# Object Tracking With Movement Prediction Algorithms

Akhilesh Raj
Dept. of Electrical Engg.
IIT Kanpur
Kanpur, India - 208016
Email:akhileshraj91@gmail.com

Abishalini Sivaraman
Dept. of Electrical Engg.
Texas A&M University
College Station, United States
Email:siva558@tamu.edu

Chandreyee Bhowmick
Dept. of Electrical Engg.
IIT Kanpur
Kanpur, India - 208016
Email:cindybhowmick@gmail.com

Nishchal K. Verma
Dept. of Electrical Engg.
IIT Kanpur
Kanpur, India - 208016
Email:nishchal@iitk.ac.in

***Abstract*—The task of tracking an object becomes tedious when the object moves through a dynamic background and the camera also has a random motion. This type of problem has three main aspects - object detection, prediction of object motion and compensation of the camera motion. In this paper, we have developed three algorithms using three different object detection algorithms, namely background subtraction, template matching and Speeded Up Robust Features (SURF). Unscented Kalman Filter (UKF) algorithm has been devised for the motion prediction of the moving object as well as to compensate for the camera movement. The proposed algorithms have been validated through extensive simulations performed on several video datasets and an analytical study has also been presented. Through the simulation results, performance of the proposed algorithms are compared.**

***Index Terms*—Object tracking, Unscented Kalman Filter, background subtraction, template matching, SURF**

## I. INTRODUCTION

Tracking a moving object is an important aspect of computer vision due to its applications in various fields ranging from security and surveillance, video communication, traffic control [1][2], medical imaging and human-machine interface. The task of tracking moving objects is complex due to the following reasons [3]: i) When computer vision is used, the real 3D world gets projected into a 2D image, which causes loss of some information, such as depth data etc. ii) Quality of images taken are very crucial. Noisy images cause problem in tracking the object. iii) Abrupt motion of objects. iv) Change in orientation of the object and image. v) artial and complete object occlusion. vi) Varying lighting conditions. vii) Camera motion. viii) Requirements for real-time processing.

Moving object tracking using computer vision [4] has attracted many researchers. Two popular methods of tracking moving object in dynamic background are Optical Flow Method [5] and Global Motion Compensation (GMC) method [6]. In mean-shift based object tracking [7], most probable target position is found in the current frame. Another method called ORB [8] is also used which is rotation invariant and noise resistant. This is a very fast binary descriptor based on BRIEF. This algorithm has been presented as an efficient alternative to SURF algorithm. Some other methods are Robust fragments-based tracking using integral histogram [9], Bayesian multiple-blob tracker (BraMBLe) [10].

The problem of tracking a moving object in dynamic background is divided into three parts - object detection using image processing algorithm, estimation of the object motion and compensation of the camera motion. Various algorithms are in use for object detection, namely, Mean based Background Subtraction, Template matching and SURF detection. Kalman filter is highly useful to predict the future states of the moving object using noisy data and the previous states, and therefore found wide application in tracking a moving object in dynamic background. We use Kalman filter for compensation of camera motion too. The contributions of this paper are the following:

(i) The future states of the moving object are predicted using Unscented Kalman Filter (UKF) which does the object tracking part.
(ii) In most of the real-time systems, a moving camera is used to capture the video. In this paper, we model the random camera motion using UKF and this motion has been compensated.
(iii) For object detection, three methods have been implemented - Mean based Background Subtraction, Template matching and SURF detection, which gives rise to three different algorithms for the whole process of tracking a moving object in a dynamic background.
(iv) These three algorithms have been validated through simulation results performed on standard video dataset.
(v) A detailed analysis of the proposed algorithms has also been presented in this paper. Three videos with different lighting conditions, camera motion and object orientation have been considered. A thorough analysis has been presented about which algorithm would perform the best in a specific video.
(vi) A quantitative analysis is given on efficiency of the proposed algorithms.
(vii) The algorithms work well in case some occlusion occurs as the future states of the object are predicted using Kalman filter algorithm.

The rest of the paper is organized as follows. Unscented Kalman filter algorithm is explained in Section II. Section III presents a detailed discussion of the object detection algorithms. Object tracking problem using object detection and motion prediction algorithms have been dealt with in Section

IV. Simulation results, analytical discussions and comparative study are presented in Section V. Finally, Section VI concludes the paper.

## II. Unscented Kalman Filtering Algorithm

UKF uses the method of unscented transformation (UT) to predict the future values using noisy measurement and past data [11, 12] in case of non-linear dynamics where the sigma points are chosen from the Gaussian distribution of each state variable and each sigma point is given a weight [13].

UT calculates the statistics of a random variable which undergoes a nonlinear transformation. Let us consider a random variable $x$ of dimension $L$, having mean $X$ and covariance $P_x$, propagates through a nonlinear function $y = g(x)$. To calculate the statistics of $y$, a matrix $\chi$ is formed with $2L+1$ sigma vectors $\chi_i$ having corresponding weights $W_i$, according to the following relations:

$$\begin{aligned} \chi_0 &= X \\ \chi_i &= X + \left(\sqrt{(L+\lambda)P_x}\right)_i, \quad i = 1, \cdots, L \\ \chi_i &= X - \left(\sqrt{(L+\lambda)P_x}\right)_{i-L}, \quad i = L+1, \cdots, 2L \\ W_0^{(m)} &= \frac{\lambda}{L+\lambda} \\ W_0^{(c)} &= \frac{\lambda}{L+\lambda} + (1 - \alpha^2 + \beta) \\ W_i^{(m)} &= W_i^{(c)} = \frac{1}{2(L+\lambda)} \end{aligned} \tag{1}$$

where $\lambda = \alpha^2(L+\kappa) - L$ is the parameter for scaling. $\alpha$ depicts the sigma variance around $X$. It is usually confined to a low positive value (e.g., $10^{-3}$). $\kappa$ is the secondary scaling parameter (usually confined to 0), $\beta$ is used to incorporate knowledge in priori of the distribution of $x$ (for Gaussian distribution, $\beta = 2$ is optimal). Sigma vectors propagate through the nonlinear function

$$\mathcal{Y}_i = g(\chi_i), \quad i = 0, \cdots, 2L \tag{2}$$

Mean and error covariance of $y$ are approximated as follows:

$$y_0 \approx \sum_{i=0}^{2L} W_i^{(m)} \mathcal{Y}_i \tag{3}$$

$$P_y \approx \sum_{i=0}^{2L} W_i^{(c)} (\mathcal{Y}_i - y_0)(\mathcal{Y}_i - y_0)^T \tag{4}$$

The UKF is demonstrated through Algorithm 1.

**Algorithm 1** The Unscented Kalman Filter algorithm [14]

1: Initialization of estimated state vector and error covariance matrix:

$$\begin{aligned} \hat{x}_0 &= E[x_0] = X \\ P_0 &= E[(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T] \\ \hat{x}_0^a &= E[x^a] = [\hat{x}_0^T \quad 0 \quad 0]^T \\ P_0^a &= E[(x_0^a - \hat{x}_0^a)(x_0^a - \hat{x}_0^a)^T] = diag(P_0, P_v, P_n) \end{aligned} \tag{5}$$

for $k \in \{1, \cdots, \infty\}$

2: Calculate sigma points: The sigma points are so chosen that higher order information is incorporated in them.

$$\chi_{k-1}^a = \left[\hat{x}_{k-1}^a \quad \hat{x}_{k-1}^a \pm \sqrt{(L+\lambda)P_{k-1}^a}\right] \tag{6}$$

3: Time update step:

(i) The sigma points are redrawn as follows:

$$\chi_{k|k-1}^x = F[\chi_{k-1}^x, \chi_{k-1}^v] \tag{7}$$

(ii) The predicted values of state vector and error covariance matrix:

$$\begin{aligned} \bar{x}_k &= \sum_{i=0}^{2L} W_i^{(m)} \chi_{i,k|k-1}^x \\ \bar{P}_k &= \sum_{i=0}^{2L} W_i^{(c)} [\chi_{i,k|k-1}^x - \hat{x}_{\bar{k}}][\chi_{i,k|k-1}^x - \hat{x}_{\bar{k}}]^T \end{aligned} \tag{8}$$

(iii) The new sigma points are propagated through the measurement model of the system:

$$\begin{aligned} \mathcal{Y}_{k|k-1} &= H\left[\chi_{k|k-1}^x, \chi_{k-1}^n\right] \\ \bar{y}_k &= \sum_{i=0}^{2L} W_i^{(m)} \mathcal{Y}_{i,k|k-1} \end{aligned} \tag{9}$$

4: Measurement update step:

(i) The innovation covariance and cross covariance matrices:

$$\begin{aligned} P_{\bar{y}_k \bar{y}_k} &= \sum_{i=0}^{2L} W_i^{(c)} [\mathcal{Y}_{i,k|k-1} - \bar{y}_k][\mathcal{Y}_{i,k|k-1} - \bar{y}_k]^T \\ P_{\bar{x}_k \bar{y}_k} &= \sum_{i=0}^{2L} W_i^{(c)} [\chi_{i,k|k-1} - \bar{x}_k][\mathcal{Y}_{i,k|k-1} - \bar{y}_k]^T \end{aligned} \tag{10}$$

(ii) The posteriori state vector and error covariance matrix are updated as

$$\begin{aligned} \hat{x}_k &= \bar{x_k} + \mathcal{K}(y_k - \bar{y}_k) \\ P_k &= \bar{P}_k - \mathcal{K} P_{\bar{y}_k \bar{y}_k} \mathcal{K}^T \end{aligned} \tag{11}$$

where the gain is

$$\mathcal{K} = P_{x_k y_k} P_{\bar{y}_k \bar{y}_k}^{-1} \tag{12}$$

where, $x^a = [x^T v^T n^T]^T$, $\chi^a = [(\chi^x)^T (\chi^v)^T (\chi^n)^T]^T$, $\lambda$ = composite scaling parameter, $L$ = dimension of augmented state, $P_v$ = error covariance of process noise, $P_n$ = error covariance of measurement noise, $W_i$ = weights

## III. Object Detection Algorithms

There are various methods for detection of object in an image. The methods which are interest to us discussed here.

### A. Background Subtraction Using a Mean Filter

Background subtraction using mean filter is a broadly used method to detect an object where the foreground of the image is extracted from the whole image [15].

(i) The background image is estimated as the mean of multiple image frames as the background image is more likely to appear in the subsequent frames.

$$B(x,y,t) = \frac{1}{n} \sum_{i=0}^{n-1} I(x,y,i) \tag{13}$$

where $x$ and $y$ correspond to the position of a particular pixel in the input image $I$ and $n$ is the number of past frames at $t$th time instant.

(ii) The absolute difference between the input image and the background image is calculated and a threshold $T$ is applied to get the foreground image.

$$|I(x,y,t) - B(x,y,t)| > T \tag{14}$$

Any blob that moves across the static background image is isolated.

(iii) The blobs are sorted in order of size and the blob of interest is selected to put a bounding box around it. The largest blob is the object to be tracked.

This method is easy to implement, but doesn't work well if

(i) background is dynamic.
(ii) the scene contains many slow moving objects.
(iii) general lighting condition changes with time.

Mean background models have high memory requirements since we need to analyze multiple image frames at the same time to get the extracted background image. Also, having a global threshold for all pixels which is not a function of time can lead to false detections in real world.

### B. Template Matching

In template matching [16], one template image of the object is procured and then using a correlation algorithm, position of the object is found in the input image. One such correlation algorithm is given below:

**Algorithm 2** Correlation algorithm [17]

1: Depending on the image size, cross-correlation in the spatial or the frequency domain is found out.
2: Local sums are calculated by pre-computing the running sums.
3: Local sums are used to normalize the cross-correlation to get correlation coefficients. The coefficient $\gamma$ at $(u,v)$ is calculated using the following formula:

$$\gamma = \frac{\sum_{x,y}\left[f(x,y) - \overline{f}_{u,v}\right]\left[t(x-u,y-v) - \overline{t}\right]}{\{\sum\limits_{x,y}\left[f(x,y) - \overline{f}_{u,v}\right]^2 \sum\limits_{x,y}\left[t(x-u,y-v) - \overline{t}\right]^2\}^{0.5}} \tag{15}$$

where $f$ = input image, $\overline{t}$ = mean of the template and $\overline{f}_{u,v}$ = mean of $f(x,y)$ under the template region.

This algorithm will output a matrix $A$ that contains values ranging from 0 to 1 which indicate the level of correlation between the template and the image segment (of the same size of the template) centered at each $(x,y)$ position of the input image. A threshold is applied to get rid of detections that might vaguely resemble the object to be tracked.

Template matching is highly effective to track a particular colored object. False detections are very common in cases of poor lighting, occlusion and noise. Thus, we need many template images to recognize the target object under diverse conditions which leads to the algorithm having a high computation time.

### C. Speeded Up Robust Features

SURF [18] is a local feature detector and descriptor, partly inspired by SIFT (Scale Invariant Feature Transform) [19] that detects blob-like features. It is invariant of lighting, scaling, partial occlusion, rotation and hence reliable. Steps [20] are:

(i) Interest point detection: The interest points in an image I are found using a blob detector based on Hessian matrix $H(p,\sigma)$, as given below

$$H(p,\sigma) = \begin{bmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{yx}(p,\sigma) & L_{yy}(p,\sigma) \end{bmatrix} \tag{16}$$

where $p(x,y)$ represents a pixel at $(x,y)$ and $\sigma$ is the scale. SURF makes use of the determinant of the Hessian matrix to account for the interest points' scale and location.

(ii) Local neigborhood descriptor: The descriptor helps to extract distinct and robust image features in terms of the pixel intensity distribution around the points of interest. Haar wavelet responses are used to obtain the orientation of the interest points in $x$ and $y$ directions inside a circular neighborhood of radius $6\sigma$, where $\sigma$ is the scale at which the interest point was perceived.

(iii) Matching: Matching pairs can be established by comparing interest points and their description from two different images.

SURF is comparatively fast and highly efficient to track object irrespective of scale, orientation and background lighting.

## IV. Object Tracking Using Object Detection And Motion Prediction Algorithms

As discussed before, the problem of tracking a moving object has three main aspects, object detection, prediction of object movement and compensation for the camera movement. Object detection is performed using the algorithms discussed in Section 3. UKF is used for movement prediction of the object and camera movement compensation.

*Compensation for the camera movement*

In most cases, the camera sensor is moving and thus the camera movement has to be compensated. We can do this either by increasing the dimensions of the state variable to compensate image frame position and velocity with respect to the object or by adding the velocity of the camera to the kinematic equations incorporated to predict the future location of the object. Both the algorithms perform efficiently.

(i) Increase state variables and include image width, height, $x$ velocity and $y$ velocity as this corresponds to the movements of the camera. This will help us reposition the frame at every instant as per the camera movement.

$$f = \begin{bmatrix} x(1) + x(3) * dt + \frac{a_x}{2} * (dt^2) \\ x(2) + x(4) * dt + \frac{a_y}{2} * (dt^2) \\ x(3) + a_x * dt \\ x(4) + a_y * dt \\ x(5) + x(7) * dt \\ x(6) + x(8) * dt \\ x(7) \\ x(8) \end{bmatrix} \tag{17}$$

where $x(1)$ = $x$ position of object, $x(2)$ = $y$ position of object, $x(3)$ = $x$ coordinate of object velocity, $x(4)$ = $y$ coordinate of object velocity, $x(5)$ = width of image frame, $x(6)$ = height of image frame, $x(7)$ = velocity of camera in $x$ direction, $x(8)$ = velocity of camera in $y$ direction.

(ii) Modify the function $f$ by incorporating camera velocity into it while implementing an UKF. The velocity of the camera will be considered as a control variable. This method is valid for small camera velocities.

$$f = \begin{bmatrix} x(1) + x(3) * dt + \frac{a_x}{2} * (dt^2) + v_c * dt \\ x(2) + x(4) * dt + \frac{a_y}{2} * (dt^2) + v_c * dt \\ x(3) + a_x * dt \\ x(4) + a_y * dt \end{bmatrix} \quad (18)$$

where $x(1)$ = $x$ coordinate of object position, $x(2)$ = $y$ coordinate of object position, $x(3)$ = $x$ coordinate of object velocity, $x(4)$ = $y$ coordinate of object velocity, $a_x$ and $a_y$ = acceleration of object in $x$ and $y$ direction. $v_c$ = velocity of camera

The whole process is summarized in Algorithm 3.

**Algorithm 3** Moving Object Tracking Algorithm

1: Load template and its features (size, pixel values etc.). Set $N_f$ (number of frames in the video),
2: Initialization of Kalman filter parameters
3: **for** $i$ = 1 to $N_f$ (each frame) **do**
- Load frame $i$ and its features (size, pixel values etc.)
- Perform appropriate object detection algorithm (background subtraction/ template matching/ SURF) to extract the actual location of the object
- **if** i = 1 **then** Initialize position state variables with actual values from the object detection algorithm
  **end if**
- Motion prediction:

$$[X^+, P^+] = unscentedKF(f, X, P, h, z, Q, R)$$

where $f$ = function defined as second order dynamics of the target object, $h$ = function defined to relate the actual values obtained by detection algorithm to estimates derived from UKF, $z$ = actual values obtained from detection algorithms, $Q$ = error covariance matrix of process noise, $R$ = error covariance matrix of measurement noise, $X$ and $P$ = apriori state vector and error covariance matrix, $X^+$ and $P^+$ = postriori state vector and error covariance matrix

4: **end for**

## V. Results and Discussions

The object tracking algorithms are validated through simulations performed on video datasets. Various object detection algorithms have been used.

For implementation of the Kalman filter algorithm, the state variable comprises of $x$, $y$ coordinates of the object position in image frame and $x$, $y$ coordinates of object velocity. This Kalman filter predicts the states of the object with respect to a still image frame. But due to motion of the camera, some compensations are to be incorporated in the algorithm. This can be done in any of the following ways:

(i) Frame parameters (width and height) and camera velocity (in x and y directions) are included as variable in the state vector.

(ii) For small camera velocities, we can add the camera velocity to the second order movement dynamics of the object.

Satisfactory results have been obtained by incorporating either of these methods. In this section, we present the results when the camera movement compensation has been done by increasing the dimension of the state vector. The first simulation is performed on a standard video dataset available at [21], in which the target is a bouncing ball. The ball movement in the subsequent frames is depicted in Fig. 1.

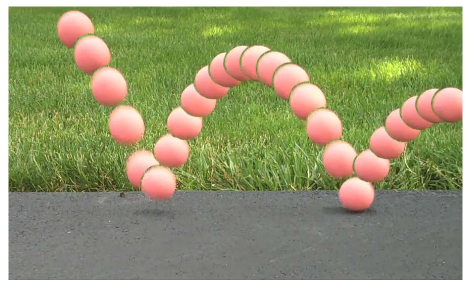
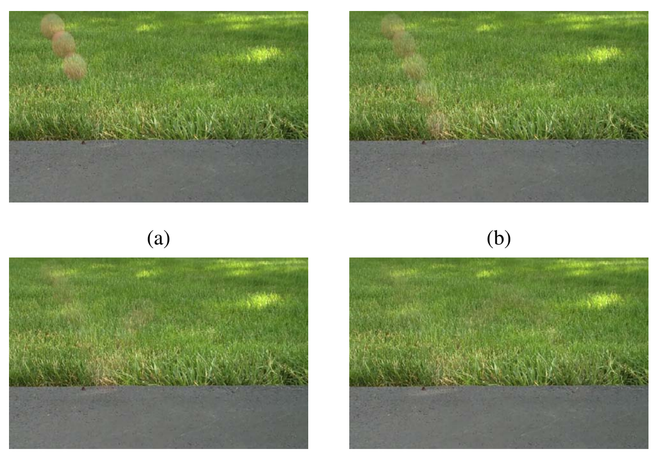


Fig. 1: The bouncing ball video frames

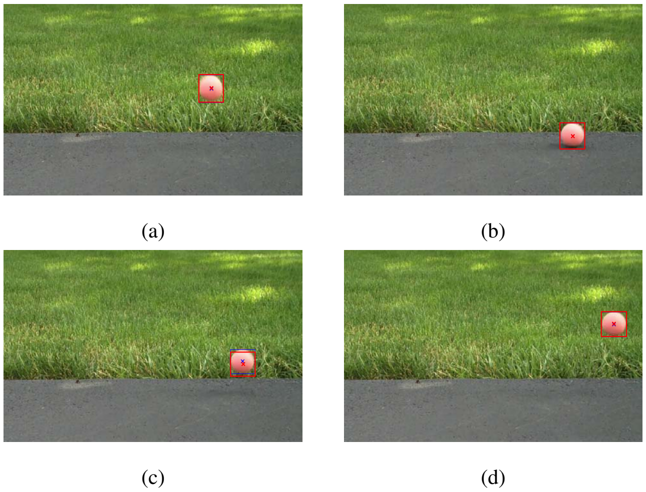
The video is captured using a static camera, so the target object moves in a constant background. Moreover, there is no other moving object in the video. This makes the background subtraction object detection algorithm the most appropriate one. The results of the background extraction filter have been presented in Fig. 2. The background is better extracted when more number of frames are considered for averaging.



(a) (b) (c) (d)

Fig. 2: Extracted background by averaging (a) 3 frames (b) 5 frames (c) 10 frames (d) 20 frames

The object is tracked in successive video frames by subtracting the extracted background image from each input frame followed by application of UKF. The object is detected by a rectangular box around it as shown in Fig. 3. The root mean square error in tracking has been plotted in Fig. 4.



(a) (b) (c) (d)

Fig. 3: Detected target object in (a) frame 18 (b) frame 20 (c) frame 21 (d) frame 25 using background subtraction

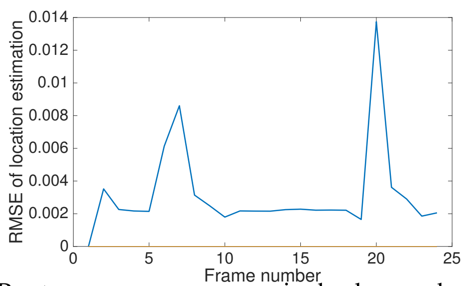



Fig. 4: Root mean square error in background subtraction based tracking

The second simulation is performed on video dataset [22] where the target object is a moving car. As the video is captured by a moving camera, background subtraction does not hold good. Template matching is used to detect the object as the car has same orientation throughout the entire video and a single template is sufficient to track the object. Compensation for the camera movement has also been taken care of. The tracked object in four frames has been shown in Fig. 5.

(a) (b) (c) (d)

Fig. 5: Detected target object in (a) frame 16 (b) frame 39 (c) frame 46 (d) frame 48 using template matching

As depicted in Fig. 5, in frame 46, it gives a false detection as template matching algorithm is not very effective when the size of the object varies with respect to template. Despite the false detection, the UKF accurately predicts the location of the object from its previous location. For this simulation too, the root mean square error in tracking has been plotted in Fig. 6.

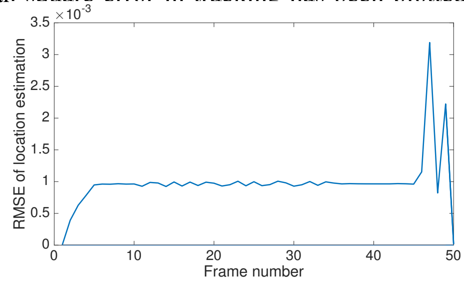



Fig. 6: Root mean square error in template match based tracking

Third simulation is performed on data generated in our laboratory environment. Target object is a small carton which moves randomly and video is recorded by a randomly moving camera. SURF is the optimal algorithm because:

(i) Carton orientation changes w.r.t. camera position.
(ii) The lighting condition varies across various frames.
(iii) Target image is not clear due to fast camera motion.

Interest points are detected using SURF both in the template image as well as in the input images and they are matched to find the exact location of the object. One such matching has been shown in the Fig. 7. Camera movement compensation has been incorporated in our algorithm. The tracking performance is demonstrated through the results in Fig. 8.

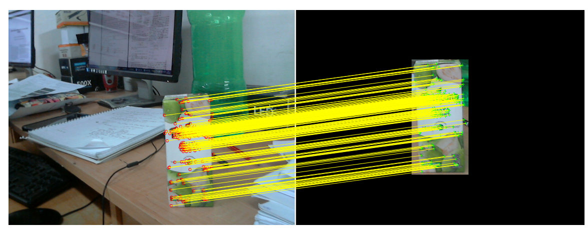
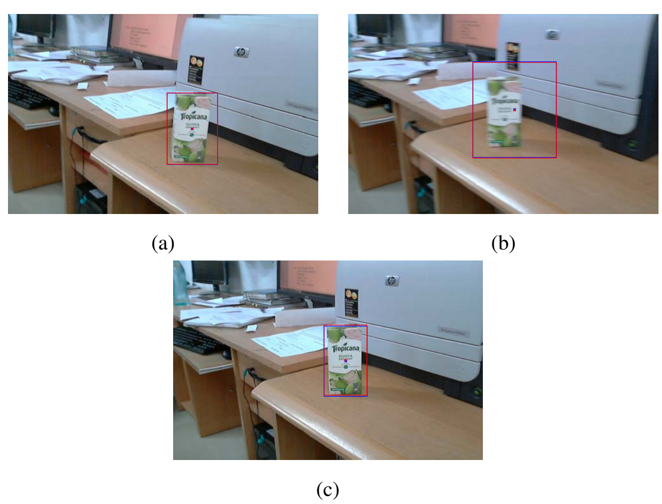


Fig. 7: Matched points between the template and input frame



(a) (b) (c)

Fig. 8: Detected target object in (a) frame 31 (b) frame 35 (c) frame 45 using template matching

The algorithm is able to track the object even in case of blurred image. The root mean square error is shown in Fig. 9.

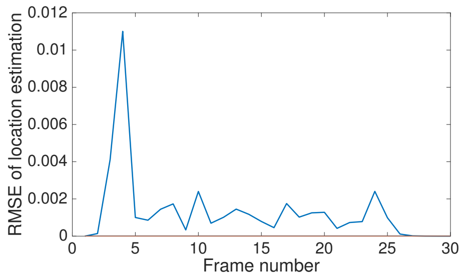



Fig. 9: Plot of root mean square error in SURF based tracking

Performance of the algorithms are compared through average root mean square error values, as tabulated in Table I.

TABLE I: Average root mean square error in tracking

| Method | Average RMSE |
|---|---|
| Background subtraction based algorithm | 0.0042 |
| Template matching based algorithm | 0.0011 |
| SURF based algorithm | 0.005 |

Background subtraction method helps us get an accurate result only when the background is stationary, whereas template matching can track the target object moving through a dynamic background accurately provided the object doesn't change its size and orientation. Therefore we see a higher RMSE for Background subtraction than template matching for tracking objects across a dynamic background. We get a higher RMSE value for SURF based tracking since it takes object rotation, scale variation and fast camera movement into account. The error is small considering the fact that the target object and the camera moved randomly at varying speeds at all time.

## VI. Conclusion

This paper focusses on the problem of computer vision based tracking of a moving object. There are three aspects of such problems - object detection, motion prediction of the object and compensation for the camera motion, which have been individually dealt with. Different video datasets have been considered and the most optimal algorithm has been selected for tracking the moving target, based on different features of the videos. The simulation results have been presented along with detailed study of the algorithms' performance. A quantitative analysis has also been given to compare the efficiency of the algorithms.

## References


[1] N. K. Verma, G. Kumar, A. Siddhant, P. Nama, A. Raj, A. Mustafa, N. K. Dhar, and A. Salour, "Vision based obstacle avoidance and recognition system," in *IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI) 2015*, pp. 1–7.

[2] N. K. Verma, P. Nama, G. Kumar, A. Siddhant, A. Raj, N. K. Dhar, A. Salour, *et al.*, "Vision based object follower automated guided vehicle using compressive tracking and stereo-vision," in *IEEE Bombay Section Symposium (IBSS) 2015*, pp. 1–6.

[3] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *Acm computing surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.

[4] N. K. Verma, S. K. Sahu, A. Mustafa, N. K. Dhar, A. Salour, *et al.*, "Priority based optimal path routing for automated guided vehicle," in *IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI) 2015*, pp. 1–7.

[5] B. D. Lucas, T. Kanade, *et al.*, "An iterative image registration technique with an application to stereo vision.," in *IJCAI*, vol. 81, pp. 674–679, 1981.

[6] H. Jia, M. Xie, and L. Ren, "An improved global motion estimation for practical objection detection," in *IEEE International Conference on Information and Automation, 2008*, pp. 1159–1162.

[7] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *IEEE Conference on Computer Vision and Pattern Recognition, 2000*, vol. 2, pp. 142–149.

[8] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *IEEE International conference on computer vision 2011*, pp. 2564–2571.

[9] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2006*, vol. 1, pp. 798–805.

[10] M. Isard and J. MacCormick, "Bramble: A bayesian multiple-blob tracker," in *IEEE International Conference on Computer Vision 2001*, vol. 2, pp. 34–41.

[11] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[12] R. Faragher *et al.*, "Understanding the basis of the kalman filter via a simple and intuitive derivation," *IEEE Signal processing magazine*, vol. 29, no. 5, pp. 128–132, 2012.

[13] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *AeroSense'97*, pp. 182–193, International Society for Optics and Photonics, 1997.

[14] E. A. Wan and R. Van Der Merwe, "The unscented kalman filter for nonlinear estimation," in *The IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000*, pp. 153–158.

[15] T. Bouwmans, F. Porikli, B. Höferlin, and A. Vacavant, *Background Modeling and Foreground Detection for Video Surveillance*. CRC Press, 2014.

[16] B. Roberto, "Template matching techniques in computer vision: theory and practice," 2009.

[17] J. Lewis, "Fast normalized cross-correlation," in *Vision interface*, vol. 10, pp. 120–123, 1995.

[18] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*, pp. 404–417, Springer, 2006.

[19] D. G. Lowe, "Object recognition from local scale-invariant features," in *CIEEE International Conference on Computer vision, 1999*, vol. 2, pp. 1150–1157.

[20] A. M. Gupta, B. S. Garg, C. S. Kumar, and D. L. Behera, "An on-line visual human tracking algorithm using surf-based dynamic object model," in *2013 IEEE International Conference on Image Processing*, pp. 3875–3879, IEEE, 2013.

[21] "Video snapshots: Creating high-quality images from video clips." http://www.eecs.harvard.edu/~kalyans/research/snapshots/supplementary/results.html.

[22] "Object recognition in video dataset." http://mi.eng.cam.ac.uk/research/projects/VideoRec/CamVid/.