# Feature-Based Object Tracking with an Active Camera

Young-Kee Jung[1], Kyu-Won Lee[2], and Yo-Sung Ho[3]

[1] Honam University
59-1 Seobong-dong Kwangsan-gu, Kwangju, 506-090, Korea
ykjung@honam.ac.kr
[2] Daejeon University
96-3 YongUn-dong Dong-gu, Daejeon, 300-716, Korea
kwlee@dju.ac.kr
[3] 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 tracking system that can track moving objects with a pan-tilt camera. After eliminating the global motion of the camera movement, the proposed tracking system traces multiple corner features in the scene and segments foreground objects by clustering the motion trajectories of the corner features. We propose an efficient algorithm for clustering the motion trajectories. Key attributes for classifying the global and local motions are positions, average moving directions, and average moving magnitude of each corner feature. We command the pan-tilt controller to position the moving object at the center of the camera. The proposed tracking system has demonstrated good performance for several test video sequences.

## 1 Introduction

Owing to rapid progress of the computer technology and its applications, computer vision systems are partly replacing our role. In practice, machine vision systems that are composed of computer vision and various kinds of machinery are ripe enough to be used in the industrial field and in our daily life. A popular example of them is the automated surveillance system that watches moving objects in the restricted area or that monitors the traffic condition for the intelligent transportations system. In those applications, object segmentation and object tracking play quite important roles.

Object motion has long been considered as a significant source of information in the natural vision system. Understanding the visual motion is necessary for both distinguishing sources of different motions and identifying moving objects relative to the surrounding environments. Object motion can be recognized by Johansson's moving light display (MLD) [1]. We can use MLD to find trajectories of a few specific points corresponding to connecting joints of the moving object, and can use them as a key to recognition of the object activity. Gould and Shah build a trajectory primal sketch that represents significant changes in motion in order to identify objects using the trajecto-

Y.-C. Chen, L.-W. Chang, and C.-T. Hsu (Eds.): PCM 2002, LNCS 2532, pp. 1137-1144, 2002.
© Springer-Verlag Berlin Heidelberg 2002

ries of a few representative points [2]. In particular, human motion has been studied extensively using model-based approaches [3].

Several motion-based tracking algorithms have been developed with motion energy in the scene. Those can be implemented with a low complexity. However, they are sensitive to noise and difficult to cope with the global motion caused by the camera movement [4].

In this paper, we are concerning with feature-based object tracking in the mobile camera environments [5-6]. We propose a new algorithm for clustering motion trajectories based on corner features. With a video camera mounted on a pan-tilt controller, we can detect motion of the moving object and then command the pan-tilt controller to follow the object such that the object is positioned at the center of the view field.

## 2 Proposed Tracking Algorithm

As shown in Fig. 1, the proposed tracking system consists of four main functional parts: camera motion estimation, feature detection and tracking, clustering motion trajectories, and control of the pan-tilt camera.

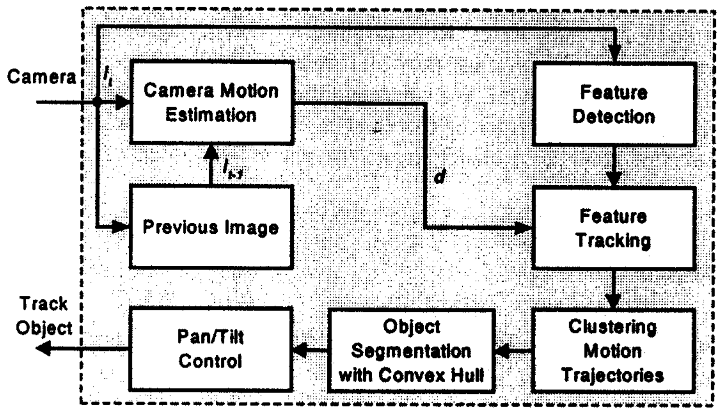



**Fig. 1.** Proposed Object Tracking Algorithm

In the first part of camera motion estimation, we compute the global motion caused by camera movement by finding the maximal matching position between two consecutive frames using a template-matching algorithm. We have taken a two-level pyramidal approach to reduce the computation cost.

After eliminating the global motion by subtracting the camera movement $\boldsymbol{d}$ from the current feature position, we employ the Kalman filter to predict the search region for each corner point. The 7x7 template, that was extracted when the corner point was detected in the previous frame, is correlated in the search region. After we locate the correlation peak, the feature template of correlation is updated.

We cluster the feature trajectories by grouping the attributes of the feature trajectories that hold similar characteristics. Positions, average moving angles and average moving magnitude of the corner points are used as key attributes for classifying the global and local motions, and regions of moving objects are segmented by forming convex hulls with the classified feature points.

Finally, we command the pan-tilt controller to follow the object such that the object will always lie at the center of the camera.

## 2.1 Camera Motion Estimation

In order to simplify the analysis of the scene from the mobile camera, we assume that the mobile camera keeps only the translation motion. With this assumption, the camera motion $d(\Delta x, \Delta y)$ is computed by finding the best matching position between the current image $I_t(=I_c)$ and the previous one $I_{t-1}(=I_p)$ using a template-matching algorithm, as shown in Fig. 2.

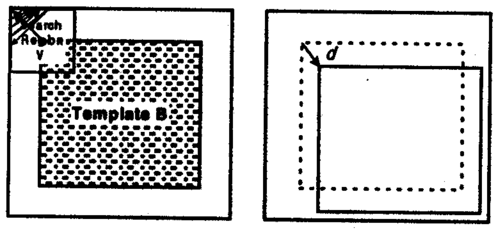



**Fig. 2.** Camera Motion Estimation

For real-time object tracking, we take a two-level coarse-to-fine pyramidal approach. At the top level, the camera displacement $d_s$ is computed for the pair of 1/3 subsampled images, which is used as the base registration for the next level. For a pair of subsampled images $(I_{cs}, I_{ps})$, template matching is performed at every pixel location within the search region $V_s$ as follows:

$$\min_{(sx,sy)\in V_s} \sum_{m,n\in B_s} \left| I_{cs}(m+sy, n+sx) - I_{ps}(m,n) \right| \tag{1}$$

where $V_s$ is the search region and $B_s$ is the region for comparison at the top level.

At the bottom level, the camera displacement $d$ is computed for the pair of the original images $(I_c, I_p)$.

$$\min_{(x,y)\in V} \sum_{m,n\in B} \left| I_c(m+y, n+x) - I_p(m,n) \right| \tag{2}$$

$$V = \{(x,y) \mid sx-1 \le x \le sx+1,\ sy-1 \le y \le sy+1\} \tag{3}$$

where the search region $V$ is determined by the matching result $(sx, sy)$ at the top level and the comparison region $B$ has nine times of the size of $B_s$.

## 2.2 Feature Detection

Since the corner feature is viewpoint invariant and naturally leads to the representation of the object shape, corner points are used as the features in the scene. 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][6].

$$Z = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix} \tag{4}$$

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 $\lambda_1$ and $\lambda_2$ are two eigenvalues of the matrix $Z$ and $\lambda_c$ is a predefined threshold value.

## 2.3 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 [7][8][9]. 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 by

$$t(k+1) = t(k) + \Delta t(k) \tag{5}$$

We know that 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 $\Delta t(k)$.

$$x(k) = \begin{bmatrix} \Delta x\_center(k) \\ \Delta y\_center(k) \end{bmatrix} \tag{6}$$

Once we define the system model and the measurement model, we apply the recursive Kalman filtering algorithm to obtain LMV estimates of motion parameters [7]. The recursive Kalman filtering algorithm consists of three steps of operations.

At the initialization step, we determine the initial state estimate that is derived from the discrete time derivatives of the feature center locations in the first two frames. We also determine the initial error covariance matrix that represents the 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 the normalized correlation. 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.

### 2.4 Clustering Motion Trajectories

There are two types of possible motions from the scene of the mobile camera. One is the global motion of the background occurred by camera movement, and the other is the local motion caused by moving objects.

Since we have selected corner points as image features, we can easily obtain a representation of the object shape and other aspects of the background movement. In addition, we can separate two heterogeneous motions by grouping attributes of the corner points according to their spatial and temporal displacements.

The key attributes for classifying the global and local motions are position $C(C_x,C_y)$, average moving direction $A_a$, and average moving magnitude $M_a$ of the corner points. Each attribute of the feature is computed by the following equations.

$$M_a = \frac{1}{N}\sum_{i=1}^{N} M_i,\ M_i = \sqrt{(C_x(i) - C_x(i-1))^2 + (C_y(i) - C_y(i-1))^2} \tag{7}$$

where $i$ is the time segment, $M_i$ is the moving distance of the corner point at time $i$, $C_x$ and $C_y$ are the horizontal and vertical positions of the corner point at current image, respectively, and $N$ is the trajectory length.

$$A_a = \frac{1}{N}\sum_{i=1}^{N} A_i,\ A_i = \arctan\frac{C_y(i) - C_y(i-1)}{C_x(i) - C_x(i-1)} \tag{8}$$

where $A_i$ is the moving direction of corner point at time $i$.

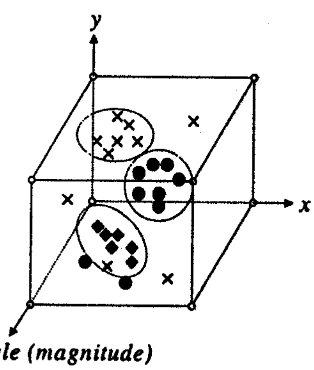



Fig. 3. Clustering in Multi-dimensional Feature Space

As shown in Fig. 3, the attributes are arranged in the three-dimensional feature space. We cluster the corner points by grouping the attributes of similar characteristics. After the dynamic range of each attribute is normalized, we cluster the attributes by the K-means algorithm that is extended to three parameters. The full set $U$ of the corner points $q$ is given by

$$U=\{q_0, q_1, q_2, \cdots, q_n\} \tag{9}$$

We compute the first-order moment from the elements of $U$ and denote it as the initial center $\vec{m}_0$. If the standard deviation $\sigma_0$ obtained from $U$ and $\vec{m}_0$ is greater than the predetermined threshold, a new center vector of a cluster $\vec{m}_1$ is determined by

$$\vec{m}_1=\vec{m}_0+\alpha\sigma_0, \ \alpha: constant \tag{10}$$

The cluster points are reassigned based on the Euclidean distances, $d(\vec{m}_0, q_k)$ and $d(\vec{m}_1, q_k)$, from $\vec{m}_0$ and $\vec{m}_1$. The criterion for reassignment of the cluster points is described by

$$\begin{aligned} C_0 &= \{q_k : d(\vec{m}_0, q_k) \geq, d(\vec{m}_1, q_k)\} \\ C_1 &= \{q_k : d(\vec{m}_0, q_k) <, d(\vec{m}_1, q_k)\} \ k = 1, 2, 3, \cdots, n \end{aligned} \tag{11}$$

where $x_0$ and $x_1$ are numbers of elements in the cluster sets $C_0$ and $C_1$, respectively.

Consequently, the sets of elements of new clusters are defined by

$$\begin{aligned} C_0 &= \{q_{00}, q_{01}, q_{02}, \cdots, q_{0x_1}\}, \ 1 \leq x_0 < n \\ C_1 &= \{q_{10}, q_{11}, q_{12}, \cdots, q_{1x_1}\}, \ 1 \leq x_1 < n \ and \ x_0 + x_1 = n \end{aligned} \tag{12}$$

where $x_0$ and $x_1$ are numbers of elements in the cluster sets $C_0$ and $C_1$, respectively.

After finding the new first moments $\vec{m}_0'$ and $\vec{m}_1'$ with elements of the sets $C_0$ and $C_1$, we perform the reassignment process for the elements classified before by computing $d(\vec{m}_0', q_k)$ and $d(\vec{m}_1', q_k)$ for all elements of the set $U$. We repeat the process recursively until each standard deviation $\sigma_k$ is smaller than the specific threshold value. Eventually, the cluster set $C_k$ comprises all the corner points.

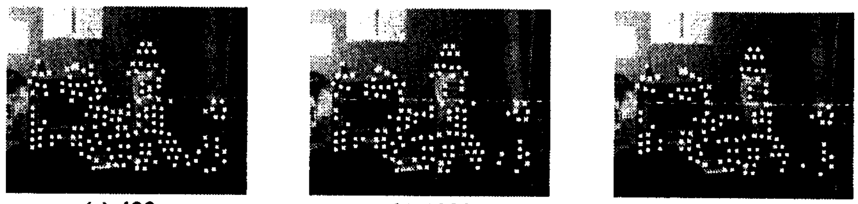
$$\begin{aligned} & C_k = \{q_{k0}, q_{k1}, q_{k2}, \cdots, q_{kx_k}\}, \\ & 1 \leq x_k < n \ and \ x_0 + x_1 + \cdots + x_k = n \end{aligned} \tag{13}$$

where $x_k$ is the number of elements in the $k$-th cluster.

## 3 Simulation Results

The proposed tracking system has been tested on several video sequences in indoor environments. The camera is mounted on the pan/tilt driver and the maximum rotation velocity of the camera is about 1.92 rad/sec.

Fig. 4 shows the feature detection results for three eigenvalue thresholds. A high eigenvalue threshold diminishes the number of the detected features. We use 1000 as the eigenvalue threshold $\lambda_c$ for the tracking system.



(a) 400 (b) 1000 (c) 1600

**Fig. 4.** Feature Detection for 3 Eigenvalue Thresholds

Three consecutive images of Fig. 5 are captured with two motions. The left panning of camera causes one motion. A moving person occurs the other motion. The camera motion $\boldsymbol{d}$ is computed by finding the maximal matching position using template matching. Subtracting the camera movement $\boldsymbol{d}$ from the current feature position eliminates the global motion. Fig. 5(a) displays the results before global motion compensation. The results after global motion compensation are represented in Fig. 5(b).

(a) before global motion compensation

(b) after global motion compensation

**Fig. 5.** Global Motion Compensation

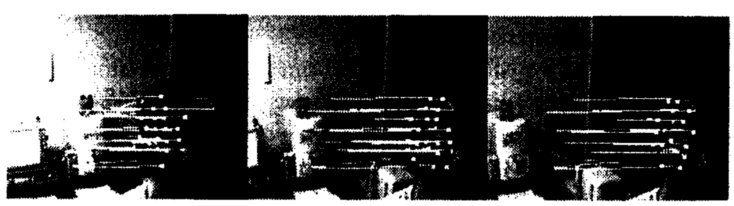


**Fig. 6.** Tracking Results for The Scene of Right Moving Person. Frames shown here are (top to bottom, left to right) numbers 202, 208, 214

Fig. 6 shows the tracking results for the scene of the person who moved to right direction. As shown in Fig. 6, a number of corners are selected as the active corners. It is seen that there are several feature paths corresponding to the person in the scene. According to the global motion by camera movement is eliminated, the result shows the only local motions of person. The pan-tilt is commanded to move the camera to the centroid of local motion.

## 4 Conclusions

In this paper, we have proposed an algorithm for moving object tracking with a mobile camera. We use a corner detector to extract features and trace the features using two-dimensional token-based Kalman filtering. Then, the foreground objects are segmented by clustering motion trajectories of the corner features. We have also proposed an efficient clustering algorithm using feature trajectory to obtain a stable local motion. In case of a single moving object, the proposed algorithm shows robust tracking results. In the future, we plan to improve our algorithm by applying active zooming and multiple objects tracking.

**Acknowledgement.** This work was supported in part by grant NO. R05-2002-000-00868-0 from the Basic Research Program of the Korea Science & Engineering Foundation. This work was also supported in part by KOSEF through UFON and in part by MOE through BK21.

## References

1. Johansson, G.: Visual Perception of Biological Motion and a Model for Its Analysis. Perception and Psychophysics, Vol.14 (1973) 201-211
2. Gould, K., Shah, M.: The Trajectory Primal Sketch: A Multi-Scale Scheme for Representing Motion Characteristics. IEEE Conf. of CVPR, (1989) 79-85
3. Rouke, O., Badler: Model-based Image Analysis of Human Motion using Constraint Propagation. IEEE Trans. on PAMI, Vol.3, No.4 (1980) 522-537
4. Lee, K.W., Kim,Y.H., Jeon, J., and Park, K.T.: An Algorithm of Moving Object Extraction Under Visual Tracking without Camera Calibration. Proceedings of ICEIC, (1995) 151-154
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. 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
7. Jung, Y.K., Ho, Y.S.: Robust Vehicle Detection and Tracking for Traffic Surveillance. Picture Coding Symposium, (1999) 227-230
8. 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
9. McFalane, N., Scholfield, C.: Segmentation and Tracking of Piglets in Images. Machine Vision and Application. Vol. 8 (1995) 187-193