IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 5, July 2014.
www.ijiset.com
ISSN 2348 – 7968

A Critical Appraisal on Engineering Kalman-Filters for Real-Time Object Tracking & Motion Detection Systems

Sanchia Harsha
B.E. (ECE)
Anna University, Chennai
Email Id: sanchia2592@gmail.com

C.V. Sarath Kumar
B.E (ECE)
Anna University, Chennai
Email Id: sarath242@gmail.com

Abstract– *Object tracking in video sequences has long been a challenging area in the field of computer vision and has many real world applications like surveillance system, robotics, missile defense system, public security system and visual information processing. A lot of research has been undergoing ranging from applications to noble algorithms. However, most works are focused on a specific application, such as tracking human, car, or pre-learned objects. The work described in this paper is focused on tracking a randomly moving object chosen by a user using Kalman filter. For simplicity, a video sequence with just one object within has been selected. The Kalman filter predicts the most probable location of a detected object in the subsequent video frame and tracks an object by assuming the initial state and noise covariance. After sufficient information about the objects is accumulated, we can exploit the learning to successfully track objects. Extended the same concept for tracking the virtual objects such as stock prices which vary randomly and non-linear in nature. Extended Kalman Filter (EKF) is used to track or predict such objects movement, by modeling its non-linear process. The Kalman filter also helps to smooth out the irregularities due to the measurement error. An experimental result from different moving object video samples shows a very good result. This filter is intended to be robust without being programmed with any environment specific rules.*

Keywords– *Kalman filter; Object detection; Background subtraction; Object tracking*

1. INTRODUCTION

Real time object tracking is arguably one of the most flexible and adaptable additions to our ever-omnipotent technological advancement. Object tracking is an important task within the field of computer vision. ***Tracking is the problem of generating an inference about the motion of an object given a sequence of images.*** The proliferation of high-powered computers, the availability of high quality and inexpensive video cameras, and the increasing need for automated video analysis has generated a great deal of interest in object tracking algorithms. It can also be used in combining information and parameters of historical and empirical analysis in predicting the change in stock prices. The most famous early use of the Kalman filter was in the Apollo navigation computer that took Neil Armstrong to the moon, and (most importantly) brought him back. There are three key steps in video analysis: detection of interesting moving objects, tracking of such objects from frame to frame, and analysis of object tracks to recognize their behavior. Good solutions to this problem can be applied to many applications.

Therefore, the use of object tracking is pertinent in the tasks of:

- Predicting the stock prices in the stock market which fully random and non-linear.
- Motion-based recognition, that is, human identification based on gait, automatic object detection, etc.
- Automated surveillance, that is, monitoring a scene to detect suspicious activities or unlikely events
- Human-computer interaction, that is, gesture recognition, eye gaze tracking for data input to computers, etc.
- Traffic monitoring, that is, real-time gathering of traffic statistics to direct traffic flow.
- Vehicle navigation, that is, video-based path planning and obstacle avoidance capabilities.
- Video indexing, that is, automatic annotation and retrieval of the videos in multimedia databases.

1.1. BACKGROUND THAT MOTIVATED US:

There are so many algorithms used for tracking purpose such as mean shift algorithm, CAMShift algorithm, Optical flow, SURF etc. Each algorithm has strength in certain environment and weakness in others. A summary of such algorithm is presented here.

1) **MeanShift Algorithm:** MeanShift algorithm is an efficient pattern-matching algorithm with no parameter estimation, and can be combined with other algorithms. It uses the kernel function histogram model of the target

object. MeanShift is not sensitive for part of block rotation and deformation, but the track is easy to lapse when blocked a large area on the target.

2) **CAMShift Algorithm:** The Continuously Adaptive Mean Shift algorithm (CAMShift) is a lightweight object-tracking algorithm based on a one-dimensional hue histogram. Originally designed for tracking faces or flesh tone, the algorithm computes the probability that any pixel is part of the tracked object as opposed to the background. The result of camshaft is not so good if the foreground is same colour as background or background's colour changes rapidly.

3) **SURF:** Speeded Up Robust Feature (SURF) is a feature detector and descriptor. It is based on sums of approximated 2D Harr wavelet responses and makes an efficient use of integral images. The result of SURF is good even if the colour of background and foreground is same or background object's colour changes rapidly but the most crucial drawback is the computational time is high so it is incapable of tracking real-time object.

4) **Optical Flow:** Optical Flow tracking techniques allow the distinction between multiple objects and the background in a scene. This is computed based on the relative motion between an observer and the scene.
A comparison chart of different methods has been given below:

Algorithm	Dim light	bright light	lighting changes	moving object	stationary object	dropped frames	low resolution	similar background shapes	similar background colors
CamShift	bad	good	o.k.	good	good	o.k.	good	good	o.k.
SURF	good	good	good	N/A	good	good	bad	o.k.	good
Optical Flow	o.k.	o.k.	o.k.	good	bad	o.k.	bad	good	good

Table 1.1

It is observed that most of the algorithm dependent on application environment and very less immune to noise. A good filtering algorithm can eliminate noise from the data and retain useful information. Kalman Filter, An optimal recursive data processing algorithm has been taken for this tracking problem. The Kalman filter not only works well in practice, but it is theoretically attractive because it can be shown that of all possible filters, it is the one that minimizes the variance of the estimation error.

2. DISCRETE KALMAN FILTER THEORY

It is a recursive predictive filter that is based on the use of state space technique and recursive algorithm. It is called recursive because it does not require to store all the previous measurement and process the data optimally. Since the time of its introduction, the Kalman filter has been the subject of extensive research and application, particularly in the area of autonomous or assisted navigation.

The main advantages of kalman filter innovations are:

(i) The variance of the Kalman filter innovations is smaller than the variance of the deterministic innovations.
(ii) The computation time of Kalman filter is less.

Generally it is two-step process:

1) Prediction
2) Correction.

First the state is predicted with the dynamic model then it is corrected with the observation model so that the error co-variance is minimized. The process is repeated with the each time with the step as the previous step as initial value.

2.1. DISCRETE KALMAN FILTER ALGORITHM

The Kalman filter estimates the process step as with the dynamic model and then take feedback in form of noisy measurements and update the estimates with the measurements. The equations for the Kalman filter fall into two groups: time update equations and measurement update equations. The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the a priori estimates for the next time step. The measurement update equations are responsible for the feedback—i.e. for incorporating a new measurement into the a priori estimate to obtain an improved a posteriori estimate. The time update equation is called predictor equation and the measurement update equation is called corrector equation.

The complete algorithm is shown in Figure 1.

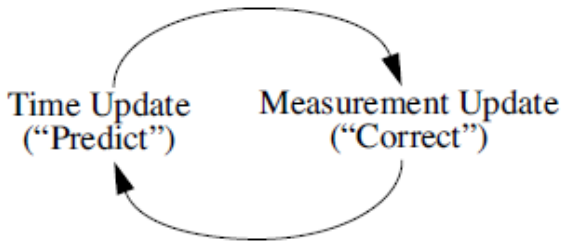


Figure 1. The complete Kalman filter algorithm

The time update equations are given as:

ISSN 2348 – 7968

$$X_k^{\hat{}-} = A\,X_{k-1}^{\hat{}} + BU_k \text{ ----------(1)}$$
$$P_k^- = A\,P_{k-1}A^T + Q \text{ ----------(2)}$$

Where,
'A' is state matrix
'B' coverts control input
'Q' is process noise covariance.

The measurement update equations are given as:

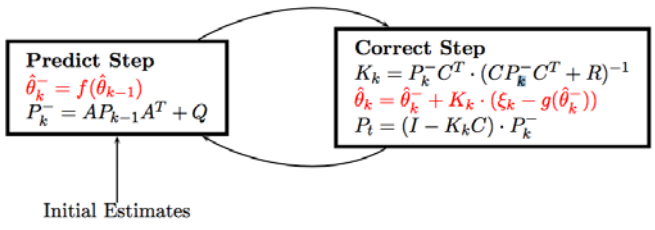
$$K_k = P_k^-\,H^T\,(HP_k^-\,H^T + R)^{-1} \text{ ----------(3)}$$
$$X_k^{\hat{}} = X_k^{\hat{}-} + K_k\,(Z_k - H\,X_k^{\hat{}-}) \text{ ----------(4)}$$
$$P_k = (1 - K_k H)\,P_k^- \text{ ----------(5)}$$

The first task during the measurement update is to compute the Kaman gain, □□ . The next step is to actually measure the process to obtain □□ , and then to generate a posterior state estimate by incorporating the measurement as in equation (4). The final step is to obtain a posterior error covariance with the equation (5). After each time and measurement update pair, the process is repeated with the previous a posteriori estimates used to project or predict the new a priori estimates.

4. EXTENDED KALMAN FILTER

So far only linear systems have been considered. But in practice the dynamic or the observation model can be nonlinear. One approach to the Kalman filter for such nonlinear problems is the so-called Extended Kalman filter. This Kalman filter linearizes about the current estimated state. Thus the system must be represented by continuously differentiable functions.

The complete algorithm is shown in figure 2.



1. **Initialization** of x_0 and P_0 and for *k* in 1...*N*.

2. **Time Update** (Prediction) equations

$\mathbf{x}^{\hat{}-}_k = \mathbf{f}\,(\mathbf{x}^{\hat{}}_{k-1}, \mathbf{0})$

$\mathbf{P}_k^- = \mathbf{A}_k\mathbf{P}_{k-1}\mathbf{A}_k^t + \mathbf{W}_k\mathbf{Q}_{k-1}\mathbf{W}_k^t$

3.1. Innovation: We define

$\mathbf{z}^{\hat{}-}_k = \mathbf{h}\,(\mathbf{x}^{\hat{}-}_k, \mathbf{0})$

$\mathbf{v}_k = \mathbf{z}_k - \mathbf{z}^{\hat{}-}_k$ as the innovation process.

3.2. Measurement Update (Filtering) equations

$\mathbf{x}^{\hat{}}_k = \mathbf{x}^{\hat{}-}_k + \mathbf{K}_k\,\nu_k$

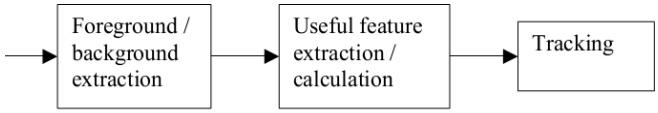
$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^-$

$\mathbf{K} = \mathbf{P}_k^-\mathbf{H}_k^t(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^t + \mathbf{U}_k\mathbf{R}_k\mathbf{U}_k^t)^{-1}$

and I the identity matrix

3. EXPERIMENT RELATED WORK USING MATLAB

A pre-captured or real-time video needs to be processed before applying it in Kalman Filter. At first we need to read and represent a movie in a matrix form. This can be done in a Matlab by using — mmreader or aviread command. It constructs a multimedia reader object associated with video file. We can extract the information content in a video like height, width, number of frame associated in a video. After getting number of frames available in a video, we need to construct Matlab movie structure from the video frames. Matlab movie structure gives colour map information and 2D array of intensity values of 3 colours (RGB). Now we assumed that foreground contains the objects of interest / moving object. For removing less interest zones from the image, we need to extract background image by using mean filter.



For calculating the image containing only the background, a series of preceding images are averaged. For calculating the background image at the instant t,

$$B(x,y) = \sum_{i=1}^{N} V(x,y,t-i)$$

Where N is the number of preceding images taken for averaging. This averaging refers to averaging corresponding pixels in the given images. N would depend on the video speed (number of images per second in the video) and the amount of movement in the video. After calculating the background B(x,y) we can then subtract it from the image V(x,y,t) at time t =t and threshold it.

Thus the foreground is, **| V(x,y,t) – B(x,y) | > Th ,** where, 'Th'

is threshold.

Similarly we can also use median instead of mean in the above calculation of B(x,y). Once background has been extracted, now we use simple difference operation to get object of interest.

Now before calling Kalman Filter, image thresholding needed for removing artifacts and image smoothing. Since colour image is taken for processing, individual thresholding for Red, Green and Blue channel data needed. Then it removes object that have fewer than 300 pixels by using Matlab command ―bwareaopen. Thus it removes the noise artifacts.

After getting the number of frames from the video background is estimated and it is updated with the background updater. Here we track any moving object by initially tracking its centre and radius. Then a circle is drawn with the help of this centre and radius. Hence the model is independent of the size of the ball. So position and velocity will be the input for a kalman filter.

In our model of moving objects on 2D camera images, state is a 4-dimentional vector [x, y, dx, dy], where x and y represent the coordinates of the object‘s centre and dx and dy represent its velocity. The transition matrix is thus simply

$$A=\begin{matrix}1 & 0 & 1 & 0\\ 0 & 1 & 0 & 1\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{matrix}$$

Here B=[1 0 0 0] because object control affects only x-position co-ordinate.

For our model the measurement update equation,

$$\begin{bmatrix}x_{t+1}\\ y_{t+1}\end{bmatrix}=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}x_t\\ y_t\end{bmatrix}+\begin{bmatrix}v\_k * randn\\ v\_k * randn\end{bmatrix}$$

Therefore,

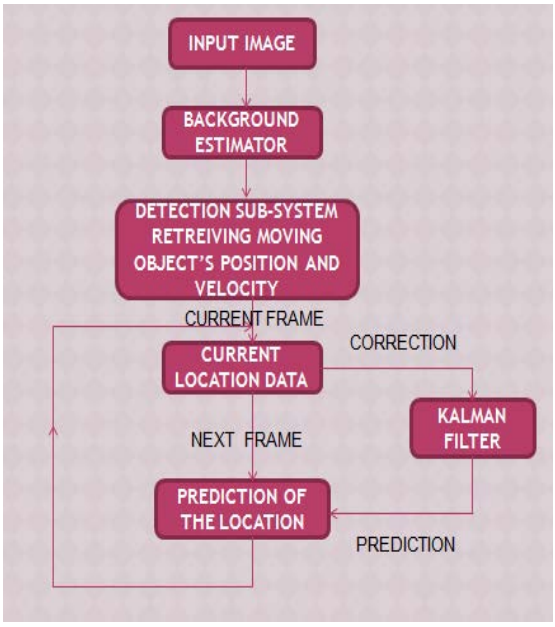
$$Q=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 1\\ 0 & 0 & 0 & 1\end{bmatrix}$$

From one frame to other frame we see only change in the position (x,y) coordinate so measure vector is two dimensional vector. So

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

The measurement noise covariance matrix is calculated using the formula R= E[□□ □□□] where □□ is the standard deviation of the measurement noise.

A flow chart for object tracking problem formulation is given in figure 3.



4. EXPERIMENTS AND RESULTS

Tracking system hardware platform consist of two parts – PC and Camera. The first part is a PC with Intel Core2Duo 1.66 GHz processor and 2GB RAM with a web-camera mounted on it.

- Using the web-cam directly to track moving objects
- Using a video which contains a moving object

Now we have collected small video samples containing a moving object to test our algorithm. Video has been taken from web-camera. Camera resolution is set to (320 x 240). Image acquisition frame rate is set to 30 frames per second. All video taken from camera is saved in Audio Video Intervened (AVI) format. Data rate is taken less than 1000kbps for faster decoding purpose.

We have taken video named Red.avi that contains a random moving ball with some flicker noise. Video dimension is 320 x 240 pixel and frame rate is 30 fps. Tracking result and analysis report has been given below.

1. TRACKING USING A VIDEO TAKEN:

ISSN 2348 – 7968

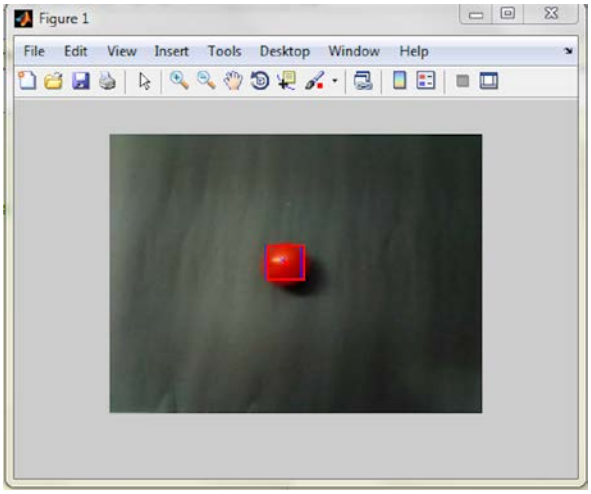


Figure 4. Snap shot of a moving ball

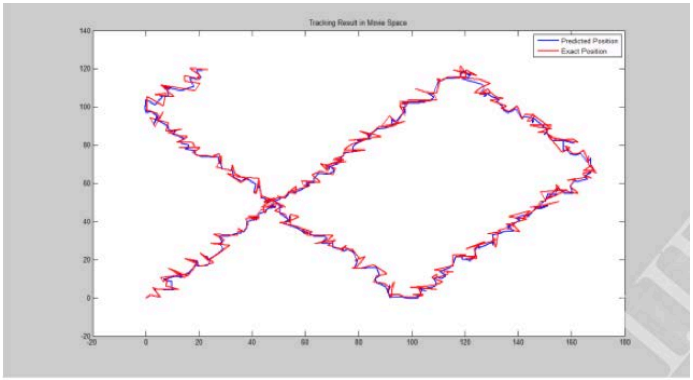


Figure 5. The final tracking result in movie space of a moving ball

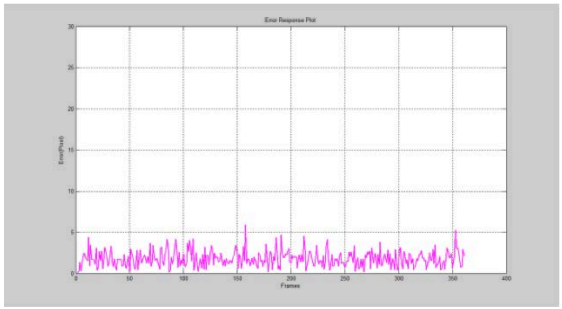


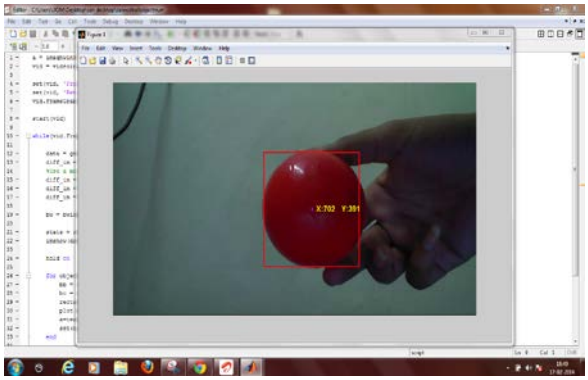
Figure 6. The error response plot of a moving ball

Kalman Filter Processing Time = 0.000058 seconds for 361 frames.

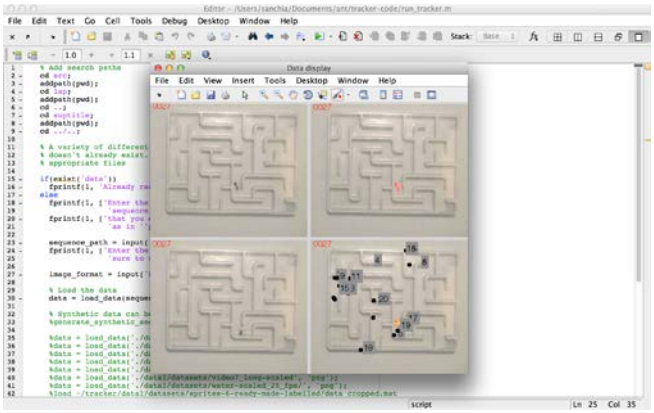
Standard deviation of errors = 0.9638

It can be seen from the result that, Kalman filter can successfully predict the path of that moving noise with very minimum noise of 5%.

2. TRACKING USING WEB-CAM:



3. TRACKING THE MOTION OF AN ANT:



5. CONCLUSION

For linear systems we have used object moving randomly i.e. a ball rolling and ant moving in random paths. A video sequence with just one object within has been selected. The Kalman filter predicted the most probable location of a detected object in the subsequent video frames and tracked an object by assuming the initial state and noise covariance. After accumulating sufficient information about the object, we then successfully tracked the object.

For non-linear object tracking we have chosen predicting the stock prices of the stock market in the short term, relying on extended Kalman filter that enables us to use Speculative model of the market as priori or back ground information which is depending on historical data. And for the observation of the market we have used Empirical model of the

ISSN 2348 – 7968

market, which does not depend on the past history but change randomly, which can be estimated by stochastic pricing models. We have used both models and combined data is given to the prediction-correction filter named extended Kalman filter. Then, after statistical and possibilities calculation, a new stock price will be estimated as the algorithm predicted price to the user. To test the model we have taken sample historical data of Intel Stocks for the past one year estimated the future price of the stock.

6. REFERENCES

1. Saeid bagheri-golzar et al., A New Method for Video Object Tracking, The Journal of Mathematics and Computer Science TJMCS Vol. 4 No.2 (2012) 122-128.
2. Jianbo Shi and Carlo Tomasi, ―Good Features to Track‖, IEEE Conference on Computer Vision and Pattern Recognition, 1994,pp. 593-600.
3. Sheldon Xu and Anthony Chang, Robust Object Tracking Using Kalman Filter with Dynamic Covariance, Cornell University.
4. Dorin Comaniciu, Peter Meer. "Mean Shift: A Robust Approach Toward Feature Space Analysis," IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol 24(5), 2002, pp: 603-619.
5. Zhang Hongzhi, Zhang Jinhuan, Yue Hui, Huang Shilin, ―Object tracking algorithm based on
6. Rachel Kleinbauer, Kalman Filtering Implementation with Matlab, University of Stuttgart, Helsinki, Nov 2004.
7. G. Welch and G. Bishop, An Introduction to Kalman Filter, proceedings of SIGGRAPH 2001,pp 19-24.