N. Shanthakumar, Girija G and Raol J.R.
Scientists, System Identification Laboratory
Flight Mechanics and Control Division
National Aerospace Laboratories-Bangalore-560017
e-rnail: jrraol@csstmmacs.ernet.in

Abstract

In this paper factorization filtering algorithms are described and used for processing data from a typical flight test range. Specifically U-D (unit upper triangular-diagonal) factorization based Kalman filtering algorithms are considered. The algorithms are validated using simulated data and implemented in MATLAB and alpha DEC machines. UDP protocols are used to transfer data from one DEC machine to another where the UD filter algorithm is activated *to* process the data. A very brief description of the fusion scheme in which the UD filtering algorithms are being used is given.

1. Introduction

Tracking is the process of obtaining values of thdse parameters which completely specify the motion of a vehicle based on the measurements [1] These measurements could be obtained from radar, electro-optical devices (EOD), sonar etc The measurements are processed using an optimal estimation algorithm. 'A precision track &termination estimator takes the measurement noise into account and determines the track that provides a "best *fit*" to the collected data. Major constituents of a track determination process are : vehicle's (system) kinematic motion model, measurements and estimation techniques. For a flight test range the tracking of a flight vehicle and sensor fusion are of great importance. In this paper, UD-factorization filtering algorithm, which directly handles bias parameters and correlated process noise in addition to the state estimates, is described. The algorithms are implemented in PC MATLAB and C language (in alpha DEC computer). The algorithms are validated using simulated and real data. Using socket-programming features the UDP protocol for data communication has also been established between two alpha DEC computers.

2. Tracking Filter

Kalman filter has found very wide application in tracking problems because of its optimal and recursive features. The conventional Kalman filter algorithm is not numerically robust due to round-off errors, and numerical accuracy can degrade to the point where results cease to be meaningful. Typical problems that could occur are loss of positive-definiteness of the covariance matrix resulting from numerical errors such as finite word length computations and cancellation errors due to the subtraction term in the Kalman covariance update. Square-root filter formulations offer a solution to this problem of numerical accuracy and hence stability of the filter. The improved numerical behavior of square root algorithms is due to a reduction of the numerical ranges of the variables. In the UD filter [2], the covariance update formulae and the estimation recursions are

reformulated so that the covariance matrix does not appear explicitly. Specifically recursions are used for **U** and **D** factors of the covariance matrix. Computing and updating with triangular and diagonal matrices involve fewer arithmetic operations. The algorithm processes measurements, one component at **a** time.

2.1 Mathematical Model

A mathematical model of the vehicle state includes position, velocity and sometimes acceleration as state variables. The measurement model relates state variables to available measurement variables. For simplicity the states of motion are defined in the spherical coordinates such that the state equation can be decoupled into three independent channels. Then the tracking filter can work independently on each channel. The problem addressed in this paper is represented **by** the following set of equations:

State Model: $x_{j+1} = \phi_{j+1,j} x_j + G w_j$ (1)

Measurement Model:

$$z_j = H x_j + v_j \quad (2)$$

Here. x is the state vector, w is the process noise with zero mean and covariance matrix Q. z is the measurement vector and v is the measurement noise with zero mean and covariance matrix R. all of appropriate dimensions.

2.2 U-D factorization Kalman filter

The tracking filter **is** implemented in the factorized form.

Time Propagation Algorithm

State vector evolution (prediction)

$$\bar{x}_{j+1} = \phi_{j+1,j} \mathbf{x}_j \quad (3)$$

Covariance update

$$\tilde{P}_{j+1} = \phi \hat{P}_j \phi^T + GQG^T \quad (4)$$

With $\hat{P} = \hat{U}\hat{D}\hat{U}^T$ and Q, the time update factors $\tilde{U}$ and $\tilde{D}$ are obtained through modified Gram-Schmidt orthogonalization process [2]. The matrix **U** is an upper triangular matrix with unit elements on its main diagonal and D is a diagonal matrix. Covariance and gain processing algorithms. operating on **U** and **D** factors of state error covariance matrix P. are a technique for implementing "square root filtering" without requiring cbmputation of square roots. The U-D Kalman filtering algorithm is considered efficient, stable and accurate for real-time applications [2].

We define $W = [\phi\hat{U} \mid G]; \quad D = diag[\hat{D}, Q]$ with $W^T = [w_1, w_2, \ldots, w_n]$,

The U,D factors of $\tilde{P} = WDW^T$ may be computed now. For j= n, n-1,..., 2, the following equations are recursively evaluated as shown below:

$$\begin{aligned}
\tilde{D}_j &= < w_j^{(n-j)}, w_j^{(n-j)} >_D \\
\tilde{U}(i,j) &= < w_j^{(n-j)}, w_j^{(n-j)} >_D / \tilde{D}_j \qquad i = 1,\ldots,j-1 \\
w_i^{(n-j+1)} &= w_i^{(n-j)} - \tilde{U}(i,j) w_j^{(n-j)} \\
\tilde{D}_1 &= < w_1^{(n-1)}, w_1^{(n-1)} >_D
\end{aligned} \qquad (5)$$

Here subscript D qualifies the weighted inner product with respect to D.

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 as follows:

$$\begin{aligned}
K &= \tilde{P}a/\alpha, \\
\hat{x} &= \tilde{x} + K(z - a^T\tilde{x}), \\
\alpha &= a^T\tilde{P}a + r \\
\hat{P} &= \tilde{P} - Ka\tilde{P}
\end{aligned} \qquad (6)$$

Here, r is the measurement noise variance (for scalar data processing). Gain K and updated covariance factors $\hat{U}$ and $\hat{D}$ can be obtained from the following equations:

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

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

$$\begin{aligned}
\alpha_j &= \alpha_{j-1} + v_j f_j \\
\hat{d}_j &= \tilde{d}_j \alpha_{j-1}/\alpha_j \\
\hat{u}_j &= \tilde{u}_j + \lambda_j K_j; \qquad \lambda_j = -f_j/\alpha_{j-1} \\
K_{j+1} &= K_j + v_j\tilde{u}_j
\end{aligned} \qquad (8)$$

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

2.3 UD Filter for Correlated Process Noise and Bias Parameters

A mathematicat model representing a trajectory is usually not exact (even in a statistical sense) and the process noise is hardly a strictly white noise process. For this case the state model is given as:

$$\begin{bmatrix} x \\ p \\ y \end{bmatrix}_{j-1} = \begin{bmatrix} V_x & V_p & V_y \\ 0 & M & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} x \\ p \\ y \end{bmatrix}_j + \begin{bmatrix} 0 \\ w_j \\ 0 \end{bmatrix} \tag{9}$$

Here x is the state vector, **p** is the (state) variable representing the correlated noise and y is the bias vector. The transition matrix is almost triangular. The mapping (time propagation) of the U-D factors is carried *out* using the following equivalence [2]:

A priori factors:

$$\begin{bmatrix} \hat{U}_x & \hat{U}_{xp} & \hat{U}_{xy} \\ 0 & \hat{U}_p & \hat{U}_{py} \\ 0 & 0 & \hat{U}_y \end{bmatrix}; \quad \hat{D} = diagonal(\hat{D}_x, \hat{D}_p, \hat{D}_y) \text{ all at } \textbf{time } j$$

$$[\bar{U}_p \bar{U}_{py} \bar{U}_y \bar{D}_p \bar{D}_y] = [\hat{U}_p \hat{U}_{py} \hat{U}_y \hat{D}_p \hat{D}_y]$$

$$\bar{U}_{xp} = V_x \hat{U}_{xp} + V_p \hat{U}_p \; ; \; \bar{U}_{xy} = V_x \hat{U}_{xy} + V_p \hat{U}_{py} + V_y \hat{U}_y$$

$$\bar{U}_x \bar{D}_x \bar{U}_x^T = (V_x \hat{U}_x) \hat{D}_x (V_x \hat{U}_x)^T$$

$$\tilde{U}_Y = \bar{U}_Y = \hat{U}_Y \; ; \; \tilde{D}_Y = \bar{D}_Y = \hat{D}_Y$$

$$\tilde{U}_{py} = M\bar{U}_{py} = M\hat{U}, \; ; \quad \tilde{U}_{xy} = \bar{U}_{xy}$$

$$\begin{bmatrix} \tilde{U}_x & \tilde{U}_{xp} \\ 0 & \tilde{U}_p \end{bmatrix} \begin{bmatrix} \tilde{D}_x & 0 \\ 0 & \tilde{D}_p \end{bmatrix} \begin{bmatrix} \tilde{U}_x^T & 0 \\ \tilde{U}_{xp} & \tilde{U}_p^T \end{bmatrix} = \begin{bmatrix} \bar{U}_x & \bar{U}_{xp} \\ 0 & M\bar{U}_p \end{bmatrix} \begin{bmatrix} \bar{D}_x & 0 \\ 0 & \bar{D}_p \end{bmatrix} \begin{bmatrix} \bar{U}_x^T & 0 \\ U_{xp}^T & \bar{U}_p^T M^T \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & Q \end{bmatrix} \tag{10}$$

The above updating is mechanized by using the modified Gram-Schmidt orthogonalization algorithm. The factors related *to* correlated process noise and the bias are mapped as final factors using the modified Gram-Schmidt algorithm *to* get:

$$\tilde{U}\tilde{D}\tilde{U}^T = W \; Diagonal(D, Q) W^T$$

$$W = \begin{bmatrix} \bar{U}_x & \bar{U}_{xp} & 0 \\ 0 & M\bar{U}_p & I \end{bmatrix}$$

3. Sensor Fusion Scheme [3]

In order to be able to. use sensor-channels such as EOT, PCMC, S-Band. two TM, RADAR 1, RADAR 2, INS, and GPS for fusion, it is necessary to develop fusion logic to use the information from these sensors, Fig. 1. The priority logic is decided based on the sensor accuracy within the range of the sensor capability. Based on this priority logic the following sequence could be given to the sensors (of the first module) within the range limit of, say RL km.

- EOT
- PCMC
- TM and S-Band fusion
- S-Band
- TM
- Track loss

For range more than RL km PCMC radar tracks the vehicle. For the second module which contains the RADARs, the following sequence is followed

- RADAR 1 and RADAR 2 fusion
- RADAR1
- RADAR2
- Track loss

For the third module which contains the INS and GPS, the following sequence is followed

- INS and GPS fusion (GPS data replaced by INS)
- INS
- GPS (GPS data replaced by INS)
- Track loss

The U-D filters described in the previous section are being used in the above sensor fusion scheme for tracking of a vehicle and data fusion.

4. Results and Discussions

Simulated data with correlated process noise have been generated. The transition matrices used are given by:

$$V_x = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \; ; M_j = \exp(-\Delta t/\tau) \quad ; \; \tau = 0 \; to \; \infty \tag{11}$$

The states are position and velocity. A constant bias has been added to the position data so that V_y=1. At is the sampling interval equal to 0.5 sec and the value of M is 0.9, indicating high correlation. The UDF (without accounting for correlated noise and bias) and the UDFCPNB (accounting for correlated process noise and bias) are used to process these data using vehicle position only as the observable. Fig. 2 shows the

comparison of the state estimates obtained by these filters. From the results of state errors (difference of true and estimated state), it is seen that the UDFCPNB performs better than UDF when the data is contaminated by correlated noise. This is also supported by the results of Table 1.

The real data of a flight vehicle (in Cartesian coordinates X,Y,Z) are processed using both the UD filters. The transition matrix of the state eqn. (9) is given as:

$$V_x = \begin{bmatrix} 1 & \Delta t & \Delta t^2/2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \qquad (12)$$

and $H=[1\ 0\ 0]$. State vector x has position, velocity, and acceleration as its components. The figs. 3 and 4 show the results of this analysis. From fig. 3 it is seen that the performance of both the filters is grossly similar. However, from fig. 4 and Table 2 it can be inferred that the performance of UDFCPNB is better than the UDF.

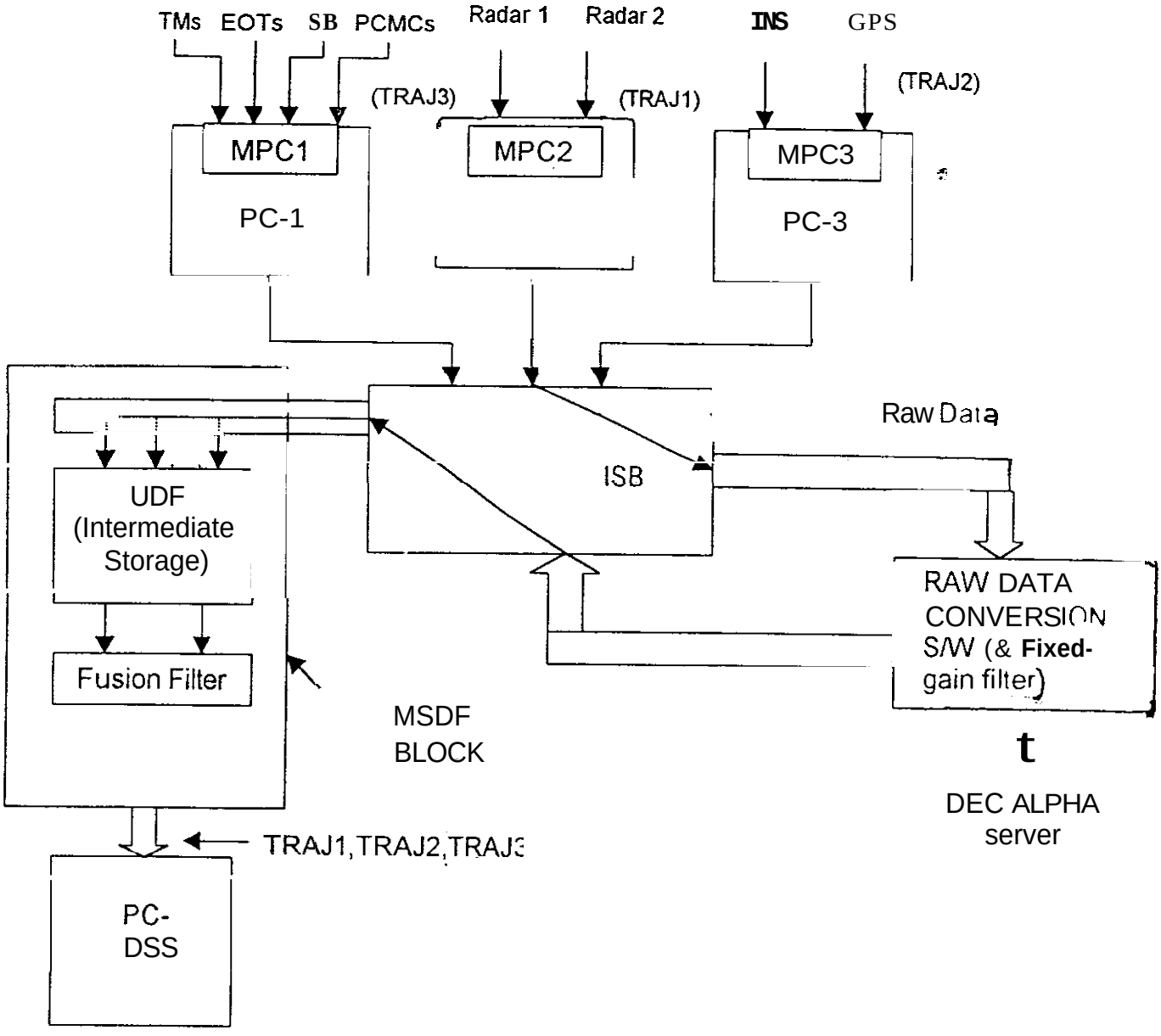
5. Concluding Remarks

	Mean	
	Position	velocity
UDF	**-8.4**	**-17.211**
UDFCPNB	**0.8489**	**1.68**

Table 2: Results of Real Data- Mean values of residuals*

	Mean of Residuals		
	X	Y	Z
UDF	-7.2* (0·88)	8.6 * (1·51)	-9.2* (1·75)
UDFCPNB	0.66* (0.03)	0.94* (0·18)	6.4* (0·2)

* w.r.t. estimated state
(·) w.r.t. predicted measurement.



UD F	U-D Filter	PC	Pesonal Computer
UDFF	U-D Fusion Filter	TM	Telemetry
MSDF	Multi Sensor Data Fusion	SBR	Side Band Radar
ISB	Intelligent Switch Board	DSS	Decision Support System

Fig. 1 Block diagram of data acquisition and tracking/fusion

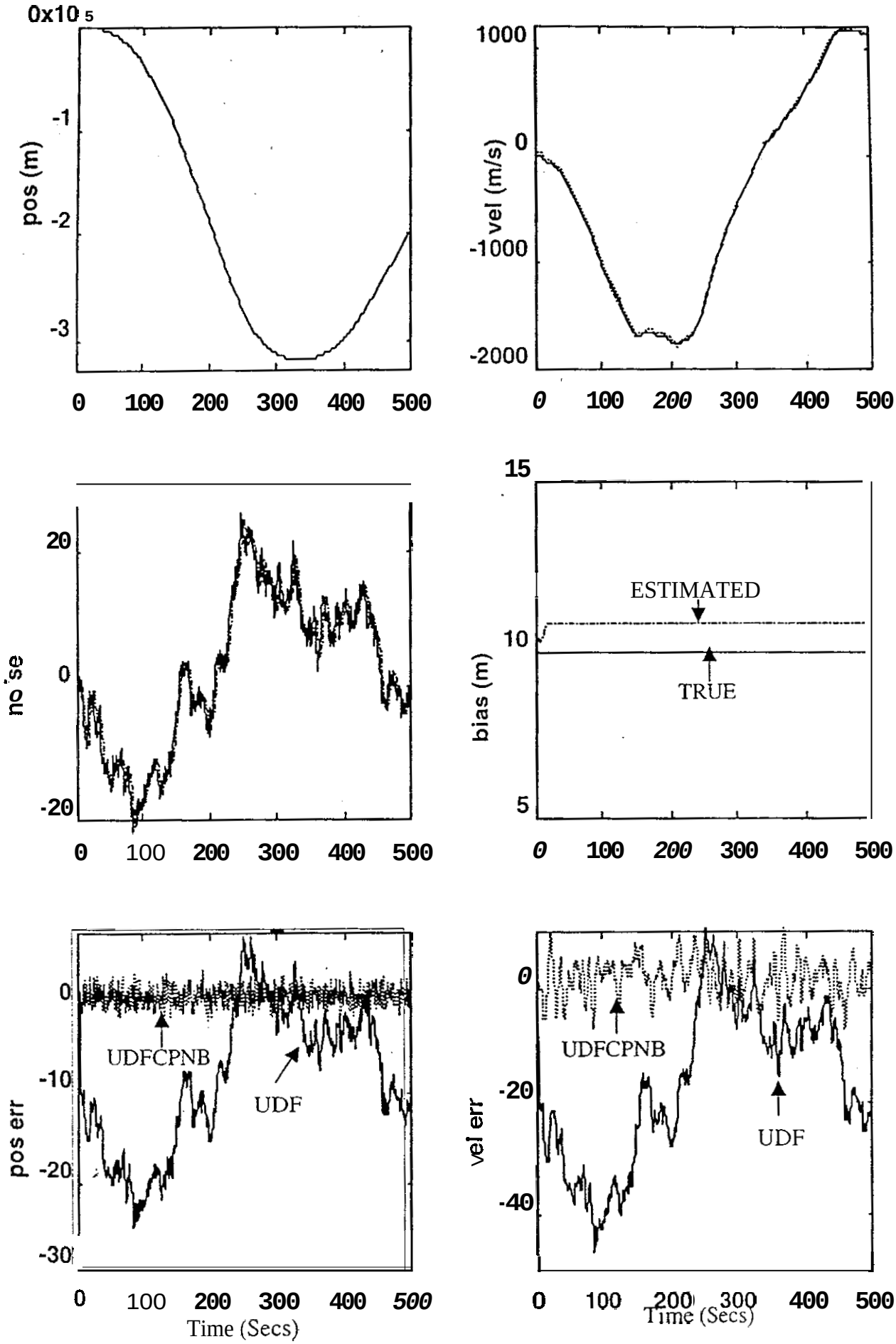


Fig. 2 State Estimates, State error - Simulated data

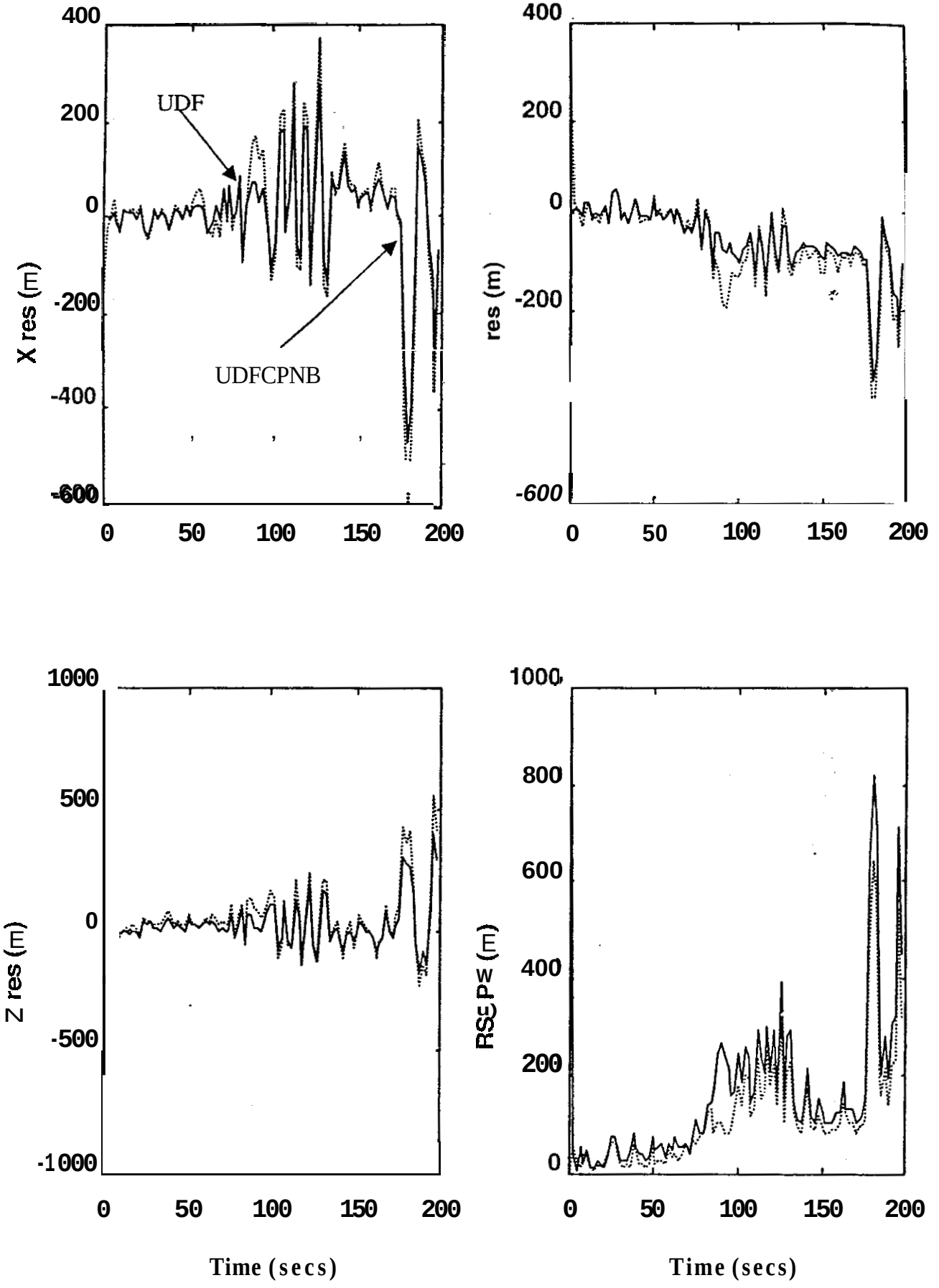


Fig. 3 Residuals and RSSPE - Real Data

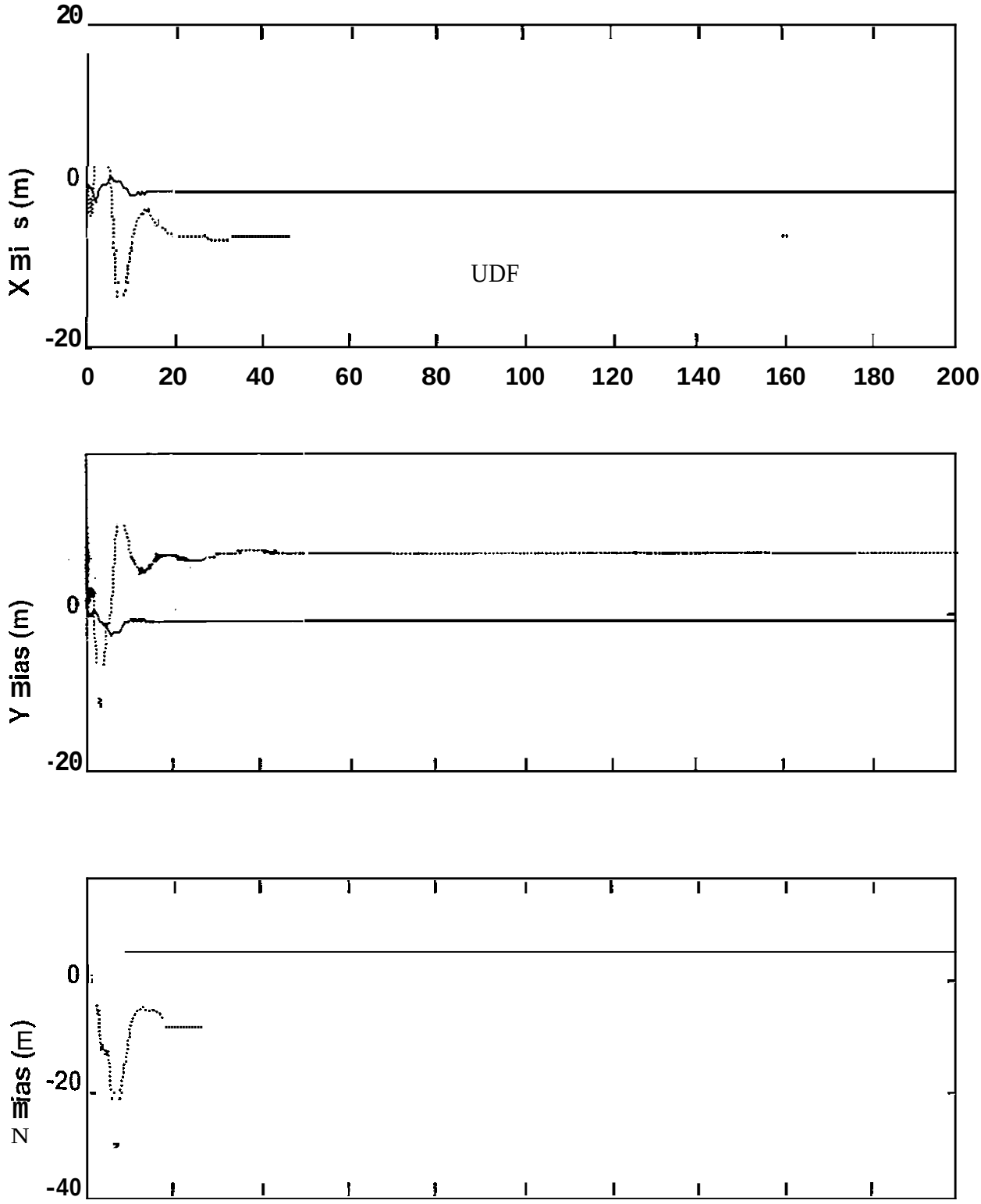


Fig. 4. Bias Estimates - Real data