Real-Time Incremental Segmentation and Tracking of Vehicles at Low Camera Angles Using Stable Features

Neeraj K. Kanhere and Stanley T. Birchfield, *Senior Member, IEEE*

***Abstract*—We present a method for segmenting and tracking vehicles on highways using a camera that is relatively low to the ground. At such low angles, 3-D perspective effects cause significant changes in appearance 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; however, in our sequences, it is not uncommon for vehicles to enter the scene partially occluded and remain so throughout. By utilizing a 3-D perspective mapping from the scene to the image, along with a plumb line projection, we are able to distinguish a subset of features whose 3-D 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. Our technique uses a single grayscale camera beside the road, incrementally processes image frames, works in real time, and produces vehicle counts with over 90% accuracy on challenging sequences.**

***Index Terms*—Feature tracking, occlusion, perspective projection, spillover, vehicle tracking.**

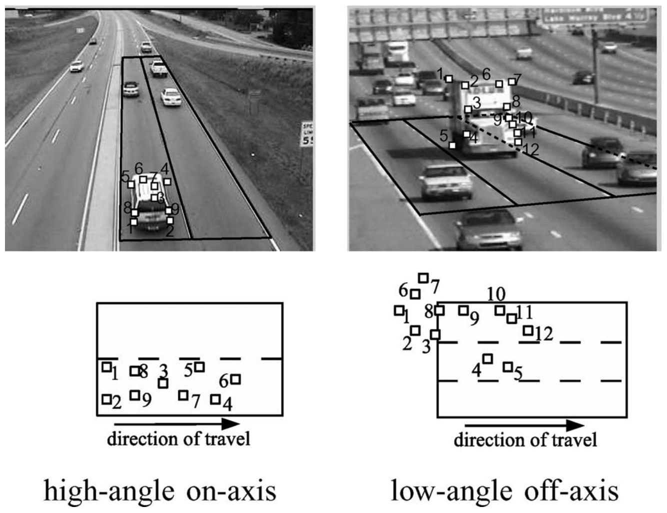


Fig. 1. (Left) When the camera is high above the ground, a homography is sufficient to map the road plane to the image plane. (Right) At lower angles, a homography may cause features on a vehicle to spill onto neighboring lanes.

I. Introduction

Traffic counts, speeds, and vehicle classification are fundamental parameters for a variety of transportation projects, ranging from transportation planning to modern intelligent transportation systems. Among the many technologies, vision-based systems are emerging as an attractive alternative due to their ease of installation, inexpensive maintenance, and ability to capture a rich description of the scene [1]. In principle, videos provide not only aggregate information such as the average speed, vehicle counts, and queue lengths but individual parameters such as trajectories, individual speeds, and classification as well.

Existing vision systems typically place cameras high above the ground, anywhere from 15 to 100 m, to provide a bird's eye view of the road [2]–[4]. At such a high vantage point, the appearance of a vehicle does not significantly change over time, and occlusion between vehicles is considerably reduced, thus simplifying the problem. However, placing cameras at such heights is not always possible. In nonurban areas, the required infrastructure is cost prohibitive, and for transient traffic studies, the expensive mounting equipment and the strategic placement of cameras are precluded by a lack of long-term commitment. In particular, we are interested in developing portable camera systems to gather data about current traffic patterns to assist in planning and safety management.

Fig. 1 illustrates the difference between the two situations. When the camera is high above the ground and near the center of the road, a homography can be defined to map the road surface to the image plane, and the height of vehicles can be safely ignored because their appearance does not significantly change over time. In contrast, when the camera is at a low angle and/or off-centered from the road, the height of the vehicle causes significant occlusion. A single homography (under the flat-world assumption) will not suffice because feature points on a vehicle will spill over into neighboring lanes. Moreover, background subtraction will combine neighboring vehicles into a single combined blob, from which it is not trivial to extract the individual vehicles.

In this paper, we present an automatic method for segmenting and tracking vehicles in video taken by a camera beside the road at a height of approximately 9 m, where occlusion is a

Manuscript received April 27, 2007; revised July 13, 2007 and September 29, 2007. This work was supported in part by a Dwight David Eisenhower Transportation Fellowship from the Federal Highway Administration of the U.S. Department of Transportation and by a Grant from the James E. Clyburn University Transportation Center of South Carolina State University. The Associate Editor for this paper was L. Vlacic.

The authors are with the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC 29634-5124 USA (e-mail: stb@clemson.edu; nkanher@clemson.edu).

Digital Object Identifier 10.1109/TITS.2007.911357

serious issue. The approach is based on sparse features that are incrementally tracked throughout a video sequence using local gradient-based search techniques. A calibration procedure provides a full 3-D to 2-D mapping between the world coordinate system and the image plane, enabling the estimation of the 3-D coordinates of the preimages of the features using a *plumb line projection* (PLP). Features are divided into two categories—stable features that are grouped and tracked to provide the vehicle count, and unstable features that are assigned to the stable groups to produce a more descriptive representation of the vehicles. Because of its reliance on 3-D estimates, the overall approach is able to follow vehicles even when they enter the scene partially occluded and remain so throughout their appearance in the video. The algorithm runs in real time (30 Hz) on a standard computer, and it incrementally processes image frames to allow for online operation.

This paper is organized as follows. After discussing the previous literature in Section II, we present the algorithm in Section III. Its performance on several image sequences of straight and curved highways, with significant shadows and occlusions, is presented in Section IV, followed by our conclusions in Section V.

II. Previous Work

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 [5]–[10]. Many advancements have been made in recent years to the adaptation of the background image to lighting changes [10]–[13] and to the reduction of the effects of shadows [14], [15]. A well-known challenge for background subtraction (as well as with the closely related approach of frame differencing [16]–[20]) occurs when vehicles overlap in the image, causing them to merge into a single foreground blob. Koller *et al.* [21] used 2-D splines to solve this occlusion problem, whereas other researchers employ graph association or split-and-merge rules to handle partial or complete occlusions [11], [12], [19], [22]. Although these solutions can disambiguate vehicles after an occlusion occurs, they require the vehicle to either enter the scene unoccluded or become unoccluded at some point during its trajectory in the camera field of view. In congested traffic, such may never be the case.

An alternative to using temporal information is to match wireframe models to video images [23]–[26]. Ferryman *et al.* [27] combined a 3-D wireframe model with an intensity model of a vehicle to learn the appearance of the vehicle over time. Kim and Malik [2] matched vehicle models with line features from mosaic images captured from cameras on top of a 30-story building next to the freeway to recover detailed trajectories of the vehicles. Alessandretti *et al.* [28] employed a simpler model, namely, the 2-D 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.

A third alternative that has been employed is the tracking of point features. Beymer *et al.* [3] described a system that tracks features throughout the video sequence and then groups the features according to motion cues 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 and Sayed [29] used 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.

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.* [30], 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.* [3] to the case of low-angle cameras, this paper introduces a simple but effective technique for estimating the 3-D coordinates of features in an incremental fashion. Our contribution in this paper 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, our approach handles features that cannot be continually tracked throughout the trajectory, which is a common occurrence in the low-angle situation.

The approach presented in this paper extends our earlier work [31], [32] in several significant ways. Whereas our earlier approach required a computationally intensive batch processing of image frames, the new algorithm described in this paper incrementally processes images in real time, thus making it applicable to online processing of arbitrarily long video sequences. The incremental approach arises from a simplified grouping procedure and a novel technique for distinguishing stable features using a single image. As a result, the algorithm described in this paper exhibits improved accuracy and decreased sensitivity to parameters compared with the earlier version.

III. Algorithm Description

An overview of the system is shown in Fig. 2. 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 PLP, the 3-D 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 that are detected in different image frames to achieve long-term tracking, and classifying vehicles based on the number of

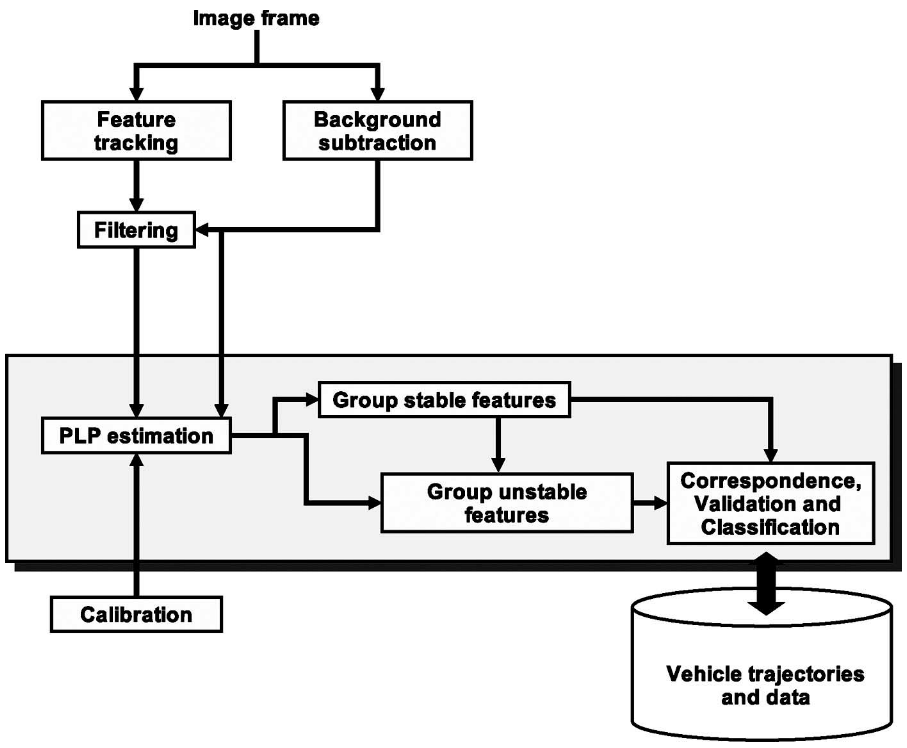


Fig. 2. System overview.

unstable features in the group. The details of these steps are described in the following subsections.

A. 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

$$\breve{\mathbf{u}} = C\breve{\mathbf{p}} \tag{1}$$

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 [33]. Since w is an arbitrary nonzero scale factor, C has 11 unique parameters. Thus, the correspondence of at least six points in a nondegenerate 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 road and one line perpendicular to the direction of travel, as shown in Fig. 3. The latter line is estimated by sequencing through the video and finding the intensity edge between the windshield and the hood of a light-colored vehicle. These three lines yield two vanishing points, from which the internal and external camera parameters are automatically computed using the mathematical formulation described by Schoepflin and Dailey [17]. The remaining six vertices of the cuboid defining the 3-D detection zone are then computed from the user-specified lane width, the number of lanes, and the desired length and height of the cuboid. For the world coordinate system, we adopt the convention that the 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.

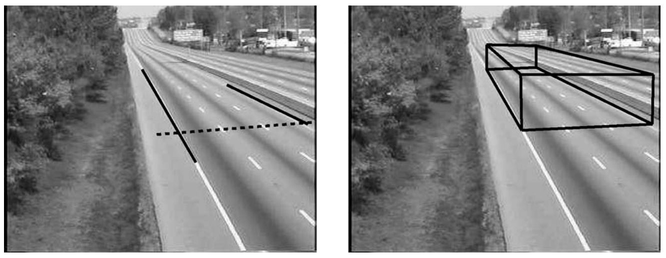


Fig. 3. Camera calibration. (Left) User draws three lines—(solid) two along the edges of the road and (dashed) one perpendicular to the direction of travel. The lines can be of arbitrary length. (Right) Three-dimensional detection zone is automatically computed.

Because the overall system is insensitive to small inaccuracies in the calibration (quantified in Section IV), this process is widely applicable to prerecorded sequences that are captured from unknown cameras. Note that the calibration procedure recovers a full 3-D to 2-D perspective mapping, which is necessary to handle the perspective effects encountered at low camera angles, unlike previous 2-D to 2-D calibration tools that recover only a planar mapping between the road surface and the image plane [3]. Also, note that perspective projection leads to more robust results than the multilayer homography used in [31] due to the reduced number of free parameters.

B. Background Subtraction

The background of the scene is learned by storing the average gray level of each pixel over a fixed period of time. With our sequences, we found 20 s of video to be sufficient for this task;

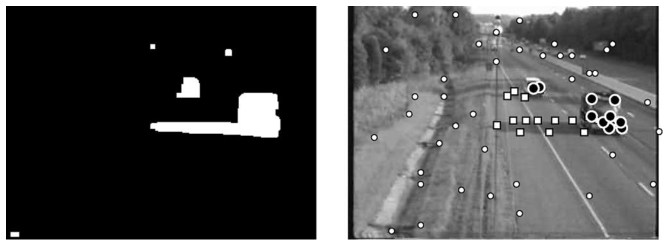


Fig. 4. (Left) Foreground mask resulting from background subtraction. (Right) Features being tracked in this frame of video, divided into three kinds: 1) those that lie on the background (shown as small dots), 2) those that lie within τ_s pixels of the background (shown as small squares), and 3) those 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.

however, 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 the road for which the traffic is moderately dense for some period of time.

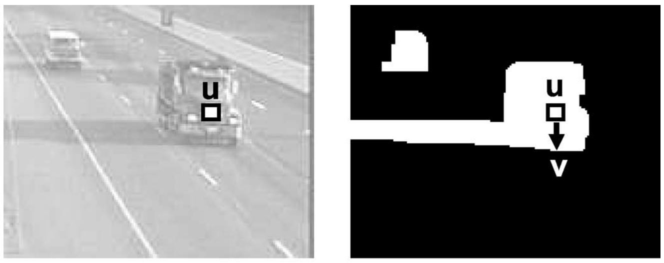
Once the background is learned offline, the technique of background subtraction, including morphological operations and thresholding, is applied to each image of the sequence to yield a binary foreground mask that indicates whether each pixel is foreground or background. To cope with lighting and environmental changes, the background is adaptively updated using this mask as the sequence is processed to preclude inadvertently adapting to foreground intensities [11]. 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 [34]–[41].

We use background subtraction to perform a simple filtering operation on the features, as shown in Fig. 4. 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, we also ignore any feature that lies within a small distance τ_s from a background pixel. (We set $\tau_s = 2$ pixels in all experiments.) This simple procedure removes many of the features that are due to shadow edges alone since the road surface tends to be fairly untextured, while removing only a small fraction of legitimate foreground features.

C. PLPs

Feature points are automatically selected and tracked using the Lucas–Kanade feature tracker [42]. We use the OpenCV implementation of the feature tracker, which uses the Sharr gradient operator [43]. A coarse-to-fine pyramidal strategy allows for large image motions, and features are automatically selected, tracked, and replaced.

Because of the dimension loss in projecting the 3-D world to a 2-D 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 computed. In this section, we present a method for exploiting this capability.



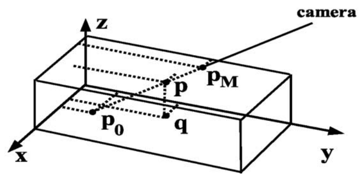


Fig. 5. Top (left) Image and (right) foreground mask F with a feature point $\mathbf{u}$ and its PLP $\mathbf{v} = \psi_F(\mathbf{u})$. (Bottom) Three-dimensional coordinates of the preimage $\mathbf{p} = \Phi(\mathbf{u})$ of the feature can be computed under the assumption that $\mathbf{q} = \Phi(\mathbf{v})$ directly lies 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.

Suppose we have a feature point $\mathbf{u}$ and a binary foreground mask F from background subtraction, as shown in Fig. 5. Projecting $\mathbf{u}$ downward in the image plane to the first encountered background pixel yields the point $\mathbf{v}$ that we call the 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 $\varphi_z : \mathbb{R}^2 \to \mathbb{R}^3$ be the mapping from a 2-D 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 (1) yield the following inhomogeneous equation:

$$\varphi_z(\mathbf{u}) = K^{-1}(\mathbf{u})\mathbf{t}_z(\mathbf{u}) \tag{2}$$

where

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

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

$\mathbf{u} = [u \quad v]^T$ is the projection of $\mathbf{p}$, and c_{ij} is the ijth element of C. (See [44] for the derivation.)

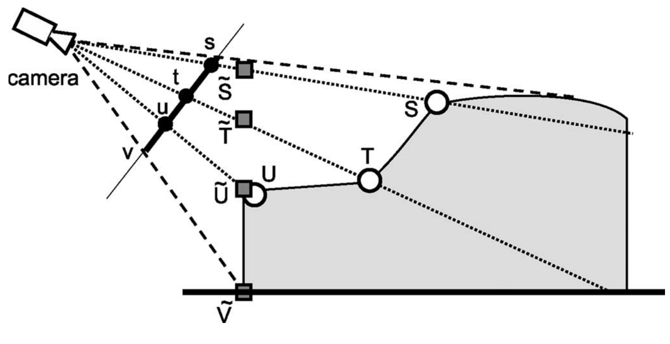


Fig. 6. Three points on the surface of a vehicle viewed by a camera with their estimated coordinates using PLP. The points lower to the ground yield less error.

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 3-D coordinates of $\mathbf{p}$, all we need is to compute its z coordinate, which is done by solving (1) in a least squares manner, i.e.,

$$\hat{z} = \frac{\mathbf{h}_p^T \mathbf{h}_c}{\mathbf{h}_p^T \mathbf{h}_p} \tag{3}$$

where

$$\mathbf{h}_p = \begin{bmatrix} u & c_{33} - c_{13} \\ v & c_{33} - c_{23} \end{bmatrix}$$
$$\mathbf{h}_c = \begin{bmatrix} c_{14} - u & c_{34} + (c_{11} - uc_{31}) & x + (c_{12} - uc_{32}) & y \\ c_{24} - v & c_{34} + (c_{21} - vc_{31}) & x + (c_{22} - vc_{32}) & y \end{bmatrix}$$

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

D. Identifying and Grouping Stable Features

The technique just presented for computing the 3-D coordinates of the preimage of a feature point $\mathbf{u}$ from its PLP relies on 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.
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.

Fig. 6 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 that the third 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

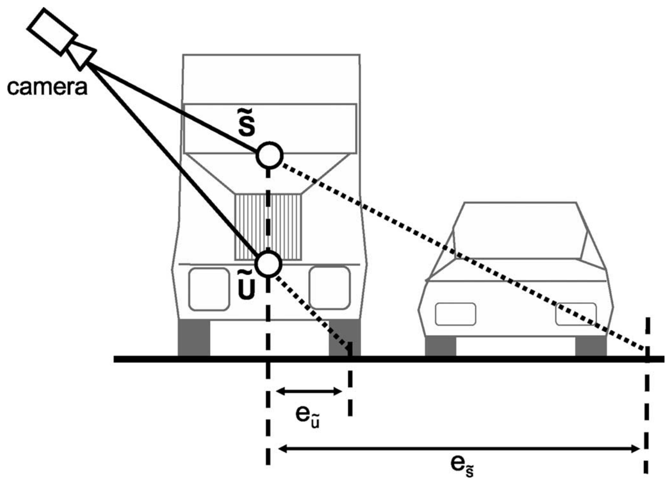


Fig. 7. Estimated coordinates of two points using PLP. Because the estimated height is nearly always greater than the true height, the higher feature is more likely to be assigned to the wrong vehicle.

contour at any point never exceeds the bound $\mu_{\max}(x, z)$ [44]. Then, we have the following observation.

Observation 1: For any two points $\mathbf{S} = (x_S, y_S, z_S)$ and $\mathbf{U} = (x_U, y_U, z_U)$ on the surface of a vehicle such that $z_S > z_U$, the Euclidean error in the estimate $\tilde{\mathbf{S}}$ will not be less than that of $\tilde{\mathbf{U}}$, i.e., $\|\tilde{\mathbf{S}} - \mathbf{S}\| \geq \|\tilde{\mathbf{U}} - \mathbf{U}\|$, 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 nondecreasing 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 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 $\tilde{\mathbf{S}} = (\tilde{x}_S, \tilde{y}_S, \tilde{z}_S)$ and $\tilde{\mathbf{U}} = (\tilde{x}_U, \tilde{y}_U, \tilde{z}_U)$ such that $z_S > z_U$, the maximum possible Euclidean error in the estimate $\tilde{\mathbf{S}}$ is greater than that of $\tilde{\mathbf{U}}$, i.e., $\max \|\tilde{\mathbf{S}} - \mathbf{S}\| > \max \|\tilde{\mathbf{U}} - \mathbf{U}\|$.

To see the validity of this observation, notice from Fig. 6 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 Fig. 7, where the camera in 3-D is aimed toward the front of the vehicles at an oblique angle. Let $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{U}}$ be the 3-D estimates of two preimages using the PLP procedure, with $\tilde{\mathbf{S}}$ higher above the road than $\tilde{\mathbf{U}}$. Using the upper bound $z_{\text{true}} \leq \tilde{z}$, 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. We see that both observations lead to the conclusion that points that are 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 $\mathbf{u} = [u \quad v]^T$, we compute the PLP of the

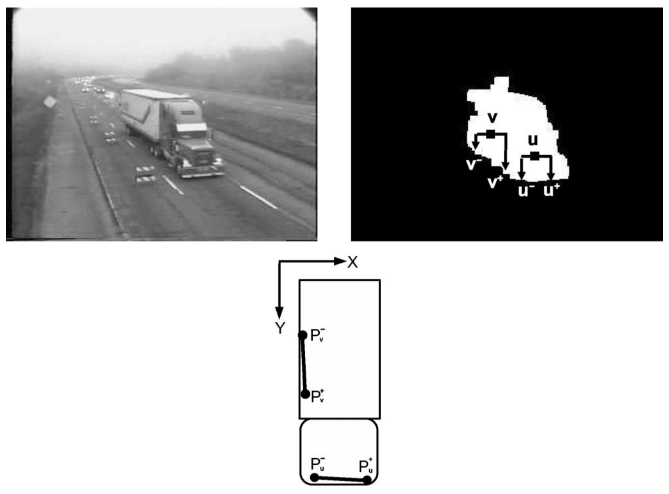


Fig. 8. Top (left) Image and (right) foreground mask, with two unrelated feature points ($\mathbf{u}$ and $\mathbf{v}$) and the PLPs ($\mathbf{u}^+$, $\mathbf{u}^-$, $\mathbf{v}^+$, and $\mathbf{v}^-$) of their perturbations. (Bottom) Points on the front of the vehicle yield a smaller slope in the road plane than points on the side of the vehicle.

two points obtained by horizontally perturbing the feature in the image plane (see Fig. 8): $\mathbf{u}^+ = \psi_F([\,u+\delta \quad v\,]^T)$ and $\mathbf{u}^- = \psi_F([\,u-\delta \quad v\,]^T)$. The 3-D 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 on the side. Since the shadows on the side tend to be more severe than those on the front, the points on the front are less likely to violate the third assumption and, hence, are more reliable.

Putting this analysis together, we distinguish between two kinds of features, namely, *stable* and *unstable*. We classify a feature point $\mathbf{u}$ as stable if it satisfies the following two conditions:

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

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 [31].

Once the stable features have been identified, they are grouped in the road plane (xy-plane), as shown in Fig. 9. Because of the criteria used in selecting stable features, points belonging to the same vehicle generally have a small deviation in their world coordinates 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

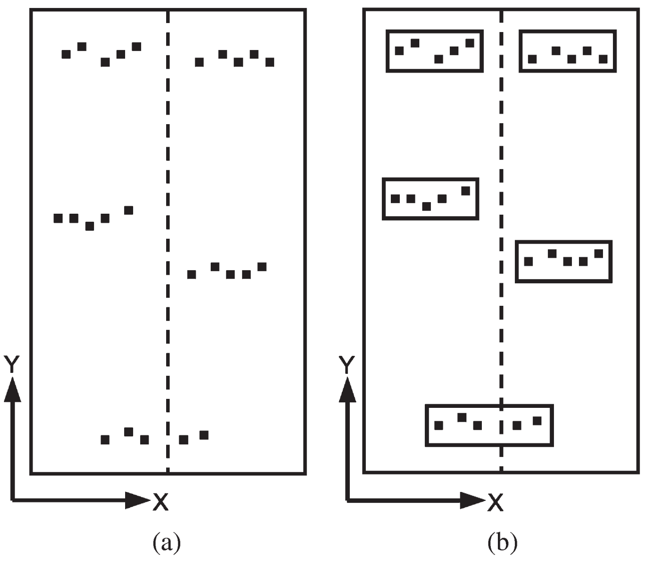


Fig. 9. Stable features are grouped in the road plane using a region-growing algorithm that compares their y coordinates.

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 [31], 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 for the vehicle to be correctly detected, although, in practice, we discard groups with fewer than three features to reduce the number of spurious false detections. We set $\epsilon_y = \epsilon_z = 0.4 w_{\text{lane}}$, $\epsilon_{\text{slope}} = 1.5$, and $\delta = 3$ pixels, where w_{lane} is the width of a lane that is computed during the calibration step.

E. 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 that 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 Fig. 10, where $z = 0$ is the road plane, and $z = M$ is 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 parametrically represented as follows:

$$\begin{aligned}\mathbf{p} &= \mathbf{p}_0 + \alpha(\mathbf{p}_M - \mathbf{p}_0)\\ \mathbf{p}' &= \mathbf{p}'_0 + \alpha'\,(\mathbf{p}'_M - \mathbf{p}'_0)\end{aligned} \tag{4}$$

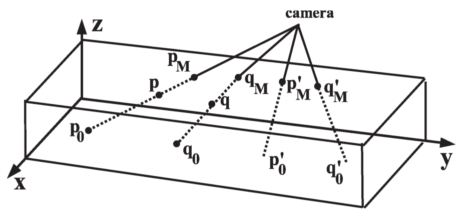


Fig. 10. Features $\mathbf{p}$ and $\mathbf{q}$ on a vehicle travel to $\mathbf{p}'$ and $\mathbf{q}'$ at a different time. If the vehicle travels parallel to the road plane, then the coordinates of the unstable feature $\mathbf{p}$ can be computed from the coordinates of the stable feature $\mathbf{q}$.

where α, $\alpha' \in \mathbb{R}$ are the fractional distances along the ray. If we further assume that the road is horizontally flat, then the z components of $\mathbf{p}$ and $\mathbf{p}'$ are equal, from which it can be easily 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

$$\alpha = \frac{(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)^T(\Delta\mathbf{q} - \Delta\mathbf{p}_0)}{(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)^T(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)} \tag{5}$$

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 is given by

$$\hat{\mathbf{p}} = \mathbf{p}_0 + \frac{(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)^T(\Delta\mathbf{q} - \Delta\mathbf{p}_0)}{(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)^T(\Delta\mathbf{p}_M - \Delta\mathbf{p}_0)}(\mathbf{p}_M - \mathbf{p}_0) \tag{6}$$

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 $\mathbf{p}$, we use the above procedure 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, we estimate the world coordinates using the PLP procedure described in Section III-D 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, we then compute a score indicating whether $\mathbf{p}$ belongs to group i, i.e.,

$$\mathcal{L}_{\mathbf{p}}^i = \mathcal{L}_x^i \mathcal{L}_y^i \mathcal{L}_z^i \mathcal{L}_\ell^i \mathcal{L}_h^i \tag{7}$$

where

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

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

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

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

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

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 that 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 that are 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 on the upper bound of z_{true} mentioned in Section III-D.

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 that are 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, respectively, 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 $\mathbf{u}_\ell^i$ that are foreground pixels (in the foreground mask) be ℓ^i, and let the same fraction along the line between $\mathbf{u}^i$ and $\mathbf{u}_h^i$ be h^i so that $0 \le \ell^i, h^i \le 1$. 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 $\mathcal{L}_\ell^i$ and $\mathcal{L}_h^i$ 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 $\mathcal{L}^a/\mathcal{L}^b > \mathcal{L}_{\text{ratio}}$. 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 $\mathcal{L}_{\text{ratio}}$). We set $\sigma_x = \sigma_y = \sigma_z = 5$ ft, $\sigma_\ell = \sigma_h = 0.1$ pixels, $\lambda_\ell = 1.2w_{\text{lane}}$, $\lambda_h = 0.8w_{\text{lane}}$, $\mathcal{L}_{\min} = 0.8$, and $\mathcal{L}_{\text{ratio}} = 2$.

F. 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 that are 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, we keep count of the total number of frames in which it was successfully tracked (η_t) and the number of recent consecutive frames where it has been missing (η_m).

After finding a match for all nonmissing vehicles, the remaining unassociated feature 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,

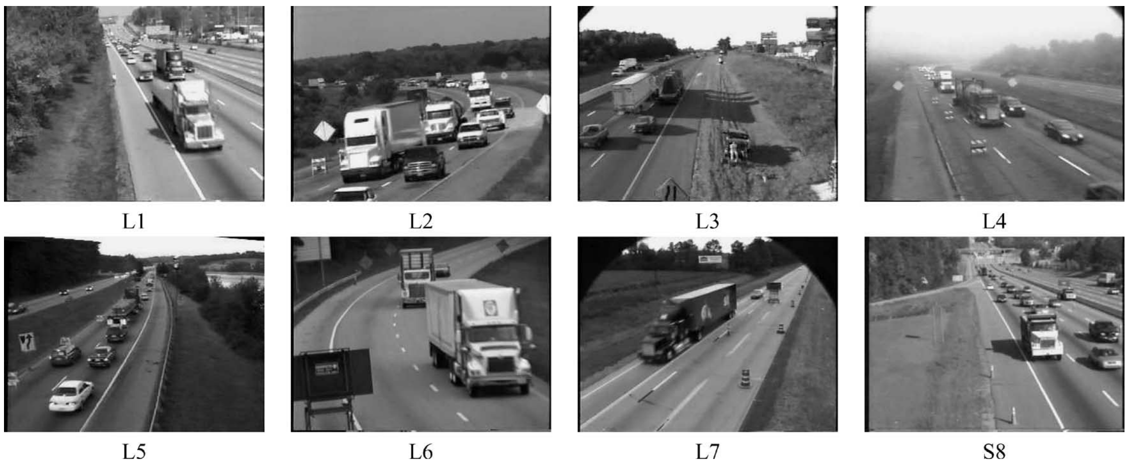


Fig. 11. Sample image frames from the 11 sequences used in evaluating the algorithm, showing the variety of scenarios considered. S1 and S4 exhibit the same conditions as L1 and L4, respectively. S9, which is omitted due to lack of space, closely resembles S8.

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 detection 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 $\kappa \geq 0$, at any time, which is important to prevent momentary false detections from being retained.

To classify a vehicle as a car or truck,[1] we simply sum the number of unstable features that are associated with that vehicle over all the frames that the vehicle is tracked. Vehicles with more than n_{truck} unstable features are classified as trucks, whereas the rest are considered cars. We only use unstable features 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 that are associated with trucks is usually much greater than that of cars (typically five to ten times higher). We set $\tau_x = 0.3w_{\text{lane}}$, $\tau_y = 0.5w_{\text{lane}}$, $\tau_\eta = 4$, $\kappa = 2$, and $n_{\text{truck}} = 20$.

IV. Experimental Results

The algorithm was tested on 11 grayscale video sequences captured by a 30-Hz camera placed on an approximately 9-m pole on the side of the road and digitized at 320 $\times$ 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 2-D image coordinates and 3-D world coordinates, as described in Section III-A. After the calibration, the system is 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 Fig. 11.

[1]We define a car as a vehicle with two axles and a truck as a vehicle with more than two axles.

TABLE I
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

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)

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 min each (18 000 image frames), whereas the "short" sequences S8 and S9 are approximately 30 s 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 I. 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

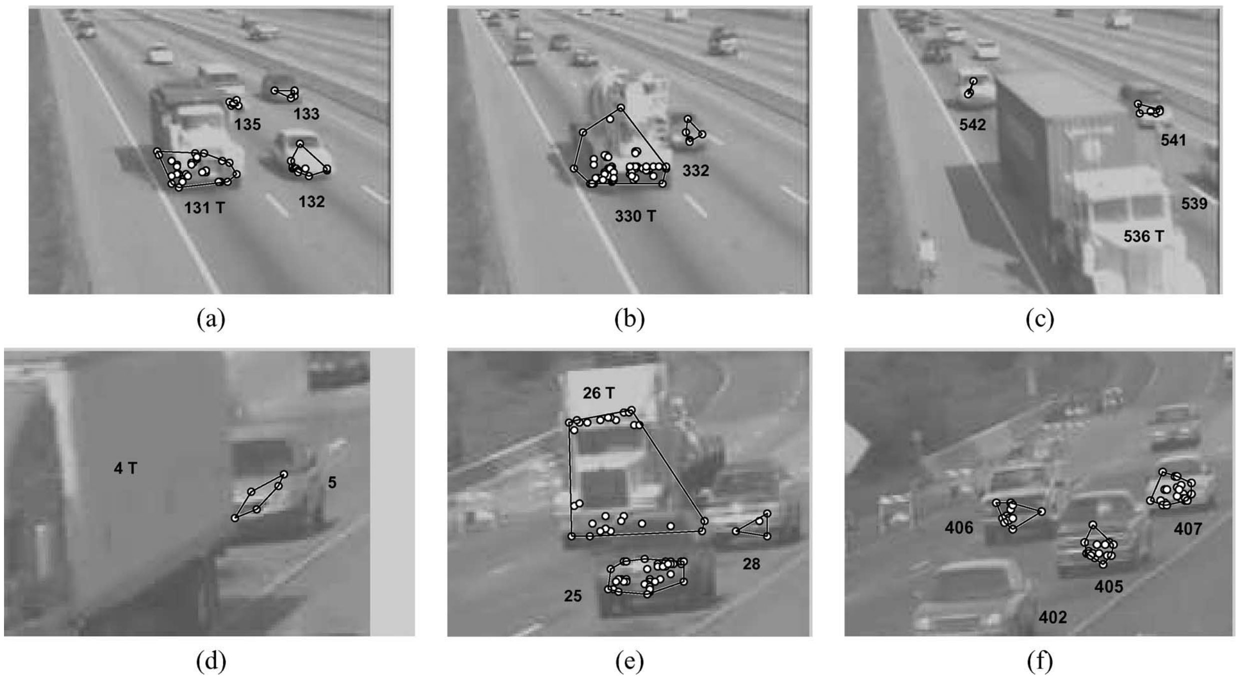


Fig. 12. Results of the algorithm on some image frames showing the ability of the algorithm to handle severe occlusions. The sequence name and the frame number for each image is given as follows: (a) L1: 03 516. (b) L1: 10 302. (c) L1: 15 440. (d) L2: 00 134. (e) L2: 00 913. (f) L2: 11 960.

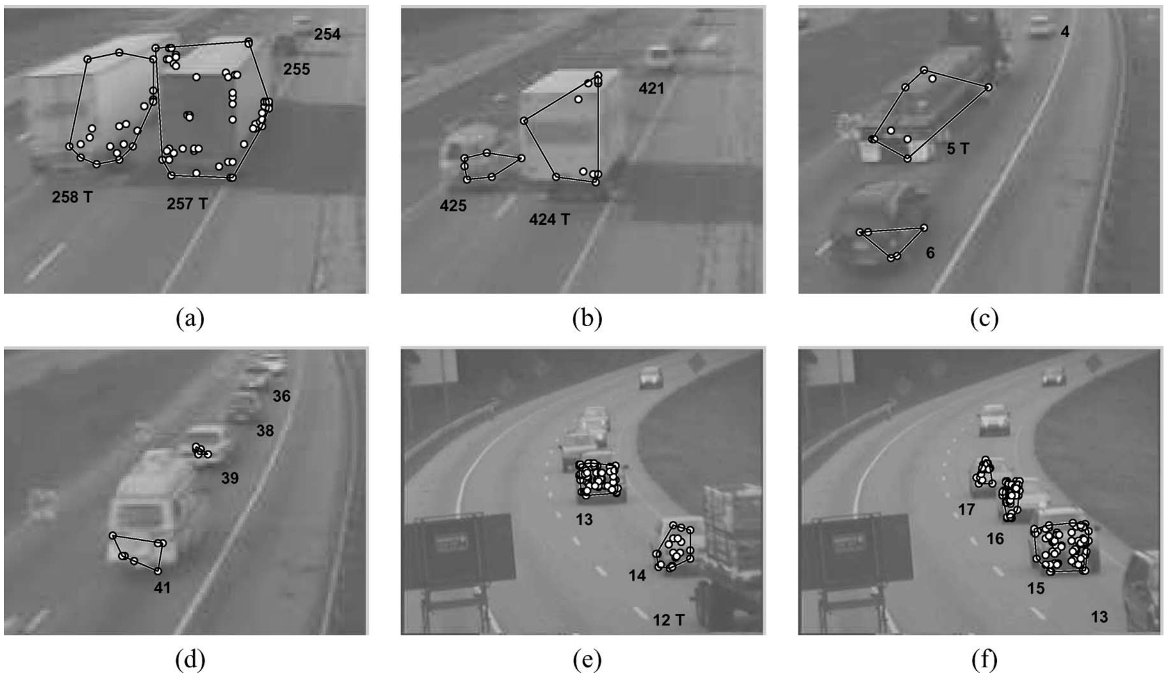


Fig. 13. Additional experimental results on sequences in which the vehicles are moving away from the camera or the road is curved. (a) L3: 08 860. (b) L3: 17 522. (c) L5: 00 439. (d) L5: 02 618. (e) L6: 01 098. (f) L6: 01 190.

detected and tracked because of the relatively low texture on the rear of the vehicles.

Figs. 12–14 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 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 that is 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 detection zone, although they have not yet left the image.

Fig. 12 demonstrates the ability of the system to segment vehicles which are severely occluded, often by larger vehicles traveling in adjacent lanes. In Fig. 12(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

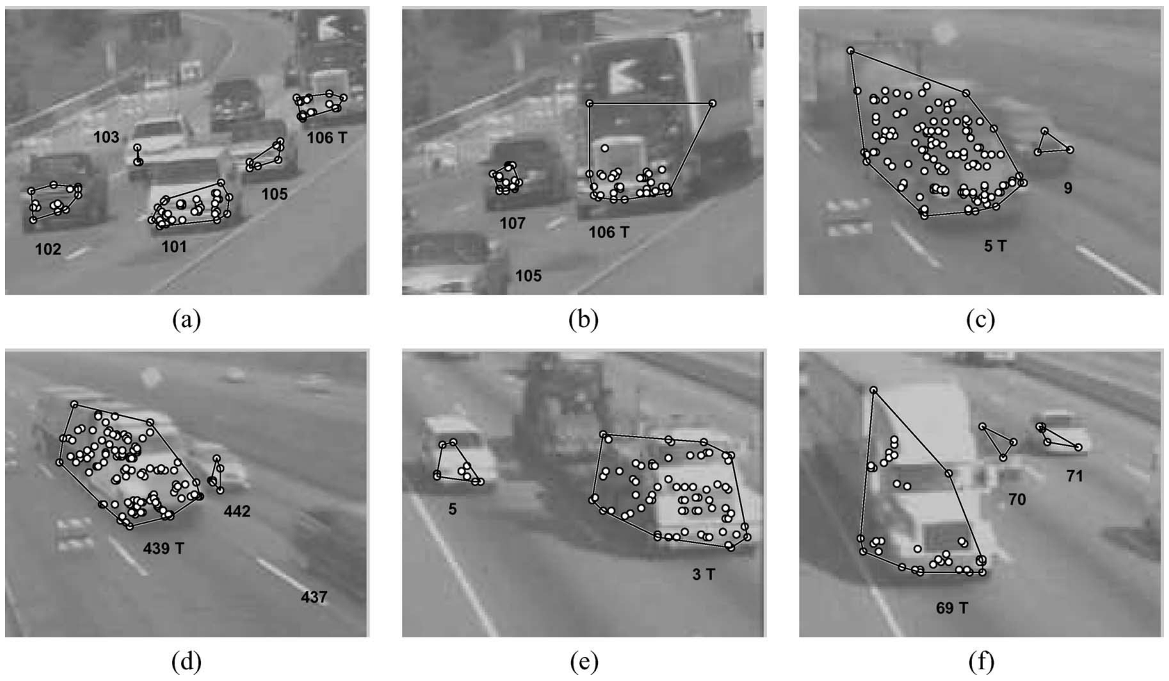


Fig. 14. More experimental results demonstrating the performance of the algorithm when large tractor trailers occlude other vehicles. (a) L2: 03 204. (b) L2: 03 260. (c) L4: 00 566. (d) L4: 14 654. (e) S8: 00 078. (f) S8: 00 506.

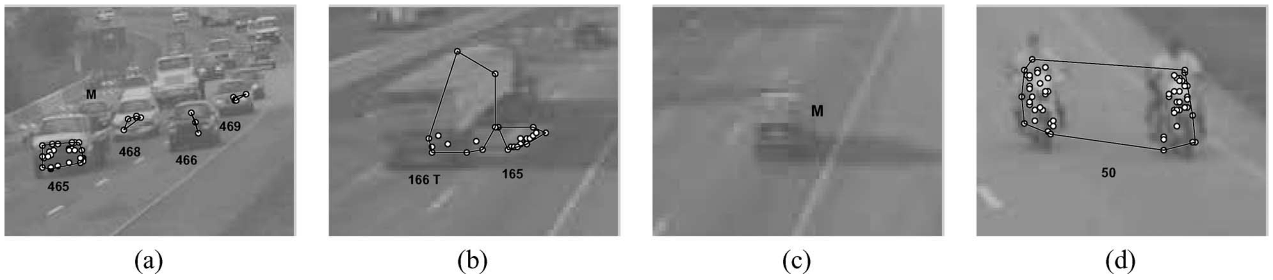


Fig. 15. Some instances in which the algorithm makes a mistake. (a) L2: 13 854. (b) L3: 06 216. (c) L3: 10 940. (d) L6: 07 240.

(#131) throughout the detection zone. In Fig. 12(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 Fig. 12(d), the vehicle (#5) is detected as it is being disoccluded by the truck in front. In Fig. 12(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.

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

Some examples involving large tractor trailers are shown in Fig. 14. In Fig. 14(a), both the vehicles (#103 and #105) that are occluded by the white van (#101) are correctly detected and tracked. Similarly, the dark-colored sport-utility vehicle (#107) traveling adjacent to the truck (#106) in Fig. 14(b) is detected after a few frames once a sufficient number of stable features is found. In Fig. 14(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 detection zone is again demonstrated. In Fig. 14(e), the features of a large tractor trailer are all correctly grouped into one vehicle, despite the large extent to which 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 2-D algorithms.

To convey a sense of the limitations of the algorithm, some mistakes are shown in Fig. 15. In Fig. 15(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 remains partially occluded by the vehicle in front (#465) throughout the detection zone so that none of the features on the vehicle qualify as stable features. In Fig. 15(b), the shadow of the tractor trailer is mistakenly detected as a car (#165), thus yielding a false positive. In Fig. 15(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 Fig. 15(d), the algorithm misinterprets two motorcycles traveling side by side as a single car, which is an error that could be avoided by including a model for motorcycles and measuring the foreground evidence to validate each vehicle.

In Fig. 16, the number of vehicles detected by the algorithm is compared with the ground truth manually obtained for the S2 sequence. Note that the 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 1-min intervals) is plotted versus time, which corresponds with the general trend evidence in the video sequence.

We have found the detection accuracy to be fairly insensitive to the calibration parameters. To quantify this conclusion, each of the endpoints 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 endpoints 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 four 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[2] and the OpenCV Lucas–Kanade tracker.[3] 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 the frame rate. To achieve this speed, the background was updated every 60 frames (2 s), 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).

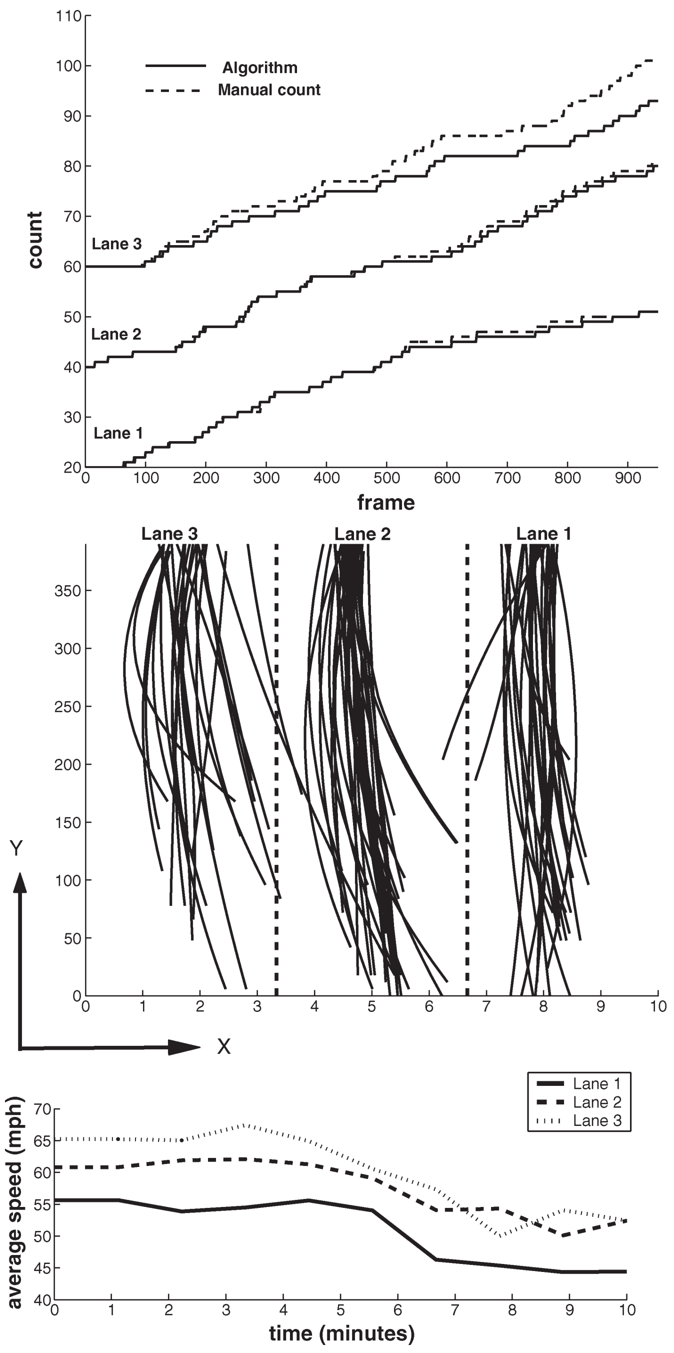


Fig. 16. Plots displaying the results of the algorithm. (Top) Total vehicles detected in each lane versus time in the S2 sequence, with lanes 2 and 3 offset by 40 and 60 for viewing clarity. (Middle) Some of the vehicle trajectories for L1 as seen in a top-down view, with vehicles that are changing lanes clearly visible. (Bottom) Mean speed (in miles per hour) for the vehicles in L1 computed over 1-min intervals.

V. Conclusion

Previous approaches to segmenting and tracking vehicles using video generally require the camera to be placed high above the ground to minimize the effects of occlusion and spillover. In this paper, we have presented a technique that overcomes this limitation, working when the camera is relatively low to the ground and beside the road. Our approach is based on identifying and grouping feature points in each image frame whose 3-D coordinates can be computed in a manner that is relatively immune to the effects of perspective projection. The novelty of this paper includes an incremental online real-time algorithm to estimate the heights of features using a combination of background subtraction, perturbed PLPs, 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

[2]http://www.ces.clemson.edu/~stb/blepo.

[3]http://www.intel.com/research/mrl/research/opencv.

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 that are unable to afford the infrastructure that is necessary for mounting cameras high above the ground. In addition, by addressing the important problem of occlusion, many of the concepts contained in this paper are directly applicable to existing high-angle scenarios with a large number of traffic lanes in which large trucks often occlude neighboring vehicles. To further improve this paper and enhance its applicability, future work should be aimed at reducing the effects of shadows, incorporating appearance models of vehicles to improve robustness, and supporting continued operation in the presence of changing weather and environmental conditions.

References

[1] L. A. Klein, *Sensor Technologies and Data Requirements for ITS*. Boston, MA: Artech House, 2001.

[2] Z. W. Kim and J. Malik, "Fast vehicle detection with probabilistic feature grouping and its application to vehicle tracking," in *Proc. Int. Conf. Comput. Vis.*, 2003, vol. 1, pp. 521–528.

[3] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, "A real time computer vision system for measuring traffic parameters," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 1997, pp. 495–501.

[4] R. Ervin, C. MacAdam, A. Vayda, and A. Anderson, "Applying the SAVME database of inter-vehicle kinematics to explore the natural driving environment," in *Proc. TRB Annu. Meeting Compendium Papers, Transp. Res. Board Annu. Meeting*, 2001. Paper 01-0496. [CD-ROM].

[5] S. Atev, H. Arumugam, O. Masoud, R. Janardan, and N. P. Papanikolopoulos, "A vision-based approach to collision prediction at traffic intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 4, pp. 416–423, Dec. 2005.

[6] D. R. Magee, "Tracking multiple vehicles using foreground, background and motion models," *Image Vis. Comput.*, vol. 22, no. 2, pp. 143–155, Feb. 2004.

[7] S. C. Cheung and C. Kamath, "Robust techniques for background subtraction in urban traffic video," in *Proc. Electron. Imaging: Visual Commun. Image Process.*, 2004, pp. 881–892.

[8] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Statistic and knowledge-based moving object detection in traffic scenes," in *Proc. IEEE Conf. Intell. Transp. Syst.*, 2000, pp. 27–32.

[9] P. Kumar, S. Ranganath, H. Weimin, and K. Sengupta, "Framework for real-time behavior interpretation from traffic video," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 1, pp. 43–53, Mar. 2005.

[10] 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 Trans. Intell. Transp. Syst.*, vol. 4, no. 3, pp. 154–167, Sep. 2003.

[11] S. Gupte, O. Masoud, R. F. K. Martin, and N. P. Papanikolopoulos, "Detection and classification of vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 1, pp. 37–47, Mar. 2002.

[12] Y.-K. Jung, K.-W. Lee, and Y.-S. Ho, "Content-based event retrieval using semantic scene interpretation for automated traffic surveillance," *IEEE Trans. Intell. Transp. Syst.*, vol. 2, no. 3, pp. 151–163, Sep. 2001.

[13] H. Veeraraghavan, O. Masoud, and N. P. Papanikolopoulos, "Computer vision algorithm for intersection monitoring," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 2, pp. 78–89, Jun. 2003.

[14] J.-W. Hsieh, S.-H. Yu, Y.-S. Chen, and W.-F. Hu, "Automatic traffic surveillance system for vehicle tracking and classification," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 2, pp. 175–187, Jun. 2006.

[15] J. Kato, T. Watanabe, S. Joga, Y. Lui, and H. Hase, "An HMM/MRF-based stochastic framework for robust vehicle tracking," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 3, pp. 142–154, Sep. 2004.

[16] D. Dailey, F. W. Cathy, and S. Pumrin, "An algorithm to estimate mean traffic speed using uncalibrated cameras," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 2, pp. 98–107, Jun. 2000.

[17] T. N. Schoepflin and D. J. Dailey, "Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 2, pp. 90–98, Jun. 2003.

[18] Y.-K. Ki and D.-Y. Lee, "A traffic accident recording and reporting model at intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 188–194, Jun. 2007.

[19] O. Masoud, N. P. Papanikolopoulos, and E. Kwon, "The use of computer vision in monitoring weaving sections," *IEEE Trans. Intell. Transp. Syst.*, vol. 2, no. 1, pp. 18–25, Mar. 2001.

[20] Y. Cho and J. Rice, "Estimating velocity fields on a freeway from low-resolution videos," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp. 463–469, Dec. 2006.

[21] D. Koller, J. Weber, and J. Malik, "Robust multiple car tracking with occlusion reasoning," in *Proc. Eur. Conf. Comput. Vis.*, 1994, pp. 189–196.

[22] J. Melo, A. Naftel, A. Bernardino, and J. Santos-Victor, "Detection and classification of highway lanes using vehicle motion trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 2, pp. 188–200, Jun. 2006.

[23] T. Zhao and R. Nevatia, "Car detection in low resolution aerial images," in *Proc. Int. Conf. Comput. Vis.*, 2001, vol. 1, pp. 710–717.

[24] D. Koller, K. Dandilis, and H. H. Nagel, "Model based object tracking in monocular image sequences of road traffic scenes," *Int. J. Comput. Vis.*, vol. 10, no. 3, pp. 257–281, 1993.

[25] T. N. Tan and K. D. Baker, "Efficient image gradient based vehicle localization," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1343–1356, Aug. 2000.

[26] M. Haag and H. Nagel, "Combination of edge element and optical flow estimate for 3-D-model-based vehicle tracking in traffic image sequences," *Int. J. Comput. Vis.*, vol. 35, no. 3, pp. 295–319, Dec. 1999.

[27] J. M. Ferryman, A. D. Worrall, and S. J. Maybank, "Learning enhanced 3D models for vehicle tracking," in *Proc. British Mach. Vis. Conf.*, 1998, pp. 873–882.

[28] G. Alessandretti, A. Broggi, and P. Cerri, "Vehicle and guard rail detection using radar and vision data fusion," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 1, pp. 95–105, Mar. 2007.

[29] N. Saunier and T. Sayed, "A feature-based tracking algorithm for vehicles in intersections," in *Proc. 3rd Can. Conf. Comput. Robot Vis.*, 2006, p. 59.

[30] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, "Traffic monitoring and accident detection at intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 2, pp. 108–118, Jun. 2000.

[31] N. K. Kanhere, S. J. Pundlik, and S. T. Birchfield, "Vehicle segmentation and tracking from a low-angle off-axis camera," in *Proc. IEEE CVPR*, Jun. 2005, pp. 1152–1157.

[32] N. K. Kanhere, S. T. Birchfield, and W. A. Sarasua, "Vehicle segmentation and tracking in the presence of occlusions," *Transp. Res. Rec.,* no. 1944, pp. 89–97, 2006.

[33] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[34] A. Prati, I. Mikic, M. M. Tridevi, and R. Cucchiara, "Detecting moving shadows: Algorithms and evaluation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 7, pp. 918–923, Jul. 2003.

[35] T. Horprasert, D. Harwood, and L. Davis, "A statistical approach for real-time robust background subtraction and shadow detection," in *Proc. IEEE Frame Rate Workshop*, Kerkyra, Greece, 1999, pp. 1–19.

[36] P. Rosin and T. Ellis, "Image difference threshold strategies and shadow detection," in *Proc. 6th BMVC*, 1995, pp. 347–356.

[37] R. Cucchiara, C. Grana, M. Piccardi, A. Prati, and S. Sirotti, "Improving shadow suppression in moving object detection with HSV color information," in *Proc. IEEE Conf. Intell. Transp. Syst.*, Aug. 2001, pp. 334–339.

[38] Y. Ivanov, A. Bobick, and J. Liu, "Fast lighting independent background subtraction," *Int. J. Comput. Vis.*, vol. 37, no. 2, pp. 199–207, Jun. 2000.

[39] K. Onoguchi, "Shadow elimination method for moving object detection," in *Proc. IAPR Int. Conf. Pattern Recog.*, 1998, pp. 583–587.

[40] J. Stauder, R. Mech, and J. Ostermann, "Detection of moving cast shadows for object segmentation," *IEEE Trans. Multimedia*, vol. 1, no. 1, pp. 65–76, Mar. 1999.

[41] X. Tao, M. Guo, and B. Zhang, "A neural network approach to elimination of road shadow for outdoor mobile robot," in *Proc. IEEE Int. Conf. Intell. Process. Syst.*, 1997, pp. 1302–1306.

[42] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 1994, pp. 593–600.

[43] J.-Y. Bouguet, "Pyramidal implementation of the Lucas Kanade feature tracker," in OpenCV documentation. Santa Clara, CA: Intel Corp., Microprocessor Res. Labs, 1999.

[44] N. K. Kanhere and S. T. Birchfield, "Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features," Clemson Univ., Clemson, SC, Tech. Rep. CU-CVL-0701, Jul. 2007.

Neeraj K. Kanhere received the B.E. degree in electronics engineering from the University of Mumbai, Mumbai, India, in 2002 and the M.S. degree in electrical engineering in 2005 from Clemson University, Clemson, SC, where he is currently working toward the Ph.D. degree in electrical engineering.

For his research on vision-based traffic surveillance, he received the Dwight David Eisenhower Transportation Fellowship and the Clemson University Outstanding Performance Fellowship. He is the author or the coauthor of more than ten conference and journal papers on computer vision, image processing, and pattern recognition applied to Intelligent Transportation Systems.

Stanley T. Birchfield (S'91–M'99–SM'06) received the B.S. degree in electrical engineering from Clemson University, Clemson, SC, in 1993 and the M.S. and Ph.D. degrees from Stanford University, Stanford, CA, in 1996 and 1999, respectively.

While at Stanford University, his research was supported by a National Science Foundation Graduate Research Fellowship, and he was part of the winning team of the American Association for Artificial Intelligence Mobile Robotics Competition of 1994. From 1999 to 2003, he was a Research Engineer with Quindi Corporation: a startup company in Palo Alto, CA. Since 2003, he has been an Assistant Professor with the Department of Electrical and Computer Engineering, Clemson University. His research interests include visual correspondence, tracking, and segmentation, particularly applied to real-time systems.