Available online at www.sciencedirect.com

SCIENCE DIRECT®

Pattern Recognition Letters 26 (2005) 1597–1607

Pattern Recognition Letters

www.elsevier.com/locate/patrec

Real time classification and tracking of multiple vehicles in highways

Roya Rad, Mansour Jamzad *

Department of Computer Engineering, Sharif University of Technology, Azadi Avenue, P.O. Box 11365-9517 Tehran, Iran

Received 3 December 2004; received in revised form 3 December 2004
Available online 7 April 2005
Communicated by E. Backer

Abstract

Real time road traffic monitoring is one of the challenging problems in machine vision, especially when one is using commercially available PCs as the main processor. In this paper, we describe a real-time method for extracting a few traffic parameters in highways such as, lane change detection, vehicle classification and vehicle counting. In addition, we will explain a real time method for multiple vehicles tracking that has the capability of occlusion detection. Our tracing algorithm uses Kalman filter and background differencing techniques. We used morphological operations for vehicle contour extraction and its recognition. Our algorithm has three phases, detection of pixels on moving objects, detection of a "Shape of Interest" in frame sequences and finally determination of relation among objects also in frame sequences. Our system is implemented on a PC with Pentium II 800 MHZ CPU. Its processing speed was measured to be 11 frames per second. The accuracy of measurement was 96%.

Keywords: Tracking; Highway; Vehicle type; Classification; Occlusion removal

1. Introduction

High speed processing of image frame sequences is highly important for many real time computer vision algorithms. Image sequence analysis provides intermediate results for conceptual description of events in a scene. In vehicle tracking applications that uses image sequences, many methods are suggested. At first we can name model-based tracking methods in which a 3D model of vehicle is extracted (Coifman et al., 1998; Malik and Russell, 1997). One of the advantages of these methods is their high accuracy in

* Corresponding author.
E-mail addresses: rad@ce.sharif.edu (R. Rad), jamzad@sharif.edu (M. Jamzad).

0167-8655/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2005.01.010

determining the vehicle type and their detail geometric model. In fact, model-based tracking methods because of their high calculation cost, can be used only for free-flowing traffic with small number of vehicles.

In some other methods that are called feature-based (Roberts, 1994), a few features such as distinguishable lines or corners are extracted for each vehicle. Some of these features are grouped together to label a vehicle (Shi and Tomasi, 1994). One of the most important advantages of this type of methods is that, even in presence of partial occlusion, some of these features remain to be visible. On the other hand, they face problems in detecting features for individual vehicles that run close to each other. In region-based methods (Coifman et al., 1998; Setchell, 1997), vehicles are presented as blobs. In these methods, at first, connected components are extracted and then regions are merged or split ted if needed. The most serious weakness of these approaches is that merging and splitting regions can cause some inaccuracy in vehicle detection.

In addition, there are some methods in which the contour of vehicles is extracted. Although contours can be detected by simple edge detection methods, but these simple methods sometimes detect false edges of the background too. However, if more complex algorithms of edge detection such as active contours or snakes (Paragios and Deriche, 2002) are used, one should find a way to optimize the coding to make them usable for real time applications with commercially available processors. In practice there are many applications such as our system, that often one does not need to know the exact detail of vehicle type, but a general type category would be sufficient. In our system the surveillance CCD camera is installed in a relatively far distance from a highway and the vehicles are visualized as small objects with minimum detail on their geometrical model.

In this paper we introduce a novel real time machine vision system for classification and tracking of multiple vehicles and also determining some traffic parameters such as lane change and counting the number of vehicles passing the highway during a desired time interval. For tracking we used Kalman filter (Grewal and Angus, 1993) and background differencing techniques. Our algorithm takes advantage of region based and contour based methods by combining their ideas in order to detect a "Shape of Interest", that in practice it is a bounding box around the vehicle. By using the bounding box and region boundary, the occlusion and overlapping of two regions are detected by examining the object shape and determining if it was the result of merging two or more vehicles and then deciding upon a proper split point to separate the merged vehicles.

Our system was implemented in Visual C++ using Matrox Meteor II frame grabber on a Pentium II 800 MHZ CPU. The input images were gray scale with eight bits per pixel resolution and of size 320×320. Our experimental results showed an accuracy of 96% when it was compared with the measurements done by a human expert. The initial version of this work is given in (Rad and Jamzad, 2003).

The rest of this paper is organized as follows: After a review of related works, our algorithm is described in three main sections, Change detection, Vehicle recognition (where our ideas for occlusion removal and vehicle classification are discussed) and Vehicle tracking. In Section 6, the experimental results is presented and finally the conclusion remarks is given in Section 7.

2. Related works

Many works have been reported for vehicle tracking from image sequences in machine vision and related topics literature. Vehicle detection is a fundamental component of image-based traffic monitoring system. Here we take a brief look at some of them. One such approach is to use background subtraction or optical flow for detection of moving objects (Javed and Shah, 2002; Gupte et al., 2002) and then tracking them. Methods based on background subtraction followed by object tracking do not suffer from the problem of false positive (Stauffer and Grimson, 2000) and can run in real-time. Cheung and Kamath (2004) compared various background subtraction algorithms for detecting moving vehicles in urban traffic video sequences. They considered approaches varying from simple techniques such as frame differencing and adaptive median filtering, to more sophisticated probabilistic modeling techniques.

In (Magee, 2004) a vehicle tracking algorithm was presented based on the combination of a per pixel background model and a set of single hypothesis foreground models based on a general model of object size, position, velocity, and color distribution. Each pixel in the scene was classified as either background, belonging to one of the foreground objects or a noise. Calibrated ground-plane information was used within the foreground model to strengthen the object size and velocity consistency assumptions. In addition in this paper, in order to provide a prior estimate of object velocity a model based on the information obtained from a learning phase designed for some specific scenes of traffic was used.

Kamijo and Sakauchi (2002) used the Spatio-Temporal Markov Random Field model (S-T MRF) for segmentation of Spatio-temporal images. This S-T MRF optimizes the segmentation boundaries of occluded vehicles and their motion vectors simultaneously (Bennett et al., 2004). They divided an image into blocks and optimized labeling of such blocks by referring to texture and labeling correlations along the temporal and spatial axes. This was done in combination with their motion vectors and stochastic relaxation method. They achieved good result for occluded vehicles, about 90–95% success rate for cross road images. But the speed of their algorithm was not acceptable for a real time approach, because it was 3 frames/s.

In (Kim and Malik, 2003), Kim presented a vehicle tracking method which used a model-based vehicle detection algorithm based on line features. It also gives a 3-D description of the vehicles that is based on probabilistic density functions (PDF's) of the 3-D distances between lines. We will compare our algorithm with this method in Section 6.

In (Zhao and Nevatia, 2001), Zhao and Nevatia presented an algorithm for detecting vehicles from aerial images by examining their rectangular shape, front and rear windshields, and the shadow. They reported good detection rate (about 90% with 5% false alarm) but with low speed (about 30 s plus the preprocessing time needed for images of size 1000×870 on a PII 400 MHZ). Although this method has a high accuracy but mainly due to the high computation needed for image-based comparison that is the main cause of its low speed, it can not be used in real time applications.

Stauffer and Grimson (1999) used a background–foreground segmentation scheme based on a mixture of Gaussian background model to track both vehicles and pedestrians. Connected component analysis was used to form foreground 'blobs' which are clustered on the basis of coherent motion using a set of Kalman filters. In (Magee, 2004), Magee took Gaussian mixture background modeling of Stauffer and Grimson, improved it's sensitivity to lighting changes and computational efficiency by combining with a novel foreground model that was based on dynamically modeling vehicle invariants such as size, color and velocity (assumed locally invariant in time) for a particular vehicle. In their method, foreground pixels (identified as outliers of the background model), rather than blobs or corner features/regions, were compared with current instances of foreground model to determine to which object they belong. This data was used to re-estimate model parameters. Moreover, they have used Kalman filter for predictive tracking, too. This algorithm is fast and runs near 20 fps (frame per second).

3. Change detection

Detection of changes between sequences of frames is a major task in many machine vision applications. Methods that are based on sequence frame differencing and moving edge detection, have aperture problem and are sensitive to vehicle speed. In the following we show how to use background differencing method to group pixels in moving and non-moving category.

In this method, first we construct a background reference image of the road that has no moving vehicles in it. In order to avoid the problem of changes in ambient light, this background reference image is updated periodically. In the update procedure, only those pixels that do not have a large difference with the corresponding pixel of the last updated version of the background image (i.e. they still belong to the background), are considered in the update procedure.

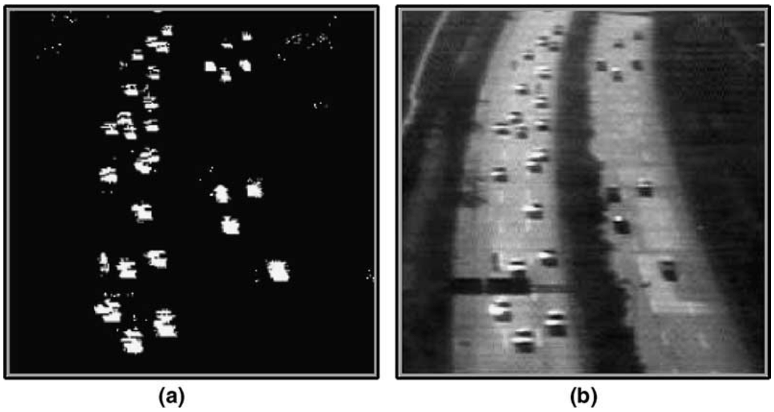

Fig. 1. (a) An example of a difference image. (b) The original frame of (a).

A mean filter is applied to the difference image for noise removal and smoothing. The reference image is used to detect changes in frame sequences. Fig. 1(b) shows a gray scale image and Fig. 1(a) is its corresponding difference image obtained at this stage.

4. Vehicle recognition

As seen in Fig. 1(a), this binary image has several small noises in it. Our vehicle recognition algorithm assumes to receive as input a binary image that only has two groups of pixels. Pixels belonging to the background, and those belonging to the moving objects. This means that we have to modify Fig. 1(a) in such a way that it becomes completely noise free. For noise removal, we used Closing and Opening morphological operators. It is known that Closing fills little apertures and Opening removes small particles depending on the size of the structuring element selected for these operations (Gonzalez and Woods, 2002). Fig. 2(b) shows the result of applying Opening followed by Closing on Fig. 1(a).

For detection of moving objects, we utilized a hybrid approach that uses features from both contour based and region based methods. To extract the boundaries of connected components (i.e. moving objects) from Fig. 2(b), the Laplacian filter

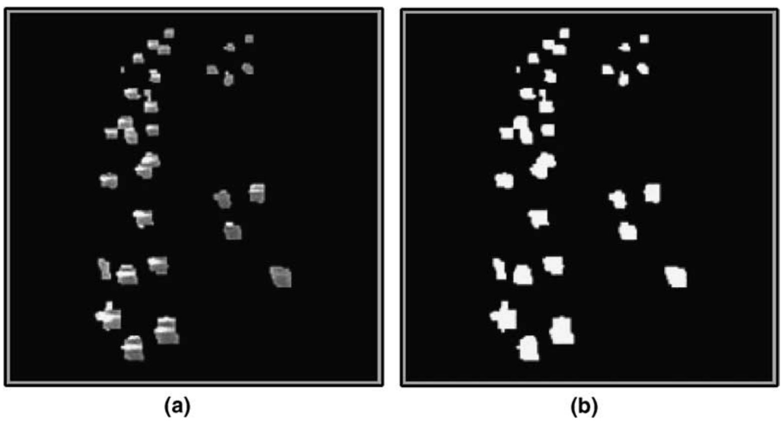

Fig. 2. (a) Vehicles that are separated from background. (b) Vehicles shown as filled in connected components that are resulted from morphological operators of Closing and Opening.

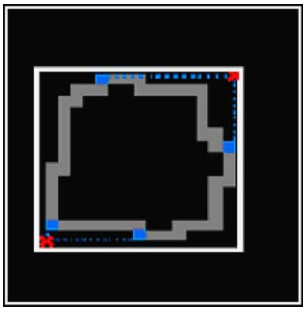

Fig. 3. Boundary of a vehicle and its bounding box.

(Gonzalez and Woods, 2002) was used. Since these objects have no holes inside, the Laplacian operator produces perfect boundaries. As a result the problem of having internal edges is avoided. Fig. 3 shows an example of the contour detected around an object (i.e. vehicle) and its corresponding bounding box.

In the following section, in order to speed up the tracking procedure, we do not process the irregular shape of the vehicle boundary but we track the bounding box corresponding to each vehicle.

4.1. Occlusion removal

Since the CCD camera of the control traffic system used in this research is installed in a relatively far location from the highway, often we will face the problem of occlusion. It means that, two vehicles running too close to each other, might be detected as one large vehicle. This is a serious problem in our application. Therefore, for each detected region, we must determine if it belongs to a single vehicle or it is a merged one. This problem is solved by examining the following conditions:

(1) Examine the trajectory of each vehicle. If the center points of two bounding boxes representing two vehicles in two consecutive frames are very close to each other, then an occlusion is predicted.
(2) Inspect the size of region. If the width of a region is more than a certain threshold, that region includes more than one object.
(3) Check the amount of change in region size. If a region suddenly becomes much larger or much smaller than its previous size, then a wrong merge or split of region has occurred.

If at least one of the above tests detects a wrong merged region, then that region must be split from an appropriate position. To find the split position, we carefully looked at the situations when the merge events have occurred. It was found that there were mainly three cases that caused a merge as follows:

(1) Horizontal merge. A vehicle merges with the one that is in its right side or left side.
(2) Vertical merge. A vehicle merges with the one located at its back or front.
(3) Diagonal merge. A vehicle merges with the one that is diagonally in its back or front.

In order to split a merged region, we used the real contour of the merged region and its bounding box. For example, to separate a horizontal merge, we follow the contour pixels of the region. On each contour pixel, we calculate its vertical distance from top and bottom sides of the bounding box corresponding to this region. These distance values are stored in two arrays called A and B using which another array C is calculated such that $C(i) = A(i) + B(i)$; $i = 1, 2, \ldots, n$, where n is the array size representing the maximum number of contour pixels.

Let the maximum value of elements in array C is detected at index k. Then we split the merged region from position k into two distinct left and right regions. This procedure is visualized in Fig. 4(a). Fig. 4(b) shows how the large bounding box is split into two smaller ones from a position k. Regions resulted from vertical and diagonal merges are split by similar approaches.

4.2. Vehicle classification

The bounding boxes extracted around vehicles are used for vehicle classification. In this research, we needed to divide the vehicles in three groups as Motorcycle & Bicycle, Car, and Bus & Minibus.

We found that for differentiating the Motorcycle & Bicycle and Car from each other, it was

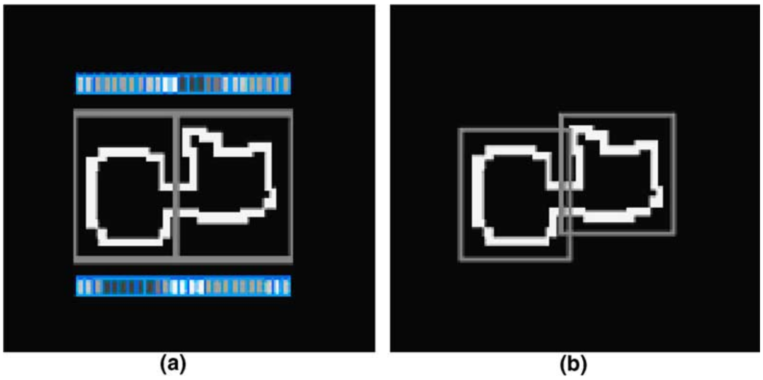

Fig. 4. (a) Finding the splitting point in a horizontal merge. (b) Splitting two horizontally merged objects into two separate ones using two bounding boxes.

sufficient to use the width of their bounding boxes. But to distinguish between Car, and Bus & Minibus, it was not sufficient to use the width or even the length of their bounding boxes. Therefore, we used the fact that usually heavy vehicles run more slowly than the cars. Thus, in this case the speed parameter was used for their separation.

In order to find vehicle types, we define a sub-window on the image frame. The size of this sub-window is determined in such as way that only one vehicle will appear inside this sub-window during the desired period of time. Then the type of all vehicles during the time they enter and leave this sub-window are determined as described in above.

If there are n frames during this desired period of time, in each frame a vehicle type is determined according to the above mentioned algorithm. Since the same vehicle is present in all n frames, in the best case, our algorithm should be able to detect a unique vehicle type in all n frames. But it is always possible that some noise can cause miss-calculation for the size of bounding box. In these situations, our algorithm might not be able to assign the correct type to those vehicles in each frame, therefore we assume to have a few miss-matches.

To find the true vehicle type, we calculate the frequency of occurrence of each vehicle type determined in all n frames. The type with highest frequency of occurrence is taken as the true vehicle type. In this way we could classify the vehicle type in three main groups such as motorcycle & bicycle, car and bus & minibus.

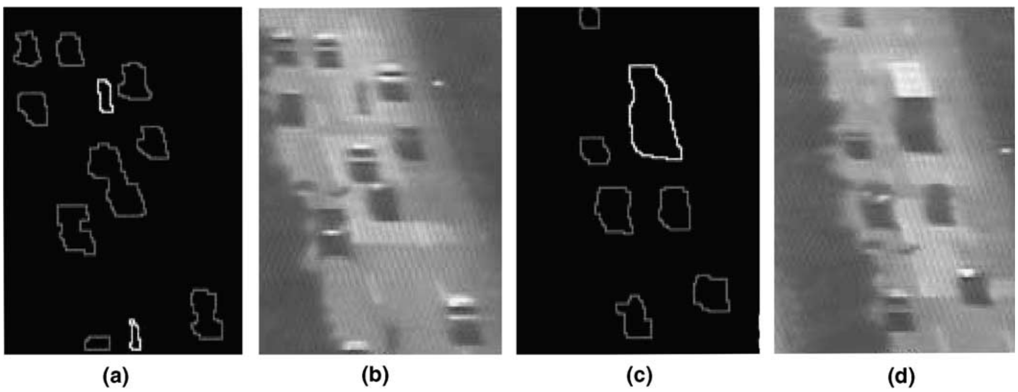

Fig. 5. (a) Comparison between two motorcycles (bold white contour) and cars (white contour). (b) The original figure of (a). (c) Comparison between a bus (bold white contour) and cars (white contour). (d) The original figure of (c).

Fig. 5(a) shows the result of classification between motorcycles and cars, and Fig. 5(c) shows how a bus and cars are classified.

5. Vehicle tracking

For tracking objects in a sequence of frames, the relation between objects in two consecutive frames must be found and recorded. Doing this we developed three modules. The first one is a complete image search, in which the whole image is searched. In the second module, only the area on the road (i.e. excluding the background) is searched. And in the last module, a small area in which the vehicle might be seen in next frames was determined and the trajectory of each vehicle was tested.

These three modules were tested. The first two modules were very time consuming and could not be used in real time, because their processing time were 3 and 6 frames per second, respectively. But the last module could be run in 25 frames per second. Therefore, it was used in our tracking procedure.

5.1. Prediction

In order to reduce the cost of search operation, we used the Kalman filter (Grewal and Angus, 1993) to predict the location of trajectory of a vehicle in frame sequences. The state vector of Kalman filter was defined as follows:

$$\hat{x}_k = (x_k, y_k, u_k, v_k) \tag{1}$$

where (x_k, y_k) is the center of bounding box and $(u_k,\ v_k)$ is the speed vector. The vehicle position is measured by Z_k that is computed from kth frame according to the following equation:

$$Z_k = H\hat{x}_k(-) + w_k \tag{2}$$

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \tag{3}$$

Eq. (2) shows the correspondence between the measured and actual position of the state vector. $\hat{x}_k(-)$ is the prior estimation of state vector, w_k is the measured error. H is the observation matrix for center of vehicle in frame number k.

5.2. Search in predicted area

In previous sections we described how to estimate a vehicle by drawing a bounding box around it. In each frame, the center points of all bounding boxes are determined. These points are used to define extended search windows in the next frame. The size of estimated search windows for next frame is obtained by adding a constant to the length and width of their corresponding bounding boxes in the present frame.

Fig. 6 shows these estimated search windows with gray lines, the bounding boxes are shown in solid white lines and the true curved shape contours of vehicles are shown inside the bonding boxes. During the search operation in search windows, we might face with the following scenarios:

(1) There is no vehicle in the search window. In this case, we use a phantom vehicle and record the predicted point as the center of this phantom vehicle.
(2) Only one vehicle is detected in the search window. In this case we record the center of that vehicle as a true center.

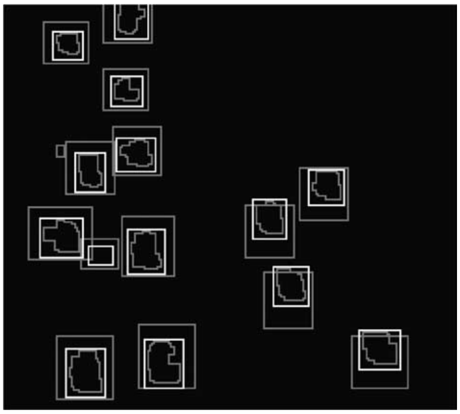

Fig. 6. Gray rectangles show the search windows. White ones are the bounding boxes estimated around the vehicle in a previous frame.

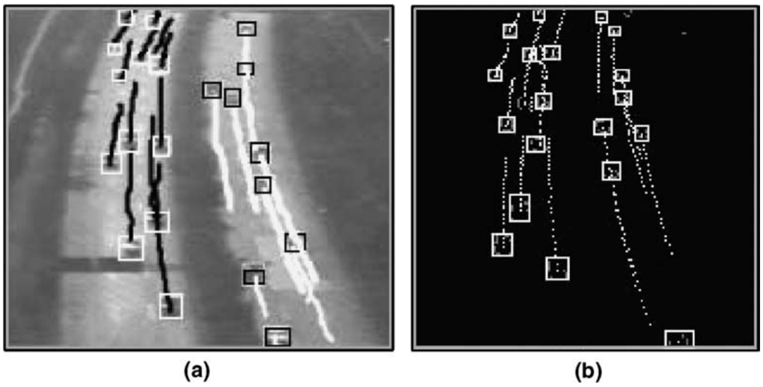

Fig. 7. (a) Tracking vehicles in a sequence of frames, visualized on an original frame. (b) Visualization on the binary image of (a).

(3) More than one vehicle is detected inside the search window. In this case, we must select the best candidate. We have used the following two parameters for decision-making. The first parameter was used to measure the similarity between sizes of bounding box found in current frame and its corresponding bounding box in previous frame. The second parameter was used to measure the distance between centers of bounding box found in current frame and that in previous frame.

In addition, we define a confidence coefficient for each vehicle. The value of this coefficient is increased or decreased depending on finding or not finding the desired vehicle in the search window, respectively.

However, a detected vehicle is ignored if the value of its confidence coefficient is too small (i.e. less than a certain threshold). In such cases these vehicles are marked as the lost ones. Fig. 7(a) shows a visualization of our tracking algorithm on an original frame and Fig. 7(b) shows the same visualization in its corresponding binary image.

The result of our tracking algorithm was used to determine lane changes for vehicles and to obtain the corresponding statistics. Detection of lane changes is done by extracting road traffic lines and calculating the number of vehicles in each frame inside every lane. The frequency of lane change is obtained by counting the number of vehicles detected in each lane in every frame.

6. Experimental results

In order to test our algorithm, we used image sequences of about 400 frames of a video tape captured by a traffic surveillance CCD camera. This camera was installed on a height and far distance from a wide highway in the city of Tehran. The recorded video showed two sides of the highway that has three lanes in each side. The average number of vehicles in a frame was measured to be 27.2. The mean processing speed of our algorithm measured on 400 consecutive frames is summarized in Table 1.

As shown in Table 1, the overall mean time for processing one frame is 91 ms. Therefore, the processing speed was 10.99 frames per second. Although this frame rate is about $\frac{1}{2.27}$ of the real time frame rate (i.e. 25 frames per second), but this speed obtained with a Pentium II 800 MHZ processor was quite acceptable for our purpose. However, using computers with higher CPU speed that are available these days and in future as well, this frame rate can be improved. In addition, the overall error of our system was measured by the following three criteria:

Table 1
Mean processing time in ms for each routine

Algorithm	Mean time (ms)
Lane change detection	20
Vehicle recognition	53
Search and prediction	18
Total time	91

(1) Wrongly measured regions. This error occurs when vehicles were very close to each other, or when the region includes more than two vehicles (i.e. our split algorithm can only split a large region into two vehicles).
(2) Wrongly split regions. This error often occurs when vehicles appear at the lower side or higher side of the image frame. In these cases only part of the vehicle is visible in that frame. In addition, wrongly split regions happen when a bus is mistaken with two vehicles.
(3) Losing vehicles in tracking. This error occurs in different situations. For example, when vehicles (i.e. specially motor cycles) are too far away from camera.

The error values of the above three categories are summarized in Figs. 8–10. The total error of our tracking algorithm is 5.4% that is the summation of the three error (i.e. 2.4% + 1.6% + 1.3% = 5.4%).

6.1. Comparison with related works

It is a fact that an exact comparison between vision-based works of traffic monitoring is nearly impassible because of difference of input stream as aspect of complexity, light condition, average number of vehicles running on the highway, quality of film, height and angle of camera, and also the quality of image acquisition. But we think that our hybrid method may give a pretty good result in comparison with others.

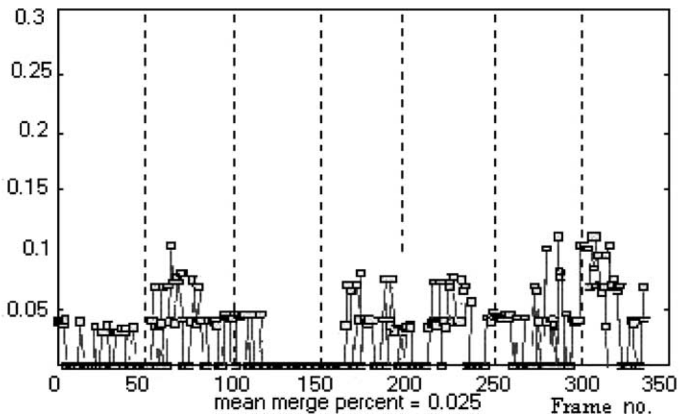

Fig. 8. Error of wrongly merged regions. The mean error is 2.5%.

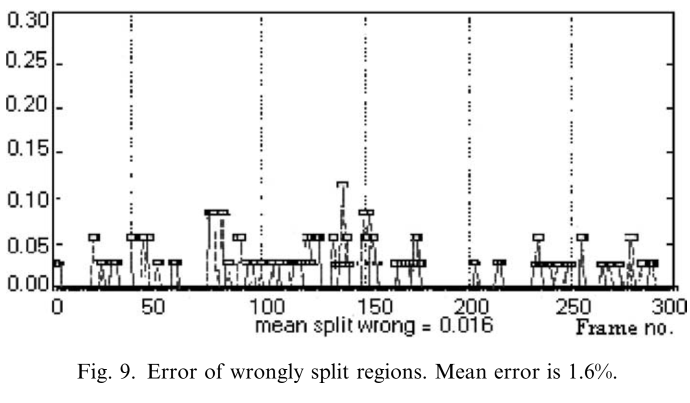

Fig. 9. Error of wrongly split regions. Mean error is 1.6%.

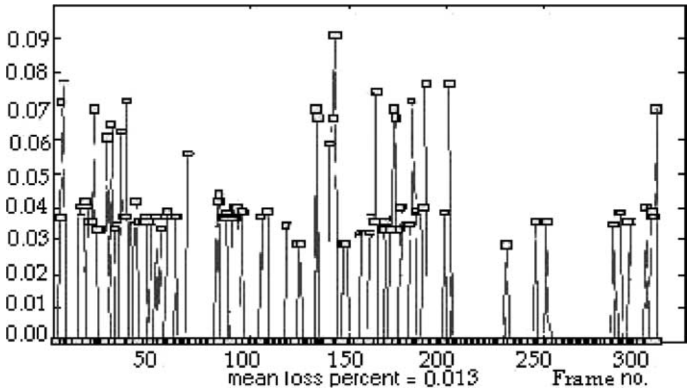

Fig. 10. Errors of losing vehicles. Mean error is 1.3%.

In order to evaluate our results, we compared it with a similar resent work that is done in Berkeley in 2003 (Kim and Malik, 2003). It is a model-based algorithm that uses line extraction and model fitting by dynamic programming. It is claimed that this algorithm has a frame rate of 10 fps for 200×200 frame size and an accuracy of 85%. The sense complexity of input stream used in this algorithm was similar to that of our method. Our method has superiority in speed: 10.99 fps for larger fame size (e.g. 320×320)), accuracy: 96% which is 11% higher than the above mentioned algorithm, and also in handling of special cases: tracking buses and trucks.

But the quality of trajectories provided by the above algorithm is superior to ours, because they detect edges from the front and rear of vehicles.

7. Conclusion

In this paper, we presented an algorithm for real time detection of vehicles, classification of their types, and tracking. Our system was implemented on commercially available PC and a frame grabber. Its processing speed is 10.99 fps. In this application, since video recording is done from a relatively far distance, the field of view of camera is large enough so that running vehicles remain in the field of view in such a period of time that the processing speed of about 11 frames per second seems to be a satisfactory real time speed. However, the processing time can be increased by code optimization and also using newly available PC's with higher CPU speed.

One of the main advantages of our system is its low cost and relatively high accuracy. It can be used for gathering statistical data for analysis in road traffic management systems. Specially the data obtained from the tracking routine can be used to study drivers behavior in highways and roads. A difficulty with vehicle tracking is the occlusion problem. It occurs when two or more vehicles merge into each other. In these cases, optical flow techniques and other approaches that can be implemented to provide better detection for occlusion, and therefore reduce the recognition error rate, can be used.

In order to reduce the effect of noise in detection routines, we need to develop methods for minimizing the size of bounding boxes drawn around vehicles in frame sequences. In addition, in future works we need to develop methods to handle the problem of shadows caused by ambient light during different hours of a day.

Acknowledgments

We would like to thank the Control Traffic Company of Tehran for their cooperation and providing the traffic video tapes.

References

Bennett, B., Magee, D., Cohn, A.G., Hogg, D.C. Using Spatio-temporal Continuity Constraints to Enhance Visual Tracking of Moving Objects, Proc. European Conference on Artificial Intelligence, pp. 922–926, 2004.

Cheung, S.-C., Kamath, C. Robust techniques for background subtraction in urban traffic video, Video Communications and Image Processing, SPIE Electronic Imaging, San Jose, January 2004.

Coifman, B., Beymer, D., Malik, J., 1998. A real-time computer vision system for vehicle tracking and traffic surveillance. Transport. Res.: Part C 6 (4), 271–288.

Gonzalez, R.C., Woods, R.E., 2002. Digital Image Processing. Addison Wesley, London.

Grewal, M.S., Angus, P.A., 1993. Kalman Filtering and Practice. Prentice Hall, Upper Saddle River, NJ, SA.

Gupte, S., Masoud, O., Martin, R.F.K., Papanikolopoulos, N.P., 2002. Detection and classification of vehicles. IEEE Trans. Intell. Transport. Syst. 3 (1), 37.

Javed, O., Shah, M. Tracking and object classification for automated surveillance, Proceedings ECCV, 2002.

Kamijo, S., Sakauchi, M. Illumination Invariant and Occlusion Robust Vehicle Tracking by Spatio-Temporal MRF Model. In: 9th World Congress on ITS, Chicago, Oct. 2002.

Kim, Z., Malik, J. Fast Vehicle Detection with Probabilistic Feature Grouping and Its Application to Vehicle Tracking. In: Proc. IEEE Int. Conf. on Computer Vision, 2003.

Magee, D., 2004. Tracking multiple vehicles using foreground, background and motion models. Image Vis. Comput. 22, 143–155.

Malik, J., Russell, S. Traffic Surveillance and Detection Technology Development: New Traffic Sensor Technology, PATH research report UCB-ITS-PRR-97-6, Institute of Transportation Studies, University of California, Berkeley, 1997.

Paragios, N., Deriche, R., 2002. Geodesic active regions: A new paradigm to deal with frame partition problems in computer vision. J. Visual Commun. Image Represent., 249–268.

Rad, R., Jamzad, M. Vehicle Tracking and Classification in Highways Using Region and Boundary Extraction. In: IASTED International conference on Visualization, Imaging and Image Processing (VIIP), Benalmedena, Spain, Sept. 8–10, 2003.

Roberts, J.M. Attentive Visual Tracking and Trajectory Estimation for Dynamic Scene Segmentation, Ph.D. thesis, University of Southampton, 1994.

Setchell, C. Applications of Computer Vision to Road-traffic Monitoring, Ph.D. thesis, Department of Computer Science, University of Bristol, Sept. 1997.

Shi, J., Tomasi, C. Good features to track. In: IEEE Conf. Comput. Vis. Pattern Recognit., 1994, p. 59300.

Stauffer, C., Grimson, W. Adaptive background mixture models for real-time tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1999, pp. 246–252.

Stauffer, C., Grimson, E., 2000. Learning patterns of activity using real-time tracking. IEEE Trans. PAMI 22 (8), 747–757.

Zhao, T., Nevatia, R. Car detection in low resolution aerial image. In: Proc. IEEE Int. Conf. Computer Vision, 2001.