Ball tracking in cricket video

Kanav Vats
Systems Design Engineering
University of Waterloo
k2vats@uwaterloo.ca

Abstract

Ball tracking in cricket videos is carried out using multiple high speed cameras involving generation of 3D trajectories using triangulation and kalman filtering. In this project, we focus on tracking the cricket ball in low quality single camera videos using the well known kalman filter and particle filter algorithms and comparing the performance of the two. For the purpose of simplicity, non broadcast video having a stationary camera is used. Ball detection is carried out using color thresholding and background subtraction. Experimental results demonstrates that the particle filter performs better than kalman filter in tracking the non-linear abrupt change in trajectory at the point of ball bouncing.

1 Introduction

Ball tracking has become an indispensable part of sports. Accurate estimation of ball position and ball trajectory is of high utility to sports analysts and coaches for accessing the performance of players. Various techniques like hawk-eye [7] have been implemented for ball tracking in sports like cricket, tennis and badminton. Cricket is also one of the sports where ball tracking is very useful. Ball tracking has helped cricket umpires to give better decisions, especially in leg-before-wicket decisions. hawk-eye in cricket is extremely useful for leg before wicket decision in order to determine whether a ball hitting the pad of a batman was about the hit the stump or not (figure 1). Also, tracking the ball and estimating the trajectory of the ball can be used to predict where the ball meets the batsman and hence, determine what can be the optimal batsman shot for that delivery. Actual ball tracking in sports is implemented using six or more television cameras, placed at strategic locations on the ground, such that a 3D trajectory of the ball can be reproduced using kalman filtering [4].

The objective of this project is to implement a very basic version of ball tracking for cricket videos. It involves using the kalman Filter and particle filter to track the position of the cricket ball in the time interval between the bowler's ball delivery and ball hitting the bat. The implementation is done on non-broadcast video with a steady camera, for the purpose of simplicity. Ball detection for getting measurements is done by old-school computer vision techniques such as ball-color detection and background subtraction (figure 3).

The main challenge involved in this problem is the abrupt change in trajectory once the ball bounces, lack of measurements due to motion blur short-

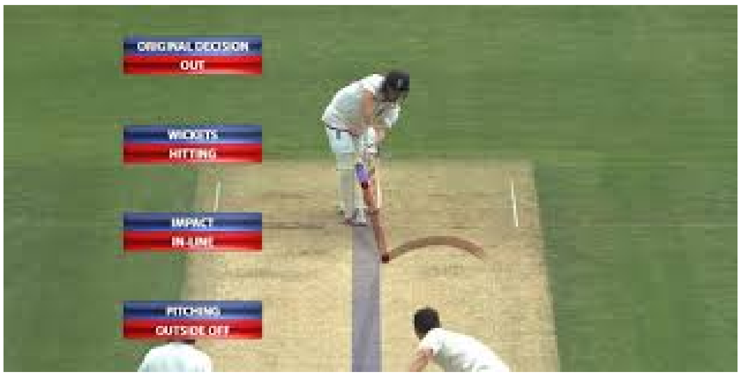


Figure 1: Ball tracking in cricket is used to aid cricket umpires in leg before wicket decision. A batsman is given out if the ball was about to hit the wicket and it touched the pads of the batsman.

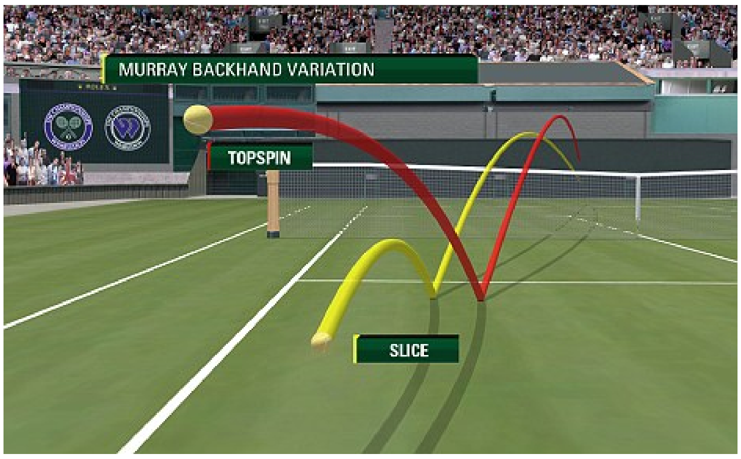


Figure 2: Ball tracking used for analyzing ball hitting patterns.

coming of the ball detection system and varying speed of the ball delivery. The experimental results demonstrate that kalman filter, being a linear algorithm is able to track the ball in the horizontal direction with a constant velocity process formulation, but it is not able to handle the abrupt change in trajectory at the point of ball bouncing. The particle filter algorithm, on the other hand, can handle such non linear situations.

2 Background

Sports analytics has widely been revolutionized by advancement in tracking methods and computer vision. This is evident by the development of ball tracking systems in sports like tennis [10] golf [9] and cricket. The most prominently used ball tracking system in cricket and tennis broadcast sports footage is the

Hawk-eye [7] ball tracking system. In this system multiple high speed cameras are strategically placed on the ground. The system first generates 2D trajectories on the screen coordinates. This process is then followed by 3D trajectory reconstruction by combining 2D tracklets into 3D trajectories by polynomial curve fitting. Two distinct 3D trajectories are combined by finding the point of ball impact on the ground between them which is achieved by using a kalman filter [4]. The system differs from the work done in this project in the way that in actual hawk-eye system, tracking is done in 3D coordinate system and 2D tracklets are only used for 3D mapping with the knowledge of camera parameters by triangulation.

Ball tracking in broadcast videos make use of multiple high speed cameras and uses high quality videos. In the case of low quality, single camera videos, ball tracking depends on accurate object detection and trajectory prediction. Kalman filter [4] and particle filter [1] are widely used in ball tracking in sport videos. Yan *et al* [10] detect the tennis ball using an SVM making use of foreground blob features and then use a particle filter to track tennis ball from low quality videos taken from a single camera. The particle filter successfully handles the bouncing of the ball from the ground. Also, in order to prevent loss of tracking, an automatic switching model is used to handle the situation of player hitting the ball with racket. Huang *et al* [3] use a combination of weak and strong detectors with a particle filter for ball tracking in soccer. Once occlusion is detected, the velocity is no longer estimated. Kim *et al* [5] attempt to track soccer ball in soccer videos using a dynamic kalman filter which automatically changes it's parameters in order to handle the problem of ball occlusion. However their algorithm fails when the players are crowded on the field. Cheng *et al* [2] use the particle filter for ball tracking and event detection in volleyball. The likelihood of getting hit by an external force is based on how close the ball is from the player and velocity variation in the past ball trajectory. Zhou *et al* [12] produce ball candidate detections by subtracting the adjacents frames of the video and form small "trajectorylets" using the ball candidate detections. They then form a weighted directed acyclic graph using the trajectorylets and use the shortest path algorithm to produce the final ball trajectory.

Unlike tennis, in the context of cricket, ball tracking in low quality, single camera videos has not been explored in a great detail. Zaveri *et al* [11] use a filter bank approach to tracking of small objects like ball in sport sequences in order to handle the maneuvering and non maneuvering tracks of the object to be tracked. Roopchand *et al* [8] use kalman filtering for bat detection and tracking which is utilized for stroke recognition.

This project focuses on investigating the advantages and shortcomings of the kalman and particle filter algorithms for ball tracking in cricket videos. The bouncing of the cricket ball on the pitch and absence of measurements are two big challenges in this scenario and the objective is to determine the performance of kalman filter and particle filter in these situations.

3 Methodology

This section deals with the steps involved in the pipeline from data collection and obtaining ball detection measurements to ball tracking. The theoretical aspects of the filtering algorithms applied are discussed along with their scope

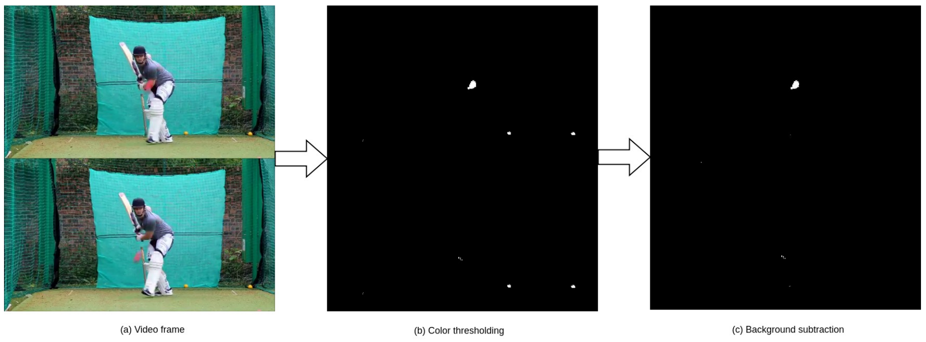


Figure 3: Pipeline for obtaining ball measurements. Red color thresholding is applied to frames extracted from cricket video after which background subtraction is done for removing false positives.

of application to the ball tracking problem.

3.1 Data collection and ball detection

In order to apply tracking algorithms, appropriate candidates for the ball's x and y coordinates need to be determined. Broadcast cricket videos have various camera positions and movement involved. In order to simplify the experiments, a non broadcast cricket video of resolution 640×360 pixels with a frame rate of 30 frames per second having a static camera is used. Since we are tracking the ball from the point when it leaves the bowler's hand and reaches the batsman, the frames corresponding to this time frame are extracted.

In the video used, the ball is red in color, similar to what is found in test cricket matches (figure 3). In order to detect ball candidate position in the video frames, the frame is converted from BGR color space to HSV color space followed by red color thresholding, which removes anything in image not red in color. The hue value of red, in OpenCV, being approximately in the range of 0 to 20 and 160 to 180. In practice, color thresholding results in a large number of false positives (figure 3). In order to reduce false positives, background subtraction using a gaussian mixture based background segmentation algorithm [cite] in done between subsequent frames. In order to calculate the performance of tracking algorithms, the actual ground truth position of the centre of the ball is determined manually. Since the time interval between the ball delivery and ball reaching the batsman is very short, we make use of all the frames with a time interval of $\frac{1}{30}$ seconds in this period of the video.

3.2 Kalman Filter

After collecting appropriate ball detection measurements, we analyze the performance of the kalman filter [4] for ball tracking. The state of the ball is a 4 dimensional vector $\mathbf{X_t} = \begin{bmatrix} x_t & y_t & \dot{x_t} & \dot{y_t} \end{bmatrix}^T$, where x_t and y_t denote the horizontal and vertical coordinates of the ball respectively. $\dot{x_t}$ and $\dot{y_t}$ denote the horizontal and vertical velocities. Let Δt be the time interval between two subsequent frames. We have taken two scenarios into account.

3.2.1 Acceleration only as dynamic noise

In this scenario, the equations of motion are:

$$x_{t+1} = x_t + \dot{x}\Delta t \tag{1}$$

$$y_{t+1} = y_t + \dot{y}\Delta t \tag{2}$$

$$\dot{x_{t+1}} = \dot{x}_t \tag{3}$$

$$\dot{y_{t+1}} = \dot{y}_t \tag{4}$$

Given that the x and y positions can be measured, the state and measurement models can be defined as:

$$X_{t+1} = AX_t + w_t \tag{5}$$

$$Y_t = HX_t + v_t \tag{6}$$

where Y_t are the measurements. w_t and v_t are the process noise and measurement noise at time step t with a covariance of Q and R respectively.

$$A = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

Let P_t be the error covariance at time t, we now have the Kalman predict and update equations:

Predict

$$\hat{X}_{t,t-1} = A\hat{X}_{t-1,t-1} \tag{7}$$

$$P_{t,t-1} = AP_{t-1,t-1}A^T + Q \tag{8}$$

Update

$$K_t = P_{t,t-1}H^T(HP_{t,t-1}H^T + R)^{-1} \tag{9}$$

$$\hat{X}_{t,t} = \hat{X}_{t,t-1} + K_t(Y_t - H\hat{X}_{t,t-1} \tag{10}$$

$$P_{t,t} = (I - K_tH)P_{t,t-1} \tag{11}$$

3.2.2 Constant acceleration control input

In this case we assume that the ball's motion has a constant acceleration a in the vertical direction. The equations of motion for this case are:

$$x_{t+1} = x_t + \dot{x}\Delta t \tag{12}$$

$$y_{t+1} = y_t + \dot{y}\Delta t + \frac{1}{2}a\Delta t^2 \tag{13}$$

$$\dot{x_{t+1}} = \dot{x}_t \tag{14}$$

$$\dot{y_{t+1}} = \dot{y_t} + a\Delta t \tag{15}$$

The new state model and state predict equation become:

$$X_{t+1} = AX_t + B_t u_t + w_t \tag{16}$$

$$\hat{X}_{t,t-1} = A\hat{X}_{t-1,t-1} + B_t u_t \tag{17}$$

where $B_t = \begin{bmatrix} 0 & \frac{1}{2}\Delta t^2 & 0 & \Delta t \end{bmatrix}^T$ and control input $u_t = a$. The measurement model and update equations remains the same. This is because the measurements and updates do not depend on the way the state changes.

3.3 Particle Filter

Particle filter [1] is a filtering algorithm suitable for non-linear and non-gaussian systems. Since the bouncing of the ball on the cricket pitch induces a non linearity in the ball's dynamics, the problems becomes a good candidate for applying the particle filter.

Succinctly, in particle filter method, we create thousands of particles, each representing a hypothesis about the state of the system we want to track. In addition to that, each particle is assigned a weight which is proportional to the probability of the particle representing the true state of the system. The mean of the estimate is the weighted mean of the particles. The crux of the particle filter algorithm is **Importance Sampling**. It is a way of estimating the properties of an unknown distribution by sampling from a known distribution. Formally, let the expected value of the function $f(x)$ with an unknown distribution $p(x)$ is given by :

$$E[f(x)] = \int f(x)p(x)dx$$

Since we do not know $p(x)$, we multiply and divide this equation by a known distribution $q(x)$ such that,

$$E[f(x)] = \int f(x)p(x)\frac{q(x)}{q(x)}dx$$

, which equals,

$$E[f(x)] = \int \frac{f(x)p(x)}{q(x)}q(x)dx$$

$$E[f(x)] = E_q\Big[\frac{f(x)p(x)}{q(x)}\Big]$$

where $\frac{p(x)}{q(x)}$ is the likelihood or the importance weight. So, the importance sampling algorithms works by sampling n samples $X_1, X_2.....X_n$ from the known (also called proposal) distribution $q(x)$. The the estimatior of $E(f(x))$ or μ is given by:

$$\hat{\mu} = \frac{1}{n}\sum_{n=1}^{n}\frac{f(X_i)p(X_i)}{q(X_i)}$$

Now we can move on to the actual particle filter algorithm. Let $\chi_t = (x_t^i, w_t^i)_{i=1..N}$ be the set of N particles and their weights at time step t. The initial weights at $t = 0$ are set to as $\frac{1}{N}$. The particles x_t^i are sampled from a proposal distribution

$p(x_t|x_{t-1}, u_t)$, where u_t is the control input. Let z_t denote the measurement at time step t,

Algorithm 1: Particle filter

```
  Result: $\chi_t$
1 Input: $\chi_{t-1}, u_t, z_t$
2 Initialize: $\chi_t = \phi$
3 for $i = 1 \, to \, N$ do
4     Predict : $sample \, x_t^i \sim p(x_t|x_{t-1}, u_t)$
5     Update : $w_t^i = p(z_t|x_t^i)$
6     Add $(x_t^i, w_t^i)$ to the set $\chi_t$
7 end
8 Normalize the weights $w_t^i$
9 Return $\chi_t$
```

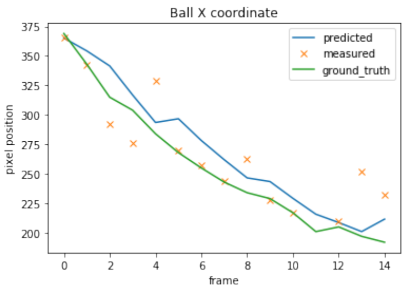
In our problem of ball tracking, each particle represents a possible state of the ball and is a 4D vector of vertical and horizontal position and velocities (as in Kalman Filter formulation in section 3.2). The proposal distribution $p(x_t|x_{t-1}, u_t)$ is the motion model, with acceleration as the control input. The main idea is that we are trying to estimate the properties of the actual distribution of the motion of the ball (say $\pi(x)$) using this proposal distribution with the help of importance sampling. The state estimate at any time t is simply the weighted mean $\sum_{i=1}^{N} w_t^i x_t^i$. We have assumed gaussian process and measurement noise.

3.3.1 Degeneracy

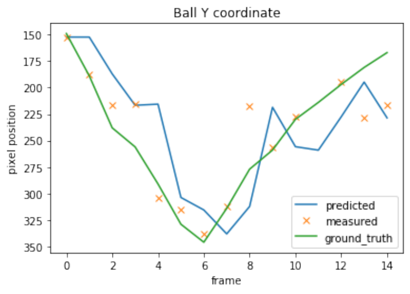
The Algorithm 1 suffers from the problem of degeneracy where after a few iterations, only a few number of particles have any significant weight (figure z). This is because with increasing iterations, the particles which do not agree with the measurements are left with extremely small weight. A measure of the degeneracy of the algorithm is to calculate the effective number of paticles N_{eff} [6], given by:

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} (w_t^i)^2} \tag{18}$$

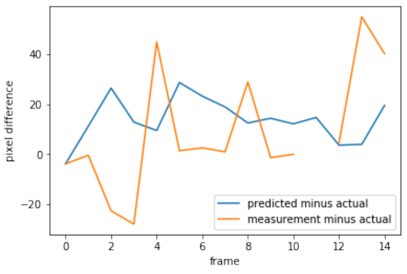
In case of severe degeneracy such that $N_{eff} < T$, where T is threshold often set to $\frac{N}{2}$, the problem of degeneracy is solved by the process of **resampling**[1]. The basic idea behind resampling is simple, the particles with low weights are discarded and the particles having a high weight are sampled more than once. After the resampling is done the weights are re-set to the value $\frac{1}{N}$. The modified particle filter algorithm with resampling is:



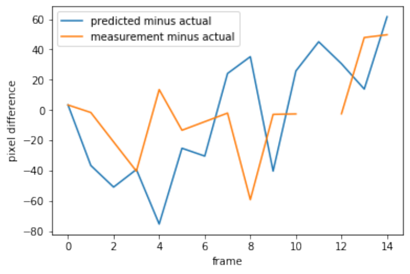
(a) Ball x coordinate vs frame



(b) Ball y coordinate vs frame



(c) Error comparison in x direction



(d) Error comparison in y direction

Figure 4: Results of the Kalman filter with acceleration only as process noise.

Algorithm 2: Particle filter with resampling

```
   Result: $\chi_t$
1  Input: $\chi_{t-1}, u_t, z_t$
2  Initialize: $\chi_t = \phi$
3  for $i = 1\, to\, N$ do
4      Predict : $sample\, x_t^i \sim p(x_t|x_{t-1}, u_t)$
5      Update : $w_t^i = p(z_t|x_t^i)$
6      $Add(x_t^i, w_t^i)$ to the set $\chi_t$
7  end
8  Normalize the weights $w_t^i$
9  Calculate $N_{eff}$ using (18)
10 if $N_{eff} < \frac{1}{N}$ then
11     $(x_t^i, w_t^i)_{i=1..N} = RESAMPLE((x_t^i, w_t^i)_{i=1..N})$
12 end
13 Return $\chi_t$
```

4 Results and discussion

We compare and discuss the performance of kalman filter and particle filter on the same sequence of cricket video frames.

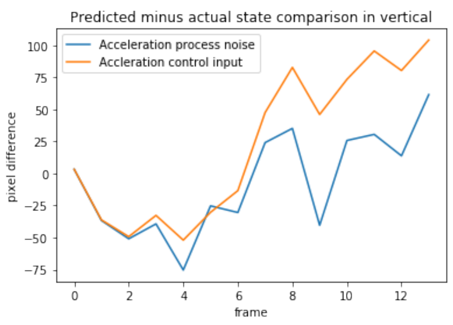


Figure 5: Comparison of the performance of the two kalman filter models in Y direction. The kalman filter with acceleration as control input performs better before the ball bounce point at the 6th frame. But after the 6th frame it becomes highly measurement dependent.

4.1 Kalman Filter

This section deals with the advantages and drawbacks of the Kalman filter applied to the problem. Throughout the results we use the image coordinate system with the origin at the top left side of the image.

4.1.1 Acceleration only as dynamic noise

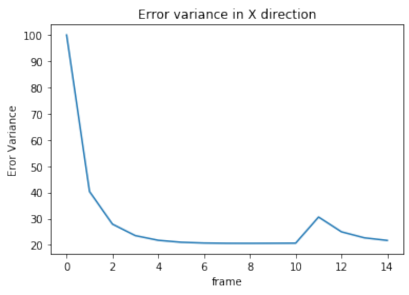
The results of the algorithm are shown in Figure 4. Because of the incorrect constant velocity process assumption in the motion model, we use a high value of process noise for the position and velocity in the Y direction. The algorithm performs much better in the X(horizontal) direction than the Y(vertical) direction, evident from Figure 4 (a) and Figure 4 (c), showing lesser error between the predicted and actual position than the measured and actual position. This is because, in the horizontal direction, the ball has a constant velocity which matches our model formulation. In Figure 4 (a), no measurement is present at the $11th$ frame due to motion blur, however from the figure it can be seen that the algorithm does a good job predicting the position of the ball at this point.

In the Y(vertical direction) the algorithm is not able to do a good job, evident from 4 (b) and (d), showing a high difference between the predicted and actual position than the measured and actual position. This is because:

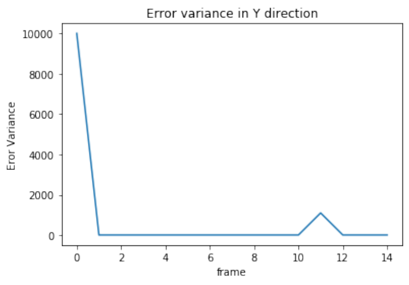
1. The motion model does not conform with the actual motion of the ball. Even after applying a large process noise and less measurement noise in the Y direction, the filter lags behind the actual process. This is because in the predict step the model always follows a constant velocity process and the only way it can correct itself is by making predictions close to measurements.

2. Because of the inherent lag present in the predictions, the filter is not able to handle the ball bouncing off the ground evident by Figure 4 (b). The velocity vector of the ball reverses it's direction after bouncing which makes it very hard for the algorithm to track.

Figure 6 shows the variation of the norm of error variance.

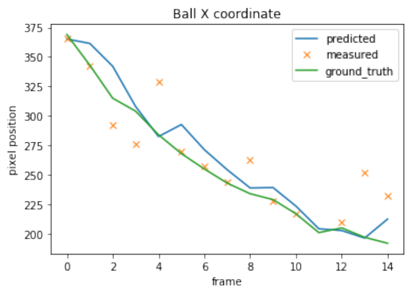


(a) Error variance in X direction

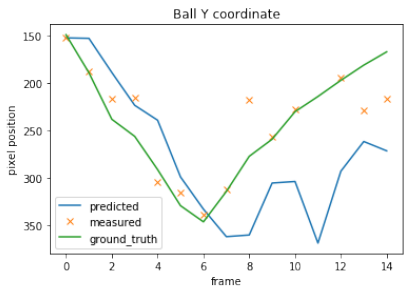


(b) Error variance in Y direction

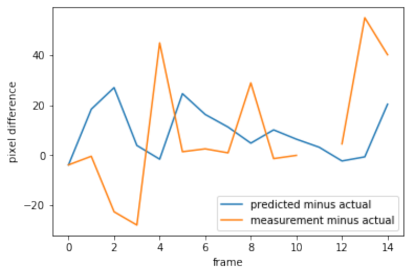
Figure 6: Norm of error variance Kalman filter with acceleration as process noise. The bump at frame 11 is due to the absence of measurement.



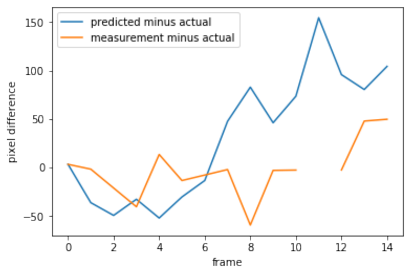
(a) Ball x coordinate vs frame



(b) Ball y coordinate vs frame



(c) Error comparison in x direction



(d) Error comparison in y direction

Figure 7: Results of the kalman filter with acceleration as control input. The algorithm shows good performance in X direction. In Y direction, before frame 6 where the ball bounces off, it performs better than kalman filter with acceleration only as process noise.

Algorithm	Absolute prediction error(pixels)	
	X direction	Y direction
Kalman1	211	492.83
Kalman2	152.13	747.5
Particle filter	**129.12**	**251.39**

Table 1: Sum of absolution prediction error across all frames. Kalman1 is the kalman filter with acceleration only as dynamic noise and kalman2 is the kalman filter with acceleration as control input. The particle filter shows the best performance in X direction and by far the best performance in the Y direction.

4.1.2 Acceleration as control input

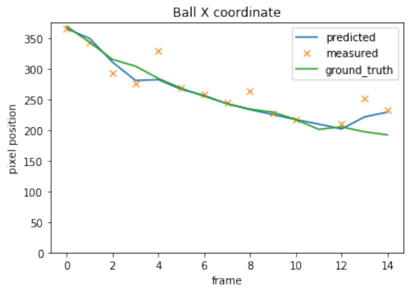
From figure 7 (a) and (c), the algorithm performs well in the X direction and the performance is similar to the previous model. This is because the motion model and filter formulation have not changed in the X direction.

Since ideally, the ball has a constant acceleration in the Y direction, we provide the same value of acceleration in the Y direction as a control input at every step. Since the motion model conforms with the actual scenario, we now provide lesser process noise and more measurement noise in Y direction when compared to the previous model. From figure 7 (b) and (d), the algorithm does a good job before the point of ball bouncing compared to the kalman filter using acceleration only as a process noise. This can be seen by the lesser error difference between the predicted state and actual state between the two models shown in figure 5. After the ball has bounced, the model does a poor job which can be seen by the big error difference after the 6th frame in figure 7 (d). This is because just like the previous model, this model is not able to handle the sudden reversal of velocity direction after the bounce point and the only way the model corrects itself is by following the measurements. If we decrease the measurement noise in Y direction the performance in Y direction becomes similar to the kalman filter with acceleration only as process noise.

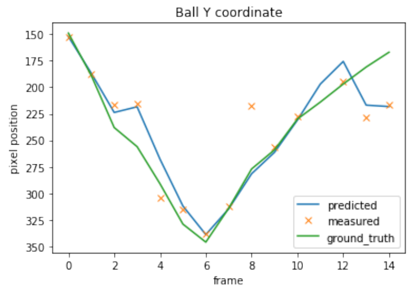
From the performance of the two kalman filter models, we observe that the model being linear performs well in the X direction with constant velocity assumption. But is not able to handle the ball bouncing on the pitch which is a non linearity. One of the solution can be to use an adaptive technique of identifying the bounce point as the point where the error start becoming big and then reset the velocity according to it. Another solution is to use a filtering algorithm which can handle non-linearities which we investigate in the next section.

4.2 Particle filter

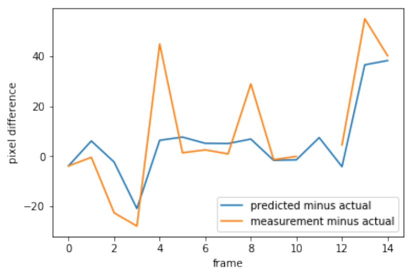
In our implementation of particle filter, we have used 3000 particles. As the number of particles increase, the performance of the algorithm becomes better. This is because more the number of particles, more is the diversity in our hypothesis space. From figure 8, the particle filter is able to perform well in both X and Y directions evident by the low prediction error. In addition, it is able track the bouncing of the ball, where the kalman filter did not perform well enough. This is because, with enough process noise, the particle filter is



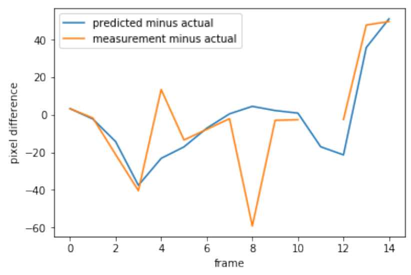
(a) Ball x coordinate vs frame



(b) Ball y coordinate vs frame



(c) Error comparison in x direction



(d) Error comparison in y direction

Figure 8: Results of the particle filter. The particle filter performs better compared to kalman filter demonstrated by the low prediction error in (c) and (d). In Y direction, the particle filter also performs better than the kalman filter on the missing measurement at the 11th frame

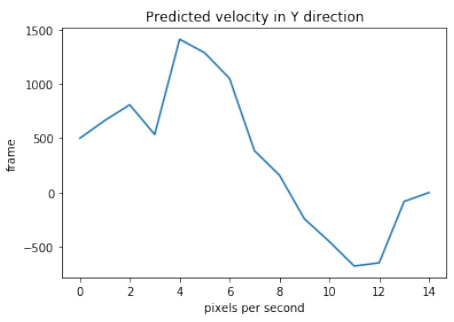
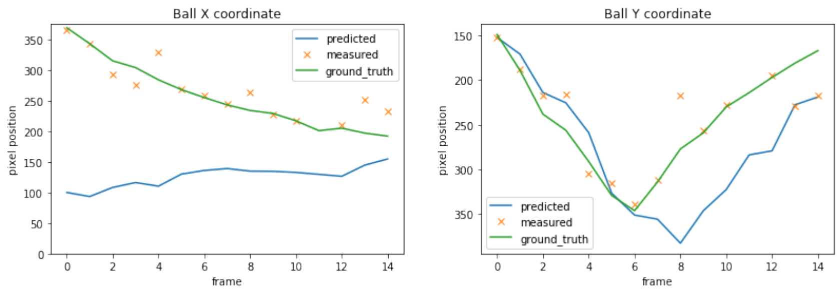


Figure 9: The predicted velocity in Y direction by particle filter. The velocity becomes negative after the 8th frame with a value of -243 pixel/second which demonstrates that the particle filer is able to estimate the reversal of the ball velocity after bouncing.



(a) Ball x coordinate vs frame

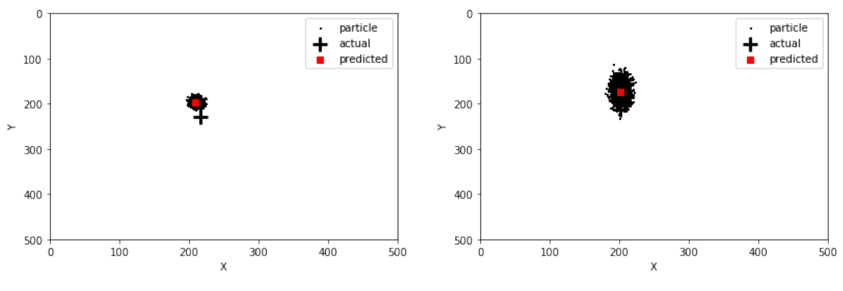
(b) Ball y coordinate vs frame

Figure 10: Bad initialization of initial particles hampers the performance of the particle filter. In this figure the particles were initialized with a value of X coordinate which hugely deviated from the initial measurement.

able to generate enough particles(hypothesis) which can match the true state of the ball. Interestingly, the particle filter is able to determine the point of ball bouncing, demonstrated by the negative velocity at the 8th frame in figure 9. A disadvantage of particle filter is that the algorithm in highly initialization dependent. If the particles are not initialized near the first measurement then poor results are obtained shown in figure 10. This condition can be solved by increasing the process noise such that particles are spread more in the predict stage. The overall error comparison between the particle filter and kalman filter is shown in table 1.

5 Conclusion

In this project, we have investigated the performance of the well known kalman filter and particle filter algorithms in the problem of ball tracking in cricket videos. It can be concluded that the Kalman filter performs well in problems



(a) (b)

Figure 11: Two consecutive iterations of the particle filter algorithm. The weighted mean of the particles is considered as the state prediction.

where non linearity is not involved evident by good performance shown in tracking the cricket ball in horizontal direction. However, our experiments demonstrate that non-linear situations like bouncing off from the ground requires a multi hypothesis adaptive kalman filtering approach or a non linear filtering algorithm such as particle filter. Future work includes using multiple dynamic particle filters to track the ball after it hits the cricket bat, using a better ball detector for better measurements and extending this work to broadcast videos with a moving camera.

References

[1] J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. *IEE Proceedings - Radar, Sonar and Navigation*, 146(1):2–7, Feb 1999.

[2] X. Cheng, N. Ikoma, M. Honda, and T. Ikenaga. Simultaneous physical and conceptual ball state estimation in volleyball game analysis. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4, Dec 2017.

[3] Y. Huang, J. Llach, and C. Zhang. A method of small object detection and tracking based on particle filters. In *2008 19th International Conference on Pattern Recognition*, pages 1–4, Dec 2008.

[4] Rudolf Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering (ASME)*, 82D:35–45, 01 1960.

[5] J. Kim and T. Kim. Soccer ball tracking using dynamic kalman filter with velocity control. In *2009 Sixth International Conference on Computer Graphics, Imaging and Visualization*, pages 367–374, Aug 2009.

[6] Jun S. Liu and Rong Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93:1032–1044, 1998.

[7] N. Owens, C. Harris, and C. Stennett. Hawk-eye tennis system. In *2003 International Conference on Visual Information Engineering VIE 2003*, pages 182–185, July 2003.

[8] R. Roopchand, A. Pooransingh, and A. Singh. Bat detection and tracking toward batsman stroke recognition. In *2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)*, pages 256–260, Dec 2016.

[9] S. Wang, Y. Xu, Y. Zheng, M. Zhu, H. Yao, and Z. Xiao. Tracking a golf ball with high-speed stereo vision system. *IEEE Transactions on Instrumentation and Measurement*, pages 1–13, 2018.

[10] Fei Yan, William J. Christmas, and Josef Kittler. A tennis ball tracking algorithm for automatic annotation of tennis match. In *BMVC*, 2005.

[11] M. A. Zaveri, S. N. Merchant, and U. B. Desai. Small and fast moving object detection and tracking in sports video sequences. In *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*, volume 3, pages 1539–1542 Vol.3, June 2004.

[12] X. Zhou, L. Xie, Q. Huang, S. J. Cox, and Y. Zhang. Tennis ball tracking using a two-layered data association approach. *IEEE Transactions on Multimedia*, 17(2):145–156, Feb 2015.