MVA '96 IAPR Workshop on Machine Vision Applications, November. 12-14, 1996, Tokyo, Japan

Real-Time Tracking with Kalman Filter

Dae-Sik Jang, Hyung-Il Choi
School of Computing
Soongsil University

Gye-Young Kim
Electronics and Telecommunications Research Institute(ETRI), Korea

Abstract

This paper describes a real-time tracking system which detects an object entering into the field of view of camera and executes the tracking of the detected object by controlling a servo device so that a target object always lies at the center of an image frame. In order to detect and track a moving object, we basically apply a model matching strategy. We allow a model to vary dynamically during the tracking process so that it can assimilate the variations of shape and intensities of a target object. We also utilize Kalman filter so that a tracking history can be encoded into state parameters of Kalman filter. The estimated state parameters of Kalman filter will then be used to reduce search areas for model matching and to control a servo device.

1. Introduction

In this paper, we describe a real-time tracking system which detects an object entering into the field of view of a camera and executes tracking of the detected object by controlling a servo device in such a way that a target always lies at the center of an image frame. In order to detect and keep tracking of a moving object, we basically apply a model matching strategy (3-5). We allow the model of a target to vary dynamically during tracking process so that it can assimilate variations of shape and intensities of a target object. We also encode a tracking history into state parameters of a Kalman filter. The estimated state parameters will then be used to reduce search areas for model matching and to control a servo device.

Fig.1 shows an overall configuration of our tracking system. A *detecting module* detects the entrance of a moving object into a FOV(Field of View) by analyzing two consecutive input images. It also forms an initial model of a target object. A *tracking module* keeps tracking of a target object by controlling a servo device. A servo device needs motion information of a target by which it may adjust its pan and tilt angles. This information is obtained by model matching with the help of a Kalman filter. A model of a target is automatically updated during the matching process.

2. Detection of moving object

The detection process is basically performed by comparing two successive images captured by a fixed camera and identifying differing areas of one against another image(6,7). Then, a question is how adjacent two successive images must be. If a time interval between two successive images is too short compared to the movement of an object, these two images may not show clear differences and the detection process may fail to identify movement itself. On the other hand, if the time interval is too long, the detection process may fail to catch up an entering object. We solve this problem by employing temporal resampling. The next question is which areas of chosen images we have to compare in which way. A detection period should be short enough not to miss an entering object during the period. So, we examine only a small portion of an entire image by spatial resampling.

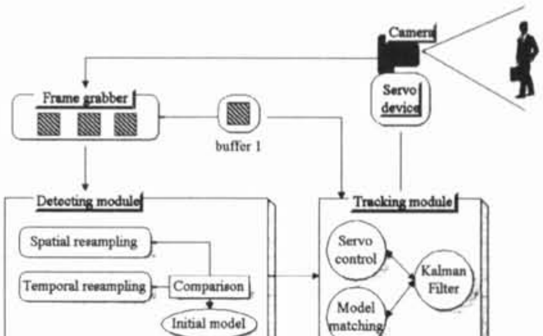


Fig1. Overall configuration

We first divide an entire image into small cells, and select some evenly distributed cells. And we compare the cells of two successive images to identify differing areas by a dissimilarity measure *DisSimil*(8). If the value of *DisSimil* is less than some predefined threshold TH_1, we call the corresponding cell as a stationary cell. Otherwise, the corresponding cell is called a nonstationary cell. When the number of nonstationary cells is greater than some predefined threshold TH_2, we presume that some object has entered into the FOV of a camera.

Once the detection of entering object is made, we generate an initial model. We define a model of a target object as a feature vector which represents spectral characteristics of the selected nonstationary area. The enclosing rectangle of the selected area has constituent cells. Some of them may be stationary, while all the others are nonstationary. We compute an average and standard deviation of gray values in each nonstationary cell. We then list up the calculated values to form a feature vector of $\boldsymbol{m(i,j)}$, where (i,j) denotes the index of a constituent cell. Stationary cells of the enclosing rectangle do not belong to a target object.

3. Estimation of motion parameters

In this section, we describe how to utilize a Kalman filter so that a tracking history is encoded into state parameters of a Kalman filter. The estimated parameters will then be used to reduce the searching scope for model matching and to control a servo device. A Kalman filter provides sequential and recursive algorithm for optimal LMV(Linear Minimum Variance of error) estimation for system states(1,2). We assume that a state model is linear and is defined by the following equation.

$$\boldsymbol{x}(t) = \Phi(\Delta t)\, \boldsymbol{x}(t - \Delta t) + \boldsymbol{w}(t - \Delta t) \quad (1)$$

where $\boldsymbol{x}(t)$ denotes system states at time instant of t, $\Phi(\Delta t)$ denotes a state transition matrix during a period of Δt, and $\boldsymbol{w}(t)$ denotes an estimation error. In this paper, we express system states as an 8-dimensional vector which represents the positional change of a target object per unit time interval and the size of a target object. (2) shows such system states and an estimation error, respectively.

We represent the positional change of a target object as displacement of the centroid of a target object per unit time interval, and the size as horizontal and vertical lengths of the enclosing rectangle of a target object. That is, Δx and Δy of (2) denote displacement of the centroid of a target object in the x-axis and y-axis respectively, and xs and ys of (2) denote horizontal and vertical length of the enclosing rectangle respectively. The notation of superbar denotes a derivative with respect to t. We also assume that the trajectory of a target object varies with a constant acceleration and the size of a target object varies linearly. We then have a state transition matrix as in (3).

$$x(t) = \begin{pmatrix} \Delta x(t) \\ \Delta y(t) \\ xs(t) \\ ys(t) \\ \Delta\overline{x}(t) \\ \Delta\overline{y}(t) \\ \overline{xs}(t) \\ \overline{ys}(t) \end{pmatrix} \quad w(t) = \begin{pmatrix} w_{\Delta x}(t) \\ w_{\Delta y}(t) \\ w_{xs}(t) \\ w_{ys}(t) \\ w_{\Delta\overline{x}}(t) \\ w_{\Delta\overline{y}}(t) \\ w_{\overline{xs}}(t) \\ w_{\overline{ys}}(t) \end{pmatrix} \quad (2)$$

$$\Phi(\Delta t) = \begin{pmatrix} 1 & 0 & 0 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad (3)$$

Kalman filter algorithm tries to estimate system states based on a set of measurements. We assume linear relation between system states and a set of measurements as in (4).

$$\boldsymbol{y}(t) = \boldsymbol{H}(t)\boldsymbol{x}(t) + \boldsymbol{v}(t) \quad (4)$$

where $\boldsymbol{y}(t)$ denotes a set of measurements, $\boldsymbol{H}(t)$ denotes an observation matrix, and $\boldsymbol{v}(t)$ denotes measurement errors. We measure the positional change and size of a target object at each time instant to get values of $\boldsymbol{y}(t)$. $\boldsymbol{y}(t)$ and $\boldsymbol{H}(t)$ are then formed as in (5).

$$\boldsymbol{y}(t) = \begin{pmatrix} \Delta x \\ \Delta y \\ xs \\ ys \end{pmatrix} \quad \boldsymbol{H}(t) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (5)$$

Once we have defined a system model and measurement model, we can now apply recursive Kalman filter algorithm to obtain LMV estimates of motion parameters. The recursive Kalman filter algorithm consists of three phases of operations; initialization, state estimation, and measurement update(2).

The phase of state estimation determines a priori LMV estimate and its error covariance matrix for current state based on previous state estimate and error covariance. (6) formalizes this phase.

$$\widetilde{\boldsymbol{x}}^{-}(t) = \Phi(\Delta t)\ \widetilde{\boldsymbol{x}}(t - \Delta t)$$

$$P^-(t) = \Phi(\Delta t)\, P(t-\Delta t)\, \Phi^T(\Delta t) + Q(t-\Delta t) \qquad (6)$$

where $\widetilde{x}^-(t)$ denotes a priori estimate for system states at the time instant of t based on measurements of $y(0), y(1), \ldots, y(t-\Delta t)$, and $\widetilde{x}^-(t-\Delta t)$ denotes optimal estimate for system states at the time instant of $t - \Delta t$ based on measurements of $y(0), y(1), \ldots, y(t-\Delta t)$. We will use the values of $\widetilde{x}^-(t)$ to reduce searching scope for model matching and determine appropriate angles of a servo device.

The phase of measurement update combines the estimated information with new measurements to provide LMV estimate and its error covariance matrix for current state. (7) formalizes this phase.

$$K(t) = P^-(t)\, H^T(t)\, (H(t)\ P^-(t)\, H^T(t) + R(t))^{-1}$$
$$P(t) = (I - K(t)\, H(t))\ P^-(t)$$
$$\widetilde{x}(t) = \widetilde{x}^-(t) + K(t)\,(y(t) - H(t)\ \widetilde{x}^-(t)) \qquad (7)$$

where the term $K(t)\,(y(t) - H(t)\ \widetilde{x}^-(t))$ provides optimal LMV estimate for $\widetilde{x}(t) - \widetilde{x}^-(t)$ based on $\widetilde{y}(t)$. That is, it represents optimal correction of the error incurred from the predicted estimate $\widetilde{x}^-(t)$ of $\widetilde{x}(t)$. We perform this correction process with measurements on the positional change and size of a target object which are to be computed through matching process.

4. Servo control and model matching

After an initial model has been generated, the system will go into tracking stage. The tracking stage can be divided into two types of substages; servo control and model matching. The former is to adjust the FOV of a camera in such a way that a target always lies at the center of an image frame, and the latter is to detect a target in the image captured by the adjusted camera.

Once we detected a target in the frame taken at $t - \Delta t$ instant, we apply Kalman filter algorithm and obtain state estimate $\widetilde{x}^-(t)$. The estimated values $\Delta x(t)$ and $\Delta y(t)$ of $\widetilde{x}^-(t)$ will then be converted to pan and tilt angles by which a servo device is to be rotated. As an illustration of such operation, let us consider Fig.2.

In Fig.2, (x_c, y_c) denotes the origin of image frame coordinates, (x_o, y_o) denotes center coordinates of a target in the frame taken at $t - \Delta t$ instant, and $\Delta x(t)$ and $\Delta y(t)$ denote the estimated amount of displacement which will occur during the period of Δt.

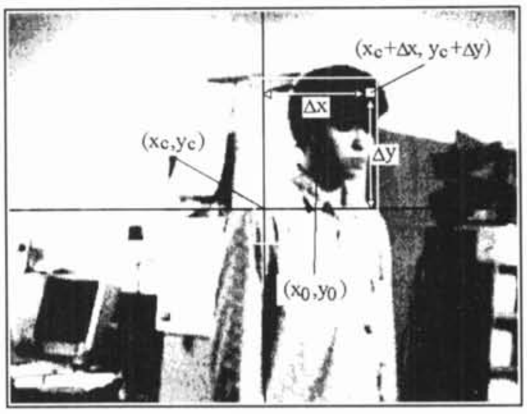


Fig.2. Example of $\Delta x(t)$ and $\Delta y(t)$ contained in $\widetilde{x}^-(t)$

The second substage of tracking process matches a current model against the input image captured by an adjusted camera and finds out a target object. This substage also updates the model for matching at the next frame, since the shape and size of a target may vary as the target moves on. Kalman filter algorithm provides estimated size of a target as well. We use the estimated size $xs(t)$ and $ys(t)$ to build a window within the input image. A target is to be searched for within the window. We set the width and height of the window one and half the values of the estimated $xs(t)$ and $ys(t)$.

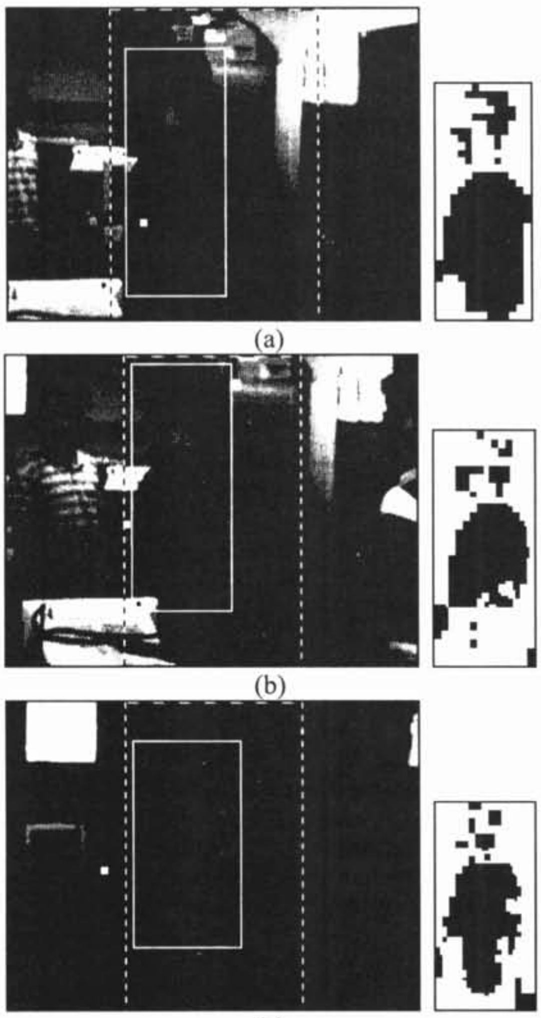
We perform model matching across the window in two stages; coarse matching and fine matching. The stage of coarse matching speeds up matching process by examining only the small portion of the window. The stage of fine matching identifies a target object and it also updates the feature vector of a model.

5. Experimental results and conclusions

We evaluated the suggested system under realistic tracking conditions. This section presents some experimental results which illustrate the operational characteristics of the proposed system. We implemented the suggested system with IBM-PC equipped with DT-2867 Frame grabber. Input images are captured with a CCTV camera mounted on PAN-TILT unit which can change the FOV of a camera by 6° per second. The captured images are 640 by 480 in size with 8 bits per pixel.

Fig.3 shows captured images during the course of tracking process, together with overlaid templates over which new models were formed. In Fig.3, the dotted white rectangle denotes the window predicted by a

Kalman filter, over which a target is to be searched for. The undotted white rectangle denotes the enclosing rectangle of a target object extracted through model matching. The white mark denotes the position to which the center of a frame needs to be adjusted at the next time instant. This position was also obtained by the prediction of a Kalman filter. We can notice that the system positions a target object around the center of the frame, even though other objects move across the target object. We can also confirm that the shape and size of the template, over which a new model is to be formed, varies as an object moves on.


(a)

(b)

(c)

Fig.3. Example of tracking process

In our implementation environment, the total processing time was around 0.09 second which is a little bit less than one period of processing cycle of Δt_{init}. One half of the processing time was spent in matching a model. We confirmed through this experiment that our system can keep tracking of a target which moves not faster than 0.96 *m/sec* and not closer than distance of 10 meters from a camera. We also confirmed that the predicted values of a Kalman filter are very useful in reducing searching scope for model matching and in controlling a servo device.

REFERENCES

[1] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, Englewood Cliffs, New Jersey (1986).

[2] G. Minkler and J. Minkler, *Theory and Application of Kalman Filtering*, Magellan (1994).

[3] Silvana Dellepiane, Giovanni Venturi and Gianni Vernazza, "Model generation and model matching of real images by a fuzzy approach", *Pattern Recognition* 25, pp. 115-137 (1992).

[4] Koichiro Akita, "Image sequence analysis of real world human motion", *Pattern Recognition* 17, pp. 73-83 (1984)

[5] Ted J. Broida and Rama Chellappa, "Estimating the kinematics and structure of a rigid object from a sequence of monocular images", *IEEE Transaction on Pattern Analysis and Machine Intelligence* 13, pp. 497-513 (1991).

[6] T. Skordas, S. de Paoli, E. Degremont and A. Chehikian, "Motion detection using a Multiscale Image Representation Strategy", *International conference on Intelligent Robots and Systems*, Yokohama, Japan, pp. 261-266 (1993).

[7] R. Jain, D. Militer, and H. H. Nagel, "Separating non-stationary from stationary scene components in a sequence of real world TV-images," *Proc. 5th Int. Joint Conf. Artificial Intelligence*, pp. 612-618 (1977).

[8] Ramesh Jain, "Extraction of motion information from peripheral processes", *IEEE Transaction on Pattern Analysis and Machine Intelligence* PAMI-3, pp. 489-503 (1981).