Defence Science Journal, Vol. 57, No. 3, May 2007, pp. 289-303

Target Tracking with Multi Acoustic Array Sensors Data

V.P.S. Naidu and J.R. Raol

National Aerospace Laboratories, Bangalore-560 017

ABSTRACT

Acoustic array sensor along with Root-MUSIC algorithm is used to estimate the direction of arrival of the acoustic signal emitted by an acoustic target. Three architectures are used to track the target in Cartesian coordinates: (i) digital filter with least square estimation, (ii) linear Kalman filter with least square estimation, and (iii) extended Kalman filter. A comparative evaluation of the three architectures in terms of performance metrics is presented.

Keywords: Multi acoustic array sensor, target tracking, Root-MUSIC algorithm, acoustic sensors, multiple signal classification

NOMENCLATURE

λ	Wavelength
θ_i	i^{th} DoA angle
Δx	Perturbation step size
σ^2	Noise variance
$A[\theta(t)]$	Vandermode matrix
$a(\theta)$	Steering vector
a	Butterworth filter's denominator coefficients
b	Butterworth filter's numerator coefficients
c	Rate of propagation
d	Spacing between sensors
$E\{\}$	Expectation operator
e_{a_j}	Complex envelope of additive Gaussian noise at j^{th} sensor
F	State transition matrix
G	Process noise gain matrix
H	Measurement matrix
$H(z)$	Filter transfer function
I	Identity matrix
i,j,k	Index numbers
K	Kalman gain
$\underline{k}_i$	i^{th} direction vector
l	Number of targets
M	Number of samples
m	Number of sensors
N	Number of snapshots
n,p,q,r	Indexes
P	State error covariance
Q	Process noise covariance matrix
R	Measurement noise covariance matrix
R_a	Correlation matrix

Revised 16 January 2007

$svd()$	Singular value decomposition operator
S	Innovation covariance matrix
S_{r_i}	i^{th} sensor
S_{a_i}	i^{th} target signal waveform
T	Sampling interval
t	Continuous time index
U	Eigenvector
U_e	Noise subspace
U_s	Signal subspace
v	Measurement noise sequence
w	Process noise sequence
X	State vector
$\hat{X}$	Estimated X
$\tilde{X}$	Predicted X
X_t	True state vector
X_{a_j}	j^{th} sensor output
x	Position in x-axis
x_t	True x-position
x_{tr}	Target x-position
$\dot{x}$	Velocity in x-axis
Y	Array output
y	Position in y-axis
y_t	True y-position
y_{tr}	Target y-position
$\dot{y}$	Velocity in y-axis
$\underline{z}_j$	j^{th} sensor position
z_m	Measurement vector
A^T	Transpose of matrix A
λ_i	i^{th} Eigen value

1. INTRODUCTION

The problem of moving target tracking using acoustic array sensor has received considerable attention during the past few years because of its use in air traffic control, air defence, mobile user location in cellular communications, military, naval, underwater tracking, and acoustic source localisation[1]. In acoustic sensor array tracking, the sensor array listens for the acoustic signal emitted by the acoustic target to determine its direction. Unlike radar, the acoustic array sensor tracking systems have stealthy operation capability. Acoustic array sensor-based target tracking is attractive because these operate passively, are inexpensive compared with other modalities and require less power. They can be used as integral part of an intelligent surveillance system, which may also include different types of sensors such as magnetic sensors, imaging sensors, and so on. When the target is in the near-field of the acoustic array, acoustic sounds can be used to determine the location (range and angle) of the target in polar coordinates. When the target is in the far-field of the acoustic array, only angle (bearing) information can be extracted and hence multiple acoustic array sensors are needed to determine the target location. Angle only tracking is very useful when the range measurement is not available[2].

Acoustic array sensor output data is processed by multiple signal classification (MUSIC) algorithm[3] to estimate the direction of arrival (DoA) of the acoustic signal emitted by the acoustic target. The direction of arrival estimation problem has been extensively studied in signal processing[4]. DoA estimation based on the batch processing such as MUSIC, minimum norm, etc. does not reuse the information from the previous batch to help refine the estimates at the current batch. Some tracking algorithms accomplish information transfer from the previous batch to *a priori* information in the current batch by imposing motion constraints on the target. Since acoustic array sensor provides only the angle (azimuth) information, it would be difficult to track the target in Cartesian coordinates with single-angle information. Hence, multi acoustic array sensors are required to provide multiple angle measurements of the target at the same instant of time and the target location can be computed by triangulation method[5,6].

In this paper, the performance of three different architectures for tracking the acoustic target in

Cartesian coordinates is evaluated in terms of percentage fit error, mean absolute error, and root sum square position error[7]. It is assumed that the target is in the far-field of the array. Two such arrays are used to get the angle (bearing) information of the target. In the first architecture, digital filter along with least square estimation is used to estimate the target position in Cartesian coordinates. This architecture is very simple and easily adoptable. In the second architecture, linear Kalman filter along with least square estimation is used to track the target in Cartesian coordinates. It is quite straight-forward since the time updation and measurement updation are in polar coordinates. Extended Kalman filter is used in third architecture to track the target in Cartesian coordinates. Here, the state estimation is in Cartesian coordinates and measurement updation is in polar coordinates.

2. MATHEMATICAL MODELLING OF ACOUSTIC SENSORS

The acoustic sensor array contains m number of sensors uniformly distributed along a straight line as shown on Fig. 1. The array consists of m sensors placed with a uniform distance d between the adjacent sensors in the array. The choice $d = \lambda/2$ prevents spatial aliasing where λ is the carrier wavelength. The sensors sense the acoustic signal produced by the target/source. The target is assumed to be in the far-field of the sensor array, which means that the distance of the target from the sensor array is much greater than the distance between the sensors. The target is also assumed to be narrow emiting band acoustic signal and the target is far from the sensor array, thus, it satisfies the plane wave assumption. The sound waves reaching each sensor are assumed parallel to each other. The direction perpendicular to the sensor array is called the broadside direction. The DoA of the target is measured wrt this direction. It is assumed that the target is moving with constant velocity is subjected to minor random perturbations, and emitting narrow band signals of wavelength λ impinge on array of acoustic sensors which are passive at an angle of θ.

The signal from the target reaches the sensors at different times. This is because each sound wave has to travel a different distance to reach the different sensors. For example, the signal incident on sensor S_{r1} has to travel an extra distance of $d \sin(\theta_1)$ as compared to the signal incident on sensor S_{r_2} (Fig.1). This means that the signal at sensor S_{r_1} is a time-delayed version of the signal at sensor S_{r_2}, with the delay being $-d \sin(\theta_1)/c$, where c is the rate of propagation of the signal. This argument can be extended to the other sensors in the sensor array. Consider $l(l<m)$ number of targets in the test scenario. The array has receiver behind each sensor element. One can express the j^{th} sensor output as the sum of the shifted versions of the source signals and is mathematically represented as:

$$X_{a_j}(t) = \sum_{i=1}^{l} S_{a_i}(t + \tau_{ij}(t)) + e_{a_j}(t) \qquad (1)$$

$$\tau_{ij}(t) = \frac{1}{c} \underline{z}_j \underline{k}_i(t) \quad i = 1,2,..., \; l = 1,2,..., \; m$$

where $X_{a_j}(t)$ is the j^{th} sensor output at time t; $S_{a_i}(t)$ is the i^{th} target signal waveform; $\tau_{ij}(t)$ is the relative time delay induced by the i^{th} target signal in the j^{th} sensor; $e_{a_j}(t)$ is the additive noise at j^{th} sensor; $\underline{k}_i(t) = [\sin \theta_i(t)]$ for $\theta \in = [-90°,...,0,...,90°]$ is the i^{th} direction vector; $\underline{z}_j$ is the j^{th} sensor position; C is the velocity of the acoustic signal in the medium; and τ is the time delay between any two neighboring sensors in the array.

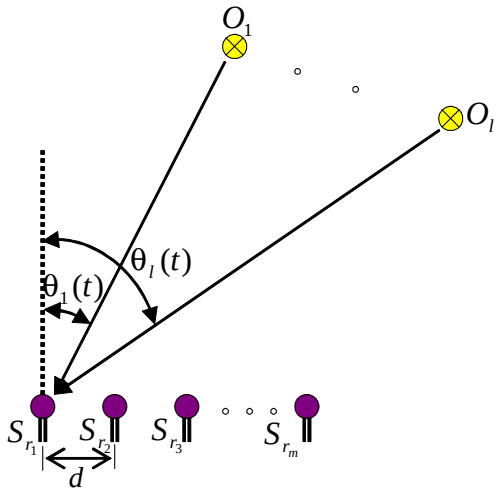


Figure 1. Acoustic sensor array geometry.

Using the analytical representation, the array outputs could be written as

$$X_a(t) = Y = A[\theta(t)]S_a(t) + e_a(t) \quad (2)$$

where

$X_a(t) = \lfloor X_{a_1}(t) \quad X_{a_2}(t) \quad \Lambda \quad X_{a_m}(t) \rfloor^T$ is an m vector of complex envelopes of the sensor outputs, superscript T stands for transpose,

$S_a(t) = \lfloor S_{a_1}(t) \quad S_{a_2}(t) \quad \Lambda \quad S_{a_l}(t) \rfloor^T$ is an l vector of complex envelopes of the target signals

$e_a(t) = \lfloor e_{a_1}(t), \quad e_{a_2}(t), \quad \Lambda \quad e_{a_m}(t) \rfloor^T$ is an m vector of complex envelopes of the additive measurement noises which are assumed to be Gaussian zero mean.

$A[\theta(t)] = [a(\theta_1) \quad a(\theta_2) \quad \Lambda \quad a(\theta_l)]$ is the $m \times l$ Vandermode matrix or steering matrix.

$$a(\theta_i) = \left[1 \quad e^{-j2\pi\frac{d}{\lambda}\sin\theta_i} \quad \Lambda \quad e^{-j(m-1)2\pi\frac{d}{\lambda}\sin\theta_i}\right]^T$$

is the steering vector for the i^{th} source/target signal and it has unit-magnitude elements.

$\theta(t) = [\theta_1(t) \quad \theta_2(t) \quad \Lambda \quad \theta_l(t)]^T$ is the unknown DoA vector (of the l targets/objects at time t) to be estimated.

It is assumed that $\theta(t)$ is a slowly varying function of time t. The change in $\theta(t)$ is assumed either zero or negligible in each interval of $[nT, (n+1)T], \quad n = 0,1,2,\ldots$. i.e.

$$\theta(t) \approx \theta(nT) \text{ for } t \in [nT, (n+1)T], \quad n = 0,1,2,\ldots \quad (3)$$

In each interval, N snapshots of sensor data are available for signal processing. Based on the assumptions made in Eqn (3), the N snapshots of sensor data could be expressed as:

$$\begin{aligned} &X_a(k) \approx A[\theta(n)]S_a(k) + e_a(k), \\ &k = n, n+1, n+2, \ldots, n+N-1 \end{aligned} \quad (4)$$

Eqn (4) is the discretised version of Eqn (2) with a sampling interval of $\frac{T}{N}$.

2.1 DoA Estimation

Here, the actual algorithm to determine the DoA is presented. The multiple signal classification (MUSIC) introduced by Schmidt is one of the popular subspace methods used in spectral estimation to estimate the frequency. Consider the array sensor output data represented in Eqn (2) as:

$$Y = AS_a + e_a \quad (5)$$

Assuming that the signals from different targets/ sources are uncorrelated, the correlation matrix of Y could be written as

$$R_a = E\{YY^T\} = E\{AS_a S_a{}^T A^T\} + E\{e_a e_a{}^T\} \quad (6)$$

$$R_a = AR_s A^T + \sigma^2 I = Z + \sigma^2 I$$

where $R_s = E\{S_a S_a{}^T\}$ is l x l auto correlation matrix with rank l, I is $m \times m$ identity matrix, σ^2 is noise variance, and Z is $m \times m$ signal covariance matrix.

The Eigenvalue decomposition could be done using *svd* operator as:

$$UDU^T = svd(R_a) \quad (7)$$

where $U = [u_1 \quad u_2 \quad \Lambda \quad u_m]$

$$D = \begin{bmatrix} \lambda_1 + \sigma^2 & 0 & \Lambda & 0 & 0 & \Lambda & 0 \\ 0 & \lambda_2 + \sigma^2 & \Lambda & M & M & \Lambda & 0 \\ M & M & O & M & M & \Lambda & M \\ 0 & 0 & \Lambda & \lambda_l + \sigma^2 & 0 & \Lambda & 0 \\ 0 & 0 & \Lambda & 0 & \sigma^2 & \Lambda & 0 \\ M & M & \Lambda & M & M & O & M \\ 0 & 0 & \Lambda & 0 & 0 & \Lambda & \sigma^2 \end{bmatrix}$$

svd() : singular value decomposition operator

The Eigenvector matrix U could be partitioned into a matrix U_s with l columns corresponding to l signal values and a matrix U_e with m–l columns corresponding to the noise Eigenvalues. U_s defines the signal subspace, while U_e defines the noise subspace and both are orthogonal to each other. All noise

Eigenvectors are orthogonal to the signal steering vectors. Pseudo-spectrum could be computed as:

$$P_{MUSIC}(\theta) = \frac{1}{A^T U_e U_e^T A} \quad (8)$$

If θ is equal to DoA, then the denominator would become zero causing peaks in function $P_{MUSIC}(\theta)$. The accuracy is limited by the discretisation at which the $P_{MUSIC}(\theta)$ is computed. It requires either human interaction or ample search algorithm to determine the largest peaks. Search algorithm is a computationally exhaustive process, and hence, estimation of DoA from the pseudo-spectrum would not be very practical. These limitations will be overcome by using Root-MUSIC algorithm. Root-MUSIC is a model-based parameter estimation technique. It uses a model (steering vector) of the received signal as a function of the DoA which is a parameter in the model. DoA, i.e., θ would be estimated based on the model and the received signal. Root-MUSIC algorithm is described as follows:

Let $z = e^{jkd\cos\theta}$

$$A[\theta] = \begin{bmatrix} 1 & z & z^2 & \ldots & z^{m-1} \end{bmatrix}^T$$

$$u_p^T A = \sum_{q=0}^{m-1} u_{pq}^* z^q = u_p(z) \quad (9)$$

The inner product of the eigenvector and the steering vector is equivalent to a polynomial in *z*. The DoA would be obtained where $u_p \perp A[\theta],\ p = l+1, l+2, \ldots, m$ i.e. roots of the polynomial. To find the polynomial:

$$P_{MUSIC}^{-1} = A^T(\theta) U_e U_e^T A(\theta) = A^T(\theta) C A(\theta) \quad (10)$$

$$P_{MUSIC}^{-1} = \sum_{p=0}^{m-1}\sum_{q=0}^{m-1} z^q C_{pq} z^{-p} = \sum_{p=0}^{m-1}\sum_{q=0}^{m-1} z^{(q-p)} C_{pq} \quad (11)$$

The double summation could be rewritten as a single sum by setting $r = q-p$. Then, Eqn (11) can be written as

$$P_{MUSIC}^{-1}(\theta) = \sum_{r=-(m-1)}^{m+1} C_r z^r \quad (12)$$

where $C_r = \sum_{q-p=r} C_{pq}$ (13)

C_r is the sum of the elements of *C* on the q^{th} diagonal. Since *z* and $\frac{1}{z^*}$ have the same phase and reciprocal magnitude, one zero is within unit circle and the other outside the unit circle. From the definition of *z* [Eqn (9)], only the phase carries the desired information. Without noise, the roots would fall on the unit circle and these roots are used to estimate the DoA. The steps involved in estimation of DoA using Root-MUSIC are as follows:

Step 1. Estimate the correlation matrix R_a

Step 2. Find the Eigenvector and Eigenvalues using singular value decomposition, i.e.,

$$\left[UDU^T\right] = svd(R_a)$$

Step 3. Partition *U* to obtain U_e that corresponds to *m–l* smallest Eigenvalues

Step 4. Compute C_r

Step 5. Find the zeros of the resulting polynomial

Step 6. From *m*–1 roots within the unit circle, choose the *l* roots closest to the unit circle ($z_q,\ q = 1,2,\ldots,l$)

Step 7. Obtain the DoA as

$$\theta_q = \sin^{-1}\left(\frac{2z_q}{d\pi}\right)\frac{180}{\pi} \quad (14)$$

3. TARGET TRACKING ARCHITECTURE

Three target tracking architectures are evaluated. In first architecture [Fig. 2(a)], second-order digital filter is used to remove the noise where the measurements are in polar coordinate and least square estimation is used to compute the target position in Cartesian coordinates. In second architecture [Fig. 2(b)], linear Kalman filter is used to estimate the state of the target in polar frame where the acoustic sensor provides measurements in the same

plane, and least square estimation algorithm is used to compute the position of the target in Cartesian coordinate. In third architecture [Fig. 2(c)], extended Kalman filter is used to estimate the target state in Cartesian coordinates while the measurements are available in polar coordinates.

3.1 Digital Filter

The acoustic array sensor provides noisy measurements in polar coordinates. Second-order low-pass digital Butterworth filter with normalised cut off frequency of 0.5 is used to remove the noise. The transfer function $H(z)$ of the filter is

$$H(z)=\frac{b_0+b_1z^{-1}+\Lambda+b_nz^{-n}}{1+a_1z^{-1}+\Lambda+a_nz^{-n}} \qquad (15)$$

where n is order of the filter; b_i are numerator coefficients and $i=0,1,2,...,n$; a_i are denominator coefficients, and z is discrete time operator.

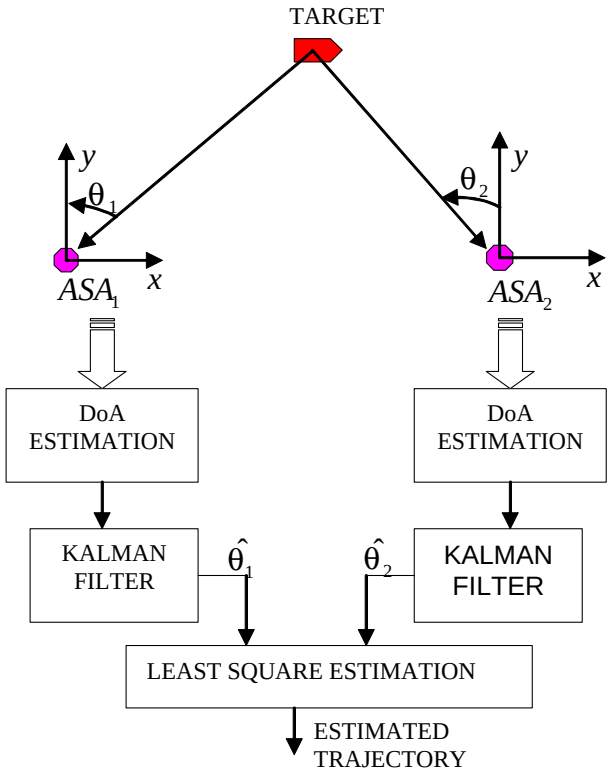


Figure 2(b). Information flow diagram of target tracker.

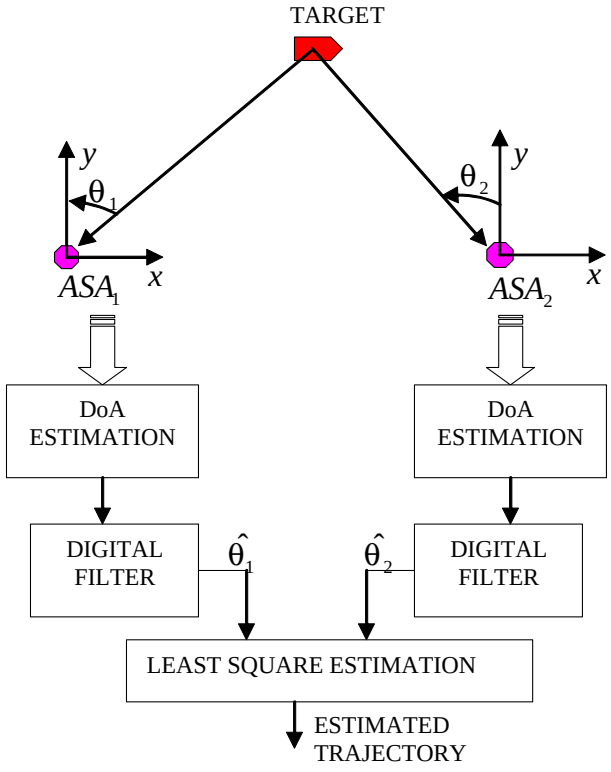


Figure 2(a). Information flow diagram of target tracker.

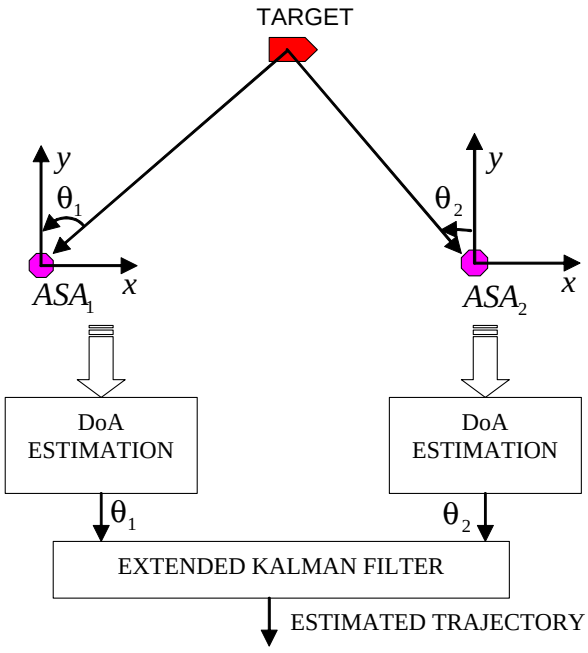


Figure 2(c). Information flow diagram of target tracker.

3.1.1 Least Square Estimation

The filtered measurements in first architecture are in polar coordinates. To get the target position in Cartesian coordinate, least square estimation[11] is used. The procedure is as follows. Let the true target position in Cartesian coordinates (x_{tr}, y_{tr}) be and the locations of two sensors S_{r_1} and S_{r_2} be $(S_{r_{1x}}, S_{r_{1y}})$ and $(S_{r_{2x}}, S_{r_{2y}})$. Let the target be at an angle of θ_1 and θ_2 w.r.t. S_{r_1} and S_{r_2}, respectively.

$$\tan(\theta_1) = \frac{y_{tr} - S_{r_{1y}}}{x_{tr} - S_{r_{1x}}} \quad \text{and} \quad \tan(\theta_2) = \frac{y_{tr} - S_{r_{2y}}}{x_{tr} - S_{r_{2x}}} \tag{16}$$

By rearranging the elements, one can get

$$\begin{bmatrix} S_{r_{1y}} - S_{r_{1x}} \tan(\theta_1) \\ S_{r_{2y}} - S_{r_{2x}} \tan(\theta_2) \end{bmatrix} = \begin{bmatrix} -\tan(\theta_1) & 1 \\ -\tan(\theta_2) & 1 \end{bmatrix} \begin{bmatrix} x_{tr} \\ y_{tr} \end{bmatrix} \tag{17}$$

This can be represented in the form of

$$f = Bg \tag{18}$$

In this Eqn, B and f are known and g is unknown. The vector g, which shows the target position, can be computed as $g=B^{-1}f$.

3.2 Liner Kalman Filter

A general motion model used in discrete Kalman filter for target tracking is[8,9,10]

$$X(k) = FX(k-1) + Gw(k-1) \tag{19}$$

$$z_m(k) = HX(k) + v(k) \tag{20}$$

where $X(k)$ is the state vector, F is the state transition matrix, G is the process noise gain matrix, and H is the measurement matrix. The process noise $w(k)$ and the measurement noise $v(k)$ are zero-mean, mutually independent, white, Gaussian with covariance Q and R respectively. $z_m(k)$ is the measurement vector at time k. The Kalman filtering is done in two steps, viz., time propagation and measurement updation.

3.2.1 Time Propagation

The state and the state error covariance matrix at time k-1 are predicted to time k as follows

$$\begin{aligned} \tilde{X}(k \mid k-1) &= F\hat{X}(k-1 \mid k-1) \\ \tilde{P}(k \mid k-1) &= F\hat{P}(k-1 \mid k-1)F^T + GQG^T \end{aligned} \tag{21}$$

where $\hat{X}$ is the estimated state vector, $\hat{P}$ is the estimated state error covariance matrix, $\tilde{X}$ is the predicted state and $\tilde{P}$ is the predicted state error covariance matrix.

3.2.2 Measurement update

Innovation: $e = z_m(k) - \tilde{z}(k \mid k-1)$ (22)

Innovation covariance:

$$S = H\tilde{P}(k \mid k-1)H^T + R \tag{23}$$

where $\tilde{z}(k \mid k-1)$ is the predicted measurement.

The measurement update part consists of the following equations:

Filter gain: $K = \tilde{P}(k \mid k-1)H^T S^{-1}$ (24)

Updated state: $\hat{X}(k \mid k) = \tilde{X}(k \mid k-1) + Ke$ (25)

Updated state covariance:

$$\hat{P}(k \mid k) = [I - KH]\tilde{P}(k \mid k-1) \tag{26}$$

where I is the identity matrix of order of state vector.

The state estimation in this architecture is in polar coordinates. To get the target position in Cartesian coordinate, least square estimation (Section 3.1.1) is used.

3.3 Extended Kalman Filter

A general motion model used in discrete extended Kalman filter for target tracking is[8,9,10]

$$X(k) = FX(k-1) + Gw(k-1) \quad (27)$$

$$z_m(k) = h(X(k)) + v(k) \quad (28)$$

where h[$X(k)$] is a nonlinear function of the states computed at time.

Equation (28) is nonlinear and it needs to be linearised to fit into the Kalman filter framework entailing the use of extended Kalman filter (EKF).

3.3.1 Time Propagation

The state and the state error covariance matrix at time k-1 are predicted to time k as follows:

$$\begin{aligned} \tilde{X}(k \mid k-1) &= F\hat{X}(k-1 \mid k-1) \\ \tilde{P}(k \mid k-1) &= F\hat{P}(k-1 \mid k-1)F^T + GQG^T \end{aligned} \quad (29)$$

3.3.2 Measurement Update

Innovation: $e = z_m(k) - \tilde{z}(k \mid k-1)$ (30)

Innovation covariance:

$$S = H(k)\tilde{P}(k \mid k-1)H(k)^T + R \quad (31)$$

where $H(k)$ is the linearised measurement matrix.

The measurement update part consists of the following equations

Filter gain: $K(k) = \tilde{P}(k \mid k-1)H(k)^T S^{-1}$ (32)

Updated state: $\hat{X}(k \mid k) = \tilde{X}(k \mid k-1) + Ke$ (33)

Updated state covariance:

$$\hat{P}(k \mid k) = [I - kH(k)]\tilde{P}(k \mid k-1) \quad (34)$$

3.3.3 Predicted Measurement and Linearised Measurement Matrix

Finite difference method is used to compute the linearised measurement matrix. Consider the state vector consisting of position and velocity components of the target in x- and y-axis as

$$[x \quad \dot{x} \quad y \quad \dot{y}] \quad (35)$$

Acoustic sensor measurement is of the form

$$z_m(k) = [\theta_1 \quad \theta_2] \quad (36)$$

The predicted state is in the form

$$[\tilde{x} \quad \dot{\tilde{x}} \quad \tilde{y} \quad \dot{\tilde{y}}] = \tilde{X}(k \mid k-1) \quad (37)$$

The predicted measurement is:

$$\tilde{z}(k \mid k-1) = h[\tilde{X}(k \mid k-1)] = [\tilde{\theta}_1 \quad \tilde{\theta}_2] \quad (38)$$

Components in the predicted measurement are computed from the predicted state vector given in Eqn (37).

$$\tilde{\theta}_1 = \tan^{-1}\left(\frac{\tilde{y} - S_{r_{1y}}}{\tilde{x} - S_{r_{1x}}}\right)$$

$$\text{and} \quad \tilde{\theta}_2 = \tan^{-1}\left(\frac{\tilde{y} - S_{r_{2y}}}{\tilde{x} - S_{r_{2x}}}\right) \quad (39)$$

Calculation of linearised measurement matrix can be accomplished by the finite difference method. This method is generalised and is flexible.

$$\begin{aligned} H(k) &= H_{ij} \quad \left.\frac{\partial h_i}{\partial x_j}\right|_{x=\tilde{X}(k \mid k-1)} \\ &= \frac{h_i(x_j + \Delta x_j) \quad h_i(x_j)}{\Delta x_j} \end{aligned} \quad (40)$$

where i=1,2,..., length of the measurement vector; j=1,2..., length of the state vector, and Δx_j=perturbation step size.

For small perturbation in each of the unknown variables, the perturbed value $h_i(x_j + \Delta x_j)$ is computed. The corresponding elements of H_{ij} are given by the finite difference in the function h [Eqn (28)] to changes in that state. In general, a perturbation step size of 10^{-7} is considered to be adequate.

4. PERFORMANCE CHECK METRICS

The target tracker performance is checked by computing[7,10] the following:

(i) The percentage fit error (PFE) in *x* and *y* positions:

It is computed as the norm of the difference between the true and estimated positions to the norm of the true positions. This will be zero when both true and estimated positions are exactly alike and it will increase when the estimated positions deviate from the true positions. In general, up to 5 per cent will be acceptable. The algorithm that gives the least PFE is preferable.

$$PFE_x = 100 * \frac{norm(x_t - \hat{x})}{norm(x_t)} \quad (41)$$

$$PFE_y = 100 * \frac{norm(y_t - \hat{y})}{norm(y_t)} \quad (42)$$

where x_t is the true *x*-position and $\hat{x}$ is the estimated *x*-position, y_t is the true *y*-position and $\hat{y}$ is the estimated *y*-position, and *norm* is the operator to find the Euclidean length of the vector.

(ii) Root mean square error in position (RMSPE):

It is computed as the root mean square error of the true and estimated *x*- and *y*-positions. It produces a single number and it will be zero when the true and estimated positions are alike. This value will increase when the estimated positions deviate from the true positions. The algorithm that gives minimum value is preferable.

$$RMSPE = \sqrt{\frac{1}{M}\sum_{i=1}^{M}\frac{(x_t(i) - \hat{x}(i))^2 + (y_t(i) - \hat{y}(i))^2}{2}},$$

i = 1,2....., M (43)

where M is number samples in the trajectory

(iii) Root sum square error in position (RSSPE):

It is computed as the root sum square error of the true and estimated *x*- and *y*-positions. It produces a sequence of numbers and these numbers will be zero when the corresponding true and estimated positions are alike. These values will increase when the corresponding estimated positions deviate from the true positions. The algorithm that gives minimum values is highly preferable.

$$RSSPE(i) = \sqrt{(x_t(i) - \hat{x}(i))^2 + (y_t(i) - \hat{y}(i))^2},$$

i = 1,2....., M (44)

(iv) Absolute error (AE) in *x* and *y* positions:

It is computed as the absolute error of the true and estimated positions. It produces a sequence of numbers and it will be zero when the true and estimated positions are alike. This value will grow when the estimated positions deviate from the true positions. When comparing different algorithms, the algorithm that produces minimum value would be highly preferable.

$$AEx(i) = \left|x_t(i) - \hat{x}(i)\right| \quad (45)$$

$$AEy(i) = \left|y_t(i) - \hat{y}(i)\right|, \quad i = 1,2....., M \quad (46)$$

(v) Mean absolute error (MAE) in *x* and *y* positions:

It is computed as the mean absolute error of the true and estimated positions. It produces a single number and it will be zero when the true and estimated positions are alike. This value will grow when the estimated positions deviate from the true positions. When comparing different algorithms, the algorithm that produces minimum value will be preferable.

$$MAEx = \frac{1}{M}\sum_{i=1}^{M}\left|x_t(i) - \hat{x}(i)\right| \quad (47)$$

$$MAEy = \frac{1}{M}\sum_{i=1}^{M}\left|y_t(i) - \hat{y}(i)\right|, \quad i = 1,2..., M \quad (48)$$

5. RESULTS AND DISCUSSION

Performance of the DoA estimation against the following parameters, viz., the noise variance level, number of snapshots, number of sensors in the array, and the direction of the impinging wave

front is presented. Then, the three architectures for target state estimation in Cartesian coordinates are validated with the DoA measurements using numerical simulation.

5.1 Performance of DoA Algorithm

Performance of the DoA estimation algorithm is evaluated using the parameters shown in Table 1. The results are obtained from the average of 50 Monte Carlo simulations. The effect of noise variance level on the DoA estimation is shown in Fig. 3. It is obtained by varying the variance level from 0 to 0.1 and keeping the other parameters constant as shown in Table 1.

It is observed that the absolute error in DoA estimation increases with increase in noise level. The effect of number of snapshots on DoA estimation is shown in Fig. 4 by keeping the other parameters as constant. As expected, the absolute error decreases with increase in the number of snapshots. Figure 5(a) and 5(b) show the effect of direction of incoming signal to the array on DoA estimation. It is observed that when the impinging wave is perpendicular to the sensor array, the algorithm fails to give accurate DoA estimation. It is due to the same signal received by the sensors in the array without time delay. Since the DoA estimation algorithm requires the delayed version of the signal, the algorithm produces more error at 90°. In Fig. 5(b), the upper window shows the estimated and true DoAs, and the corresponding error is shown in the lower window. It is clear from Fig. 5(b) that the error is very less except at 90°.

The effect of number of sensors in the array on DoA estimation is shown in Fig. 6. It is observed that the absolute error decreases with increase in the number of sensors. The number of signals provides the average effect that nullifies the noise effect.

Table 1. Parameters used in evaluation of DoA estimation algorithm performance

Parameter	Value
No. of sensors (m)	5
No. of snapshots (N)	50
Noise variance (σ^2)	0.00001
No. of targets (l)	1
Theta (θ)	30°
Spacing between the sensors (d)	0.5 λ

5.2 Target Tracking

The 2-DOFs kinematic model, with position and velocity components in each of the two Cartesian coordinates x and y, has the following transition and process noise gain matrices:

$$F = diag[\Phi \quad \Phi] \text{ and } G = diag[\varsigma \quad \varsigma]$$

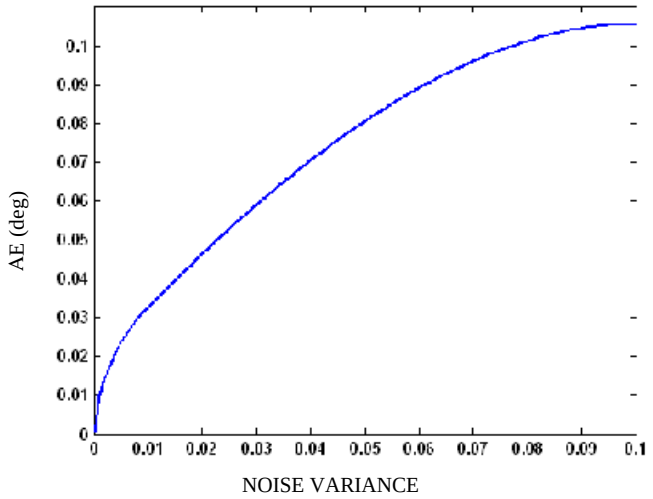


Figure 3. Effect of noise variance level on DoA estimation.

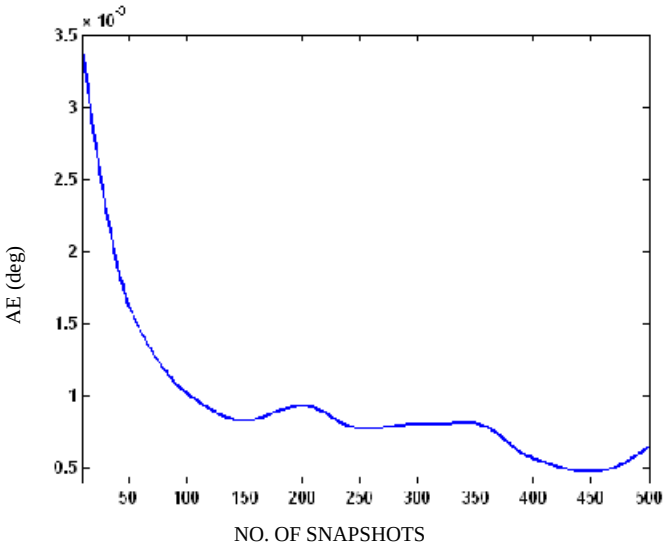


Figure 4. Effect of number of snapshots on DoA estimation.

where

$$\Phi = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \text{ and } \varsigma = \begin{bmatrix} T^2/2 \\ T \end{bmatrix}$$

The algorithm is validated using the simulated data. The simulation utilises the following parameters.

Sampling interval: T = 0.1 s

Process noise variance: 0.001

Duration of simulation: 100 s

Initial state vector is $[x \quad \dot{x} \quad y \quad \dot{y}] = [0 \quad 1 \quad 10 \quad 0]$

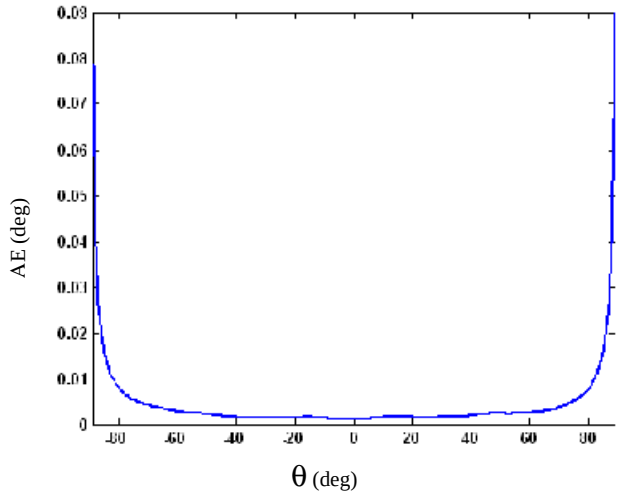


Figure 5(a). Effect of magnitude of θ on DoA estimation.

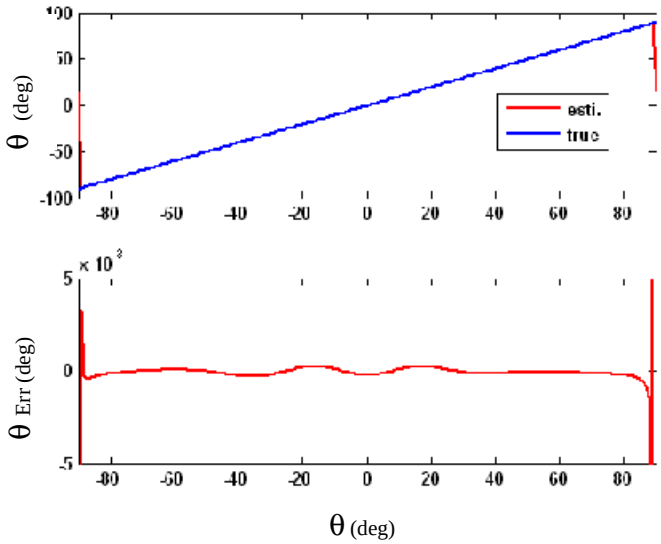


Figure 5(b). Effect of magnitude of θ on DoA estimation.

Sensor locations: $S_{1r} = [S_{r_{1x}} \quad S_{r_{1y}}] = [0 \quad 0]$ and $S_{r2} = [S_{r_{2x}} \quad S_{r_{2y}}] = [50 \quad 0]$

The target trajectory in Cartesian coordinate is shown in Fig. 7(a). The trajectory of the target in Cartesian coordinates is converted into polar coordinates using Eqn (16) and array sensor output is generated for each sample. The sensor array contains five sensors and produces 100 snapshots with noise variance of 0.00001. Root-MUSIC algorithm is used to estimate DoA at each interval. Measurement noise with variance of 0.001 is added to the estimated DoA's. The simulated DoA's data is shown in Fig. 7(b) and enlarged view of a portion is shown in Fig. 7(c).

The initial state vector for both linear and extended Kalman filters is computed with the first two measurements. Identifying matrices of the order of state vector and multiplying with 0.01 are taken as initial state error covariance matrices for both the filters. True and estimated *x*- and *y*-positions are shown in Fig. 8(a) and enlarged view of a portion is shown in Fig. 8(b). The estimated trajectories show good agreement with the true trajectories. The metrics to evaluate the performance of the tracking algorithm are shown in Table 2. These metrics are within the acceptable limits. Absolute errors in *x*- and *y*-position are shown in Fig. 9(a) and enlarged view of a portion is shown in Fig.

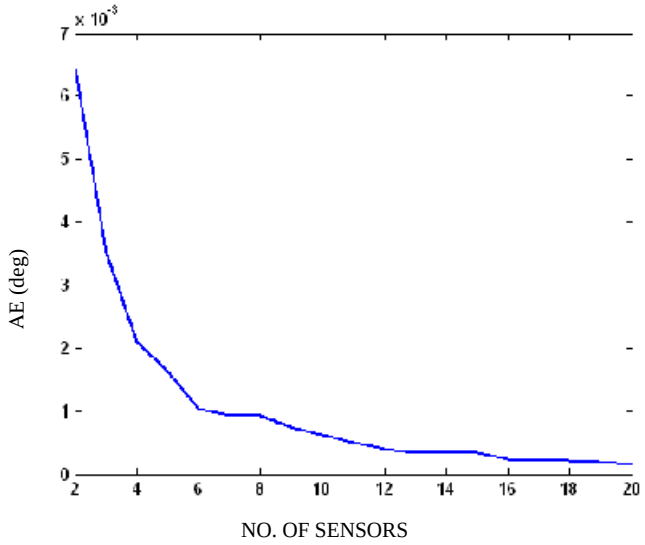


Figure 6. Effect of number of sensors on DoA estimation.

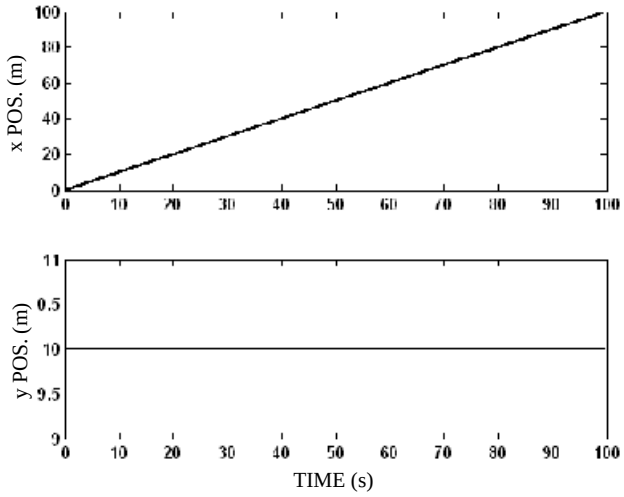


Figure 7(a). Target trajectory in Cartesian coordinates.

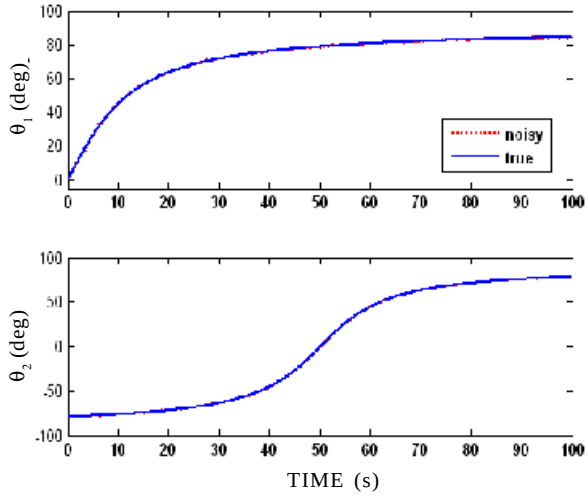


Figure 7(b). True and noisy DoA measurements.

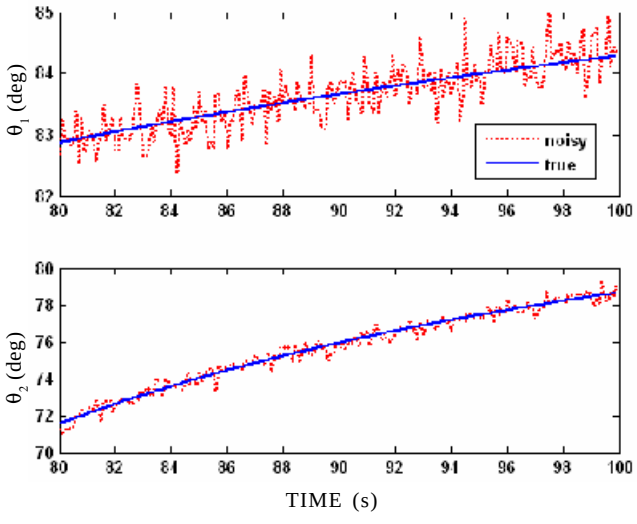


Figure 7(c). Enlarged view of some portion in Fig. 7(b).

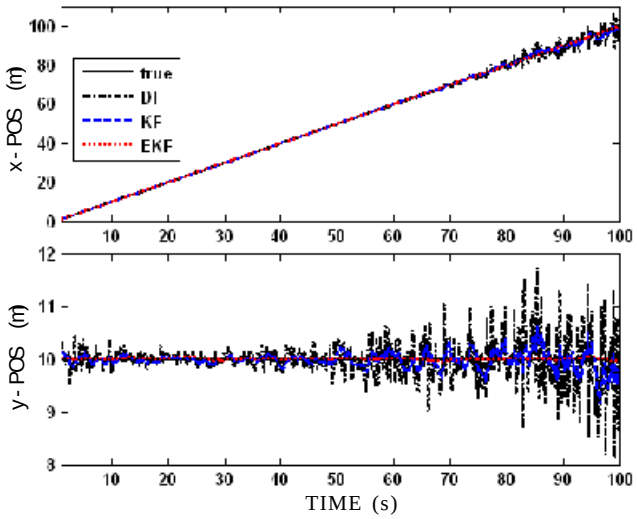


Figure 8(a). True and estimated *x*- and *y*-position.

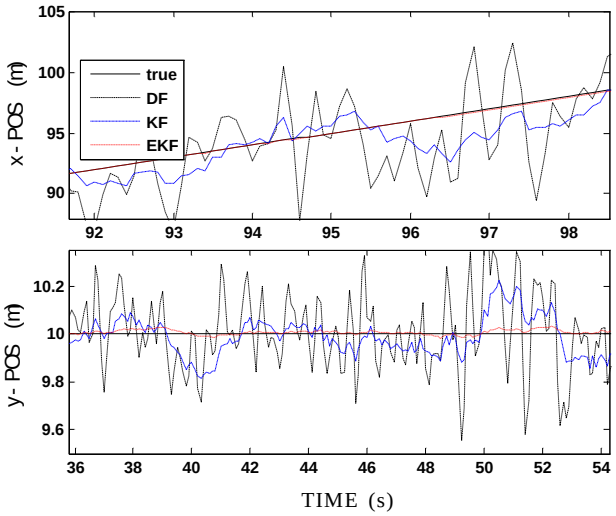


Figure 8(b). Enlarged view of some portions in Fig. 8(a).

9(b). Fig. 10(a) shows the root sum square position errors and enlarged view of a portion is shown in Figs 10(a) and 10(b). It is observed that the errors are minimal when the target moves between the sensors and these are increasing when the target moves away from the sensors.

Table 2. Performance check metrics

	PFE_x	PFE_y	MAE_x	MAE_y	RMSPE	Execution time (s)
DF	2.43	5.349	0.67	0.251	1.06	0.04
KF	0.98	1.503	0.29	0.101	0.41	0.15
EKF	0.06	0.173	0.03	0.013	0.03	0.18

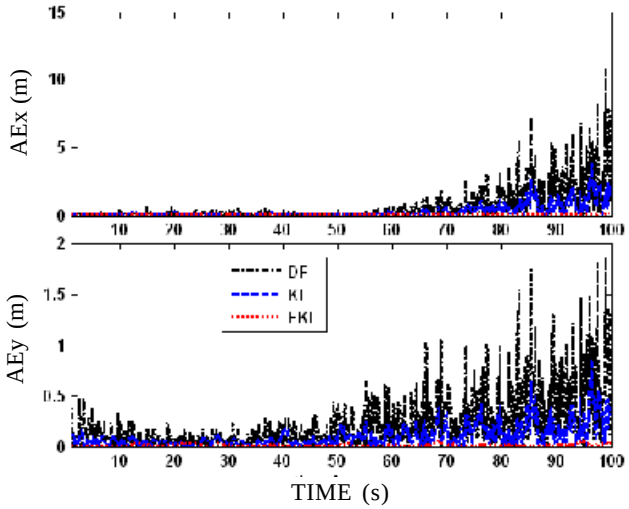


Figure 9(a). Absolute error in *x*- and *y*-positions.

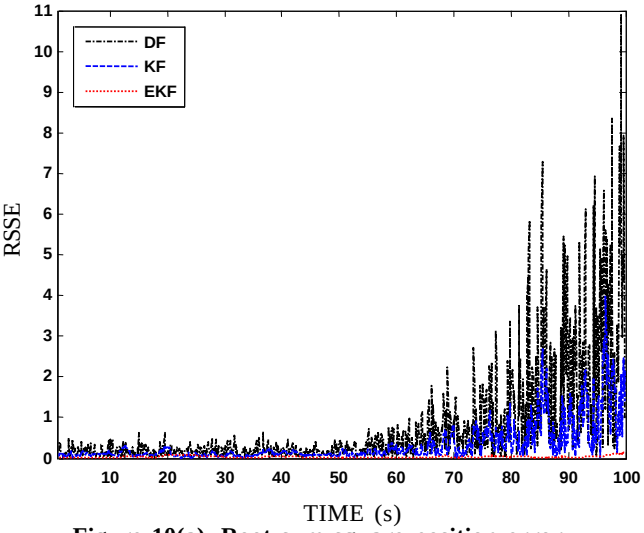


Figure 10(a). Root sum square position error.

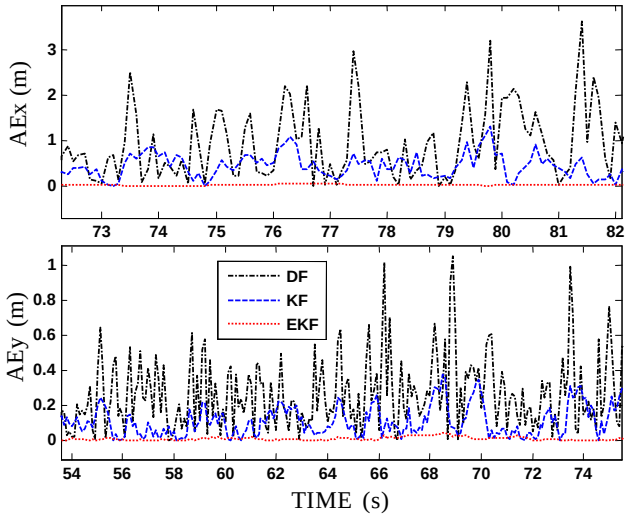


Figure 9(b). Enlarged view of some portion in Fig. 9(a).

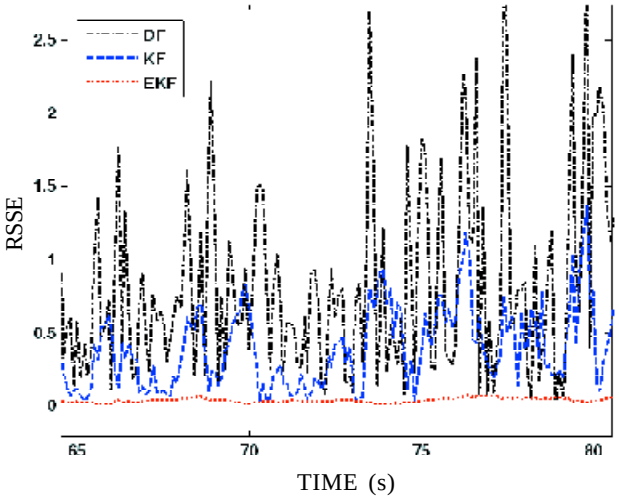


Figure 10(b). Enlarged view of some portion in Fig. 10(a).

5.3 Observations

It is observed that good DoA estimation can be achieved with increased number of snapshots and more number of sensors in the array. The effect of noise on DoA estimation would be minimised using number of snapshots. The effect of impinging wave at 90° would be avoided by using recursive target state estimator.

Digital filter along with least square estimation showed bad performance when the target moves in between the sensors and it is worse when the target moves away from the sensors. This shortcoming may be due the lack of target model in the state estimation process. Linear Kalman filter along with least square estimation showed some degraded performance when the target moves away from the sensors. The degraded performance may be due the nonlinear transformation involved in the estimation process. Extended Kalman filter is not much degraded as compared to the previous architectures. From the Table 2, it is observed that the execution time for first architecture is the lest among the three. In overall estimation, EKF showed superior performance among the three architectures albeit at the cost of execution time.

6. CONCLUSION

Mathematical model of acoustic sensor array has been used to produce uniform linear array

data. Root-MUSIC algorithm has been used for DoA estimation. The performance of DoA estimator against various parameters is presented. It is concluded that more number of snapshots and more number of sensors in the array would improve the DoA estimation.

Three architectures based on digital filter along with least square estimation, linear Kalman filter along with least square estimation and extended Kalman filter have been evaluated to track the target with only angle measurements obtained by two uniform linear array data obtained from two acoustic array sensors. From the results it is concluded that EKF showed better results than linear Kalman filter along with least square estimation and digital filter with least square estimation at the cost of execution time.

REFERENCES

1. Dogancay, Kutluyil & Hashemi-Sakhtsari, Ahmad. Target tracking by time difference of arrival using recursive smoothing. *Signal Processing*, 2005, **85**, 667-79.

2. Keche, M.; Ouamri, A.; Harrison I. & Woolfson, M.S. Performance evaluation of an algorithm for multiple target angle tracking. *IEE Proc. Radar, Sonar Navig.*, October 1997, **144**(5).

3. Stoica, P. & Nehorai, A. MUSIC, maximum likelihood and Cramer-Rao bound. *IEEE Trans. Acous. Speech Signal Proc.*, 1989, **37**(5), 720-41.

4. Johnson, D.H. & Dudgeon, D.E. Array signal processing: Concepts and techniques. Prentice Hall Signal Processing Series, 1993.

5. Wijk, Olle & Christensen, Henrik I. Triangulation-based fusion of sonar data with application in Robot pose tracking. *IEEE Trans. Robotics Automation*, 2000, **16**(6), 740-52.

6. Dufour, Francois & Mariton, Michel. Tracking a 3D maneuvering target with passive sensors. *IEEE Trans. Aero. Elec. Sys.*, 1991, **27**(4), 725-38.

7. Naidu, V.P.S. & Raol, J.R. Evaluation of data association and fusion algorithms for tracking in the presence of measurement loss. AIAA Conference on Navigation, Guidance and Control, Austin, USA, 11-14 August 2003.

8. Blackman, Samuel & Popoli, Robert. Design and analysis of modern tracking systems. Artech House, London, 1999.

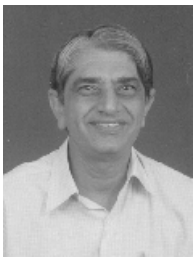
9. Bar-Shalam, Yaakov & Li, X. Estimation and tracking: Principles, techniques, and softwares. Artech House, London, 1993.

10. Naidu, V.P.S. & Raol, J.R. Fusion of radar and infrared search and track data using Kalman filter. NAL PD FC 0517, 2005.

11. Raol, J.R.; Girija G. & Singh, Jatinder. Modelling and parameter estimation of dynamic system. IEE Control Engineering Series Book, **65**, IEE, London, August 2004.

Contributors

Mr V.P.S. Naidu obtained his ME (Medical electronics) from the Anna University, Chennai, in 1997. He is working as a Scientist at the National Aerospace Laboratories (NAL), Bangalore, since 2001 in the area of multisensor data fusion and target tracking. His areas of interest are image registration, tracking, and data fusion.



Dr J.R. Raol obtained PhD from McMaster University, Canada, in 1986. He worked at NAL, Bangalore, and was actively involved in the multi-disciplinary control group's activities on human pilot modelling in fixed- and motion-based research simulators. He is Head of the Flight Mechanics and Control Division of NAL. His current activities include: Modelling, multisensor data fusion, fuzzy systems, genetic algorithms and neural networks. He is a senior member of the IEEE (USA), fellow of IEE, life fellow of the Aeronautical Society of India and life member of System Society of India. He has authored a book on Modelling and Parameter Estimation of Dynamic System, published by *IEE*, UK, in 2004. He has more than 100 publications to his credit.