm to learn the background model every time the camera is moved. In addition, the technique is insensitive to shadows, spillover, occlusion, and environmental conditions, and it is applicable in daytime or nighttime scenarios.

It is evident from the results presented in Chapter 2 that the system for detection and tracking of vehicles using stable features works under wide variety of camera placement. However the approach based on stable features has its limitations, one of them being the inability to detect vehicles due to headlight reflections. On the other hand BCVD performed better than stable features in adverse weather conditions, however BCVD performs poorly when the camera placement is considerably different from that during training. By definition, stable features are detected on either the front side or the back side of the vehicle. The pixel-area in the image corresponding to the front side of an approaching vehicle (back side in case of a receding vehicle) decreases as the pan angle increases. As a result

the performance of tracking using stable feature degrades for large pan angles, whereas for very small pan angles measuring lengths of vehicles becomes challenging. The accuracy of tracking is also affected by distance of the camera from the closest lane. A larger pan angle is required to cover all lanes when the camera is placed far from the closest lane, so the camera should be placed as close to the closest lane as possible. For the experiments conducted during this research the highest placement of the camera was about 30 feet from the ground which was sufficient to cover four 12-feet lanes. In all experimental results, pan and tilt angles were in the range of 10° to 30° and distance of the camera from the closest lane was 10-20 feet. It should be noted that the algorithm presented for automatic camera calibration fails for the case of zero pan angle because the vanishing point orthogonal to the direction of travel goes to infinity. In practice the height of the camera computed during non-zero pan angle can be used to calibrate the camera when the pan angle is zero (as shown in 4.2.5).

To further improve this work and enhance its applicability, future work should be aimed at reducing the effects of shadows, supporting continued operation in the presence of changing weather and environmental conditions and more robust strategies for modelling and maintaining the background. Expecting a single pattern detector to perform well under significant variations in vehicle appearances is unrealistic. From the experience of this work, we envision a bank of pattern detectors trained over a small number pan angle, tilt angle, and camera height to cover a wide range of appearance changes. With availability of more processing power, color information can be incorporated for suppressing shadows and for computing feature similarity. Expanding the automatic calibration technique to work with rear-facing vehicles receding from the camera, augmenting the pattern detector with other modalities to decrease convergence time, and introducing partial calibration when some camera parameters are already known from previous iterations of the algorithm.

We believe that this work demonstrates the potential for combining feature tracking-

based and pattern detection-based approaches to detect and track vehicles in highway scenarios, and that it enhances the usefulness of cameras by obviating the need for tedious manual calibration procedures.

Appendices

Appendix A

Derivations

A.1 Derivation for equations of plumb line projections

To derive Equation 2.2 from Equation 2.1, expanding the latter:

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

In inhomogeneous coordinates, this is

$$u = \frac{uw}{w} = \frac{c_{11}x + c_{12}y + c_{13}z + c_{14}}{c_{31}x + c_{32}y + c_{33}z + c_{34}} \tag{A.2}$$

$$v = \frac{vw}{w} = \frac{c_{21}x + c_{22}y + c_{23}z + c_{24}}{c_{31}x + c_{32}y + c_{33}z + c_{34}}. \tag{A.3}$$

Rearranging terms yields:

$$c_{11}x + c_{12}y + c_{13}z + c_{14} = c_{31}xu + c_{32}yu + c_{33}zu + c_{34}u \tag{A.4}$$

$$c_{21}x + c_{22}y + c_{23}z + c_{24} \quad = \quad c_{31}xv + c_{32}yv + c_{33}zv + c_{34}v \tag{A.5}$$

or

$$(c_{31}u - c_{11})x + (c_{32}u - c_{12})y \quad = \quad (c_{13} - c_{33}u)z + (c_{14} - c_{34}u) \tag{A.6}$$

$$(c_{31}v - c_{21})x + (c_{32}v - c_{22})y \quad = \quad (c_{23} - c_{33}v)z + (c_{24} - c_{34}v). \tag{A.7}$$

Without loss of generality (because the projection matrix is only defined up to a scale factor) setting $c_{34} = 1$. Rearranging terms again yields

$$\begin{bmatrix} c_{31}u - c_{11} & c_{32}u - c_{12} & 0 \\ c_{31}v - c_{21} & c_{32}v - c_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c_{14} - u + z(c_{13} - c_{33}u) \\ c_{24} - v + z(c_{23} - c_{33}v) \\ z \end{bmatrix}, \tag{A.8}$$

which is the desired result. ■

To derive Equation 2.5 from Equation 2.1, rearranging the terms in Equations A.6 and A.7 to yield

$$\begin{bmatrix} uc_{33} - c_{13} \\ vc_{33} - c_{23} \end{bmatrix} z = \begin{bmatrix} c_{14} - u\,c_{34} + (c_{11} - u\,c_{31})\,x + (c_{12} - u\,c_{32})\,y \\ c_{24} - v\,c_{34} + (c_{21} - v\,c_{31})\,x + (c_{22} - v\,c_{32})\,y \end{bmatrix} \tag{A.9}$$

or

$$\mathbf{h}_p z = \mathbf{h}_c. \tag{A.10}$$

To solve for z, then, left-multiplying both sides by $\mathbf{h}_p^T$, yields

$$z = (\mathbf{h}_p^T \mathbf{h}_p)^{-1} \mathbf{h}_p^T \mathbf{h}_c, \tag{A.11}$$

which is the desired result. ■

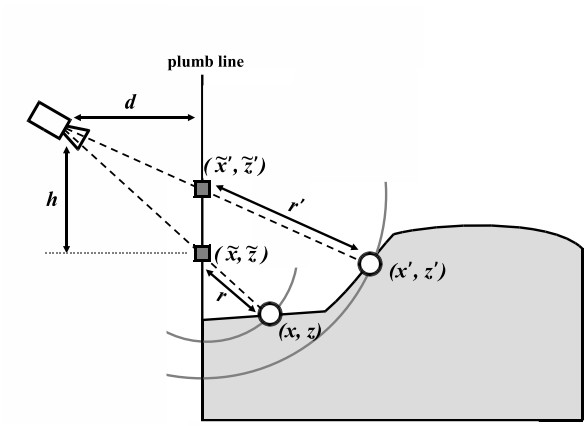


Figure A.1: Derivation of the maximum slope which defines the set Ω.

A.2 Derivation of Maximum Slope Defining Ω

To define Ω, consider a pinhole camera viewing a vehicle, as shown in Figure A.1. Let us set the origin to the camera focal point, with the positive x axis to the right and the positive z axis up. Select an arbitrary point (x, z) on the vehicle whose estimated location using PLP is the point $(\tilde{x}, \tilde{z})$. Let d be the horizontal distance from the camera focal point to the plumb line (i.e., the distance $\tilde{x}$), h the vertical distance from the camera focal point to the point $(\tilde{x}, \tilde{z})$, and r the distance along the projection ray between the actual point and the estimated point. For convenience, define $m = \sqrt{d^2 + h^2}$.

The point (x, z) is the intersection of the projection ray with a circle centered at location $(d, -h)$ of radius r:

$$(x, z) = \left(d(1 + \frac{r}{m}), -h(1 + \frac{r}{m}) \right).$$

In order for the error to be a monotonically non-decreasing function of z, another point (x', z') higher on the vehicle (i.e., $z' > z$) must have a radius $r' \geq r$,
where $r = \sqrt{(x - \tilde{x})^2 + (z - \tilde{z})^2}$ is the Euclidean error in the estimate. As a result, con-

sider another circle at location $(d, -(h - \Delta h))$, whose intersection with the corresponding projection ray yields the other point:

$$(x', z') = \left(d(1 + \frac{r'}{m'}), -h'(1 + \frac{r'}{m'}) \right),$$

where $h' = h - \Delta h$, $m' = \sqrt{d^2 + h'^2}$, and $r' = \sqrt{(x' - \tilde{x}')^2 + (z' - \tilde{z}')^2}$.

The slope of the curve is given by $\frac{dz}{dx} = \lim_{\Delta h \to 0} \frac{z' - z}{x' - x}$. To achieve the maximum bound on the allowed slope, setting $r' = r$, yielding

$$\begin{aligned}
\frac{dz}{dx} \leq \mu_{\max}(x, z) &= \lim_{\Delta h \to 0} \frac{\Delta h(1 + \frac{r}{m'}) - rh(\frac{1}{m'} - \frac{1}{m})}{dr(\frac{1}{m'} - \frac{1}{m})} \\
&= \lim_{\Delta h \to 0} \frac{\Delta h + (r\Delta h - rh)(m^2 - 2\Delta hh + (\Delta h)^2)^{-1/2} + rh/m}{dr(m^2 - 2\Delta hh + (\Delta h)^2)^{-1/2} - dr/m} \\
&= \lim_{\Delta h \to 0} \frac{1 + rA^{-1/2} + (r\Delta h - rh)A^{-3/2}(-1/2)(-2h + 2\Delta h)}{-\frac{1}{2}drA^{-3/2}(-2h + (\Delta h)^2)} \\
&= \lim_{\Delta h \to 0} \frac{1 + rm^{-1} + rh^2 m^{-3}}{-\frac{1}{2}dr(m^2)^{-3/2}(-2h)} \\
&= \frac{m^3 + rm^2 - rh^2}{2drh} \\
&= \frac{m^3 + r(d^2 + h^2) - rh^2}{2drh} \\
&= \frac{m^3 + rd^2}{drh},
\end{aligned}$$

where the last equality is obtained by l'Hôpital's rule, and where $A = (m^2 - 2\Delta hh + (\Delta h)^2)$. As long as the slope of the vehicle shape is bounded by this number, the estimation error by PLP is a monotonically non-decreasing function of height. This bound includes all convex shapes not crossing the plumb line, as well as many shapes with significant concavities, thus covering nearly all vehicles encountered in practice.

Bibliography

[1] The Infra-Red Traffic Logger (TIRTL), http://www.ceos.com.au/products/tirtl-content.htm.

[2] Intel OpenCV library, http://www.intel.com/research/mrl/research/opencv/.

[3] New Jersey department of transportation, Roadway Design Manual, http://www.state.nj.us/transportation/eng/documents/RDME/sect2E2001.shtm.

[4] Evaluation of non-intrusive technologies for traffic detection. Minnesota department of transportation, office of traffic engineering/its section, october 2001. http://www.dot.state.mn.us/guidestar/pdf/niteval/vol1test.pdf.

[5] G. Alessandretti, A. Broggi, and P. Cerri. Vehicle and guard rail detection using radar and vision data fusion. *IEEE Transactions on Intelligent Transportation Systems*, 8(1):95–105, Mar. 2007.

[6] S. Atev, H. Arumugam, O. Masoud, R. Janardan, and N. P. Papanikolopoulos. A vision-based approach to collision prediction at traffic intersections. *IEEE Transactions on Intelligent Transportation Systems*, 6(4):416–423, 2005.

[7] 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.

[8] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A real time computer vision system for measuring traffic parameters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 495–501, 1997.

[9] J.-Y. Bouguet. Pyramidal implementation of the lucas kanade feature tracker (description of the algorithm).

[10] J.-Y. Bouguet. Pyramidal implementation of the Lucas Kanade feature tracker. OpenCV documentation, Intel Corporation, Microprocessor Research Labs, 1999.

[11] A. Chachich, A. Pau, A. Barber, K. Kennedy, E. Olejniczak, J. Hackney, Q. Sun, and E. Mireles. Traffic sensor using a color vision method. In *Proceedings of the SPIE*, volume 2902, pages 155–164, 1997.

[12] S.-C. Chen, M.-L. Shyu, S. Peeta, and C. Zhang. Learning-based spatio-temporal vehicle tracking and indexing for transportation multimedia database systems. *IEEE Transactions on Intelligent Transportation Systems*, 4(3):154–167, Sept. 2003.

[13] S. C. Cheung and C. Kamath. Robust techniques for background subtraction in urban traffic video. In *Proceedings of Electronic Imaging: Visual Communications and Image Processing*, 2004.

[14] 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.

[15] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Statistic and knowledge-based moving object detection in traffic scenes. In *IEEE Conference on Intelligent Transportation Systems*, 2000.

[16] R. Cucchiara, C. Grana, M. Piccardi, A. Prati, and S. Sirotti. Improving shadow suppression in moving object detection with HSV color information. In *Proceedings of the IEEE Conference on Intelligent Transportation Systems*, pages 334–339, Aug 2001.

[17] 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.

[18] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, second edition, 2000.

[19] J. M. Ferryman, A. D. Worrall, and S. J. Maybank. Learning enhanced 3D models for vehicle tracking. In *Proceedings of the British Machine Vision Conference*, pages 873–882, 1998.

[20] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[21] 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.

[22] S. Gupte, O. Masoud, R. F. K. Martin, and N. P. Papanikolopoulos. Detection and classification of vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 3(1):37–47, Mar. 2002.

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

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

[25] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[26] X. C. He and N. H. C. Yung. New method for overcoming ill-conditioning in vanishing-point-based camera calibration. *Optical Engineering*, 46(2), 2007.

[27] T. Horprasert, D. Harwood, and L. S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *Proceedings of the IEEE Frame Rate Workshop*, pages 1–19, Kerkyra, Greece, 1999.

[28] J.-W. Hsieh, S.-H. Yu, Y.-S. Chen, and W.-F. Hu. Automatic traffic surveillance system for vehicle tracking and classification. *IEEE Transactions on Intelligent Transportation Systems*, 7(2):175–187, 2006.

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

[30] Y.-K. Jung, K.-W. Lee, and Y.-S. Ho. Content-based event retrieval using semantic scene interpretation for automated traffic surveillance. *IEEE Transactions on Intelligent Transportation Systems*, 2(3):151–163, Sept. 2001.

[31] 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 (ITS)*, 2001.

[32] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi. Traffic monitoring and accident detection at intersections. *IEEE Transactions on Intelligent Transportation Systems*, 1(2):108–118, June 2000.

[33] N. Kanhere, S. Birchfield, and W. Sarasua. Automatic camera calibration using pattern detection for vision-based speed sensing. In *TRB 87th Annual Meeting Compendium of Papers, Transportation Research Board Annual Meeting*, 2008.

[34] N. Kanhere, S. Birchfield, W. Sarasua, and T. Whitney. Real-time detection and tracking of vehicle base fronts for measuring traffic counts and speeds on highways. In *TRB 86th Annual Meeting Compendium of Papers, Transportation Research Board Annual Meeting*, 2007.

[35] N. K. Kanhere and S. T. Birchfield. Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features. *IEEE Transactions on Intelligent Transportation Systems*, 9(1):148–160, Mar. 2008.

[36] N. K. Kanhere, S. T. Birchfield, and W. A. Sarasua. Vehicle segmentation and tracking in the presence of occlusions. *Transportation Research Record, Journal of the Transportation Research Board*, 1944:89–97, 2006.

[37] N. K. Kanhere, S. J. Pundlik, and S. T. Birchfield. Vehicle segmentation and tracking from a low-angle off-axis camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1152–1157, June 2005.

[38] J. Kato, T. Watanabe, S. Joga, Y. Lui, and H. Hase. An HMM/MRF-based stochastic framework for robust vehicle tracking. *IEEE Transactions on Intelligent Transportation Systems*, 5(3):142–154, Sept. 2004.

[39] 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.

[40] Z. Kim. Realtime object tracking based on dynamic feature grouping with background subtraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[41] Z. W. Kim and J. Malik. Fast vehicle detection with probabilistic feature grouping and its application to vehicle tracking. In *Proceedings of the International Conference on Computer Vision*, volume 1, pages 521–528, 2003.

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

[43] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In *Proceedings of the European Conference on Computer Vision*, pages 189–196, 1994.

[44] P. Kumar, S. Ranganath, H. Weimin, and K. Sengupta. Framework for real-time behavior interpretation from traffic video. *IEEE Transactions on Intelligent Transportation Systems*, 6(1):43–53, 2005.

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

[46] D. R. Magee. Tracking multiple vehicles using foreground, background and motion models. *Image and Vision Computing*, 22(2):143–155, Feb. 2004.

[47] 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.

[48] O. Masoud, N. P. Papanikolopoulos, and E. Kwon. The use of computer vision in monitoring weaving sections. *IEEE Transactions on Intelligent Transportation Systems*, 2(1):18–25, Mar. 2001.

[49] J. Melo, A. Naftel, A. Bernardino, and J. Santos-Victor. Detection and classification of highway lanes using vehicle motion trajectories. *IEEE Transactions on Intelligent Transportation Systems*, 7(2):188–200, 2006.

[50] P. G. Michalopoulos. Vehicle detection video through image processing: the autoscope system. *IEEE Transactions on Vehicular Technology*, 40(1):21–29, Feb 1991.

[51] D. Middleton, D. Gopalakrishna, and M. Raman. Advances in traffic data collection and management (white paper). http://www.itsdocs.fhwa.dot.gov/JPODOCS/REPTS_TE/13766.html.

[52] K. Onoguchi. Shadow elimination method for moving object detection. In *Proceedings of the IAPR International Conference on Pattern Recognition*, pages 583–587, 1998.

[53] A. Prati, I. Mikic, M. M. Tridevi, and R. Cucchiara. Detecting moving shadows: Algorithms and evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7), July 2003.

[54] P. Rosin and T. Ellis. Image difference threshold strategies and shadow detection. In *Proceedings of the 6th British Machine Vision Conference (BMVC)*, pages 347–356, 1995.

[55] N. Saunier and T. Sayed. A feature-based tracking algorithm for vehicles in intersections. In *Proceedings of the The 3rd Canadian Conference on Computer and Robot Vision*, 2006.

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

[57] R. J. Schalkoff. *Pattern Recognition: Statistical, Structural and Neural Approaches*. John Wiley & Sons, Inc., first edition, 1991.

[58] 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.

[59] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[60] K.-T. Song and J.-C. Tai. Dynamic calibration of pan-tilt-zoom cameras for traffic monitoring. *IEEE Transactions on Systems, Man, and Cybernetics*, 36(5), Oct. 2006.

[61] J. Stauder, R. Mech, and J. Ostermann. Detection of moving cast shadows for object segmentation. *IEEE Transactions on Multimedia*, 1(1):65–76, Mar. 1999.

[62] T. N. Tan and K. D. Baker. Efficient image gradient based vehicle localization. *IEEE Transactions on Image Processing*, 9(8):1343–1356, 2000.

[63] X. Tao, M. Guo, and B. Zhang. A neural network approach to elimination of road shadow for outdoor mobile robot. In *Proceedings of the IEEE International Conference on Intelligent Processing Systems*, pages 1302–1306, 1997.

[64] M. Trajkovic. Interactive calibration of a PTZ camera for surveillance applications. In *Asian Conference on Computer Vision*, 2002.

[65] H. Veeraraghavan, O. Masoud, and N. P. Papanikolopoulos. Computer vision algorithm for intersection monitoring. *IEEE Transactions on Intelligent Transportation Systems*, 4(2):78–89, June 2003.

[66] P. Viola and M. J. Jones. Robust real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[67] G. B. Z. Sun and R. Miller. On-road vehicle detection using gabor filters and support vector machines. In *IEEE 14th International Conference on Digital Signal Processing*, 2002.

[68] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[69] 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.

[70] T. Zhao and R. Nevatia. Car detection in low resolution aerial image. In *Proceedings of the International Conference on Computer Vision*, volume 1, pages 710–717, 2001.