Video-Based Vehicle Detection and Tracking Using Spatiotemporal Maps

Yegor Malinovskiy, Yao-Jan Wu, and Yinhai Wang

Surveillance video cameras have been increasingly deployed along roadways over the past decade. Automatic traffic data collection through surveillance video cameras is highly desirable; however, sight-degrading factors and camera vibrations make it an extremely challenging task. In this paper, a computer-vision–based algorithm for vehicle detection and tracking is presented, implemented, and tested. This new algorithm consists of four steps: user initialization, spatiotemporal map generation, strand analysis, and vehicle tracking. It relies on a single, environment-insensitive cue that can be easily obtained and analyzed without camera calibration. The proposed algorithm was implemented in Microsoft Visual C++ using OpenCV and Boost C++ graph libraries. Six test video data sets, representing a variety of lighting, flow level, and camera vibration conditions, were used to evaluate the performance of the new algorithm. Experimental results showed that environmental factors do not significantly impact the detection accuracy of the algorithm. Vehicle count errors ranged from 8% to 19% in the tests, with an overall average detection accuracy of 86.6%. Considering that the test scenarios were chosen to be challenging, such test results are encouraging.

Automated vehicle detection has been an important component of freeway and intersection operation systems for decades. Inductance loop detectors have been the most popular form of detection systems since they were introduced in the early 1960s (*1*). They are relatively inexpensive in unit cost when compared with other detector types and can produce reliable traffic counts under most flow scenarios. However, loop detectors have their drawbacks. First, maintenance and installation of loop detectors require lane closures that may generate significant indirect costs (*2*). Such indirect costs may indeed make loop detectors more expensive than many other detector types. Second, loop detectors are point detectors. Several loop detectors are required to obtain advanced traffic parameters, such as vehicle speed and queue lengths, and such loop configurations further increase the costs. Furthermore, embedding inductance loops in pavement often causes damage to the pavement structure and therefore shortens the lifetime of pavements (*1*).

All these disadvantages have spurred further research in vehicle detection, with computer vision approaches quickly becoming popular alternatives. Video sensors not only have lower maintenance costs, but are also capable of providing richer traffic information than their inductance loop counterparts. Speed, queue lengths, and individual vehicle delay can be extracted from video images with proper video detection algorithms. However, since video-based vehicle detection algorithms are based on visual data, environmental factors and occlusions play significant roles in detection accuracy.

Good visibility of objects of interest is a key assumption in any video-based detection mechanism. Environmental impacts may degrade the visibility or alter the appearance of the objects in the scene. A robust video detection system should be insensitive to the impacts of shadows, sun glare, rapidly changing lighting, and sight-disturbing conditions, such as heavy rain. Additionally, vibration is a common problem for pole-mounted cameras. The resulting movements of camera vibration often cause displacements of static objects between current frame and background frame and therefore trigger a significant amount of false alarms in vehicle detection.

Vehicle occlusions are prevalent in most observation angles and are perhaps the most challenging to overcome. Occlusions result when one vehicle appears next to another and obscures it partially or completely. Typically, video-based vehicle detection systems will interpret two occluded vehicles as one, leading to undercounting errors. Therefore, occlusion issues must be properly addressed in video-based vehicle detection algorithms to improve detection accuracy.

This paper presents a novel approach for mitigating the environmental and occlusion impacts on video-based vehicle detection accuracy. The paper is organized as follows: It begins with a brief overview of the state of the art, followed by a detailed description of the proposed algorithm. Next, some testing results of the proposed algorithm are presented. The final section presents the conclusions of the study and recommendations for future studies.

STATE OF THE ART

Video-based vehicle detection has received much attention in the past two decades (see, e.g., *3–6*). Various algorithms have been developed and implemented and have resulted in several off-the-shelf commercial products, such as AutoScope and Traficon (*7, 8*). Unfortunately, many of these existing systems require ideal camera settings that are difficult to achieve, uncongested traffic flow conditions, or clear weather conditions for accurate detection. Most systems also typically require extensive calibration before being used for traffic data collection.

Automatic vehicle detection mainly consists of three steps: detection, classification, and tracking. The detection step segments objects of interest from the background. The classification step recognizes the types of the segmented objects and puts them into appropriate categories. The tracking step reidentifies the same object in a sequence of frames and enables motion data collection over a period

Department of Civil and Environmental Engineering, University of Washington, Box 352700, Seattle, WA 98195-2700. Corresponding author: Y.-J. Wu, yaojan@u.washington.edu.

Transportation Research Record: Journal of the Transportation Research Board, No. 2121, Transportation Research Board of the National Academies, Washington, D.C., 2009, pp. 81–89.
DOI: 10.3141/2121-09

of time. Through the above three steps, a complete spatiotemporal trajectory for each vehicle appearing in the field of view can be collected.

Various visual cues and patterns have been explored to accomplish the above steps. A common approach for vehicle detection is background subtraction. This method is based on the subtraction of a "static background" image from the current frame, thus revealing the objects in motion. The background image is commonly generated by processing several previous frames. This approach only performs well when each vehicle object can be completely segmented and there are no sight-degrading factors present, such as heavy rain, shadow, camera vibration, and sun glare. More complete overviews of background subtraction and some issues associated with this method can be found in Zhang et al. (*6*), Avery et al. (*9*), and Sun et al. (*10*).

Model-based approaches have also been popular means of detecting and classifying vehicles (*11, 12*). These approaches rely on a library of vehicle images as well as a model-searching algorithm. The results of these approaches can also be significantly affected by sight-degrading factors.

Kanade–Lucas–Tomasi (KLT) feature tracking has been a popular technique for vehicle tracking because of its relative insensitivity to noise and environmental effects (*13*). This method relies on motion-based features in the image and their respective locations for both tracking and detection. Motion-based feature points are those points with high gradient values in both the *X* and *Y* direction. These points can be found regardless of environmental conditions or camera movement. Hence, the KLT algorithm has been selected as one of the ideal candidates for vehicle tracking. Grouping the points, however, can be a challenging problem. Beymer et al. (*14*) suggested aggregating the points based on relative speeds, but those can often be too similar in relative speed to distinguish and distorted due to perspective. Kanhere and Birchfield (*15*) utilized a similar notion but used background subtraction to locate ground-plane features as a more accurate measurement of vehicle location, effectively reducing perspective distortion. Results obtained using this method are very promising, yet background subtraction is still subject to some environmental constraints.

Scan-line–based approaches gained early attention in vehicle detection because these approaches provide a convenient way of reducing input data and are suitable for real-time applications. Niyogi and Adelson (*16*) presented a unique approach to detecting pedestrian motion. The approach used a scan line to obtain the spatiotemporal information of moving objects. These values retrieved from a scan line were composed together along the time axis to create "XT-lines," a spatiotemporal map. Another scan line concept was adopted by Zhang et al. (*6*) to detect vehicles by comparing the pixel values along a detection line between the composed background image and the current input frame. Liu and Yang (*17*) recently attempted to extend this notion to vehicle tracking. However, they resort to background subtraction to segment the resulting strands, leaving the system vulnerable to environmental factors.

METHODOLOGY

Based on the strengths and weaknesses of the above-mentioned methods, the authors propose a new computer-vision–based algorithm for producing vehicle trajectories. Once vehicle trajectories are available, accurate volume counts can be extracted even when vehicles are occluded. This new algorithm contains four primary steps: user initialization, spatiotemporal (ST) map generation, strand analysis, and vehicle tracking. Each step may be composed of several substeps. These primary steps are marked in dashed boxes in the flowchart shown in Figure 1.

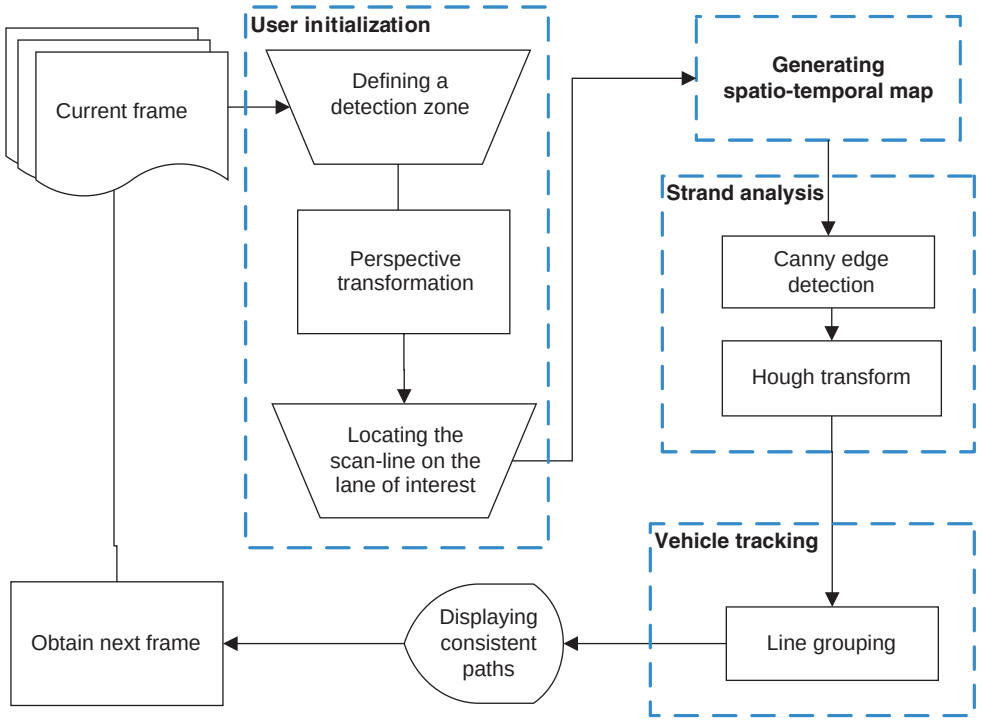


FIGURE 1 Flowchart of proposed algorithm.

The first step is performed only once and contains three substeps: defining a detection zone, perspective transformation, and locating the scan line on the lane of interest. A user must specify the detection zone of interest before proceeding on anything else. A perspective transformation is then calculated for the specified detection zone. The perspective transformation desired for our purposes changes the existing view angle of the surveillance camera to an overhead view of the specified detection zone. A detection zone may contain multiple travel lanes. The user needs to draw a scan line on each lane from which vehicle trajectory data will be collected in the transformed detection zone. After these user initializations, an ST map can be generated for vehicles traversing each data collection lane in the second primary step of the algorithm. Concurrently with the second step, the Hough line transform is used to retrieve a set of lines that describe the ST map.

The third primary step includes two substeps: Canny edge (*18*) detection and Hough transform (*19*). Through these substeps, lines representing ST movements of vehicles are obtained. When these line are grouped, individual vehicles can be identified and tracked in the last step of the proposed algorithm.

In this four-step approach, the ST map plays an important role. Although it is the second step in the algorithm, this paper introduces it first. The authors believe that a good understanding of the ST map will be helpful for understanding the user initialization process, strand analysis, and vehicle tracking steps.

Generating the ST Map

An ST map shows the time progression of a particular pixelwide slice of the image. This slice of the image corresponds to the user-defined scan line, as illustrated in Figure 2*a*. The pixel intensity values along this line are captured at every frame and are stacked onto the ST map along the time axis, as shown Figure 2*b*. All pixels along the scan line will leave traces. A group of traces captured from a moving object will form a diagonal strand. This implies that each strand represents a separate object moving along the scan line.

Compared with other vehicle detection and tracking methods, the ST map is fairly robust to sight-degrading factors, minor camera vibrations, and level of scene luminance. In addition, because an ST map can be used to track the spatial movement of an object, different types of occlusions may be distinguished as well.

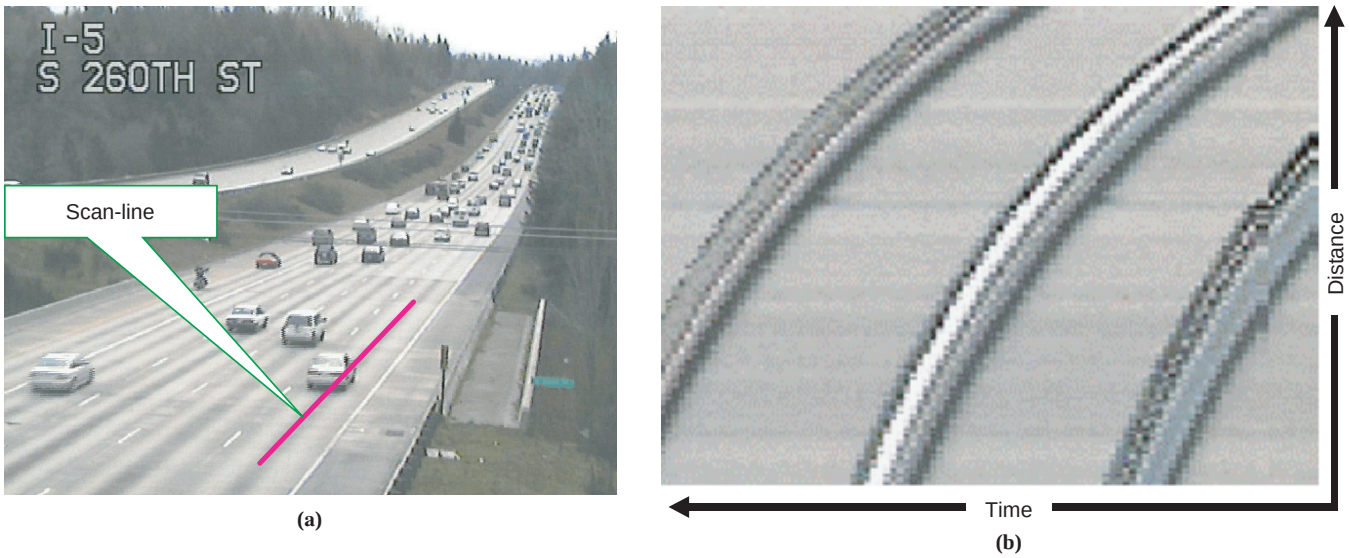
User Initialization

As is shown in Figure 2*b,* if a scan line is drawn on an input image without proper image transformation, the resulting ST map will be distorted due to the perspective effect. A distorted ST map creates more difficulties in strand analysis and does not accurately reflect the true trajectories of the vehicles. Hence, a perspective transformation is necessary to reduce distortion caused by the perspective effect on a two-dimensional image.

To accomplish this, the user should define a detection zone on the road surface in the input image. This detection zone should correspond to a square in the real world, with two edges parallel to the vehicle's travel direction and the other two edges perpendicular to it. As shown in Figure 3*a,* the user-defined detection zone, marked by the red quadrilateral on the image, can be transformed to a top-view image (see Figure 3*b*). The lane division markers are approximately parallel to each other, indicating that the perspective effect has been corrected through the transformation. After constructing a scan line on the top-view image, an ST map can be generated. The ST map expands from the left to the right with time. Figure 3*c* demonstrates an example ST map that shows that three vehicles have passed the detection zone via the rightmost lane at a nearly constant speed.

When the perspective transformation is performed through this approach, only the image points on the ground have accurate transformations. Points above the ground plane, however, are distorted, as shown in Figure 3. In addition, the height distortion of a vehicle enlarges as the distance between the vehicle and the camera increases.

Transformation of the scene image to the top-view image is performed through the homography matrix H_{ab}. To compute H_{ab}, it is necessary to know the real-world coordinates for at least four points in the two-dimensional scene image and four more in the real world. A homography is a mapping relationship between a point on



(a) **(b)**

FIGURE 2 An ST map example: (*a*) scan line and (*b*) ST map for rightmost lane.

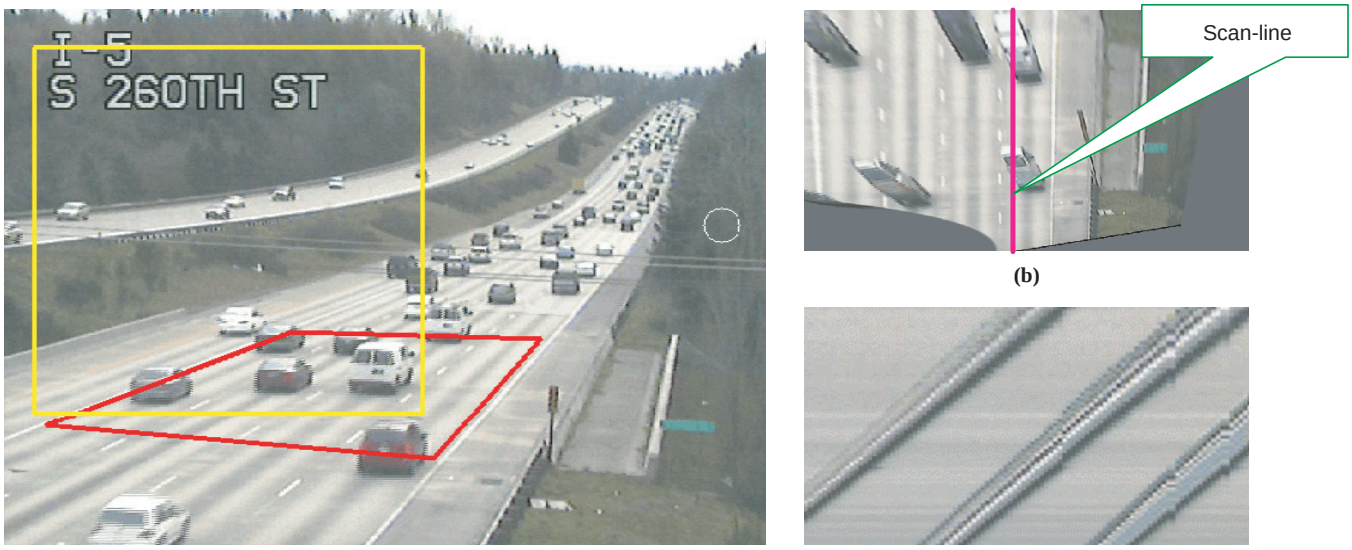


FIGURE 3 User initialization: (*a*) defining a detection zone, (*b*) user-defined detection zone after perspective transformation, and (*c*) ST map retrieved from scan line.

a ground plane and the same point on an image plane. The perspective transformation is computed using a 3×3 homography matrix H_{ab}:

$$H_{ab} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \quad (1)$$

If a point p_a in the image scene can be represented as $p_a = [x_a \quad y_a \quad 1]'$ and its corresponding matching point p_b in the real world can be represented as $p_b = [x_b \quad y_b \quad 1]'$, then the homography matrix can be computed using eight known points and the relationships

$$p_a = H_{ba} p_b \text{ and } p_b = H_{ab} p_a$$

Once elements in H_{ab} are fully determined, coordinate conversion relationship between the image coordinates and the top-view image can be established. The required points are obtained directly from user input—the square detection zone defined by the user serves as the four image coordinate points. Using the length of the bottom edge of the user drawn zone to create a hypothetical separate perfect square yields the set of real-world coordinates, which is represented by the yellow square in Figure 3*a*. This set of points only yields a rough, relative conversion, but that is all that is necessary to obtain linear trajectories at constant speeds.

Figure 4 shows the ST maps obtained from a variety of scenes. Figures 4*a* and 4*c* show the input images, and Figures 4*b* and 4*d* show the corresponding results of the ST maps.

Strand Analysis

Once the ST map is obtained, vehicle trajectories can be retrieved through strand analysis. This analysis aims at recognizing the strands present in the ST map and obtaining the coordinates along every strand to reconstruct the vehicle trajectories.

As mentioned earlier, strand analysis will be accomplished through two finer steps: Canny edge detection and Hough transform. Figure 5 illustrates the procedure of the proposed approach. Figure 5*a* shows the top-view detection zone with a scan line on the rightmost lane. Figure 5*b* shows a snapshot of the extracted ST map. There are three strands in this snapshot, indicating that three vehicles passed in the rightmost lane along the scan line. As shown in Figure 5*c*, Canny edges on the ST map are extracted by applying the Canny filter (*18*). The Hough line transform (*19*) is then applied to the Canny edges to retrieve complete lines from the strands. These identified complete lines are hereafter referred to as "Hough lines." In Figure 5*d*, these Hough lines are superimposed on the ST map to demonstrate the accuracy of the algorithm.

As mentioned earlier, the height distortion of a vehicle grows as the vehicle moves away from the camera. This implies that the extensions of all the Hough lines for each individual vehicle should theoretically converge at one point, illustrated in Figure 6. This characteristic of the obtained Hough lines serves as an important cue for detection and occlusion reasoning. The vehicle tracking problem now becomes an exercise in clustering the Hough lines.

One should note that a vehicle's ST trajectory is linear only when the vehicle is traveling straight and maintaining a constant speed through the detection zone. This approach is not valid in the cases in which the vehicle speed would vary significantly or a significant curvature is present in the highway geometry through the detection zone. To solve this problem, the detection zone can be defined to a smaller subset of the image, or a curve detection algorithm should be used. In this paper, only linear strands are considered based on the assumption that the vehicle travels with an approximately constant speed in the detection zone.

Vehicle Tracking

In reality, the Hough lines generated by the same vehicle may not converge to a single point due to the inherent error and redundancy

(a)

(b)

(c)

(d)

FIGURE 4 ST maps for various traffic environments: (*a*, *c*) input images; (*b*, *d*) corresponding ST maps.

of the Hough transform. A simplified sample problem is illustrated in Figure 7*a*. The dashed rectangle is the ST map being processed and Lines 1 through 7 are the extracted Hough lines. Grouping these lines by clustering these intersection points is a feasible solution but can be complicated, particularly when the number of the existing clusters is unknown.

Analyzing the intersections of the Hough lines is the key to determining the Hough line groups. Here the concept of "first intersection," a notion used to group related Hough lines, needs to be introduced. First intersection for a Hough line is defined as the first intersection with another Hough line that happens below the bottom of the ST map, toward the hypothetical point of convergence. A Hough line may intersect with multiple other Hough lines, but only the first intersection for a particular line is of interest.

Once the first intersection is found for a Hough line, the two intersecting lines are regarded as a Hough line pair. The Hough line pair relationships can be represented by a connected graph with undirected edges. Each node represents a Hough line. An edge represents a first intersection relationship between the connected two nodes. Hough lines generated by the same vehicle can be grouped together through a connected component analysis (*20*). Figure 7 demonstrates the concept of connected component analysis. The proposed algorithm searches from the bottom of the ST Map to find the first intersection point for each Hough line. As shown in Figure 7*a*, all the first intersection points are highlighted with large dots. For example, Line 7 first intersects Line 6; thus Lines 6 and 7 are regarded as a Hough line pair. In the connected graph, this new Hough line pair is represented by adding a new edge to connect Nodes 6 and 7. Following the same procedure, all Hough line pairs can be identified and represented in the graph as shown in Figure 7*b*.

Once the graph has been completely constructed, a depth-first search algorithm is used to determine the connected components which represent the line groups. The depth-first algorithm, commonly used in Graph theory, is a powerful tool for graph traversal and search (*21*). The BOOST C++ graph library (*22*) is used in this implementation to construct these graphs and determine the connected components.

Once the line groups have been established, the current and past positions of a vehicle can be obtained by taking the average slope of all the Hough lines in the same group. The vehicle positions

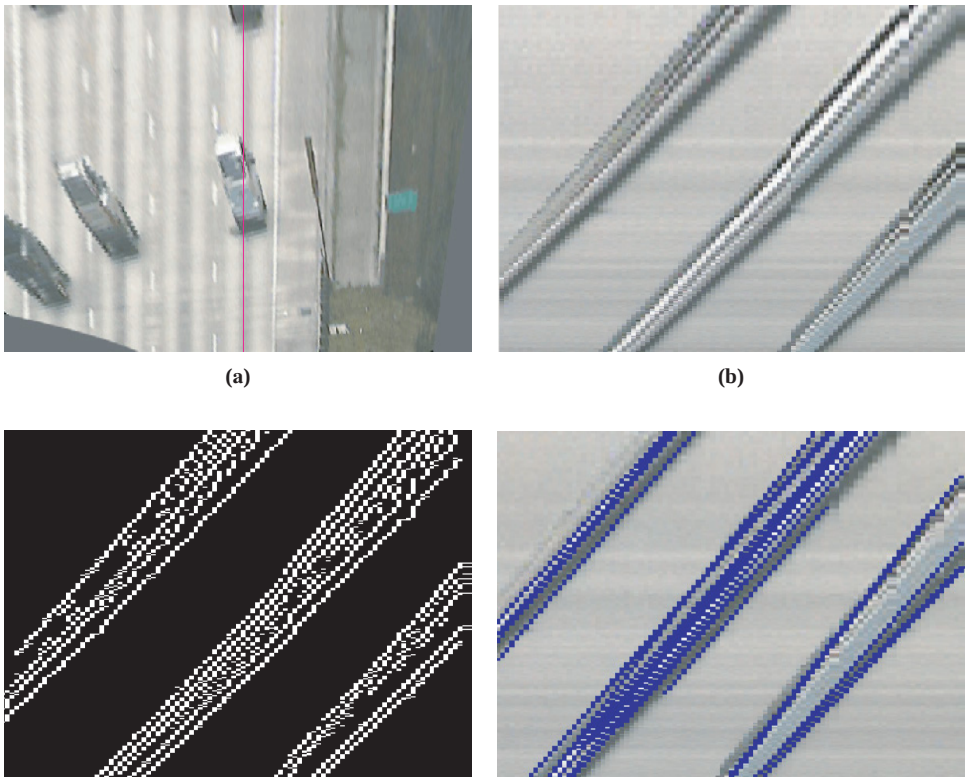


FIGURE 5 Strand analysis: (*a*) top-view detection zone with scan line, (*b*) ST map, (*c*) Canny edges of the ST map, and (*d*) result of Hough transform.

from the trajectories are mapped to the scan line to display the detected vehicles in motion. In Figure 8, Vehicle 24 and Vehicle 25 are tracked along the scan line. This demonstrates the advantage of the ST map because the historical vehicle trajectories can be easily found by using the ST map without implementing other tracking procedures.

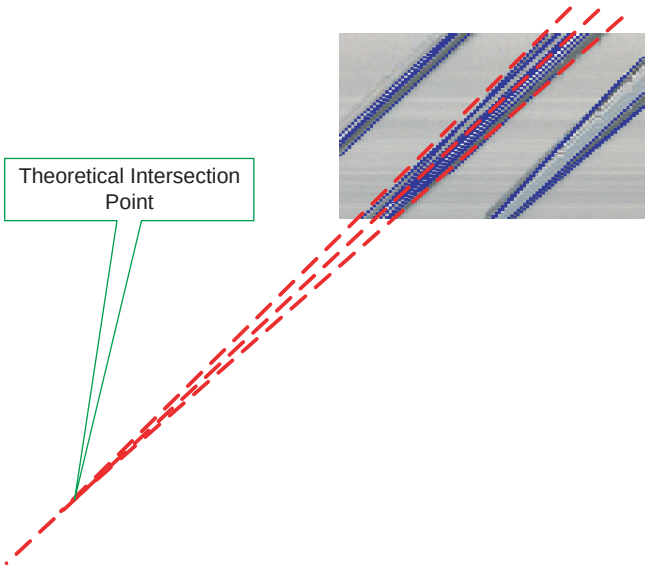


FIGURE 6 Extensions of Hough lines.

EXPERIMENTAL RESULTS

Proposed Algorithm

The proposed algorithm was implemented in C++ using OpenCV (*23*) and BOOST C++ (*22*) libraries. The algorithm runs in real time on a 1.83-Ghz Intel Centrino mobile processor.

The proposed algorithm was tested using six 10-min video sequences collected at different locations with different traffic flow conditions. Various environments and traffic flow conditions were tested to verify the robustness of the algorithm. As shown in Figure 9, the test sites were chosen from Washington State Department of Transportation surveillance cameras mounted along SR-520 and I-5 in the Greater Seattle area. Figures 9*a*, 9*b*, and 9*c* show snapshots captured from the SR-520 cameras mounted on a floating bridge. These locations can be especially challenging for computer-vision–based vehicle detection and tracking because of camera vibrations caused by wind and structure shake. Furthermore, Figure 9*b* displays a field of view that shows a curved roadway segment. Even though this sequence can be challenging for the vertical scan-line–based algorithm, vehicles were still properly detected and tracked using the proposed algorithm. Nighttime vehicle detection has been a difficult task in computer vision applications. The algorithm was also tested in a nighttime scenario, as shown in Figure 9*c*.

Figures 9*d*, 9*e*, and 9*f* show the captured images from the surveillance cameras deployed along I-5. These cameras also suffer from minor vibration problems, shadows, and sun glaring. For all test scenarios except for that shown in Figure 9*e*, the rightmost lane of the nearest observed direction was chosen as the data collection lane.

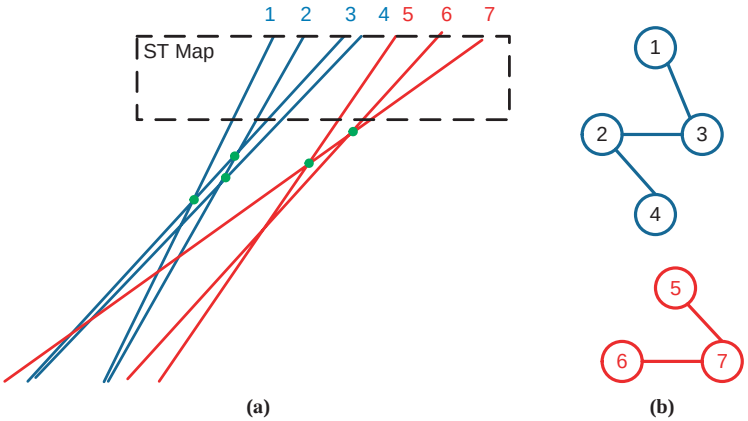


FIGURE 7 Demonstration of line grouping for vehicle detection: (*a*) Hough lines and (*b*) result of the constructed graphs.

For the I-5 50th Street site (Figure 9*e*), the leftmost lane of reversible section (the inner group) was selected to determine the effects of a static occlusion—in this case a light pole.

Experimental results for all the six test scenarios are summarized in Table 1. The count accuracies range from 81% to 92%. Considering that the test scenarios are chosen to be challenging, such results are encouraging. Of the six test scenarios examined, five undercounted vehicles by 8% to 19%. Only the night scene on SR-520 or the test scenario shown in Figure 9*c* overcounted vehicles by 15.4%. This overcount error was largely caused by the invisible (nontexture) areas between the lighted front and rear ends of some vehicles. The invisible areas generate gaps in the ST map, which can cause the grouping algorithm to prematurely group the front and rear areas separately into two vehicles. Besides the high nighttime false-positive rate, the algorithm tended to undercount vehicles in most test scenarios. This was determined to be caused by the inconsistencies created by the probabilistic Hough transform implementation in OpenCV. The probabilistic implementation of the Hough transform provided different lines at every frame, sometimes providing inconsistent results that were responsible for the loss of the moving objects.

Occlusions and Vehicle Shadows

Occlusions and vehicle shadows, the most prevalent causes of misdetections, are handled through the inherent characteristics of the proposed algorithm without any additional reasoning. The use of vertical scan lines for vehicle detection results in two types of encountered occlusions. The first type is the longitudinal occlusion that occurs between vehicles traveling in the same lane. The second type is the latitudinal occlusion that occurs between vehicles traveling in adjacent lanes. To some extent, the longitudinal occlusions were handled by the proposed Hough line grouping algorithm. Most of the occlusions encountered in all the test scenarios were of the longitudinal type because only one outer lane was selected for each test scenario. For the vehicles with the same height traveling closely at identical speeds, defining a longer scan line can increase the chances of successful segmentation. Latitudinal occlusions can be avoided by setting a scan line on a proper location without being occluded by vehicles in adjacent lanes. Such scan line placement is possible for most lanes in common observation angles, because the mounting positions are generally high enough to provide a wide field of view.

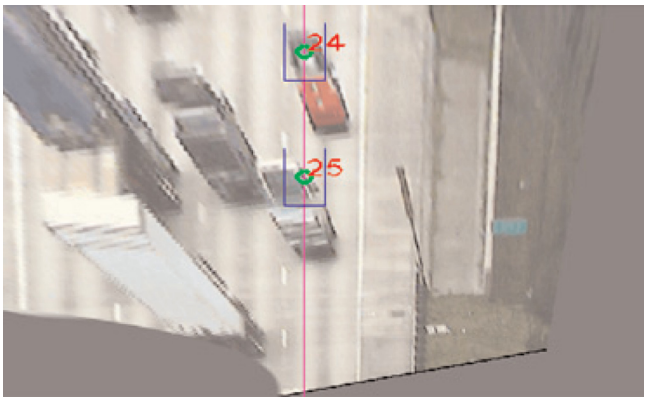


FIGURE 8 Result of vehicle tracking.

Vehicle shadows often pose problems as they are cast by and move with vehicles themselves. These shadows are often mistakenly regarded as additional vehicles by many other computer vision approaches. Because shadows typically do not contain any inner texture, the output of the ST map will have only two Canny edges for each vehicle shadow. One edge happens on the transition from the regular environment to the shadow region, and the other edge happens on the transition back. Therefore, in order to prevent a shadow from being identified as a vehicle, each line group should contain at least three lines. In the authors' experiments, a vehicle often has more than three Hough lines, as the windshield edges and bumper lines create numerous lines in addition to the vehicle borders. Although this three Hough line threshold may miss vehicles with extremely low-texture intensity, the chance of having such an event should be sufficiently small. Thus, the application of this threshold has effectively reduced false alarms caused by shadow effects in these experiments. Headlight blooms and reflections on wet pavement have similar attributes and were handled in the same manner.

SUMMARY AND CONCLUSIONS

Surveillance video cameras have been increasingly deployed along roadways over the past decade. Automatic traffic data collection through surveillance video cameras is highly desirable. However, sight-degrading factors and camera vibrations make it an extremely challenging task.

FIGURE 9 Selected test sites: (*a*) SR-520, West Highrise, looking east; (*b*) SR-520, West Highrise, looking west; (*c*) SR-520, East Highrise, looking west; (*d*) I-5, Southcenter; (*e*) I-5, Northeast 50th Street; and (*f*) I-5, Klickitat Road.

In this paper, a computer-vision–based algorithm for vehicle detection and tracking is presented, implemented, and tested. This new algorithm consists of four steps: user initialization, ST map generation, strand analysis, and vehicle tracking. It relies on a single, environment-insensitive cue that can be easily obtained and analyzed without camera calibration. The approach uses spatiotemporal slices that combine to create diagonal strands for every passing vehicle. The strands are then analyzed using the Hough transform to obtain groups of lines. A connected graph of the line objects is constructed for a connected-component analysis. Each connected line group represents one vehicle. Line group data can also be used to reconstruct vehicle trajectories and therefore track vehicles. Six test video data sets, representing a variety of lighting, flow level, and camera vibration conditions, were used to evaluate the performance of the new algorithm. Experimental results showed that environmental factors do not significantly impact the detection accuracy of the algorithm. Vehicle count errors ranged from 8% to 19% in the tests, with an overall average detection accuracy of 86.6%. Considering that the test scenarios were chosen to be challenging, such test results are encouraging.

Several conclusions can be drawn from the results. ST maps provide a consistent cue for vehicle detection, particularly after a perspective transformation is done. Analyzing the ST maps can be done by grouping linear segments found in the ST maps, but that restricts detection to constant flow. If such constraints are acceptable, the resulting algorithm is not only resistant to environmental effects, such as camera vibration and lighting changes, but also robust to moderate occlusions. Higher-volume flows, resulting in a larger number of vehicles, will generally have a lower accuracy due to a higher number of longitudinal occlusions.

TABLE 1 Test Results

Figure	Location	Conditions	Duration (min)	Manual Count	Algorithm Count	Count Error (%)
9*a*	SR-520 West Highrise E	Camera vibration	10	187	161	−13.9
9*b*	SR-520 West Highrise	Curvature, vibration	10	225	207	−8.0
9*c*	SR-520 East Highrise	Nighttime	10	130	150	+15.4
9*d*	I-5 Southcenter	Shadows, vibration	10	264	222	−15.9
9*e*	I-5 50th St	Light pole, shadows	10	100	81	−19.0
9*f*	I-5 Klickitat	Glare, vibration	10	165	151	−8.5
Total			60	1,071	972	

NOTE: Average absolute error = 13.4.

In terms of future work, the algorithm can be further improved by modifying the Hough transform implementation to make the detection results more consistent. Also, for more severe occlusions, placing scan lines on several lanes would be helpful to determine the origin of the occlusion. Heavy latitudinal occlusions can possibly be handled by analyzing all potentially occluding lanes and determining the relationship between the vehicles in each lane. If the relationship appears to be direct, a horizontal scan line can be used to determine whether the object extends into the adjacent lane. These directions should be investigated in future studies.

REFERENCES

1. *Traffic Detector Handbook,* 2nd ed. Institute of Transportation Engineers, Washington, D.C., 1998.
2. Wang, Y., and N. L. Nihan. Can Single-Loop Detectors Do the Work of Dual-Loop Detectors? *ASCE Journal of Transportation Engineering,* Vol. 129, No. 2, 2003, pp. 169–176.
3. Michalopoulos, P. G. Vehicle Detection Video Through Image Processing: The Autoscope System. *IEEE Transactions on Vehicular Technology,* Vol. 40, No. 1, 1991, pp. 21–29.
4. Gupte, S., O. Masoud, R. F. K. Martin, and N. P. Papanikolopoulos. Detection and Classification of Vehicles. *IEEE Transactions on Intelligent Transportation Systems,* Vol. 3, No. 1, 2002, pp. 37–47.
5. Kanhere, N. K., S. T. Birchfield, W. A. Sarasua, and T. C. Whitney. Real-Time Detection and Tracking of Vehicle Base Fronts for Measuring Traffic Counts and Speeds on Highways. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1993,* Transportation Research Board of the National Academies, Washington, D.C., 2007, pp. 155–164.
6. Zhang, G., R. P. Avery, and Y. Wang. Video-Based Vehicle Detection and Classification System for Real-Time Traffic Data Collection Using Uncalibrated Video Cameras. In *Transportation Research Record: Journal of the Transportation Research Board, No. 1993,* Transportation Research Board of the National Academies, Washington, D.C., 2007, pp. 138–147.
7. Autoscope. Image Sensing Systems, Inc. www.autoscope.com/. Accessed July 14, 2008.
8. Traficon. www.traficon.com/. Accessed July 14, 2008.
9. Avery, R. P., G. Zhang, Y. Wang, and N. L. Nihan. Investigation into Shadow Removal from Traffic Images. In *Transportation Research Record: Journal of the Transportation Research Board, No. 2000,* Transportation Research Board of the National Academies, Washington, D.C., 2007, pp. 70–77.
10. Sun, Z., G. Bebis, and R. Miller. On-Road Vehicle Detection Using Optical Sensors: A Review. *Proc., 7th International IEEE Conference on Intelligent Transportation Systems,* Washington, D.C., 2004, pp. 585–590.
11. Pang, C. C. C., W. W. L. Lam, and N. H. C. Yung. A Method for Vehicle Count in the Presence of Multiple-Vehicle Occlusions in Traffic Images. *IEEE Transactions on Intelligent Transportation Systems,* Vol. 8, No. 3, 2007, pp. 441–459.
12. Kollery, D., K. Daniilidisy, and H.-H. Nagelyz. Model-Based Object Tracking in Monocular Image Sequences of Road Traffic Scenes. *International Journal of Computer Vision,* Vol. 10, No. 3, 1993, pp. 257–281.
13. Tomasi, C., and T. Kanade. *Detection and Tracking of Point Features.* Carnegie Mellon University Technical Report CMU-CS-91-132. Carnegie Mellon University, Pittsburgh, Pa., 1991.
14. Beymer, D., P. McLauchlan, B. Coifman, and J. Malik. A Real-Time Computer Vision System for Measuring Traffic Parameters. *Proc., IEEE Conference on Computer Vision and Pattern Recognition,* Washington, D.C., 1997, pp. 495–501.
15. Kanhere, N. K., and S. Birchfield. Real-Time Incremental Segmentation and Tracking of Vehicles at Low Camera Angles Using Stable Features. *IEEE Transactions on Intelligent Transportation Systems,* Vol. 9, No. 1, 2008, pp. 148–160.
16. Niyogi, S. A., and E. H. Adelson. Analyzing and Recognizing Walking Figures in XYT. *Proc., IEEE Conference on Computer Vision and Pattern Recognition,* Washington, D.C., 1994, pp. 469–474.
17. Liu, A., and Z. Yang. Video Vehicle Detection Algorithm Through Spatio-Temporal Slices Processing. *Proc., 2nd IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications,* Washington, D.C., 2006, pp. 1–5.
18. Canny, J. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 8, No. 6, 1986, pp. 679–698.
19. Gonzalez, R. C., and R. E. Woods. *Digital Image Processing.* Prentice Hall, Upper Saddle River, N.J., 2000.
20. Shapiro, L. G., and G. C. Stockman. *Computer Vision.* Prentice Hall, Upper Saddle River, N.J., 2001.
21. Cormen, T. H., C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms,* 2nd ed. MIT Press and McGraw-Hill, New York, 2001, pp. 540–549.
22. *BOOST C++ Library.* www.boost.org/. Accessed July 13, 2008.
23. *Open Source Computer Vision Library* (OpenCV). Intel, Inc. www.intel.com/technology/computing/opencv/. Accessed July 13, 2008.

The Highway Traffic Monitoring Committee sponsored publication of this paper.