Traffic Parameter Extraction using Video-based Vehicle Tracking

Young-Kee Jung and Yo-Sung Ho
Kwangju Institute of Science and Technology
1 Oryong-Dong Puk-Gu, Kwangju, 500-712, Korea
ykjung@gogh.kjist.ac.kr, hoyo@eunhasu.kjist.ac.kr

Abstract

In this paper, we propose a vehicle tracking algorithm that takes a new occlusion reasoning approach. We consider two different types of occlusions: explicit occlusion and implicit occlusion. We also propose a traffic flow extraction method with the velocity and trajectory of the moving vehicles. The proposed vehicle tracking system is composed of three parts: vehicle segmentation, vehicle tracking, and traffic parameter extraction. The vehicle segmentation part separates moving vehicles from their background. We employ the adaptive background approach, which does not update the background of moving objects. We also design a 2D token-based algorithm for vehicle tracking using Kalman filtering that has a modest amount of computational complexity. The traffic parameter extraction part estimates the traffic parameters, such as the vehicle count and the average speed. It also extracts the traffic flows. Finally, we have evaluated the proposed algorithm with some MPEG-7 test sequences.

1. Introduction

A video-based vehicle tracking system detects and tracks individual vehicle that is moving through the camera scene. This system can provide not only basic traffic parameters, including the vehicle count, average speed, occupancy, but also traffic flows such as normal or slow traveling of vehicles, vehicle traveling in the wrong direction, and stopped vehicles. Recently, traffic surveillance becomes an important topic in the intelligent traffic system (ITS). In addition, MPEG-7 is developing a framework for analyzing traffic video sequences in order to get high-level semantic scene interpretation based on segmentation and tracking of moving vehicles [1].

For multiple vehicle tracking, it is very important to maintain a tracking of moving vehicles before, during, and after an occlusion of vehicles. Several systems for multiple object tracking address the occlusion problem.

A contour-based tracking system, developed by USC [2], utilizes convex polygons as description of moving vehicle regions and keeps dynamic updating. In this system, they proposed an approach for tracking vehicles in road traffic scenes using an occlusion reasoning algorithm. The algorithm analyzes moving vehicles in the order of their depth (distance to the camera) and preserves the shape mask within the overlapping region. However, this approach did not resolve the initial occlusion problem where vehicles appear too close to one another in the scene. Therefore, vehicles can be labeled as one single object. This approach is appropriate only for the contour-based tracking system.

A tracking system of multiple humans under occlusion conditions is proposed [3]. In this occlusion reasoning algorithm, collision of two or multiple objects is predicted by estimated bounding boxes of objects. Once a collision of objects is detected by object segmentation in the previous frame, prediction of the next position is in progress until the end of occlusion. If merged objects are separated, the correspondence could be solved using the predicted object locations. However, this approach is likely to have a problem of correspondence mismatch when behavior of the object is nonlinear or prediction parameters before the occlusion have some errors. This approach also did not consider the initial occlusion problem.

In order to solve the correspondence mismatch and the initial occlusion problems, we propose two types of occlusion reasoning algorithms: explicit occlusion and implicit occlusion. Explicit occlusion represents the following situation: after two or more vehicles appear separately, they are merged due to some occlusion conditions. For the case of explicit occlusion, we apply a feature trajectory matching technique to reduce the possibility of the correspondence mismatch. Since the vehicle object is mostly occluded by large or fast moving vehicles, we should examine the speed, the size and gray level intensity values of the vehicle as features for matching. Implicit occlusion represents the initial occlusion where two or multiple vehicles appear as a single object.

Fig. 1 shows an overall block diagram of the proposed vehicle tracking system, which consists of three functional parts: vehicle segmentation, vehicle tracking, and traffic parameter extraction. The vehicle segmentation part separates moving vehicles from their background. In our scheme, we take the adaptive background approach, which does not update the background of the moving object region [4,5]. In the vehicle tracking part, we track the segmented object using a predictor. We design a 2D token-based algorithm for vehicle tracking based on Kalman filtering, which has a modest amount of computation requirements. The traffic parameter extraction part estimates traffic parameters, such

0-7803-4975-X/98/$10.00 © 1999 IEEE

as the vehicle count and the average speed for the vehicle. It also extracts the condition of traffic flow, such as normal speed traveling, slow speed traveling, wrong direction traveling, and stopped vehicles, using a classification rule for traffic motion events.

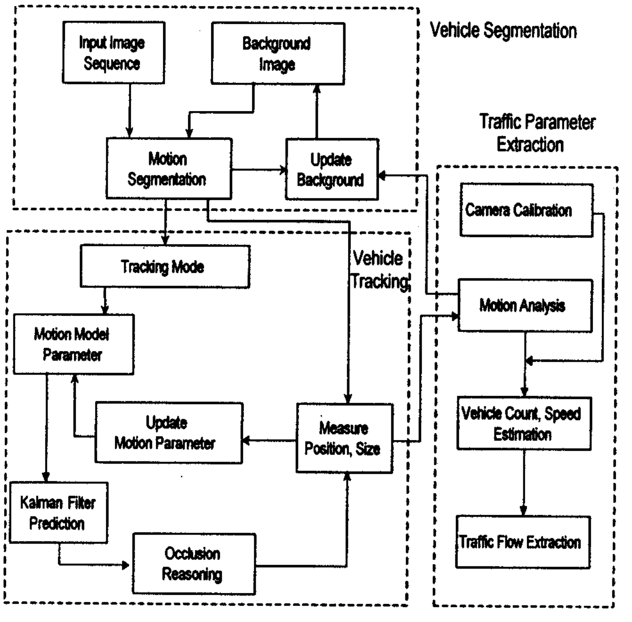


Figure 1. Overview of Vehicle Tracking System

2. Vehicle Segmentation

For adaptive background update, we utilize the temporal median operation. Fig. 2 depicts a proposed scheme for vehicle segmentation.

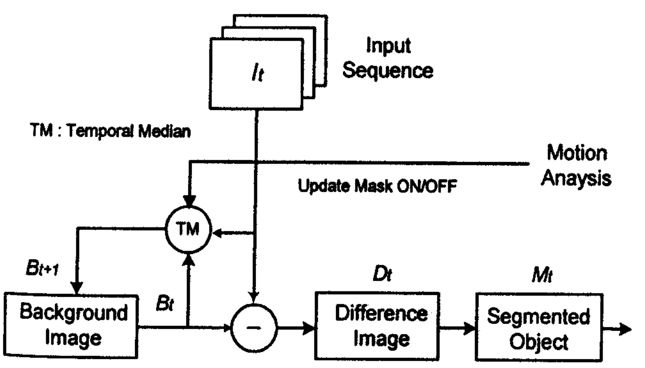


Figure 2. Vehicle Segmentation

For a frame I_t in the sequence, the background B_t is updated as follows:

$$B_{t+1}(x) = \begin{cases} B_t(x) + 1, & \text{if } B_t(x) < I_t(x) \\ B_t(x) - 1, & \text{otherwise} \end{cases} \quad (1)$$

Over a period of time, the background image contains the temporal median values of pixels. The moving blob mask M_t is obtained by thresholding the absolute difference of input image I_t and its background B_t at time t.

Since we are interested in vehicle motion of stopped or moving vehicles for traffic surveillance, we should be able to distinguish between the noise motion of temporal clutter and the motion of vehicles. For this purpose, we utilize the motion classification rule [5]. We then change the updated mask according to the object motion, and remove abrupt background changes or noise blobs of the stop motion.

3. Vehicle Tracking

Once we detect a moving vehicle, we can track the moving object efficiently by predicting the next center coordinate from the observed coordinate of the moving object. We design a token-based tracking scheme using Kalman filtering [6]. As shown in Fig. 3, the center position and the size of the moving vehicle are used as the token $t(k)$.

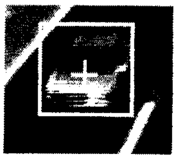


Figure 3. Token for Tracking

We assume the next token $t(k+1)$ is the sum of current token $t(k)$ and the token change $\Delta t(k)$. We can define a simplified polynomial motion model as follows:

$$t(k+1) = t(k) + \Delta t(k) \quad (2)$$

Kalman filtering provides a sequential and recursive algorithm for optimal linear minimum variance (LMV) estimation of system states $x(k)$.

We define the state variable as a four-dimensional vector, which represents the positional change of the target object per unit time interval and the size change of the target object.

$$x(k) = \begin{pmatrix} \Delta x\_center(k) \\ \Delta y\_center(k) \\ \Delta xsize(k) \\ \Delta ysize(k) \end{pmatrix} \quad (3)$$

The Kalman filter algorithm tries to estimate system states based on a set of measurement errors. We assume that a state model is linear and it is defined by

$$x(k+1) = \Phi(k,k+1)x(k) + w(k) \quad (4)$$

$$\Phi(k,k+1) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (5)$$

where $x(k)$ denotes the system state at time instant k, $\Phi(k,k+1)$ denotes a state transition matrix during the unit time interval, and $w(k)$ denotes an estimation error. Assuming that the trajectory of the target object varies with a constant acceleration and the size of the target object varies linearly, we can write the state transition matrix as in Eq. (5).

We also assume a linear relationship between the system states and a set of measurements.

$$z(k) = H(k)x(k) + v(k) \quad (6)$$

$$H(k) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (7)$$

where $x(k)$ denotes a set of measurements, $H(k)$ an observation matrix, and $v(k)$ measurement errors.

Once we define a system model and a measurement model, we can apply a recursive Kalman filtering algorithm to obtain LMV estimates of motion parameters. As shown in Fig. 4, the recursive Kalman filtering algorithm consists of three steps of operations: initialization, state prediction, and measurement update.

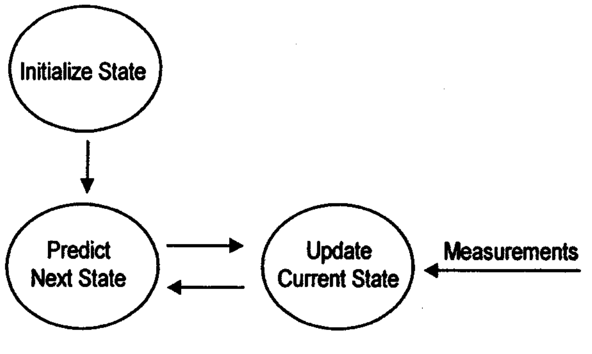


Figure 4. Tracking Operation

The initialization step determines the initial state estimate and the initial error covariance matrix which represents deviation of the initial state estimate from actual initial state. After the initialization step, we switch to the tracking mode. In the state prediction step, we determine a priori LMV estimate and its error covariance matrix for the current state, based on the previous state estimate and error covariance. In the measurement update step, we combine the estimated information with new measurements in order to refine the LMV estimate and its error covariance matrix for the current state. We perform this correction process with measurements on the positional change and size of the target object.

4. Occlusion Detection and Reasoning

For multiple vehicle tracking, the tracking algorithm needs to make data association in order to decide which of the observed objects should be associated with which track.

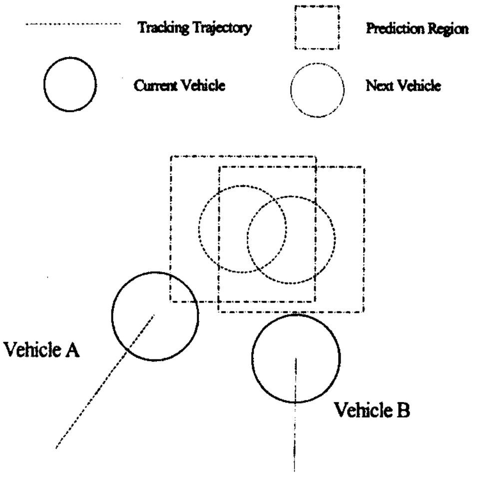


Figure 5. Nearest-Neighbor Rule

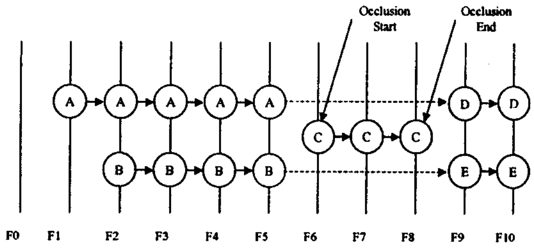
This data association problem can be solved by a nearest-neighbor prediction approach, which associates the nearest-neighbor object within the prediction region to each track [7]. However, in the occlusion situation shown in Fig. 5, the nearest-neighbor prediction is not good for data association.

We propose a new occlusion reasoning algorithm. We first define two types of occlusions: explicit occlusion and implicit occlusion. Each node of a trajectory is then defined as follows:

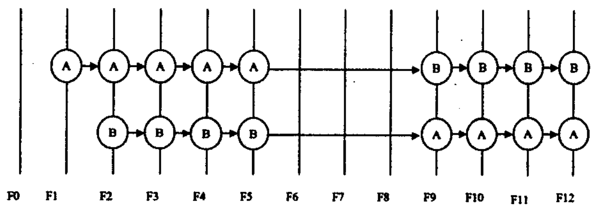
$$O_n^p(p=1,\ldots,K) = f(M,C,R,V) \quad (3)$$

where n denotes a discrete time, K is the number of moving blobs, $f(\cdot)$ is a set of features, such as shape mask(M), center position(C), bounding rectangle(R), and velocity(V).

After the merged blobs are separated, we can check whether the occlusion type is explicit or implicit. If the merged blobs had a collision, the occlusion type is explicit. In the case of the explicit occlusion, we applied the feature matching algorithm to make correspondence between the objects before and after the occlusion.



(a) merging and separation



(b) trajectory matching using features

Figure 6. Explicit Occlusion Reasoning

Explicit Occlusion Reasoning (Fig. 6)

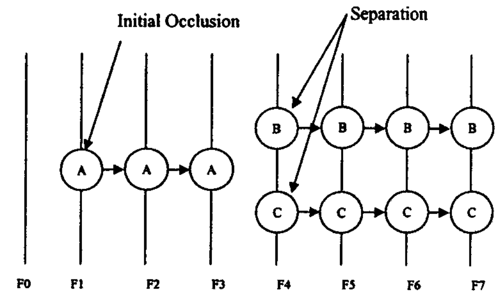
(a) The tracking algorithm predicts a collision of two or multiple objects using the estimated bounding boxes of objects.
(b) Once a collision of the objects A and B is detected, the reasoning algorithm creates a new trajectory of the merged object C
(c) The reasoning algorithm tracks the merged object C until the end of occlusion, and simultaneously predicts the objects A and B using parameters before occlusion.
(d) If the merged objects are separated, the nearest-neighbor object within the prediction region is associated with each track. Then, Eq. (10) is used for feature matching to validate the data association. The formula decides two trajectories that have the minimum feature distance.

$$
\begin{aligned}
&\underset{j}{\operatorname{argmin}} M_{i,j} = \sum_k w_k \left| f_i(k) - f_j(k) \right| \\
&i = D, E \\
&j = A, B \\
&M_{i,j} = \text{mesure for matching} \\
&\qquad \text{between i and j trajectories} \\
&w_k = \text{weighting factor} \\
&f(k) = \{\text{velocity, size, gray intencity}\}
\end{aligned}
\qquad (10)
$$

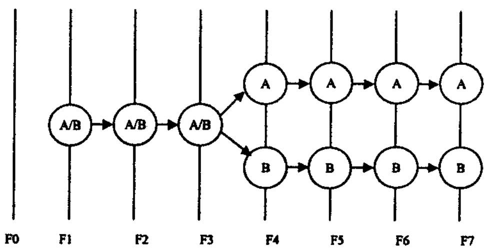
In the case of implicit occlusion, the initial occlusion reasoning algorithm is used.

Implicit occlusion reasoning (Fig. 7)

(a) If the merged objects are separated, the reasoning algorithm creates the new trajectories of the separated objects (B and C)
(b) The reasoning algorithm connects the common trajectory A to trajectories of B and C.



(a) separation



(b) connection

Figure 7. Implicit Occlusion Reasoning

5. Traffic Parameter Extraction

In order to measure the passing vehicle count and the average speed of vehicles, we define tracking ranges as solid lines in Fig. 8. If a new vehicle is detected in the detection area of white solid line box, the tracking algorithm tracks the vehicle until the end of the white solid line.

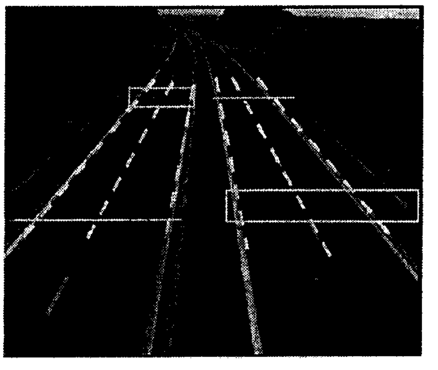


Figure 8. Definition of Tracking Ranges

Based on the velocity and trajectory of vehicles, and the geometric information of lanes of white dotted lines in Fig. 8, we determine the type of the vehicle behavior out of four categories: NORMAL TRAVELING, SLOW TRAVELING, WRONG DIRECTION TRAVELING, and STOP. We design the following classification rule for traffic motion events.

(a) NORMAL TRAVELING

$$\begin{aligned}&\text{for a vehicle } p\\&(O_n^p(c) = \mathit{within\,Lanes})\,\&\,(O_n^p(v) \geq TH_v)\\&n = n_0, \ldots, n_l\\&n_l : \text{trajectory length}\end{aligned}$$

(b) SLOW TRAVELING

$$\begin{aligned}&\text{for a vehicle } p\\&(O_n^p(c) = \mathit{Within\,Lanes})\,\&\,(O_n^p(v) < TH_v)\\&n = n_0, \ldots, n_l\end{aligned}$$

(c) WRONG DIRECTION TRAVELING

$$\begin{aligned}&\text{for a vehicle } p\\&(O_n^p(c) \neq \mathit{within\,Lanes})\end{aligned}$$

(d) STOPPED TRAFFIC

$$\begin{aligned}&\text{for a vehicle } p\\&O_{n_l}^p(v) \approx 0\end{aligned}$$

6. Simulation Results

The computer simulations have been performed on five video sequences of highway traffic scenes, which are used in the MPEG-7 evaluation. Each sequence has a MPEG-1 format of 25 Hz and 352x288 pixels/frame. The road contains two traffic lanes and one emergency lane for each traveling direction The sequences are taken from the same camera and contain some abnormal behavior of vehicles, such as emergency lane traveling and stopped.

Table 1. Test Video Sequences

	Sequence Length (Time /Frame)
Speedwa1.mpg	00'56" / 1419
Speedwa2.mpg	08'47" / 13183
Speedwa3.mpg	09'20" / 14017
Speedwa4.mpg	05'00" / 7494
Speedwa5.mpg	05'00" / 7495

We have simulated the proposed tracking algorithm to estimate the vehicle count and the average speed of moving vehicles. The vehicle count is computed for directions and compared with manually inspected data. The result is summarized in Table 2, where we note that the proposed algorithm works very well with 99% detection rate for five test sequences.

Table 2. Results of The Vehicle Count

Test Sequence	Manual Inspection	Proposed Algorithm
Speedwa1.mpg	15	15
Speedwa2.mpg	157	156
Speedwa3.mpg	179	177
Speedwa4.mpg	84	82
Speedwa5.mpg	72	72

The average speed of each vehicle is estimated in the unit of 200 ms using the moving distance in the world coordinate. We use 16 calibration points at four vertical poles, which are circled in Fig. 9. The least mean squares (LMS) method is then used to find the translation vector and the projection matrix from the image and the world coordinates of the calibration points [8].

As shown in Fig. 9, one vehicle drives towards the camera at constant speed of 120km/h and its interval speed for 200ms is estimated during the period of time 2000ms. The estimated results are at 110, 111, 112, 115, 116, 113, 115, 114, 114, and 113 km/h. Test results show that the speed error is about 5~10 km/h. Especially, the speed error is large when the location of the car is far from the camera, since incorrect image points are used for camera calibration.

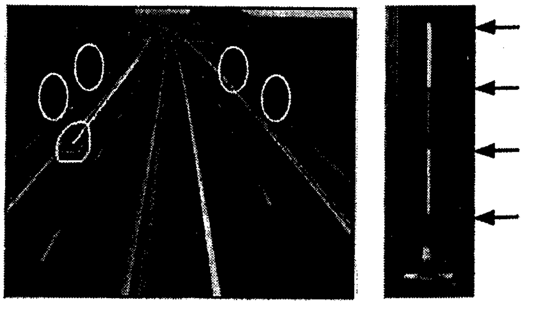
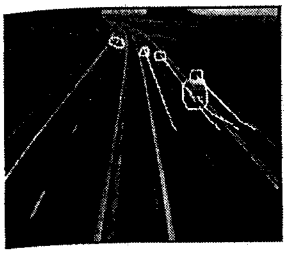
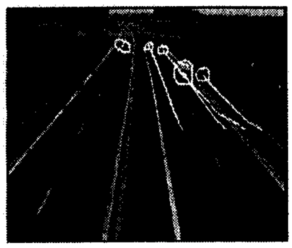


Figure 9. Interval Speed Estimation

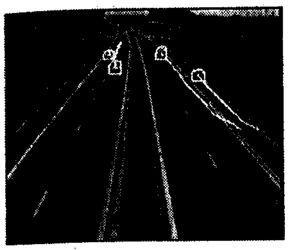
We also evaluate the proposed occlusion reasoning algorithm for explicit and implicit occlusion cases. In Fig. 10(a) and Fig. 10(b), one vehicle enters the scene on the right emergency lane, stops and occludes with the other vehicle. The tracking trajectory of the right-side road in Fig. 10(b) is the result that we obtain by the explicit occlusion reasoning algorithm. In this case, because two vehicles have large difference of speeds and gray intensity values, the feature matching algorithm easily validates the correspondence.



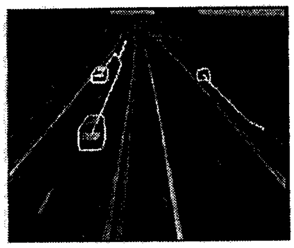
(a) Frame # 3526



(b) Frame # 3539

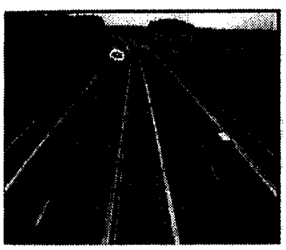


(c) Frame # 3569

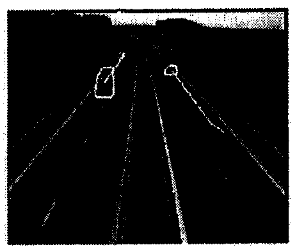


(d) Frame # 3598

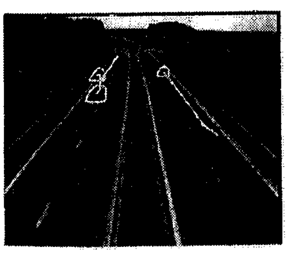
Figure 10. Explicit/Implicit Occlusion Reasoning



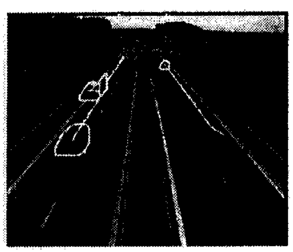
(a) Frame # 1765



(b) Frame # 1823



(c) Frame # 1824



(d) Frame # 1841

Figure 11. Implicit Occlusion Reasoning

In Fig. 10(a), two vehicles appear too close to each other in the left side of the road; therefore, it is difficult to separate the two vehicles by the vehicle segmentation algorithm. In Fig. 11(a), one vehicle passes over another vehicle, driving towards the camera, and creating a visual occlusion. Therefore, the implicit occlusion reasoning algorithm tracks the vehicles until the end of initial occlusion. When two vehicles are separated, the reasoning algorithm generates two tracking trajectories and connects the previous trajectory to new trajectories, as shown in Fig. 10(c) and Fig. 11(c).

7. Conclusion

In this paper, we propose a vehicle tracking algorithm that takes a new occlusion reasoning approach. We also suggest a traffic flow extraction method that has four categories of vehicle behaviors, such as normal traveling, slow traveling, wrong direction traveling, and stop. Our vehicle tracking system consists of three modules: vehicle segmentation, vehicle tracking, and traffic parameter extraction. Five MPEG-7 test video sequences are used for evaluating the performance of the proposed scheme. From the speed estimation simulation, the speed error was about 5~10 km/h. However, we expect that the speed error could be reduced to 0~5 km/h for better image coordinates of calibration points using higher resolution images. The proposed occlusion reasoning algorithm worked well for two types of occlusions. Since various kinds of occlusions may occur by speed and volume differences of moving vehicles in the traffic scene, our algorithm for feature matching could lower the possibility of correspondence mismatch.

Acknowledgements

This work is supported in part by LG Industrial Systems R&D Center in Korea.

References

[1] K.W. Lee, J.W. Kim, "An Automated Abstraction and Search for Surveillance Video based on Spatio-Temporal Events of Objects", ISO/IEC JTC1/SC29/WG11 MPEG99/p615, Feb. 1999.

[2] J. Malik and S. Russell, "A Machine Vision Based Surveillance System for California Roads", PATH Project MOU- 83 Final Report, University of California, Berkeley, 1994.

[3] R. Rosales and S. Sclaroff, "Improved Tracking of Multiple Humans with Trajectory Prediction and Occlusion Modeling", IEEE Conf. On Computer Vision and Pattern Recogntion, June 1998.

[4] N. McFalane and C. Scholfield, "Segmentation and Tracking of Piglets in Images", Machine Vision and Application, Vol. 8, pp. 187-193, Oct. 1995.

[5] Y.K. Jung and Y.S. Ho, "Robust Vehicle Detection and Tracking for Traffic Surveillance", Picture Coding Symposium'99, pp. 227-230, April 1999.

[6] A. M. Tekalp, Digital Video Processing, Prentice Hall, 1993.

[7] B.S.Y.Rao, H.F Durrant-Whyte, J. A. Sheen, "A Fully Decentralized Multi-Sensor System For Tracking and Surveillance", The International Journal of Robotics Research, Vol. 12, No. 1, pp.20-44, February 1993

[8] R. Jain, R. Kasturi, B. Shunk, Machine Vision, McGraw-Hill, 1995