Determination of Start and end Points of a Flight Trajectory using UD Filter and R-T-S Smoother

V. P. S. Naidu, Girija G. and J. R. Raol

***Abstract*—The** start and end points **of** a flight trajectory are predicted using **UD** Kalman filter and R-T-S smoother. Algorithms are implemented **in PC** MATLAB and validated using simulated data. The R-T-S smoother was found to generate more accurate state estimates, which led to better start point prediction. Start and end point prediction from real data **of** a guided target in ballistic mode **is also** evaluated.

Index Terms—Trajectory estimation, **UD** Kalman . filter, **R-T-S** smoother, Launch and impact point estimation.

I. INTRODUCTION

Many applications in target tracking systems require an accurate extrapolation of flight trajectory using only a segment of the recorded trajectory. The start point information is required to know from where the target was launched and the end point information is required to know where the target would fall. Kalman tracking filters have been used to predict such trajectories. The advantage of Kalman filter is that along with the position estimates, it gives estimates of the accuracy of the predictions. If the entire flight trajectory is observed by sensors, the estimation of the start and end points **is** straightforward and a Kalman filter could be used for the estimation. However, there could be situations where the flight trajectory is available only for a part of its flight. The estimation of end and start points in such a situation would require an extrapolation of the estimated trajectory backward in time to predict the start point and forward in time to predict the end point.

The extrapolation is carried out in **two** phases. In the first phase where the moving object **is** observed by sensors, its trajectory is estimated using point mass model and a Kalman filter [1]. Smoothing techniques significantly improve the initial condition of the state estimates **[2]** and hence smoothed State estimates are obtained using Rauch-Tung-Striebel (R-T-S) smoother during the first phase. In the second phase, where the target is unobservable by sensors, the trajectory is estimated by using point mass model and covariance propagation calculation by forward propagalion to get information about the end point of the target [3]. The smoothed state estimate at the start time of the data is used to predict the start point of the trajectory using backward integration.

In this paper, UD factorization based Kalman filter and forward prediction are used to predict the end point. UD factorization based Kalman filter and fixed interval Rauch-Tung-Streibal (R-T-S) smoother and backward integration are used to predict the start point. The probable use of UD filter is in its application to real-time tracking [4]. Algorithms are implemented in PC MATLAB and validated using simulated data of a target moving with constant velocity The algorithms are also used to predict start and end points from real data of a flight trajectory for which these points are known by considering certain segments of the data as being observable by the sensors. Results are presented in terms of the accuracy of prediction of the start and end points, time history comparisons, autocorrelation of the residuals with bounds, innovation sequence with the theoretical bounds **[5]** and state error with bounds for the simulated data.

II. UD FACTORIZATION BASED KALMAN FILTER AND RTS SMOOTHER

The two phases involved in the estimation of the start point and end point are illustrated in Fig.1. **A** typical trajectory ***of*** a target from **A** to **D is** shown with A as the start point and D as the end point. Assuming that measurements are available only between points B to ***C*** where sensors are tracking the target, it **is** proposed to estimate the end point D and start point **A.** In the 1st phase data between B and ***C*** is used to generate estimated states and covariances using UD filter. An R-T-S smoother working between points **B** and ***C*** utilizing the outputs of the Kalman filter generates smoothed states in a backward pass (i.e. going from C to B) operation. In the 2nd phase, point D is obtained by using the forward prediction of the **UD** filter output at point C and point **A** is estimated by backward integration starting with the smoother output at B.

1) UD Kalman filter

The general form of the state model of the tracked target is given by

$$x(k+1) = \Phi\, x(k) + G w(k) \qquad (1)$$

with discrete measurements

$$z(k+1) = H\, x(k+1) + v(k+1),$$

V P. S. Naidu, Girija. G, and Raol J. R, are with the Flight Mechanics and Control Division, National Aerospace Laboratories, Bangalore, INDIA. (e-mail: vpsn@css.nal res.in)

$k = 0,1,2,...,N$ (2)

where, x = State vector, z = Measurement vector, Φ = State transition matrix, G = Process noise related matrix, H = Observation mutrix. v = Measurement noise, w = Process noise and N = Number ofmeasurements. It is assumed that the noisc sequences $\{w(k)\}$ and $\{v(k)\}$ are zero mean white Gaussian and are mutually independent and have diagonal covariances $Q\,(=\sigma_w^2)$ and $R\,(=\sigma_v^2)$ respectively. The choice of G assures that Q being diagonal is not a loss of generality and by Cholesky decomposition whitening, it can be assumed with no loss of generality that R is diagonal [4].

The term "U-D covariance factorization" comes from a property of non-negative definite symmetric matrices, according to which covariance matrix 'P' can be factored into $P = UDU^T$, where 'U' is the upper triangular matrix with unit elements on its main diagonal and 'D' is a diagonal matrix [4]. Covariance and gain processing algorithms, operating on 'U' and 'D' factors of state error covariance matrix $\mathbf{P}$, are a technique for implementing "square root filtering" without requiring computation of square roots. The U-D Kalman filtering algorithm is considered efficient, stable and accurate for real-time applications [4].

The UD factorization based Kalman filtering algorithm is given in two parts namely Time propagation and Measurement update algorithms:

2) U-Dfactor Time Propagation Algorithm

State estimate extrapolation:

$$\tilde{x}_{k+1/k} = \Phi \hat{x}_{k/k} \tag{3}$$

Error covariance extrapolation:

$$\tilde{P}_{k+1/k} = \Phi \hat{P}_{k/k} \Phi^T + GQG^T \tag{4}$$

Given $\hat{P} = \hat{U}\hat{D}\hat{U}^T$ and Q as the process noise covariance matrix, the time update factors $\tilde{U}$ and $\tilde{D}$are obtained through modified Gram-Schmidt orthogonalization process. Defining $W = [\Phi \hat{U} \mid G_A]$, $\bar{D} = diag[\hat{D}, Q]$ with $W^T = [w_1, w_2, ..., w_n]$, where '$n$' is the number of states in state vector. P is reformulated as $\tilde{P} = \tilde{W}\bar{D}\tilde{W}^T$. The U and D factors of $\tilde{W}\bar{D}\tilde{W}^T$ may be computed as described below. For $j = n, n-1, ..., 2$ the following equations are recursively evaluated.

$$\begin{aligned}
\tilde{D}_j &= < w_j^{(n-j)}, w_j^{(n-j)} >_{\bar{D}} \\
\tilde{U}(i,j) &= < w_i^{(n-j)}, w_j^{(n-j)} >_{\bar{D}} / \tilde{D}_j \\
w_i^{(n-j+1)} &= w_j^{(n-j)} - \tilde{U}(i,j) w_j^{(n-j)}, \quad i = 1,...,(j-1) \\
\tilde{D}_1 &= < w_1^{(n-1)}, w_1^{(n-1)} >_{\bar{D}}
\end{aligned} \tag{5}$$

I) *U-D Factor Measurement Update Algorithm*

The measurement update in Kalman filtering combines a priori estimate $\tilde{x}$ and error covariance $\tilde{P}$ with scalar observation $z = a^T x + v$; $a^T = H$ to construct an updated (filtered state) estimate and covariance:

$$\begin{aligned}
K &= \tilde{P}_{k+1/k} a / \alpha \\
\hat{x}_{k+1/k+1} &= \tilde{x}_{k+1/k} + K(z(k+1) - a^T \tilde{x}_{k+1/k}) \\
a &= a^T \tilde{P}_{k+1/k} a + r \\
\hat{P}_{k+1/k+1} &= \tilde{P}_{k+1/k} - K a \tilde{P}_{k+1/k}
\end{aligned} \tag{6}$$

where $k = N_B, ..., N_C$, $\tilde{P} = \tilde{U}\tilde{D}\tilde{U}^T$, 'a' is the measurement vector, 'r' is the measurement noise covariance, 'z' is the string of noisy measurements, N_B and N_C denote start and end of the data measurements as available from the tracking sensors.

Kalman gain K and updated covariance factors $\hat{U}$ and D can be obtained from the following equations:

$$\begin{aligned}
f &= \tilde{U}^T a; \quad f^T = (f_1, ..., f_n) \\
v &= \tilde{D} f; \quad v_i = \tilde{d}_i f_i \quad i = 1,2,...,n \\
\hat{d}_1 &= \tilde{d}_1 r / \alpha_1; \quad \alpha_1 = r + v_1 f_1; \quad K_2^T = (v_1\ 0 ... 0)
\end{aligned} \tag{7}$$

For $j = 2,...,n$ recursively the following equations are evaluated:

$$\begin{aligned}
\alpha_j &= \alpha_{j-1} + v_j f_j; \quad d_j = d_j \alpha_{j-1} / \alpha_j \\
\hat{u}_j &= \tilde{u}_j + \lambda_j k_j; \quad \lambda_j = -f_j / \alpha_{j-1} \\
K_{j+1} &= K_j + v_j \tilde{u}_j
\end{aligned} \tag{8}$$

where $\tilde{U} = [\tilde{u}_1, ..., \tilde{u}_n]$, $U = [\hat{u}_1, ..., \hat{u}_n]$ and Kalman gain is given by $K = K_{n+1} / \alpha_n$ where $\tilde{d}$ is predicted diagonal element and d_j is the updated diagonal element of the D matrix.

4) R - T - S (Rauch. Tung and Streibal) Smoother

Smoothing is a non-real time data processing scheme that uses all measurements starting from T to 0 to estimate the state of a system at a certain time t, where $0 \le t \le T$. The smoothing algorithm implemented here is a backward-pass sequel to a forward pass Kalman filter outputs (filter covariances, Kalman gains etc). The R-T-S smoother formulation is considered. because of the simplicity [2,4]. The R-T-S recursions that generate smoothed estimates and error covariances are

$$x_{k/N}^* = \hat{x}_{k/k} + G_k^* (x_{k+1/N}^* - \hat{x}_{k+1/k}) \tag{9}$$

$$P_{k/N}^* = \hat{P}_{k/k} + G_k^* (P_{k+1/N}^* - \tilde{P}_{k+1/k}) G_k^{*T} \tag{10}$$

Where

$$G_k^* = \hat{P}_{k/k} \Phi_k^T \tilde{P}_{k+1/k}^{-1} \tag{11}$$

The recursion is backward sweep from

$k = (N_C - 1),...,N_B$. The state estimates and covariances $\tilde{x}_{k+1/k}$ and $\tilde{P}_{k+1/k}$ are found using 'the UD filter in the forward pass.

5) End point prediction

Forward prediction is used to predict the end point D (whose index is N **j**

$$\tilde{x}_{k+1/k} = \Phi \hat{x}_{k/k} \quad (12)$$

$$\tilde{P}_{k+1/k} = \Phi \hat{P}_{k/k} \Phi^T, \; k = N_C,...,N \quad (13)$$

The initial value of $\hat{x}_{k/k}$ is the output of the UD filter at point C.

6) Start point prediction

Backward integration was carried out to predict the start point.

$$x^*_{k/N} = \Phi x^*_{k+1/N} \quad (14)$$

$$P^*_{k/N} = \Phi^{-1} P^*_{k+1/N} (\Phi^T)^{-1}, k = (N_B - 1),...,1 \quad (15)$$

The initial value of $x_{k+1/k}$ is the smoother output at point B or UD filter output at point B.

7) Filter performance evaluation

The filter performance is ascertained by checking (i) the estimated states and the bounds for convergence (ii) residuals and their bounds for convergence and (iii) the autocorrelation of the residuals for whiteness. The performance of UD filter has been evaluated by checking whether

i. The state error $(\mathbf{x} - \hat{x})$ falls within the theoretical bound of $\pm 2\sqrt{\hat{P}_{(k,k)}}$, where $\hat{P} = \hat{U}\hat{D}\hat{U}^T$ (for simulated data only)

ii. The innovation $\varepsilon = z - H\tilde{x}$ falls within the theoretical bound of $\pm 2\sqrt{S_{(k,k)}}$, where $S = H\tilde{P}H^T + R$ and

iii. The autocorrelation of residuals fall within the theoretical bound of $\pm 1.96/\sqrt{N}$, where 'N is the number of samples.

III. RESULTS AND DISCUSSION

The UD filter and smoother techniques are implemented in PC MATLAB. The performance is evaluated using simulated data of a target moving with constant velocity. It is then applied lo real data for extrapolating the estimated/smoothed trajectory during a segment of the flight trajectory.

1) Validation using simulated data

For generating the simulated data, the following state variables of the target are considered: x-position. x-velocity, y-position and y-velocity i.e. the state vector is represented by $[x, \dot{x}, y, \dot{y}]$. Equations (1) and (2) give the model of the system considered. Sampling time $\Delta T = 0.1$ Sec is used to simulate data for 50 secs. resulting in a total of 500 data points. The transition matrix and the other related matrices are given by:

$$\Phi = \begin{pmatrix} 1 & \Delta T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta T \\ 0 & 0 & 0 & 1 \end{pmatrix} G = \begin{pmatrix} \Delta T^2/T & 0 \\ 0 & \Delta T \\ \Delta T^2/T & 0 \\ 0 & \Delta T \end{pmatrix} R = \begin{pmatrix} r_x^2 & 0 \\ 0 & r_y^2 \end{pmatrix}$$

$$ti = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad Q = \begin{pmatrix} q_x^2 & 0 \\ 0 & q_y^2 \end{pmatrix} \quad (16)$$

The initial state vector x_o = [1200, 8.3, 3000, 10.61. Measurements of x-position and y-position are generated using equation **(2)** by adding random noise with variance of $r_x = 1$ and $r_y = 100$ respectively. The process noise variance used in the simulation is $q_x = q_y = 0.01$. The results of estimation are the average of 10 Monte Carlo simulations. Fig. **2** shows the x and y position state estimates using the data between the points B and ***C*** (in the left half of the figure) assuming that measured data is available between 10 sec. and 40 sec. The state estimates with prediction of start point (of point A) using the smoother output as well as the filter output and end point (point D) using the filter output are compared with the true states in the right half of the Fig. **2.** It is clear from the figure that the start point prediction using UD filter is away from the true value. However, the smoother gives an accurate start point prediction. The innovations state errors and autocorrelation of residuals with hounds are shown in Fig. 3. The innovations, residuals and auto correlation function fall within the theoretical bounds, which indicate satisfactory filter performance. The results of end point prediction using the UD filter output at point C is shown in Table 1. **A** number of cases are presented where it is assumed that measured data is available for different time segments (indicated by points B and C in the Tables). Results are presented in terms of the end point prediction error; the standard deviation of the end point prediction, the mean of the residuals and the fit error. It is clear that even with only 10 secs of data (case ***6),*** the end point prediction is very accurate thereby proving that when the model of the target and the noise statistics are known accurately, the prediction of the end point using UD filtered output is accurate.

The results of start point prediction using the UD filter state estimate at point B and the smoother output at point B are presented in Table **2.** It is clear that in all the cases, the use of the smoother results in prediction with bener accuracy than with the use of the filter alone. Also, it is clear from Table 2 that the start point prediction accuracy degrades considerably when smaller segments of data are observed by the sensors. However, use of UD filter with smoother is able to overcome this problem in start point prediction accuracy, which is clear from Table **3.** The simulated data of a target moving with constant velocitv was chosen for validation of the technique/algorithm. The application of this technique to real data could involve nonlinear state equations & maneuvering targets. This problem would require an extensive modeling effect.

2) Application to Real data

The UD Kalman filler and smoothing algorithm are used to predict **the** start and end points ofa guided target trajectory in ballistic mode. The start point and end point for this data

arc known. In order to evaluate the accuracy of predictions when only a part of the trajectory is tracked by the sensors, a number of segments of the data are considered. The trajectory data is measured in terms of x, y and z positions in Cartesian frame of reference. The state variables of the start vehicle are X co-ordinate position (x), velocity ($\dot{x}$) and acceleration ($\ddot{x}$), Y co-ordinate position (y), velocity ($\dot{y}$) and acceleration ($\ddot{y}$), Z co-ordinate position (z), velocity ($\dot{z}$) and

$$\begin{bmatrix} 1 & \Delta T & \frac{\Delta T^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta T & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta T & \frac{\Delta T^2}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta T & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta T & \frac{\Delta T^2}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta T \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

on trial and error using post flight measurement data.

Fig. 6 shows the x, y and z-position state estimations using the data between the points B and C (in left half of the figure) assuming that the measured data is available between 10 sec and 110 sec. The state estimates with prediction of start point (point A) using the smoother output and end point (point D) using the filter output are compared with the true states in the right half of the Fig. 6. Tables 3 gives the quantitative results of start point and end point estimation. Figs 4-5 show the innovations with 2σ bounds and autocorrelation of residuals with bounds. PIOB (Percentage Innovation Gut of Bounds), PAOB (Percentage Autocorrelation Out of Bounds) are close to the acceptable theoretical limit of 5%, except for PAOB in z-position. This could be because of the maneuver observed in the z-position. The autocorrelation function is well within the theoretical bounds. However, it also indicates that the measurement noise statistics may not be white and hence inclusion of a model for this may improve the results. This is being investigated. For this set of real data. it was observed that when the measurement data length observed by the sensors was less than 115 secs, the start point estimation led to unacceptable results. The results indicate higher errors in end point prediction in the z direction because of the maneuver in that axis and if the measurements are missing during this portion, the prediction accuracy degrades considerably. Also, the start point error is higher than the end point error.

The results for real data are for the case when the target is in the ballistic mode. In general when any target is launched, it is not in the ballistic mode and certain other factors like thrust and drag must be considered to get accurate estimates of both start and end points. Thus estimation of start point would require further study including these effects.

IV. CONCLUSIONS

UD Kalman filter and R-T-S smoother were implemented in PC MATLAB and their performance studied using simulated data. The R-T-S smoother was found to generate more accurate state estimates, which lead to better start point prediction accuracies. End point and start point prediction from real data of a guided target in ballistic mode was also evaluated. For real data, the inaccuracies in both start point and end point predictions can be reduced using improved models that account for drag and atmospheric effects. This requires further study. It will be worthwhile to explore the application of the presented approach to the cases where the state equations are highly nonlinear and the data spans are shorter.

V. REFERENCES

[1] Eli Brookner, Tracking and Kalman filtering made easy, Jhon Wiley & Sons, New York, 1998.

[2] Arthur Gelb. Applied Optimal Estimation, seventh Edition, The M.I.T press. Massachusetts, 1982.

[3] Candillo. G.P, Mrstik. A V, Plambeck. Y, A track filter lor reentry objects with uncertain drag. IEEE Transactions on Aerospace and Electronic system. Vol. 35. No.2, April 1999.

[4] Bierman G.J, A new computationally efficient fixed - interval, discrete-limesmoother. Automatica. Vol.19, No.5,pp 505-511. 1983.

[5] James V.Candy, Signal processing, The model based approach. Second printing. McGraw-Hill International Edition, Singapore. 1987.

[6] J.R. Raol and Girija G , Evaluation of adaptive Kalman filtering methods for target tracking applications. AIAA Guidance. Navigation and Control Conference and Exhibit, Montreal, Canada. Paper No. AIAA 2001-4106, 6-9 Aug 2001

TABLE I

Case	Time at B	Time at C	x-pre	x-std	%xerr	y-pre	y-std	%yerr	mx	my	PFE x	PFE y
1	0 sec.	50 sec.	1613.8	0.20	-0.002	3528.6	1.19	0.006	0.0019	-0.13	0.068	0.298
2	0.5 sec.	49.5 sec.	1613.7	0.23	0.005	3528.4	1.23	0.01	-0.006	-0.375	0.069	0.299
3	2.5 sec.	47.5 sec.	1613.8	0.34	-6E-04	3528.4	1.41	0.009	0.0307	0.762	0.068	0.299
4	5 sec.	45 sec.	1613.5	0.50	0.019	3529.1	1.67	-0.01	0.0395	0.391	0.068	0.295
5	10 sec.	40 sec.	1613.5	0.82	0.017	3530.0	2.28	-0.035	0.0652	1.036	0.068	0.291
6	20 sec.	30 sec.	1613.5	0.82	0.015	3531.6	2.73	-0.08	0.1044	0.454	0.064	0.277

END POINT PREDICTION USING UD FILTER – SIMULATED DATA (POINT-D (X-TRUE=1613.77, Y-TRUE=3528.77), AT 50SEC)

x, y-pre = predicted x, y-position, Σ y-std = standard deviation of x, y-position prediction, x, y-err = x, y-position error =((x, y-true) – (x, y-pre)), %x, yerr = (x, y-err * 100)/(x, y-true), m Σ my = mean of Σ y residuals, PFE x, y = Percentage fit error in x, y-position.

TABLE II

START POINT PREDICTION USING UD FILTER & USING SMOOTHER – SIMULATED DATA (POINT-A (X-TRUE=1200.83,Y-TRUE=3001.06) AT 0SEC)

			UD filter						Smoother					
Case	Time at B	Time at C	x-pre	x-std	%xerr	y-pre	y-std	%yerr	x-pre	x-std	%xerr	y-pre	y-std	%yerr
1	0 sec.	50 sec.	1200.4	1	0.03	2998	9.0	0.09	1199.9	0.2	0.08	2999.1	1.2	0.01
2	0.5 sec.	49.5 sec.	1200.3	2.6	0.04	2966.6	34.5	1.15	1199.9	0.2	0.01	2999.2	1.2	0.06
3	2.5 sec.	47.5 sec.	1238.4	16.2	-3.13	3186	103.4	-6.16	1200.4	0.3	0.04	3001.3	0.7	-0.01
4	5 sec.	45 sec.	1313.3	43.9	-9.36	3092	306	-3.03	1200.6	0.5	0.02	3000.2	1.6	0.03
5	10 sec.	40 sec.	1258.7	62.9	-4.82	4381.5	1049	-45.9	1199.8	0.8	0.08	3000.8	2.3	0.01
6	20 sec.	30 sec.	1357.1	104	-13	1569.7	1085	47.7	1200.6	1.4	0.02	2996.7	4.1	0.12

TABLE III

PREDICTION OF END POINT USING UD FILTER & START POINT USING SMOOTHER - REAL DATA

			UD filter						Smoother		
Case	Time at B	Time at C	x-std	PFE x	y-std	PFE y	z-std	PFE z	x-std	y-std	z-std
1	0 sec	125 sec	31.31	0.05	31.313	0.03	31.31	0.27	21.12	21.12	21.12
2	2.5 sec	122.5 sec	102.65	0.22	102.65	0.21	102.65	0.91	72.76	72.76	72.76
3	5 sec	120 sec	232.99	0.15	232.99	0.25	232.99	0.48	174.9	174.9	174.9
4	10 sec	115 sec	673.12	0.24	673.12	0.22	673.12	0.66	534.1	534.1	534.1

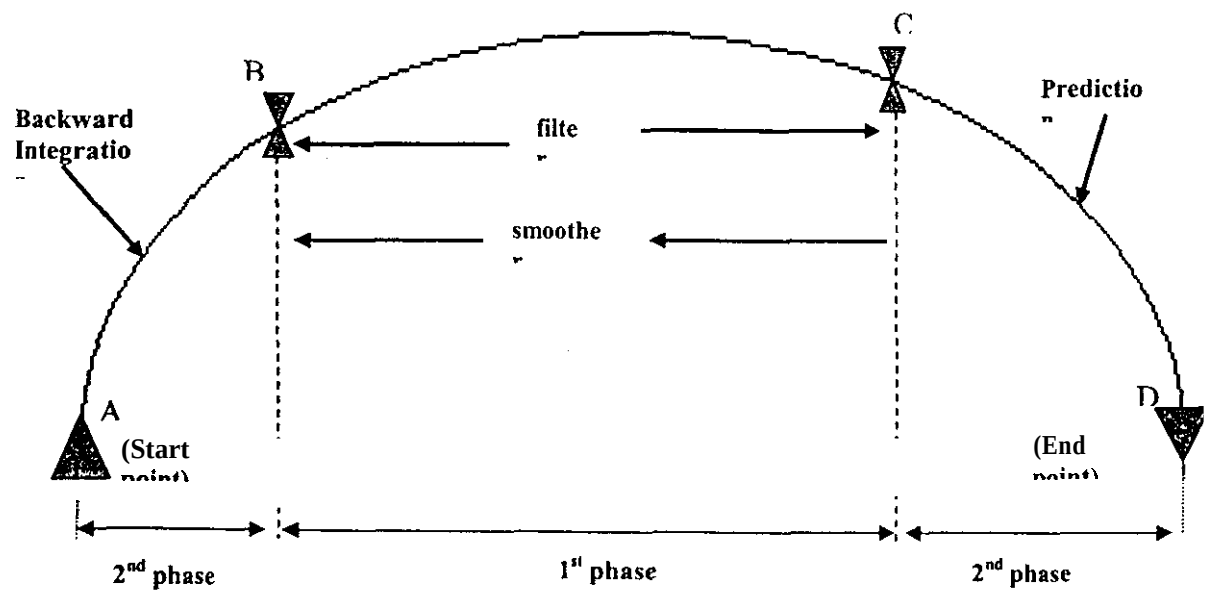

Fig. 1 Typical target trajectory

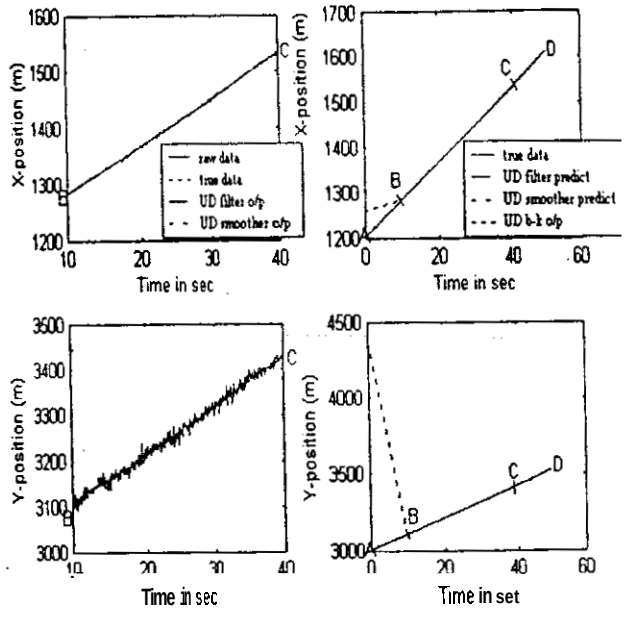

Fig. 2 Estimated states of both filter and smoother

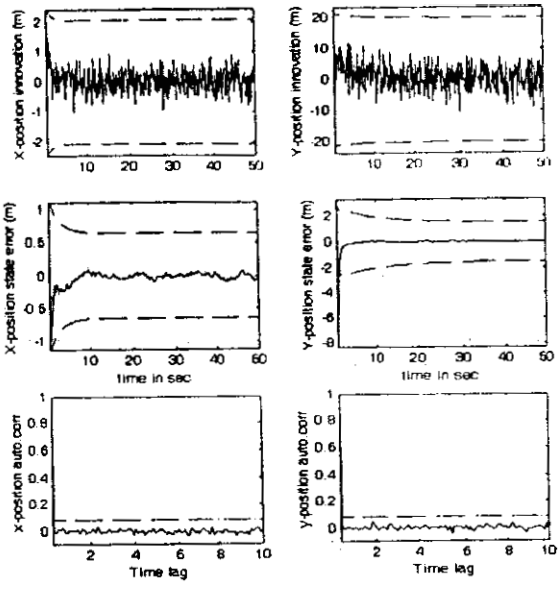

Fig. 3 Innovation, state error and autocorrelation with respective bounds

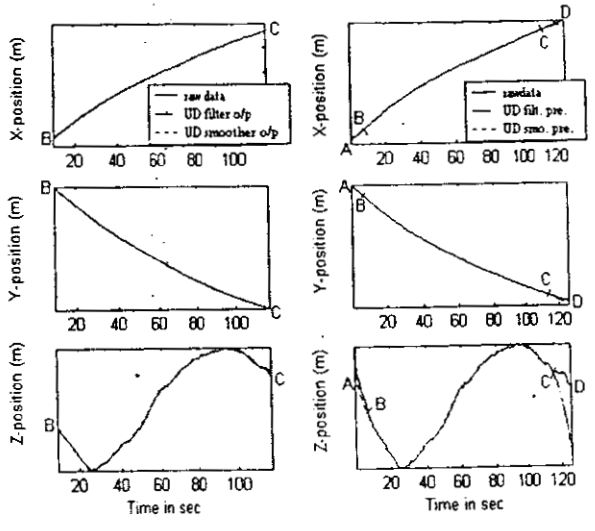

Fig. 4 x, y and z-position state estimations

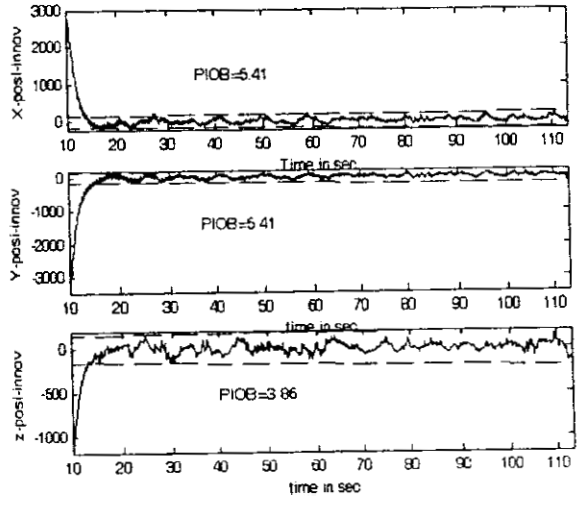

Fig. 5 Innovation with bounds

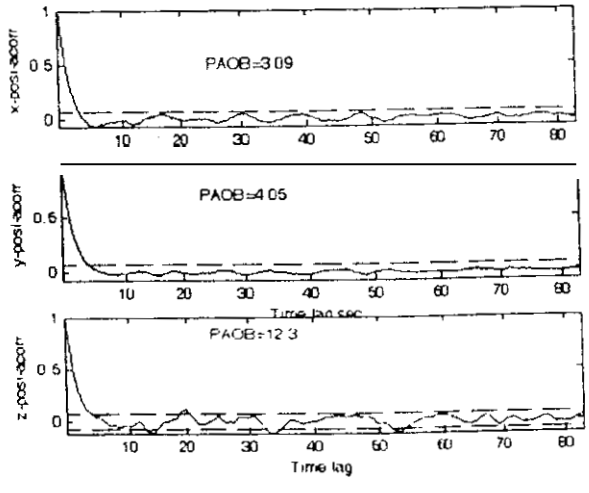

Fig.6 Autocorrelation with bounds