A Feature-Based Vehicle Tracking System in Congested Traffic Video Sequences

Young-Kee Jung[1] and Yo-Sung Ho[2]

[1] Honam University
59-1 Seobong-Dong Kwangsan-Gu, Kwangju, 506-090, Korea
ykjung@mail.honam.ac.kr

[2] Kwangju Institute of Science and Technology
1 Oryong-Dong Puk-Gu, Kwangju, 500-712, Korea
hoyo@kjist.ac.kr

Abstract. This paper describes a new feature-based vehicle tracking system using trajectory matching, which extracts corner features of the vehicle and tracks the features using linear Kalman filtering, where features from the same vehicle are grouped together. We also propose a new grouping algorithm using trajectory matching to make our tracking system robust enough for segmenting different vehicles in the congested traffic situation. The proposed system has demonstrated good performance for crossway traffic video sequences.

1 Introduction

The red-light camera is popularly applied for traffic surveillance. It helps communities enforce traffic laws by automatically photographing vehicles whose drivers run red lights or do lane violation. A red-light camera system operates with a video-based vehicle tracking system to decide red-light or lane violation by its tracking trajectory.

In traffic surveillance applications, a video-based vehicle tracking system detects and tracks an individual vehicle that is moving through the camera scene. This system can provide traffic flows, such as normal traveling of vehicles, vehicle traveling in the wrong direction, and stopped vehicles.

Various tracking systems have been developed for detecting moving vehicles and tracing their locations based on the linear predictor model [1], [2], [3], [4]. One typical approach of video-based tracking is the feature-based tracking system where sub-features, such as distinguishable points or lines of the object, are traced [4]. The main advantage of this approach is that even in the presence of partial occlusion, some sub-features of the moving object remain visible. Therefore, this approach is appropriate for congested traffic in the crossway. Since a vehicle could have multiple sub-features, we have to group a set of features belongs to the same object.

Grouping of sub-features of the vehicles is based on common motion constraints. Previous grouping algorithms utilize only the spatial information that link sub-features together within a limited range [4]. In order to make the grouping robust enough for segmenting different moving vehicles, they keep track of relative distances of all feature pairs. If a feature pair has large variation of the relative distance, the grouping of the pair is broken. However, when the vehicle is turning left or right, the shape of the vehicle can be changed and tracking positions of the features become incorrect. Therefore, the conventional grouping rule does not guarantee stable vehicle tracking. In addition, the previous approaches are not appropriate for real-time object tracking due to their computational complexity for grouping.

In this paper, we propose a new feature-based tracking system where we design a robust grouping algorithm using trajectory matching., Fig. 1 shows key functional blocks of the proposed tracking system.

2 Feature-based Tracking System

As shown in Fig. 1, the proposed vehicle tracking system consists of three functional parts: feature detection, feature tracking and feature grouping.

Fig. 1. Feature-based Tracking System

After we select features based on the measure of cornerness of the moving vehicle, we trace the detected features using linear Kalman filtering, which requires a modest amount of computation. In the grouping part, we group sub-features together that come from the same vehicle, which helps us to extract the vehicle trajectory.

2.1 Feature Detection

Corner points that can be traced reliably are chosen as sub-features of the vehicle. For corner point detection, we take gradient operations along the x and y directions over the 9x9 window, and compute the second moment matrix Z by taking average of the gradient values [5].

$$Z = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix} \tag{1}$$

where g_x and g_y are the average gradient values along the x and y directions, respectively. If the matrix Z has two large eigenvalues, the original window contains a corner feature of high spatial frequency. Therefore, we can declare the corner point if $\min(\lambda_1,\lambda_2) > \lambda_c$, where λ_1 and λ_2 are two eigenvalues of the matrix Z and λ_c is a predefined threshold value.

2.2 Feature Tracking

Once a corner point is detected, we can track the feature efficiently by predicting the next coordinate from the observed coordinate of the feature point. We design a 2D token-based tracking scheme using Kalman filtering [1], [6]. The center position of the feature is used as the token $t(k)$. We assume the next token $t(k+1)$ is a sum of the current token $t(k)$ and the token change $\Delta t(k)$. We can define a simplified polynomial motion model as follows:

$$t(k+1) = t(k) + \Delta t(k) \tag{2}$$

Kalman filtering provides a sequential and recursive algorithm for optimal linear minimum variance (LMV) estimation of the system state $x(k)$. We define the state variable $x(k)$ as a two-dimensional vector, which represents the positional change of the token $\Delta t(k)$.

$$x(k) = \begin{bmatrix} \Delta x\_center(k) \\ \Delta y\_center(k) \end{bmatrix} \tag{3}$$

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

$$x(k+1) = \Phi(k,k+1)x(k) + w(k) \tag{4}$$

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

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, we can write the state transition matrix by Eq. (5).

We can also assume a linear relationship between the system state and a set of measurements as follows:

$$z(k) = H(k)x(k) + v(k) \tag{6}$$

$$H(k) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{7}$$

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

After we define a system model and a measurement model, we apply a recursive Kalman filtering algorithm to obtain LMV estimates of motion parameters. The recursive Kalman filtering algorithm consists of three steps of operations: initialization, state prediction, and measurement update.

In the initialization step, we determine the initial state estimate that are derived by discrete time derivatives of the feature center locations in the first two frames and determine the initial error covariance matrix which represents deviation of the initial state estimate from the actual initial state.

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 to refine the LMV estimate and its error covariance matrix for the current state. We perform this correction process based on a set of measurement errors using normalized correlation. A small 9x9 gray-level template is extracted and used for calculating normalized correlation.

At each time frame, we use the Kalman filter to predict the search region for each corner point. The template that was extracted when the corner point was originally detected, is correlated in the search region. After we locate the correlation peak, we can update the system state and the error variance using Kalman filtering.

2.3 Feature Grouping using Trajectory Matching

The purpose of feature grouping is to cluster sub-features together that come from the same vehicle. Corner features that move together are linked into a single vehicle. Since there are many vehicles in traffic scenes, it is difficult to group subfeatures. In order to resolve this segmentation problem, we develop trajectory approximation and trajectory matching algorithms.

For trajectory approximation, after we detect moving features in the traffic video sequence, we extract feature trajectories. A trajectory is aligned by the trail of the centroid of the feature in successive picture frames. Therefore, the feature trajectories that come from the same vehicle have similar shapes. We can approximate the x and y positions of the centroids over the frame time t by

$$\begin{aligned} x(t) &= a_{x0} + a_{x1}t + a_{x2}t^2 + \cdots + a_{xn}t^n \\ y(t) &= a_{y0} + a_{y1}t + a_{y2}t^2 + \cdots + a_{yn}t^n \end{aligned} \tag{8}$$

where $x(t)$ and $y(t)$ represent the x and y positions of the centroids, respectively, n is the approximation order, and a_{xk} (k=0,...,n) and a_{yk} (k=0,...,n) are approximated coefficients of $x(t)$ and $y(t)$, respectively.

If we approximate m known centroid points by a polynomial of order 3, the unknown coefficients, a_{x0}, a_{x1}, a_{x2}, and a_{x3}, can be found by least squares curve fitting, which minimizes the sum of the squares of the deviations of the data from the model. Clearly, this fitting does not make an accurate approximation of the data. However, the fitted curve represents a rough shape of the moving trajectory. This means that we can describe the moving trajectory only by a few polynomial coefficients.

Once all the feature points within a limited area are grouped into a single vehicle, we exclude some points that have different shapes of trajectories from the group. In this paper, we design the following trajectory matching rule to discriminate dissimilar trajectories from the group.

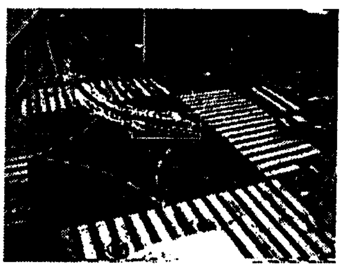
$$SM_x([P_x],[P'_x]) = w_0(a_{x0} - a'_{x0})^2 + w_1(a_{x1} - a'_{x1})^2 + w_2(a_{x2} - a'_{x2})^2 + w_3(a_{x3} - a'_{x3})^2 \quad (9)$$

$$SM_y([P_y],[P'_y]) = w_0(a_{y0} - a'_{y0})^2 + w_1(a_{y1} - a'_{y1})^2 + w_2(a_{y2} - a'_{y2})^2 + w_3(a_{y3} - a'_{y3})^2 \quad (10)$$

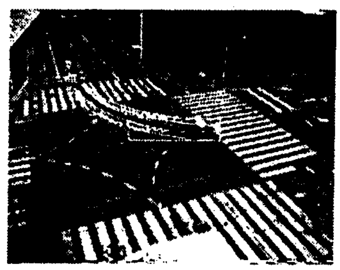
where SM_x and SM_y are defined as the x and y directional similarity measures (SM), respectively, $[P_x]$, $[P'_x]$, $[P_y]$ and $[P'_y]$ denote the coefficient value sets for the approximated trajectories to be compared, and w_k(k=0,1,2,3) are weight factors. If both SM_x and SM_y are lower than a predefined threshold value, we combine two feature points into the same group.

3 Simulation Results

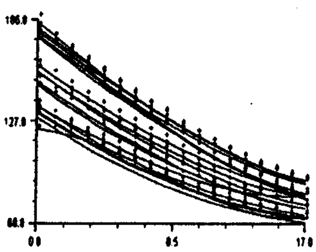
We have performed computer simulations on several crossway traffic sequences and estimated the feature distance in the world coordinate system. We employ the least mean squares (LMS) method to find the translation vector and the projection matrix using 16 calibration points. In order to evaluate the performance of the proposed grouping algorithm, we compare tracking trajectories with other results.



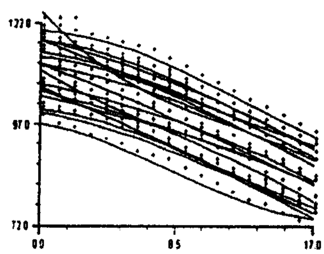
(a) Frame time #100



(b) Frame time #107



(c) x directional curve

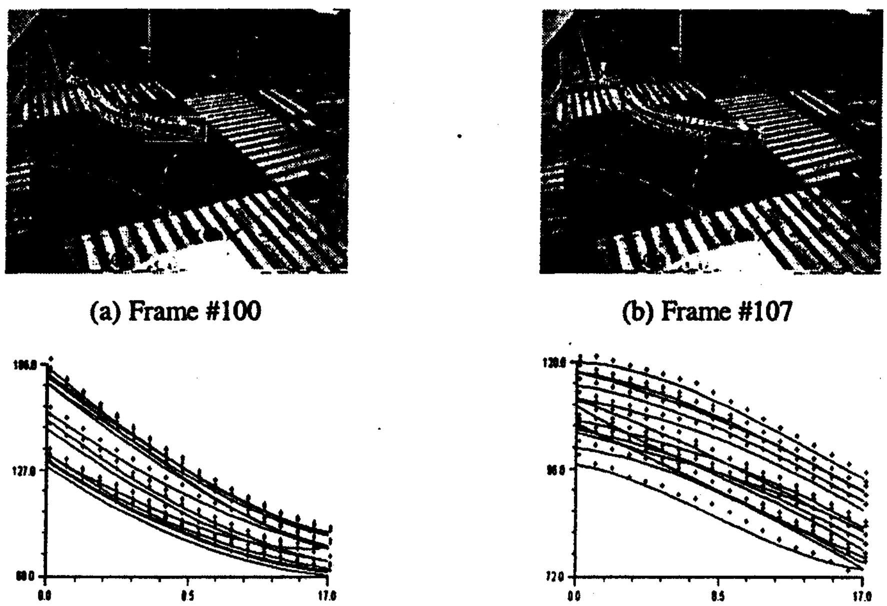


(d) y directional curve

Fig. 2. Tracking results of the grouping using only spatial information

As shown in Fig. 2(a), when we apply the grouping using only spatial information to the scene of a left-turning vehicle, one feature trajectory at the left lower region of the vehicle is deviated due to correlation mismatching between two adjacent frames. The feature point has been linked to other points within the vehicle until distances from this point to all the other features are larger than a given threshold value, which makes the grouping incorrect as shown in Fig. 2(b). Fig. 2(c) and Fig. 2(d) display approximated curves of the feature trajectories in the x and y directions, respectively, at frame time #100.

Fig. 3 demonstrates the tracking result by the grouping method using both the spatial information and trajectory matching for the same scene as in Fig. 2. As shown in Fig. 3(a), the feature point of a different shape of the trajectory is separated from the feature group of the left-turning vehicle since the similarity measure is smaller than a given threshold value. The result in Fig. 3(b) shows that the grouping is kept correctly. Fig 3(c) and Fig 3(d) display approximated curves of the feature trajectories in the x and y directions, respectively, at frame time #100.



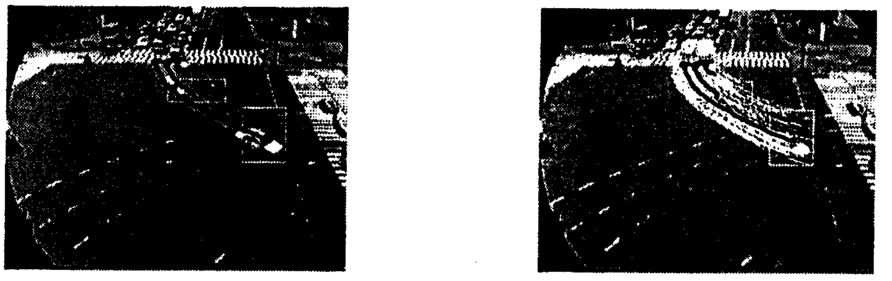
(a) Frame #100 (b) Frame #107

(c) x directional curve (d) y directional curve

Fig. 3. Tracking results of the grouping using spatial information and trajectory matching

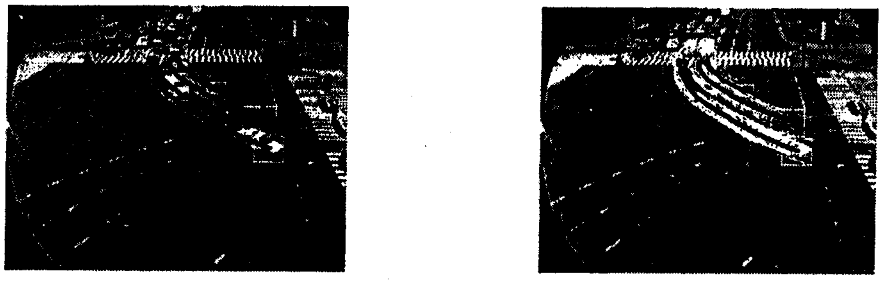
In Fig. 4, we apply the grouping method using only spatial information to a traffic scene with partial occlusion and neighboring condition. There are two over-grouping errors in Fig. 4(a). Since two vehicles appear close to each other, it is difficult to separate two vehicles by the grouping method using only spatial information. If one vehicle passes by another vehicle, a visual occlusion can occur. Fig. 4(b) is the result including feature trajectory drawings.

Fig. 5 shows the tracking result by the grouping method using both the spatial information and trajectory matching for the same scene as in Fig. 4. When two vehicles appear close to each other, the trajectory matching algorithm can separate them easily at the initial time. For the partial occlusion, two vehicles are being merged during the initial period. However, as the two vehicles move down the road, they can be separated when they exhibit distinguishing motions.



(a) (b)

Fig. 4. Tracking results of traffic scene with partial occlusion and neighboring condition by the grouping using only spatial information



(a) (b)

Fig. 5. Tracking results of traffic scene with partial occlusion and neighboring condition by the grouping using spatial information and trajectory matching

Fig. 6. Tracking results of night traffic scene

Fig. 6 shows the tracking result of night traffic scene by using spatial information and trajectory matching. Especially, in case of night traffic, many noise trajectories are occurred by headlight. However, the noise trajectories are removed by trajectory matching, and our tracking algorithm tracks the vehicles successfully.

4 Conclusions

In this paper, we have proposed a feature-based vehicle tracking system with a new grouping scheme. We extract image feature points by a corner detection algorithm and trace the features using two-dimensional token-based Kalman filtering. The new grouping algorithm using trajectory matching makes the proposed tracking system robust even in the partial occlusion and neighboring conditions. We can extend the proposed system for more congested traffic scenes.

Acknowledgement

This work was supported in part by the Korea Science and Engineering Foundation (KOSEF) through the Ultra-Fast Fiber-Optic Networks (UFON) Research Center at Kwangju Institute of Science and Technology (K-JIST), and in part by Ministry of Education (MOE) through the Brain Korea 21 (BK21) project.

References

1. Jung, Y.K., Ho, Y.S.: Robust Vehicle Detection and Tracking for Traffic Surveillance. Picture Coding Symposium'99, (1999) 227-230
2. McFalane, N., Scholfield, C.: Segmentation and Tracking of Piglets in Images. Machine Vision and Application. Vol. 8 (1995) 187-193
3. Malik, J., Russell, S.: A Machine Vision Based Surveillance System for California Roads. PATH Project MOU- 83 Final Report, University of California, Berkeley (1994)
4. Beymer, D., McLauchlan, P., Malick, J.: A real-time computer vision system for measuring traffic parameters. Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 12 (1997) 495-501
5. Forstner, W., Gulch, E.: A Fast Operator for Detection and Precise Location of Distinct Points, Corners, and Centers of Circular of Features. Proc. of the Intercommission Conf. On Fast Processing of Photogrammetric Data, (1987) 281-305
6. Rao, B.S.Y., Durrant-Whyte, H.F., Sheen, J.A.: A Fully Decentralized Multi-Sensor System For Tracking and Surveillance. The International Journal of Robotics Research, Vol. 12 (1993) 20-44