Word count – 6030 + 1 Table + 4 Figures **TRB Paper Number: 06-2943**

Vehicle Segmentation and Tracking in the Presence of Occlusions

Neeraj K. Kanhere
Department of Electrical and Computer Engineering
207-A Riggs Hall
Clemson University, Clemson, SC 29634
Phone: (864) 650-4844, FAX: (864) 656-5910
E-mail: nkanher@clemson.edu

Stanley T. Birchfield
Department of Electrical and Computer Engineering
207-A Riggs Hall
Clemson University, Clemson, SC 29634
Phone: (864) 656-5912, FAX: (864) 656-5910
E-mail: stb@clemson.edu

Wayne A. Sarasua
Department of Civil Engineering
312 Lowry Hall, Box 340911
Clemson University, Clemson, SC 29634
Phone: (864) 656-3318, FAX: (864) 656-2670
E-mail: sarasua@clemson.edu

Nov 14, 2005

ABSTRACT

A novel method is presented for automatically visually monitoring a highway when the camera is relatively low to the ground and on the side of the road. In such a case, occlusion and the perspective effects due to the heights of the vehicles cannot be ignored. Using a single camera, the system automatically detects and tracks feature points throughout the image sequence, estimates the 3D world coordinates of the points on the vehicles, and groups those points together in order to segment and track the individual vehicles. Experimental results on different highways demonstrate the ability of the system to segment and track vehicles even in the presence of severe occlusion and significant perspective changes. By handling perspective effects, the approach overcomes a limitation of commercially available machine vision-based traffic monitoring systems that are used in many intelligent transportation systems (ITS) applications. The researchers are targeting this system as a step toward a next generation ITS sensor for automated traffic analysis.

INTRODUCTION

Traffic counts, speed and vehicle classification are fundamental data for a variety of transportation projects ranging from transportation planning to modern intelligent transportation systems (ITS). Most ITS applications are designed using readily available technology (e.g., sensors and communication), such as the inductive loop detector. Other sensing technologies include radar, infrared (IR), lasers, ultrasonic sensors and magnetometers. In (*1*) and (2), state of the art traffic sensing technologies are discussed along with the algorithms used to support traffic management functions.

Since the late 1980s, video imaging detection systems have been marketed in the U.S. and elsewhere. One of the most popular video based traffic counting systems is Autoscope, which is currently distributed by Econolite. Autoscope uses high-angle cameras to count traffic by detecting vehicles passing digital sensors. As a pattern passes over the digital detector, the change is recognized and a vehicle is counted. The length of time that this change takes place can be translated into speed estimates. Autoscope includes significant built-in heuristics to differentiate between shadows and vehicles in various weather conditions. The accuracy of Autoscope and other commercially based video detection systems is compromised if the cameras are mounted too low or have poor perspective views of traffic. If the camera is not mounted high enough, a vehicle's image will "spill over" onto neighboring lanes, resulting in double counting.

A more promising approach to traffic monitoring is to track vehicles over time throughout an image sequence. This approach yields the trajectories of the vehicles, which are necessary for applications such as traffic flow modeling and counting turn movements. In this paper we present an automatic technique for detecting and tracking vehicles at a low angle, even in the presence of severe occlusion, to obtain those trajectories. Although the problem of tracking is much more difficult at a low angle because of perspective effects, a solution to this problem would be greatly beneficial for data-collecting applications in which the permanent infrastructure necessary for high-angle cameras is not feasible.

Research in Vehicle Tracking

Over the years researchers in computer vision have proposed various solutions to the automated tracking problem. These approaches can be classified as follows:

Blob Tracking. In this approach, a background model is generated for the scene. For each input image frame, the absolute difference between the input image and the background image is processed to extract foreground blobs corresponding to the vehicles on the road. Variations of this approach have been proposed in (*3, 4, 5*). Gupte et al. (*3*) perform vehicle tracking at two levels: the region level and the vehicle level, and they formulate the association problem between regions in consecutive frames as the problem of finding a maximally weighted graph. These algorithms have difficulty handling shadows, occlusions, and large vehicles (e.g., trucks, and trailers), all of which cause multiple vehicles to appear as a single region.

Active Contour Tracking. A closely related approach to blob tracking is based on tracking active contours (also known as *snakes*) representing the boundary of an object. Vehicle tracking using active contour models has been reported by Koller et al. (*6*), in which the contour is initialized using a background difference image and tracked using intensity and motion boundaries. 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 with the objects. The intersection is excluded in the shape and motion estimation. As with the previous technique, results are shown on image sequences without shadows or severe occlusions, and the algorithm is limited to tracking cars.

3D-Model Based Tracking. Tracking vehicles using three-dimensional models has been studied by several research groups (*7, 8, 9, 10*). Some of these approaches assume an aerial view of the scene which virtually eliminates all occlusions (*10*) and match the three-dimensional wireframe models for different types of vehicles to edges detected in the image. In (*9*), a single vehicle is successfully tracked through a partial occlusion, but its applicability to congested traffic scenes has not been demonstrated.

Markov Random Field Tracking. An algorithm for segmenting and tracking vehicles in low-angle frontal sequences has been proposed by Kamijo et al. (*11*). In their work, the image is divided into pixel blocks, and a spatiotemporal Markov random field (ST-MRF) is used to update an object map using the current and previous image. One drawback of the algorithm is that it does not yield 3D information about vehicle trajectories in the world coordinate system. In addition, in order to achieve accurate results the images in the sequence are processed in reverse order to ensure that vehicles recede from the camera. The accuracy decreases by a factor of two when the sequence is not processed in reverse, thus making the algorithm unsuitable for on-line processing when time-critical results are required.

Feature Tracking. In this approach, instead of tracking a whole object, feature points on an object are tracked. The method is useful in situations of partial occlusions, where only a portion of an object is visible. The task of tracking multiple objects then becomes the task of grouping the tracked features based on one or more similarity criteria. Beymer et al.(*12,13*) have proposed a feature tracking based approach for traffic monitoring applications. In their approach, point features are tracked throughout the detection zone specified in the image. Feature points which are tracked successfully from the entry region to the exit region are considered in the process of grouping. Grouping is done by constructing a graph over time, with vertices representing sub-feature tracks and edges representing the grouping relationships between tracks. The algorithm

was implemented on multi-processor digital signal processing (DSP) board for real-time performance. Results were reported for day and night sequences with varying levels of traffic congestion.

Color and Pattern-Based Tracking. Chachich et al. (*14*) 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 (*15*). 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).

APPROACH

In comparison with the research just described, the novelty of the proposed approach lies in the estimation of the 3D coordinates of feature points in order to track vehicles at low angles, when a single homography is insufficient. This research contains several extensions over our previous work (*16*): a technique for associating tracks is proposed to facilitate long-term tracking, a method we call *incremental normalized cuts* is introduced to improve the quality of the segmentation, and a single perspective mapping is used instead of a multi-level homography, which increases robustness by taking advantage of the inherent constraints of the data.

The sequence is assumed to be taken from a single grayscale camera pointing at the road from the side. The task of segmenting and tracking vehicles in cluttered scenes is formulated as a feature tracking and grouping problem. Feature points are tracked in the image sequence, followed by estimation of the 3D world coordinates for those points, which are then grouped using a segmentation algorithm. The novelty of this work lies primarily in the technique for estimating the 3D coordinates from a single camera.

Offline Calibration

Calibration is required to estimate the 3D world coordinates for corresponding 2D points in the image. It should be emphasized that the calibration process described below is for a single camera and does not require knowledge about the camera specifications such as focal length or sensor dimensions, which makes it possible to process pre-recorded sequences captured from unknown cameras. The only information that is needed is six or more point correspondences.

We assume a pinhole camera model exhibiting perspective projection. 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 x 4 homogeneous transformation matrix (*17*). This matrix will be referred to as the camera calibration matrix **C**.

$$u = x\,c_{11} + y\,c_{12} + z\,c_{13} + c_{14} - u\,x\,c_{31} - u\,y\,c_{32} - u\,z\,c_{33} \quad [1]$$

$$v = x\,c_{21} + y\,c_{22} + z\,c_{23} + c_{24} - v\,x\,c_{31} - v\,y\,c_{32} - v\,z\,c_{33} \quad [2]$$

These equations define a mapping from the world coordinates *(x,y,z)* to the image coordinates *(u,v)* as described in (*18*).

Calibration Process

The camera calibration matrix **C** can be computed from the correspondence of 2D image points with the 3D coordinates of the associated world points. Each correspondence yields two equations of the form [1] and [2]. Six or more correspondences from a non-degenerate configuration lead to an over-determined system which can be solved using a standard least squares technique.

The offline calibration process depends upon the user-specified point correspondences for the calibration process. Although it would be ideal to have known markers placed at known locations in the scene, in practice this is often not feasible (e.g., on prerecorded data). In addition, as we will show, sufficient accuracy can be obtained simply by using standard specifications such as the width of a lane and the length of a truck, because the algorithm is not sensitive to slight errors in calibration.[1] We have developed a calibration application that is straightforward, simple to learn and use, and provides adequate results. Our calibration tool is similar to that developed by Gupte et al. (*3*), except that their system finds a planar mapping between the points on the road and the image points, whereas ours estimates a full perspective mapping, leading to 3D coordinates.

An example of the calibration process is shown in Figure 1. First, the user places a marker across the width of the road and perpendicular to the lane markings as shown in Figure 1 (a). With the marker position unchanged, the 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 rear 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 rear end of the truck. 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, lane width (e.g. 12 feet on an interstate), and number of lanes yields an approximate method for estimating the world coordinates of the control points. The calibration process is simple and takes only a couple of minutes to complete.

Backprojections

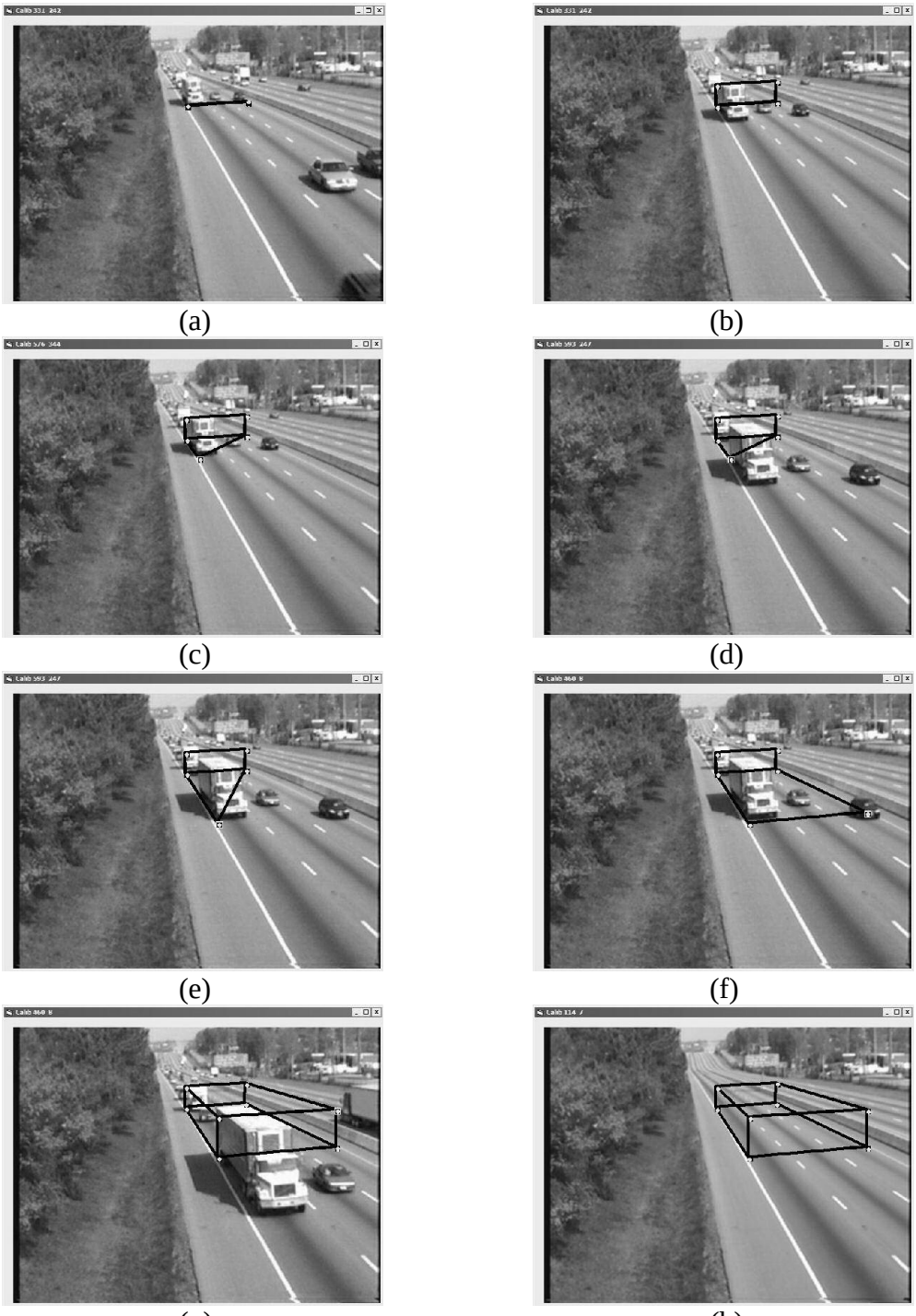
The imaging process maps a point in three dimensional space into a two dimensional image plane. The loss of dimension results in a non-invertible mapping. The calibration parameters for the camera and the image coordinates of a single point determine a ray in space passing through the optical center and the unknown point in the world. Rearranging equations [1] and [2] yields equations for two planes in 3D space.

$$(u\,c_{31} - c_{11})\,x + (u\,c_{32} - c_{12})\,y + (u\,c_{33} - c_{13})\,z + (u - c_{14}) = 0 \qquad [3]$$

$$(v\,c_{31} - c_{21})\,x + (v\,c_{32} - c_{22})\,y + (v\,c_{33} - c_{23})\,z + (v - c_{24}) = 0 \qquad [4]$$

1. Three untrained users were asked to perform the offline calibration as described. For the standard deviation (in random direction) of three pixels for each control point, the segmentation accuracy was observed to fall by 5 %.

FIGURE 1 Offline calibration process using our calibration application.



(a) – (h) show steps in offline calibration as described in the text.

The intersection of these two planes is the ray in 3D passing through the point in the world that is projected onto the image plane. Because there are two equations and three unknowns, the problem is under-constrained. If we know either $\boldsymbol{x}$, $\boldsymbol{y}$, or $\boldsymbol{z}$ we can solve for the other two world coordinates using the image coordinates and $\mathbf{C}$. In the sections to follow, we present a simple technique to overcome the under-constrained nature of the problem, enabling us to solve for all three world coordinates of a point.

Processing a Block of Frames

In the algorithm proposed by Beymer et al. (*12*) and Malik et al. (*13*), the point features tracked successfully from the entry region to the exit region are considered in the grouping step, which does not pose a problem when the camera is placed at a high vantage point looking down on the road. In our scenario, however, frequent occlusions and appearance changes (as vehicles approach the camera) result in the loss of a large number of features. As a result, the number of features that are tracked for the whole extent of the detection zone is not enough to achieve useful results. This problem is overcome by processing a block of consecutive image frames (typically twenty frames per block) in order to segment the vehicles in each block, and then to associate the segmented vehicles between the successive blocks. Features are tracked throughout a block of F image frames, overlapping with the previous block by N frames. The length of a block is determined by the average speed of the vehicles and the placement of the camera with respect to the road. If the number of frames in a block is too small, then although a large number of features will be tracked successfully throughout the frames in the block, the motion information will be insufficient for effective segmentation. On the other hand, using more frames in a frame-block will yield more reliable motion information at the expense of losing important features. The proposed algorithm relies on human judgment to balance between these tradeoffs. The steps described in the following sections are performed on the features tracked over a single block.

Tracking Features

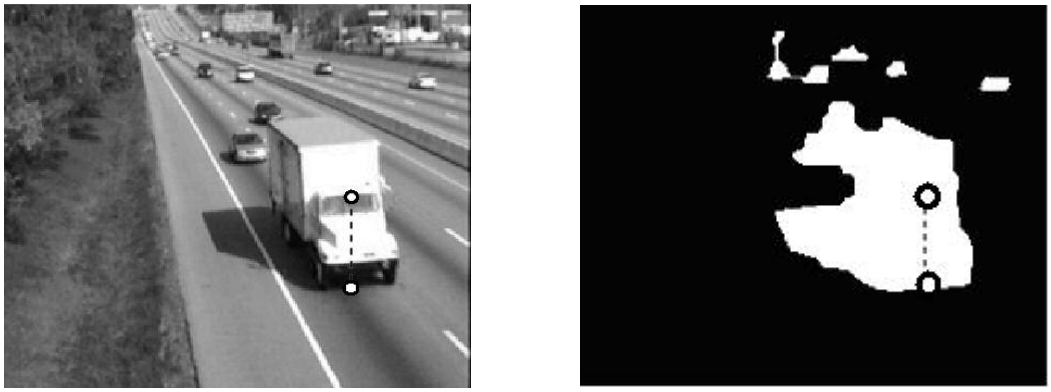
Feature points are automatically selected and tracked using the Kanade-Lucas-Tomasi (KLT) feature tracker (*19*) based on the algorithm proposed in (*20*).

Background Subtraction

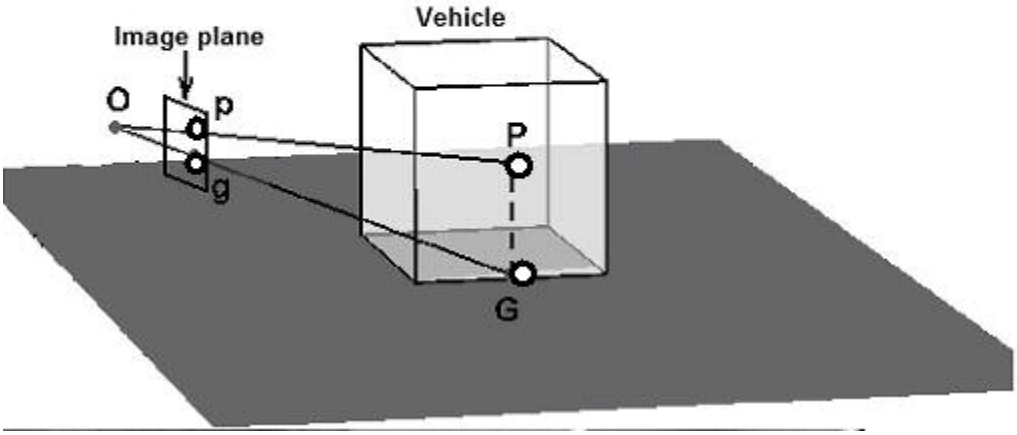
Background subtraction is a simple and effective technique for extracting foreground objects from a scene. The process of background subtraction involves initializing and maintaining a background model of the scene off-line. At run time, the estimated background image is subtracted from the image frame being processed, followed by thresholding the absolute value of the difference image, along with morphological processing to reduce the effects of noise, to yield foreground blobs. A review of several background modeling techniques is presented in (*21*).

For the scope of this research, the median filtering technique was chosen for its simplicity and effectiveness. The median filter belongs to a general class of *rank filters*. It is frequently used in image processing for removing noise in an image. For background modeling, we perform one dimensional median filtering in the time domain. For each pixel in the background image, the median value is selected from the set of values observed at the same pixel location in the previous $\boldsymbol{n}$ frames.

FIGURE 2 Road projection.



(a) **(b)**



(c)

Projecting a feature on the road surface in the image for estimating its height. (a) the first frame of a frame-block, with a feature point on a truck; (b) the foreground mask obtained by background subtraction, with the projection of the feature down to the bottom of the blob; (c) the corresponding 3D illustration, in which *p* and *g* are image points corresponding to *P* and *G* respectively, and *O* is the optical center of the camera.

Selecting Stable Features

It was shown earlier that the 3D coordinates of a world point can be estimated using its corresponding image coordinates, the calibration parameters, and at least one component of the world coordinates. A simple technique to achieve the same is presented here which involves finding the vertical projection of a point on the road surface in the image. The foreground mask generated in the previous step is used to find the projection as shown in Figure 2. ***P***=[*x y z*] is a vector of world coordinates corresponding to the point ***p***=[*u, v*] in the image. ***O*** is the optical center of the camera. ***G***=[*x y* z_g] is a 3 x 1 vector containing world coordinates of ground projection of ***P***. From equations [3] and [4] it follows that

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} c_{31}u - c_{11} & c_{32}u - c_{12} \\ c_{31}v - c_{21} & c_{32}v - c_{22} \end{bmatrix}^{-1} \begin{bmatrix} c_{14} - u + z(c_{13} - c_{33}u) \\ c_{24} - v + z(c_{23} - c_{33}v) \end{bmatrix} \quad [5]$$

Since ***G*** lies on the ground (or at least sufficiently close), we can compute its 3D coordinates by assuming $z_g = 0$ (corresponding to the road plane) in the above equation. ***P*** and ***G*** have the same ***(x, y)*** coordinates. Now, we know the image coordinates ***p*** of the world point ***P*** along with its ***(x, y)*** coordinates, and the camera calibration matrix ***C***. Substituting these values into equations [1] and [2], we solve for ***z***:

$$z = \frac{h_p^T h_c}{h_p^T h_p} \quad [6]$$

$$h_p = \begin{bmatrix} u c_{33} - c_{13} \\ v c_{33} - c_{23} \end{bmatrix}$$

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

For this technique to work, a simple box-model for the vehicles is assumed. A vehicle is modeled using five orthogonal, rectangular surfaces as shown in Figure 2 (c). Two such models have been used to represent cars and heavy vehicles. Dimensions of the corresponding models are computed using the calibration information (in proportion to the lane width). In addition, the technique assumes that the world point ***G*** is directly below ***P***. In practice these assumptions generally hold, because our threshold for the maximum height of a stable feature is 0.8 m, below which vehicles are well-modeled as boxes with vertical sides. Shadows tend not to cause a problem because features are rarely detected on the road itself due to lack of texture; as a result, shadows usually cause the height prediction of a feature to exceed the threshold, causing the

feature to be ignored. In a similar manner, although occlusions cause the point ***G*** to be estimated incorrectly, the estimated height will be above the threshold, since the problem only occurs when the feature is higher than the hood (or trunk) of the occluding vehicle. In addition, features rarely belong to the top surface, primarily due to insufficient texture and a relatively small projection in the image. Even when the assumptions are violated, the segmentation error is just 7% when the image height of ***G*** is perturbed by an error with standard deviation of 5 pixels, which we computed using simulations on equations [5] and [6]. Moreover, it is important to keep in mind that our technique only requires that at least one feature on a vehicle satisfy the assumption of lying on one of these four surfaces, so that multiple features with erroneous values do not pose a problem.

After estimating height of all the features using this technique, features which are close to the road surface are selected as *stable features*. In our previous work (*16*), stable features were selected based on an additional condition of low variance in height estimation for each frame of the block, but we have since found that removing this criterion reduces the number of computations without any noticeable degradation in the segmentation results.

World Coordinates from Multiple Frames

Factors like occlusion and shadows introduce significant error in the height estimates of many of the feature points obtained using the technique presented in the previous section, but as long as an accurate height is obtained for at least one feature per vehicle, the coordinates of that point can be used to estimate the world coordinates of the rest of the features on the vehicle. To accomplish this task, we employ rigidity constraints and a translational motion model.

A line in 3D can be represented in a parametric form as:

$$\boldsymbol{P = P_R + \alpha\,[P_H - P_R]}$$

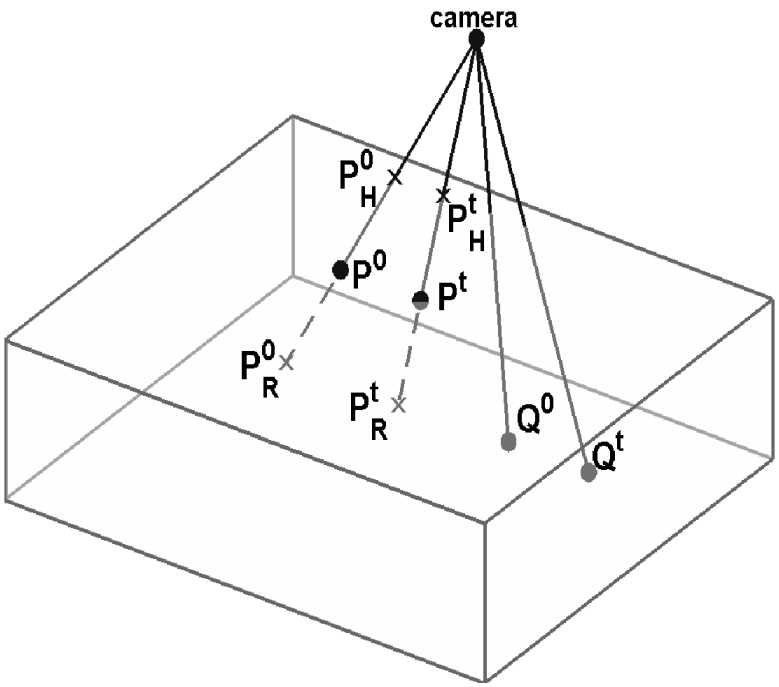
where $\boldsymbol{P_H}$ and $\boldsymbol{P_R}$ are 3 x 1 vectors representing any two points on the line, and $\boldsymbol{\alpha}$ is a scalar which defines location of a point along the line. The above representation simplifies the mathematical analysis to follow.

As shown in Figure 3, we consider two points, ***P*** and ***Q*** which undergo a translational motion from $\boldsymbol{P^0}$, $\boldsymbol{Q^0}$ in initial frame to $\boldsymbol{P^t}$, $\boldsymbol{Q^t}$ after duration ***t.*** If ***Q*** is one of the stable features, then its real world coordinates are known. As derived in (*18*), α can be solved as

$$\alpha = \frac{[\Delta_{P_H} - \Delta_{P_R}]^T [\Delta_Q - \Delta_{P_R}]}{[\Delta_{P_H} - \Delta_{P_R}]^T [\Delta_{P_H} - \Delta_{P_R}]} \qquad [7]$$

where Δ represents translational motion for the respective point.

FIGURE 3 Estimating world coordinates using rigid motion.



Coordinates of *P* are unknown. *Q* is a stable feature point with known world coordinates.

Estimates for the world coordinates of non-stable feature points can be computed with respect to each stable feature point using the two equations above. Among all the estimates, we select the estimate that minimizes the weighted sum of Euclidean distance in ***(x, y)*** and squared trajectory error over all *s* frames in the block.

$$\mathbf{P} = \min_k \left\{ w_d \; || \; \widetilde{\mathbf{P}}_k - \widetilde{\mathbf{Q}}_k \; ||_2 + w_e \left[\Delta_{P_k} - \Delta_{Q_k} \right]^T \left[\Delta_{P_k} - \Delta_{Q_k} \right] \right\}$$

$$\text{where, } k = 1, 2, \ldots, s \qquad [8]$$

Affinity Matrix and Normalized Cuts

We form the affinity matrix composed of three components, namely, the 3D Euclidean distance in world coordinates, the difference in trajectory and the *background content measure*. Euclidean distance and background content are measured using the coordinates of feature points in the first frame of the block. The details of the affinity matrix formulation can be found in (*18*). In Shi et al.(*22, 23*), it is mentioned that for the normalized cut algorithm to be computationally efficient, the affinity matrix (also called the weight matrix) should be sparse. Experiments were performed using sparse affinity matrices as well, i.e. using only local edge connections for a feature, but it was observed that using full matrices produced better results without a significant increase in the computing time.[2]

Grouping With Incremental Normalized Cuts

Based on Shi and Malik's normalized cuts, we have developed a grouping procedure that we call *incremental normalized cuts* for segmenting a set of features into meaningful groups. The process involves applying normalized cuts to the affinity matrix with increasing number of cuts until a valid group is found. The corresponding entries for the features in the detected group are removed from the affinity matrix and the process is repeated. The key part of this step is to use the calibration information to accept or reject a feature group based on following three criteria:

- The group has a minimum number of required features.
- The centroid (in 3D coordinates) lies inside the detection zone.
- Dimensions of the group are within a valid range.

Using incremental normalized cuts avoids explicitly specifying the number of cuts required for a good segmentation, which depends upon quantities such as the number of features and the number of vehicles. Further details about incremental normalized cuts can be found in (*15*).

2. For sparse affinity matrices using 70, 60 and 50 pixel neighborhood, the computational time saved was found to be 3.8%, 4.6%, 5.8% with the drop in accuracy of 14%, 22% and 31% respectively.

Correspondence Between Frame Blocks

In the previous sections, we looked at how to track feature points through a block of *F* frames, estimate the corresponding world coordinates, and group features using incremental normalized cuts. This procedure is applied to all the blocks of frames (overlapping by *F-1* frames), using the same set of parameters. Long-term tracking requires us to compute the correspondence between the vehicles detected in consecutive frame blocks. In this section we describe our approach for correspondence.

Consider two consecutive frame-blocks **A** and **B** overlapping by *F-1* frames. Let A denote the feature groups segmented in a frame-block **A**, and let B denote the feature groups segmented in frame-block **B**. An undirected graph is formed with the segmented feature groups in both frame blocks as nodes and the number of common feature points shared by a pair of groups as the weight of an edge connecting the respective nodes. If a group in the previous block shares features with only a single group in the current block, then we call this a one-to-one unique correspondence. A group in **A** sharing features with more than one group from **B** indicates splitting. Similarly, two or more groups in **A** sharing common features with a group in **B** indicates merging. A group in **A** having no association is considered a missing event, and a group in **B** having no association with any of the groups in the previous block is considered as a new detection. If a group is associated with a one-to-one correspondence over consecutive blocks, it is labeled as a reliable group. If a group is missing for consecutive blocks, it is labeled as inactive. During initialization, each group in the first frame-block is assigned a unique label. For each consecutive frame-block, a graph is constructed as mentioned above. To neglect minor segmentation errors, all the edges having weights are removed. This is followed by searching for the unique one-to-one correspondences between the groups of previous and current frame-blocks. Groups of the current block having unique correspondences are assigned the labels of respective groups in the previous block. After processing all the unique associations, the graph is searched for splits. For a split event, the edge with maximum weight is used for correspondence and the remaining edges are removed. Merge events are handled the same way. Groups in **A** which are no longer connected to any of the groups in **B** and are labeled as reliable, are declared missing. Groups in **B** which are not connected with any of the groups in **A** are declared as new detections. Each group that is declared as a new detection is matched with all the active missing groups to find a possible correspondence. If a correspondence with missing groups is not found, the group is assigned a new label.

EXPERIMENTAL RESULTS

The algorithm was tested on four grayscale image sequences, each containing 1200 frames captured at 30 frames per second. The camera was placed on an approximately 10 m pole on the side of the road. The sequences were digitized at 640 x 480 resolution. No preprocessing was done to suppress shadows or to stabilize occasional camera jitter. For each sequence, offline camera calibration was performed once, as explained earlier.

The first sequence was captured on a clear day. Vehicles are traveling in three lanes and there are moderate moving shadows. The second sequence shows a four-lane highway with the last lane blocked for maintenance work. The lane closure results in slow moving traffic with vehicles traveling close to each other. The sequence was captured during data collection for studying the effect of a workzone on freeway traffic (*24*). The third sequence was found to be even more challenging because the vehicles cast long shadows, making the process of segmentation based of size constraints harder. In Figure 4(b) the small vehicle traveling next to a trailer is correctly

segmented and tracked demonstrating the system's ability to handle severe partial occlusions. One simple method was tested for detecting and removing groups that belong to shadows as shown in Figure 4 (f). If the height of a group is below a threshold value, it is classified as a shadow group and is discarded. Having zero as the threshold (which is theoretically correct) does not yield the desired results, since the estimation process is based on the approximate calibration along with simple assumption for the shape of vehicles resulting in height estimation error. If the threshold is set higher, more shadow-groups are detected and discarded at the cost occasionally detecting a small vehicle (e.g. a compact sports car) as a shadow group. The fourth and the last sequence was also captured for the workzone study, and the vehicles are traveling close to each other at low speeds. Because of the presence of fog, the images in this sequence are noisy compared with those in the other sequences.

A quantitative assessment of the results on all the sequences is presented in Table 1. The algorithm is able to detect all of the cars and trucks in Sequence 4, and nearly all of them in Sequence 1. Sequences 2 and 3 proved to be more challenging, but the system nevertheless detects over 90% of the cars, and over 85% of the trucks. The detection rate for occluded vehicles ranges from 55% to 84%, showing that significant progress over previous systems has been made, while pointing to the need for further research. The false positive rate is generally 5 to 10% of the total number of vehicles and is usually caused by shadows.

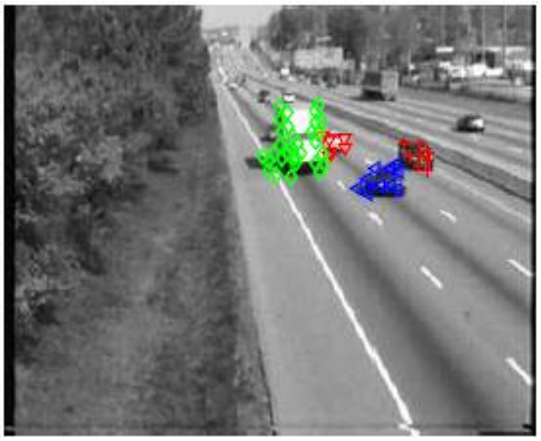
Video sequences demonstrating the performance of the system can be found at *http://www.ces.clemson.edu/~stb/research/vehicle_tracking.*

TABLE 1 Accuracy of the algorithm on the test sequences

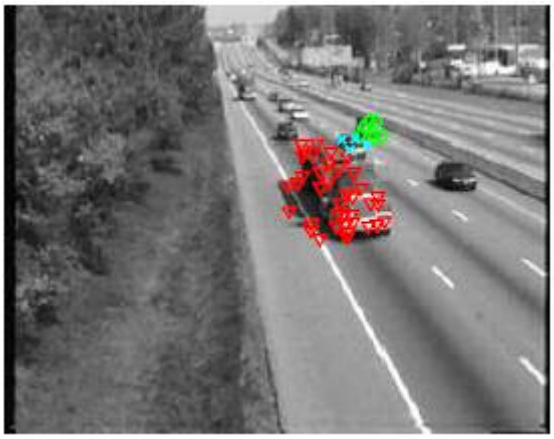
Sequence	C	T	O	DC	DT	DO	FP
1	116	9	19	114 (98%)	9 (100%)	16 (84%)	4
2	120	8	17	115 (96%)	7 (88%)	11 (65%)	4
3	57	7	11	53 (93%)	6 (86%)	6 (55%)	5
4	43	3	9	43 (100%)	3 (100%)	6 (67%)	2

The columns show the sequence, number of cars (C), number of trucks (T), number of vehicles among the cars and trucks that were significantly occluded (O), number of cars tracked (DC), number of trucks tracked (DT), number of occluded vehicles detected and tracked (DO) and number of false detections (FP) respectively.

FIGURE 4 Experimental Results.



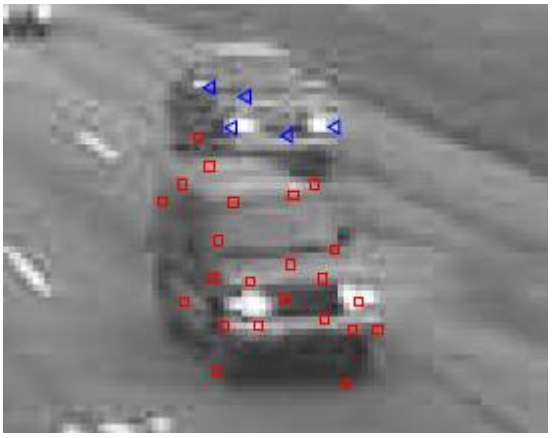
(a) Sequence 1, frame 35



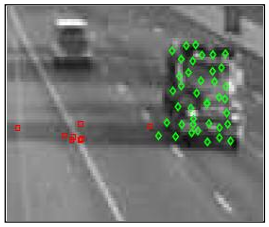
(b) Sequence 1, frame 592

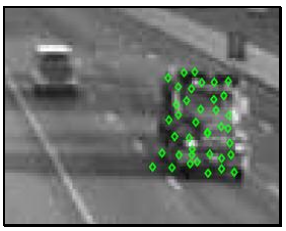
(c) Sequence 2, frame 330

(d) Sequence 3, frame 311



(e) Sequence 4, frame 415





(f) removing shadow groups

CONCLUSIONS AND RECOMMENDATIONS

Most approaches to segmenting and tracking vehicles from a stationary camera assume that the camera is high above the ground, thus simplifying the problem. A technique has been presented in this paper that works when the camera is relatively at a low angle with respect to the ground and/or is on the side of the road, in which case occlusions are more frequent. In such a situation, the planar motion assumption for vehicles is violated, especially in case of heavy vehicles like trailers. The approach proposed is based upon grouping tracked features using a segmentation algorithm called incremental normalized cuts, which is a slight variation of the popular normalized cuts algorithm. A novel part of the technique is the estimation of the 3D world coordinates of features using a combination of background subtraction, offline camera calibration (for a single camera), and rigidity constraints under translational motion. Experimental results on real sequences show the ability of the algorithm to handle the low-angle situation, including severe occlusion. This is a significant achievement in automated vehicle detection over commercially available systems. With further development, the proposed approach may lead to a next generation ITS sensor as well as an automated turn movement counter for use in conducting various traffic studies. In this later scenario, a camera mounted on a tripod set back from an intersection can collect video that can be digitally post-processed to determine turn-movement counts.

Some of the aspects of the proposed algorithm need further analysis and improvement. A simple approach has been adopted for associating the results between the frame-blocks which is based solely upon the number of common features. Using the spatial proximity, color, and motion information will help in making a more robust association. In addition, future research should be aimed at handling low-angle image sequences taken at intersections, where the resulting vehicle trajectories are more complicated than those on highways.

References

1. Lawrence A. Klein, Sensor Technologies and Data Requirements for ITS, Boston: Artech House, 2001.

2. Dan Middleton, Deepak Gopalakrishna, and Mala Raman. Advances in traffic data collection and management (white paper), January 2003. http://www.itsdocs.fhwa.dot.gov/JPODOCS/REPTS_TE/13766.html (Accessed on June 23, 2005)

3. 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, March 2002.

4. D. Magee. Tracking multiple vehicles using foreground, background and motion models. In *Proceedings of ECCV Workshop on Statistical Methods in Video Processing, (2002).*

5. D. Daily, F.W. Cathy, and S. Pumrin. An algorithm to estimate mean traffic speed using uncalibrated cameras. In *IEEE Conference for Intelligent Transportation Systems*, pages 98–107, 2000.

6. D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In *European Conference on Computer Vision*, pages 189–196, 1994.

7. 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.

8. 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.

9. J. M. Ferryman, A. D. Worrall, and S. J. Maybank. Learning enhanced 3d models for vehicle tracking. In *British Machine Vision Conference*, pages 873–882, 1998.

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

11. 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.

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

13. Jitendra Malik and Stuart Russel. Final Report for Traffic Surveillance and Detection Technology Development New Traffic Sensor Technology, December 19, 1996. http://citeseer.ist.psu.edu/malik96final.html (Accessed on October 21, 2005).

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

15. Zehang Sun, George Bebis and Ronald Miller. Improving the Performance of On-Road Vehicle Detection by Combining Gabor and Wavelet Features. *In Proceedings of the 2002 IEEE International Conference on Intelligent Transportation Systems*, September 2002.

16. Neeraj K. Kanhere, Shrinivas J. Pundlik, and Stanley T. Birchfield. Vehicle segmentation and tracking from a low-angle off-axis camera. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1152–1157, 2005.

17. Robert J. Schalkoff. *Digital Image Processing and Computer Vision*. New York: John Wiley & Sons, Inc., first edition, 1989.

18. Neeraj K. Kanhere, Vehicle Segmentation and tracking from a low-angle off-axis camera, Master's thesis, August 2005, Clemson University. http://people.clemson.edu/~nkanher/vehicle_tracking/media/kanhere_thesis_2005.pdf (Accessed on July 20, 2005)

19. S. Birchfield. KLT: An implementation of the Kanade-Lucas-Tomasi feature tracker. http://www.ces.clemson.edu/~stb/klt/ (Accessed on January 15, 2004)

20. Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, Apr 1991.

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

22. J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, Aug 2000.

23. Jianbo Shi and Jitendra Malik. Motion segmentation and tracking using normalized cuts. In *IEEE International Conference on Computer Vision*, pages 1154–1160, 1998.

24. Wayne Sarasua. Traffic Impacts of Short Term Interstate Work Zone Lane Closures: The South Carolina Experience. http://ops.fhwa.dot.gov/wz/workshops/accessible/Sarasua.htm (Accessed on June 23, 2005)