Defence Science Journal, Vol. 59, No. 3, May 2009, pp. 205-214
© 2009, DESIDOC

Modelling and Simulation of Multi-target Multi-sensor Data Fusion for Trajectory Tracking

A.K. Singh and N. Sood

Defence Institute of Advanced Technology, Girinagar, Pune-411 025

ABSTRACT

An implementation of track fusion using various algorthims has been demonstrated . The sensor measurements of these targets are modelled using Kalman filter (KF) and interacting multiple models (IMM) filter. The joint probabilistic data association filter (JPDAF) and neural network fusion (NNF) algorithms were used for tracking multiple man-euvring targets. Track association and fusion algorithm are executed to get the fused track data for various scenarios, two sensors tracking a single target to three sensors tracking three targets, to evaluate the effects of multiple and dispersed sensors for single target, two targets, and multiple targets. The targets chosen were distantly spaced, closely spaced and crossing. Performance of different filters was compared and fused trajectory is found to be closer to the true target trajectory as compared to that for any of the sensor measurements of that target.

Keywords: Multi-sensor data fusion (MSDF), multiple-target tracking (MTT), data association, interacting multiple models.

1. INTRODUCTION

Multi- sensor data fusion (MSDF) combines data from multiple sensors and related information to achieve specific inferences that may not be possible from a single sensor or source[1]. Multi sensor systems provide operational benefits to an specific application[2], which include robust operational performance, extended spatial temporal coverage, reduced ambiguity, improved detection performance, enhanced spatial resolution, increased system operation reliability, confidence and dimensionality. In recent years, significant attention has been focused on MSDF for both military and industrial applications. Military applications for information fusion include a wide range of command, control, communication, and intelligence (C^3I) missions for both tactical and strategic warfare[3], like air-to-air defence, surface-to-air defence, ocean surveillance, battlefield intelligence and surveillance, target acquisition, strategic warning and defence[1,4]. Industrial and commercial applications of data fusion include automated manufacturing[5], medical diagnostic[6], image processing[7], remote sensing[8], robotics[9] and machine intelligence[10].

The main concern for multi-target tracking is the data association which includes observation-to-track association and track maintenance. The objective of observation-to-track association is to decide if the new sensor observation corresponds to an existing track, and that of track maintenance is to decide the confirmation or the deletion of each existing track, and the initiation of new tracks. Gating is one of the most popular methods for data association, used to narrow the search around a predicted target position for the next update or measurement, i.e., used to eliminate unlikely observation-to-track pairings[11]. Three approaches to data association within a tracking gate are nearest-neighbour, global nearest-neighbour and all-neighbours' method.

In this paper, target as well as sensor modelling has been done, using MATLAB with user-friendly GUI for model selection, data alignment settings, system parameters and target model settings, tracking filter settings, simulation and results plot settings and target trajectory plots window. After setting the parameters, the user runs the Monte Carlo simulation to generate the target measurements. Thereafter track association and fusion algorithm is executed to get the fused track. The output is a trajectory plot constituting a comparison of the generated fused track with the actual target position and the tracks generated separately by each individual sensor. Measurement to track allocation is performed local to the sensors, so that each sensor maintains its own tracks. These tracks are then associated with each other. Instead of fusing the sensor tracks to form system tracks, the measurements allocated to the associated tracks are fused through measurement fusion.

Two tracking algorithms have been implemented, viz., Kalman filter and interacting multiple models (IMM) filter. The user has the option to select any of the filters for simulating a sensor while modelling the system. Track-to-track fusion for two sensors tracking a single target has been done using simple fusion (SF) and neural network fusion (NNF). Track-to-track fusion for multiple targets

Received 22 April 2008

tracking with multiple sensors has been done using JPDAF and NNF.

2. THEORETICAL ANALYSIS

Dynamic targets require continuous or discrete, time sampled measurement of the target location and the ability to estimate the kinematics of the target to predict future positions for continued sensor coverage. This process requires iterative association of each set of sensor data with predicted location of the known target tracks to determine current track, new target or false alarm. In the formulation, assumptions made are (a)The number of targets is *N* with their tracks initialised (b)The targets are detected independently at each scan (c)All the sensors are synchronised (d) Only one of the measurements can be target originated, remaining measurements are assumed due to false alarm or clutter (e)At each time, a validation region (gate) is set up to select the measurement to be used for the state update (f) The number of false measurements obeys a Poisson distribution with known mean, and (g)The target originated measurement is Gaussian with mean and covariance.

2.1 Target Modelling

A discrete time-linear dynamic system described by a vector difference equation with additive white Gaussian noise that models unpredictable disturbances was used for modelling the targets[12,13]. The dynamic equation of the target is

$$x(k+1) = F(k)x(k) + G(k)u(k)+v(k) \tag{1}$$

where the state vector at time k, $x(k) = [x_k \;\; x'_k \;\; y_k \;\; y'_k\,]'$ and *v(k)* denotes the sequence of the zero mean white Gaussian process noise with covariance

$$Q(k) = E[v(k)\, v(k)'] \tag{2}$$

The measurement equation is

$$z(k) = H(k)x(k) + w(k) \tag{3}$$

where *w(k)* denotes the sequence of zero mean white Gaussian measurement noise with covariance

$$R(k) = E[w(k)\, w(k)'] \tag{4}$$

The matrices *F*, *G*, *H*, *Q* and *R* are assumed to be known and time varying, i.e., the system can be time varying and the noise non-stationary. The initial state *x*(0), in general, is modeled as random vectors, Gaussian distributed with known mean and covariance. The two noise sequence and initial states are assumed to be mutually independent. This constitutes the linear-gaussian (LG) assumption.

- MMSE Estimate of the state if *j*=*k* (also called filtered value)
- MMSE Smoothed (filtered) value of the state if *j*<*k*
- MMSE Predicted value of the state if *j*>*k*

2.2 Sensor Modelling

Recursive dynamic target tracking is a state estimation problem that requires (a) A sensor to take the series of observations upto time *t* and process the data to estimate the state vector at time *t*.(b) A predictor to take the series of observations up to time t and process the data to predict the state vector at time $t+\tau$. The discrete time recursive solution to the linear, minimum variance estimation problem is provided by the Kalman filter. Another filter used is the IMM which switches between multiple filters.

2.2.1 Kalman Filter

A simple Kalman filter target tracker, that has been implemented, is based on the assumption that the motion can be modeled as a point target moving in a straight line with constant velocity[14]. A noise component is included in the model to allow for the target motion which is not constant. The measurements are made in Cartesian coordinates and the measurement errors are regarded as independent and normally distributed. It is assumed that the target behavior is known *a priori*. The Kalman filter provides a minimum mean square error estimate (MMSE) of target position and velocity in this system setup. The estimation algorithm starts with initial estimate and the associated initial covariance is assumed to be available.

2.2.2 Interacting Multiple Models Filter

The IMM tracker is used to predict the current state of the target using two or more different models. For example, if the target is expected to be a maneuvering target, the model used could be a straight line motion (SLM) model and a turning motion model. Other models used could be turn rate models or climbing/descending models. The number of models used is application dependent. In this work, a 2-model IMM is used, where the two models differ only by the noise term (one for SLM, and one for turning motion) and are equally probable. Similar to the soft-switching, the model probabilities are updated at each new measurement, and the resulting weighting factors are used in calculating the state, i.e., no gating decision is required for the tracker. In the IMM approach at time k, the state estimate is computed under each possible current model using *r* filters, with each filter using a different combination of the previous model–conditioned estimate–mixed initial condition[15,16]. This algorithm consists of r interacting filters operating in parallel. The mixing is done at the input of the filters with the probabilities conditioned on Z^{k-1}.

IMM Algorithm: One cycle of the algorithm consists of the following[16,17]:

(a) Calculation of the mixed probabilities
(b) Mixing (*j*=1, 2…*r*)
(c) Mode matching filtering (*j*=1, 2….*r*)
(d) Estimate and covariance combination

3. SINGLE TARGET TRACKING WITH TWO SENSORS

Sensors are assumed to have an individual measurement statistics described by the covariance of their respective track filters. As process noise introduced by the target behavior is observed by all sensors observing a common track, their correlated covariance's must be considered in correlation metrics. The issue of track to track correlation[18,19] arises when several sensors carry out surveillance over a certain area and each sensor has its own data processing system and number of tracks. Singer and Kanyuck[20] assumed the target to be estimated by a Kalman filter. Taking two sites each with its sensor and data processor, the difference between the two estimates is tested for the hypothesis whether the two underlying states are the same. However, it is tacitly assumed that the estimation errors of the same target at two sites are uncorrelated. Although, the measurement noise of the two sensors can be taken as independent, there is still the same process noise in the dynamic model. This makes the two estimation errors correlated. The statistics used is

$$R^2 = (\hat{\mathbf{x}}^i_{k|k} - \mathbf{x}^j_{k|k})'(\mathbf{P}^i_{k|k} + \mathbf{P}^j_{k|k})(\hat{\mathbf{x}}^i_{k|k} - \mathbf{x}^j_{k|k}) \qquad (5)$$

where $\hat{\mathbf{x}}^i_{k|k}$ is the estimated state of a target by sensor i at time k . Let $\hat{\mathbf{x}}^j_{k|k}$ be the estimate from sensor j of a target for the same time. $\mathbf{P}^i_{k|k}$ is the covariance associated with the estimate $\hat{\mathbf{x}}^i_{k|k}$. The hypothesis that the two targets are the same holds if R^2 is below a certain threshold obtained through the Chi Square distribution. The implicit assumption of the independence of the two estimate errors is obviously incorrect as the same process noise enters into the evolution equations of the two estimates if they belong to the same target. The proposed statistical tests using the estimates $\hat{\mathbf{x}}^i_{k|k}$, $\hat{\mathbf{x}}^j_{k|k}$ are based on the difference between the two estimates

$$\Delta^{i,j}_k = (\hat{\mathbf{x}}^i_{k|k} - \hat{\mathbf{x}}^j_{k|k}) \qquad (6)$$

to be normalised by its covariance matrix. The covariance of Eqn.(6) is

$$\begin{aligned} E\{\Delta^{i,j}_k \Delta^{i,j\prime}_k\} = E\{&[\hat{x}^i_{k|k} - x_k - (\hat{x}^j_{k|k} - x_k)] \\ &[\hat{x}^i_{k|k} - x_k - (\hat{x}^j_{k|k} - x_k)]'\} \end{aligned} \qquad (7)$$

$$= \mathbf{P}^i_{k|k} + \mathbf{P}^j_{k|k} - \mathbf{P}^{ij}_{k|k} - \mathbf{P}^{ji}_{k|k} \qquad (8)$$

where

$$\begin{aligned} \mathbf{P}^{ij}_{k|k} &\triangleq E\{(\hat{\mathbf{x}}^{ij}_{k|k} - \mathbf{x}_k)(\hat{\mathbf{x}}^j_{k|k} - \mathbf{x}_k)'\} \\ &= \mathrm{E}\{\tilde{\mathbf{x}}^i_{k|k} - \tilde{\mathbf{x}}^{j\,\prime}_{k|k}\} \end{aligned} \qquad (9)$$

reflects the correlation between the two estimates. The test statistics that replaces $\mathbf{x}^j_{k|k}$ is therefore

$$R^2 = (\hat{\mathbf{x}}^i_{k|k} - \hat{\mathbf{x}}^j_{k|k})'(\mathbf{P}^i_{k|k} - \mathbf{P}^j_{k|k} - \mathbf{P}^{ij}_{k|k} - \mathbf{P}^{ji}_{k|k})^{-1}(\hat{\mathbf{x}}^i_{k|k} - \hat{\mathbf{x}}^j_{k|k}) \qquad (10)$$

3.1 Fusion of the Estimates

If the common origin hypothesis is accepted, then one can fuse[21] the two estimates $\hat{\mathbf{x}}^i_{k|k}$ and $\hat{\mathbf{x}}^j_{k|k}$ of $\mathbf{X}(k)$. Use is made of the static linear estimation equation

$$\hat{\mathbf{x}} = \overline{\mathbf{x}} + \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}(\mathbf{z} - \overline{\mathbf{z}}) \qquad (11)$$

where the prior mean is mapped into the posterior mean using the measurement **z**. Denoting the information from sensor i as the prior data (D^i) one has, assuming normal distribution, omitting the time argument for simplicity.

$$\mathrm{P}(\mathbf{x}|\mathbf{D}^i) = \mathrm{N}(\hat{\mathbf{x}}^i, \mathbf{P}^i) \qquad (12)$$

Then a measurement

$$\hat{\mathbf{x}}^j = \mathbf{x} - \tilde{\mathbf{x}}^j \qquad (13)$$

is made, which represents the data $\mathbf{D}^j$. The error $\tilde{\mathbf{x}}^j$ is zero mean with covariance $\mathbf{P}^j$ and cross-covariance $\mathbf{P}^{ij}$ with the error $\tilde{\mathbf{x}}^i$, also normally distributed.

Dependent Tracks

The fusion equation becomes

$$\hat{\mathbf{x}} = \hat{\mathbf{x}}^i + (\mathbf{P}^i - \mathbf{P}^{ij})(\mathbf{P}^i + \mathbf{P}^j - \mathbf{P}^{ij} - \mathbf{P}^{ji})^{-1}(\hat{\mathbf{x}}^j - \hat{\mathbf{x}}^i) \qquad (14)$$

The equivalent of the covariance updates equation

$$\mathbf{P}_{xx|z} = \mathbf{P}_{xx} - \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1}\mathbf{P}_{zx} \quad \text{becomes} \qquad (15)$$

$$M = \mathbf{P}^i - (\mathbf{P}^i - \mathbf{P}^{ij})(\mathbf{P}^i + \mathbf{P}^j - \mathbf{P}^{ij} - \mathbf{P}^{ji})^{-1}(\mathbf{P}^i - \mathbf{P}^{ji}) \qquad (16)$$

which is the covariance of the fused estimate.

Independent Tracks

In the absence of dependence, i.e. if $\mathbf{P}^{ij} = 0$, equation will be

$$\begin{aligned} \hat{\mathbf{x}} &= \hat{\mathbf{x}}^i + \mathbf{P}^i(\mathbf{P}^i + \mathbf{P}^j)^{-1}(\hat{\mathbf{x}}^j - \hat{\mathbf{x}}^i) \\ &= \mathbf{P}^j(\mathbf{P}^i + \mathbf{P}^j)^{-1}\hat{\mathbf{x}}^i + \mathbf{P}^i(\mathbf{P}^i + \mathbf{P}^j)^{-1}\hat{\mathbf{x}}^j \end{aligned} \qquad (17)$$

and

$$\begin{aligned} M &= \mathbf{P}^i - \mathbf{P}^i(\mathbf{P}^i + \mathbf{P}^j)^{-1}\mathbf{P}^i \\ &= \mathbf{P}^i(\mathbf{P}^i + \mathbf{P}^j)^{-1}\mathbf{P}^j \end{aligned} \qquad (18)$$

4. MULTIPLE TARGETS TRACKING USING JPDAF

JPDAF implementation of multi-target tracking using MSDF is shown in Fig. 1. The association probabilities for every track with every measurement in the present scan is computed and subsequently used as weighing coefficients in the formation of a weighted average measurement for updating each track. The problem of associating data with targets in cluttered multi-target environment is discussed by Tugnait[16]. The probabilistic data association method[17,22], based on computation of the posterior probability of each candidate measurement found in the validation gate, assumes that only one real target is present and all other measurements are Poisson's distributed clutter. In JPDA algorithm, the joint posterior association probabilities are computed for multiple targets in Poisson clutter. The algorithm is applied to a tracking problem with multiple targets and sensors. The initial state is assumed to be Gaussian with mean $\hat{\mathbf{x}}_{0|0}$ and covariances $\mathbf{P}_{0|0}$. The tracked estimate of the target

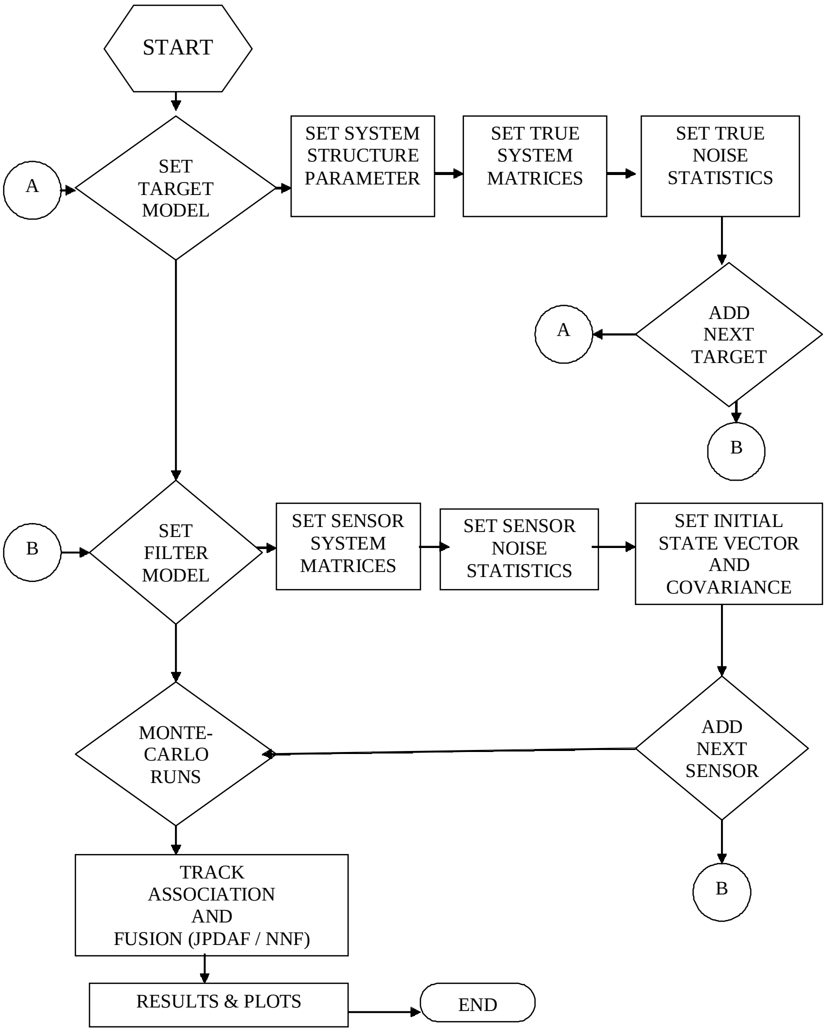
START
SET TARGET MODEL
A
SET SYSTEM STRUCTURE PARAMETER
SET TRUE SYSTEM MATRICES
SET TRUE NOISE STATISTICS
ADD NEXT TARGET
A
B
B
SET FILTER MODEL
SET SENSOR SYSTEM MATRICES
SET SENSOR NOISE STATISTICS
SET INITIAL STATE VECTOR AND COVARIANCE
MONTE-CARLO RUNS
ADD NEXT SENSOR
B
TRACK ASSOCIATION AND FUSION (JPDAF / NNF)
RESULTS & PLOTS
END

Figure 1. Flow chart for multiple targets tracking using multi-sensor data fusion.

state $\mathbf{x}_k$ at time *k*, given data up to time *i*, is $\mathbf{x}_{k|i}$ and the corresponding estimate of the output $\mathbf{z}_k$ is $\hat{\mathbf{z}}_{k|i}$. The error in the state estimate is $\tilde{\mathbf{x}}_{k|i} \triangleq \mathbf{x}_k - \hat{\mathbf{x}}_{k|i}$ with error covariance matrix $\mathbf{P}_{k|i} \triangleq E\{\mathbf{x}_{k|i}\tilde{\mathbf{x}}_{k|i}\}'$, where E denotes the expectation. In the absence of measurement origin uncertainty, the discrete time filter yields the state estimate and covariance via the recursions

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{W}_k\tilde{\mathbf{z}}_k = \mathbf{F}\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{W}_k\tilde{\mathbf{z}}_k \tag{19}$$

$$\begin{aligned}\mathbf{P}_{k|k} &= \mathbf{P}_{k|k-1} - \mathbf{W}_k\mathbf{S}_k\mathbf{W}_k' \\ &= \mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}' + \mathbf{G}\mathbf{Q}\mathbf{G}' - \mathbf{W}_k\mathbf{S}_k\mathbf{W}'_k\end{aligned} \tag{20}$$

where the innovation vector

$$\tilde{\mathbf{z}}_k \triangleq \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1} \tag{21}$$

has the covariance matrix

$$\mathbf{S}_k \triangleq E\{\tilde{\mathbf{z}}_k\tilde{\mathbf{z}}_k'\} = \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}' + \mathbf{R} \tag{22}$$

and the filter gain matrix is

$$\mathbf{W}_k = \mathbf{P}_{k|k-1}\mathbf{H}'\mathbf{S}_k^{-1} \tag{23}$$

The resulting state estimate, under the above assumptions, is the conditional mean

$$\hat{\mathbf{x}}_{k|k} = E\{\mathbf{x}_k \mid \mathbf{Z}^k\} \tag{24}$$

where $\boldsymbol{Z}^k$ denotes the set of data vector $\boldsymbol{z}_i$ for $i \le k$.

The state estimation is done for each target, but the measurement to target association probabilities are computed jointly across the targets. To account for this interdependence, consider a cluster of targets numbered $t = 1, \ldots T$ at a given time *k*. The set of *m* candidate measurements associated with this cluster (i.e. validation gates for targets $1, \ldots T$) is denoted y_j $j = 1, \ldots m$ as above. Each measurement belongs either to one of the *T* targets or to the set of false measurements which is denoted by target number *t* = 0. Denoting the predicted measurement for target *t* by the $\hat{\mathbf{z}}^t$ innovation corresponding to measurement *j* will be

$$\tilde{\mathbf{z}}_j^t \triangleq \mathbf{z}_j - \hat{\mathbf{z}}^t \tag{25}$$

and the combined innovation will be

$$\tilde{\mathbf{z}}^t = \sum_{j=1}^{m} \beta_j^t \tilde{\mathbf{z}}_j^t \qquad (26)$$

where β_j^t is the posterior probability that measurement j originated from target t and β_0^t is the probability that none of the measurements originated from target t . Thus, the JPDAF and the PDA approach utilise the same estimation equation, the difference being in the computation of association probabilities. The PDA computes β_j^t separately for each t under the assumption that all measurements not associated with target are false (i.e. Poisson distributed clutter) whereas, the JPDAF computes β_j^t jointly across the set of T targets and clutter. The key to the JPDA algorithm is the evaluation of the conditional probabilities of the following joint association events pertaining to the current time k

$$\Theta = \bigcap_{j=1}^{m_k} \theta_{jt_j} \qquad (27)$$

$$\theta_{jt} \triangleq$$

where {measurement j originates from target t}
j =1,2,.......m_k ; t = 0,1,.......T

where t_j is the index of the target to which measurement j is associated. For deriving the joint probabilities, each measurement will be assumed validated for each target, i.e. every validation gate coincides with the entire surveillance region. Thus, the PDF of each false measurement will be uniformly distributed in the entire surveillance region, yielding simpler expressions for the probabilities of the events.

5. MULTIPLE TARGETS TRACKING USING NEURAL NETWORK FUSION

The NNF filter uses a feed forward back propagation network with pure linear transfer function and network training function TRAINLM that updates weight and bias values according to “Levenberg-Marquardt” optimisation. The steps involved in the implementation of ANNs are

Selection of a network structure: (number of hidden layers, hidden nodes, and connectivity) Feed forward back propagation network was selected for the multi target tracking problem using multi sensor data fusion. The number of hidden layers is one and the number of hidden nodes is equal to the total number of scans for which the target is simulated. The input has nodes depending upon the total number of sensors and output layer has two nodes for x and y coordinates of the target.

Selection of transfer functions: The input nodes are connected through a pure linear transfer function to the hidden nodes and so are the output nodes to the hidden nodes. This is selected after lot of trials for optimum and most efficient results.

Define training function: TRAINLM is used as the training function that updates weight and bias values according to “Levenberg-Marquardt” optimization.

Train the network: The network is trained for the given inputs from sensors and the true target data as the desired output for each of the targets.

Simulate the network: The network is then simulated with the inputs from sensors and the output for each of the target is stored.

The implementation of the NNF for multi target tracking using MSDF is explained with the flowchart in Fig 2. TRAINLM is a Network training function that updates weights and bias as per “Levenberg-Marquardt” optimization. Training occurs according to the TRAINLM’s training parameters shown below with their set values in this implementation

net.trainParam.epochs	= 10	(Maximumnumber of epochs to train)
net.trainParam.goal	= 1e-2	(Performance goal)
net.trainParam.max_fail	= 5	(Maximum validation failures)
net.trainParam.mem_reduc	= 1	(Factor to use for memory/speed trade off.)
net.trainParam.min_grad	= 1e-10	(Minimum performance radient)
net.trainParam.mu	= 0.001	(Initial Mu)
net.trainParam.mu_dec	= 0.1	(Mu decrease factor)
net.trainParam.mu_inc	= 10	(Mu increase factor)
net.trainParam.mu_max	= 1e10	(Maximum Mu)
net.trainParam.show	= 25	(Epochs between displays)
net.trainParam.time	= inf	(Maximum time to train in seconds)

6. RESULTS

Various scenarios are generated to evaluate the effects of multiple and dispersed sensors for single target, two targets and multiple targets. The targets chosen are distantly spaced, closely spaced and crossing. The trajectory plots using Kalman filter for single target using two sensors with simple fusion (Chi Square statistics) are shown in Fig. 3. The fused trajectory is closest to the true trajectory. The trajectory plots using Kalman filter, with Neural Network Fusion are shown in Fig. 4 and the corresponding position errors are plotted in Fig. 5. The trajectory plots using IMM Filter with Neural Network Fusion are shown in Fig. 6 and the corresponding position errors in Fig. 7. The fused trajectory with NNF overlaps with the true trajectory as confirmed by the X and Y position errors plotted in Figs. 5 and 7, respectively. This shows that the NNF gives more accurate results when compared with the Chi Square statistics fusion technique for single target tracking

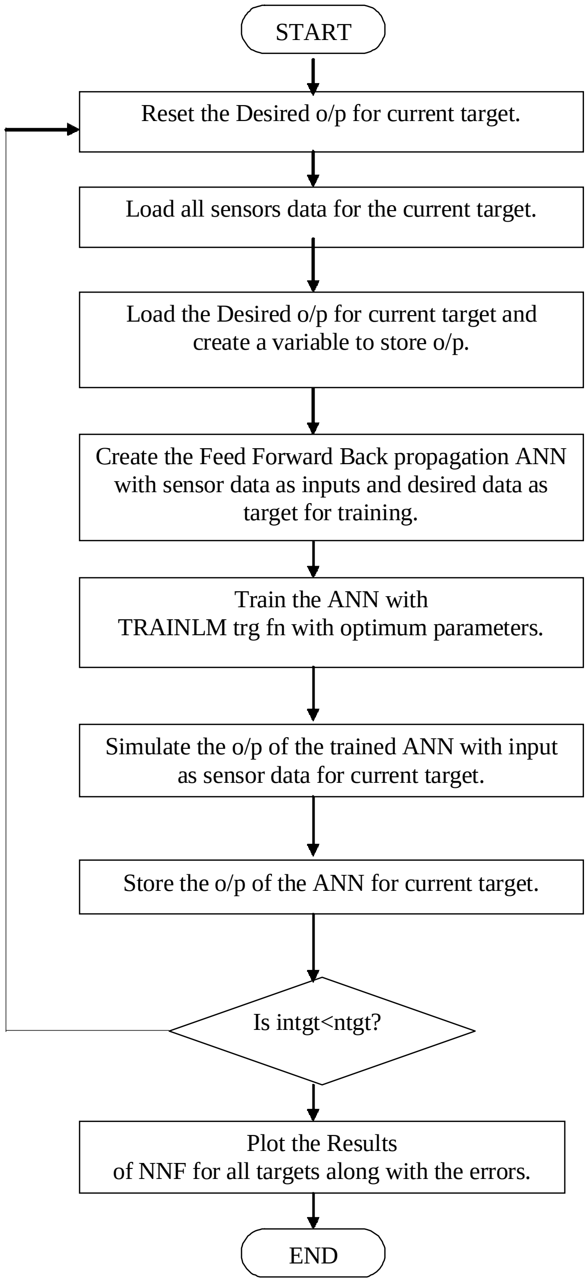
START
Reset the Desired o/p for current target.
Load all sensors data for the current target.
Load the Desired o/p for current target and create a variable to store o/p.
Create the Feed Forward Back propagation ANN with sensor data as inputs and desired data as target for training.
Train the ANN with TRAINLM trg fn with optimum parameters.
Simulate the o/p of the trained ANN with input as sensor data for current target.
Store the o/p of the ANN for current target.
Is intgt<ntgt?
Plot the Results of NNF for all targets along with the errors.
END

Figure 2. Flow chart for NNF for MTT-MSDF.

using two sensors.

The trajectory plots for two crossing targets being tracked by three sensors and their fused tracks with JPDAF are shown in Fig. 8. The JPDAF fused trajectory of each target is found to be closest to the respective true trajectories. The track switching also does not occur at the crossing point for the fused tracks. When fused track of sensor one and two is used for fusion with sensor three, the results improve as compared to the case where it is assumed that sensor one is the most accurate and does not miss any target and is used for fusion with all sensors. The trajectory plots for the same scenario with NNF are shown in Fig. 9 and the corresponding position errors are plotted in Fig.10. The NN fused trajectory of each of the targets

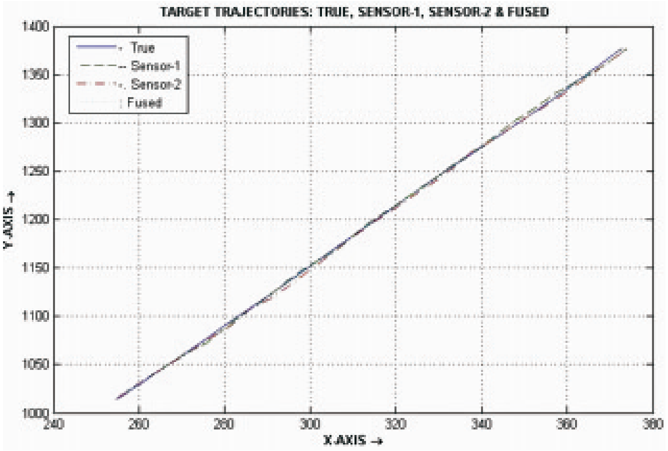

Figure 3. Single target trajectory plots (KF) for simple fusion.

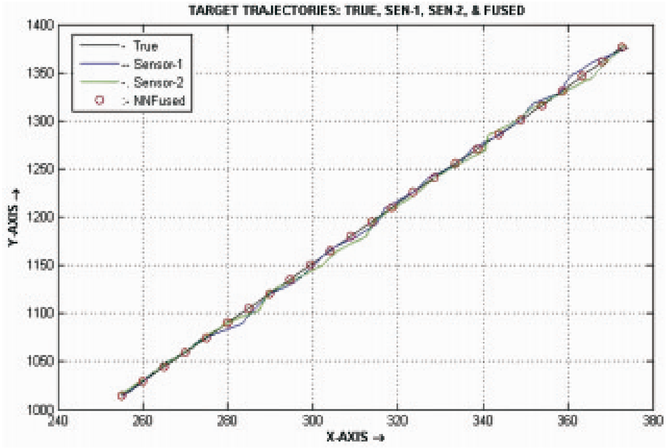

Figure 4. Single target trajectory plots (KF) for neural network fusion.

closely follows the respective true trajectory as confirmed by the *X* and *Y* position errors plots.

The trajectory plots for two and three target tracking using two sensors with JPDAF are shown in Figs 11 and 12. The JPDAF fused trajectory of each target is closest to the respective true trajectories. The trajectory plots for

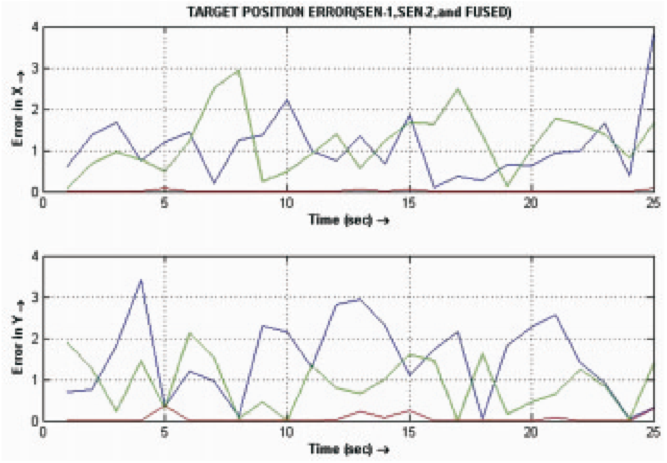

Figure 5. Single target position error plots (KF) for neural network fusion.

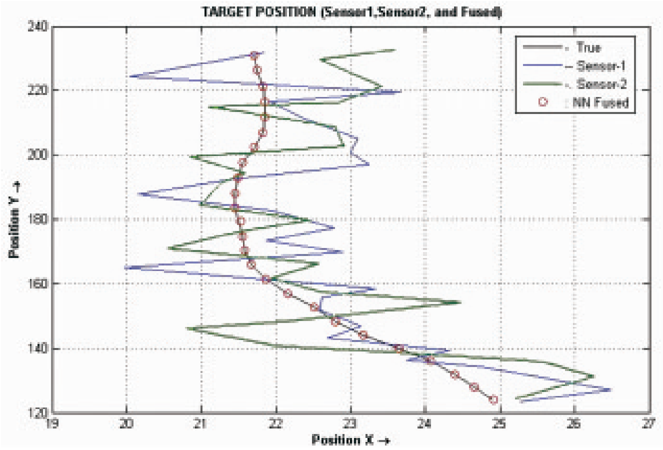

Figure 6. Single target trajectory plots (IMM) for neural network fusion.

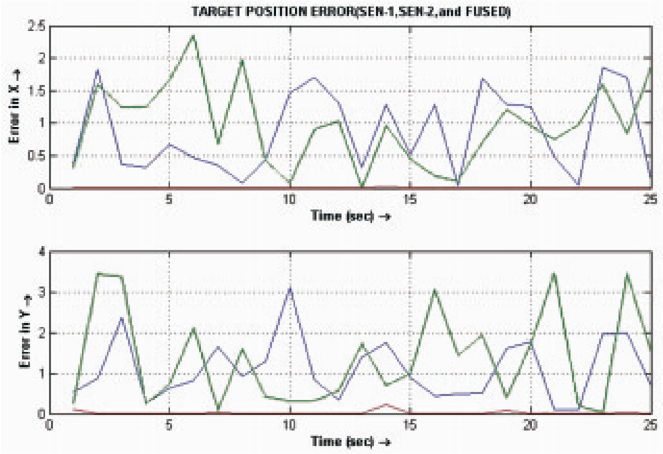

Figure 7. Single target position error plots (IMM) for neural network fusion.

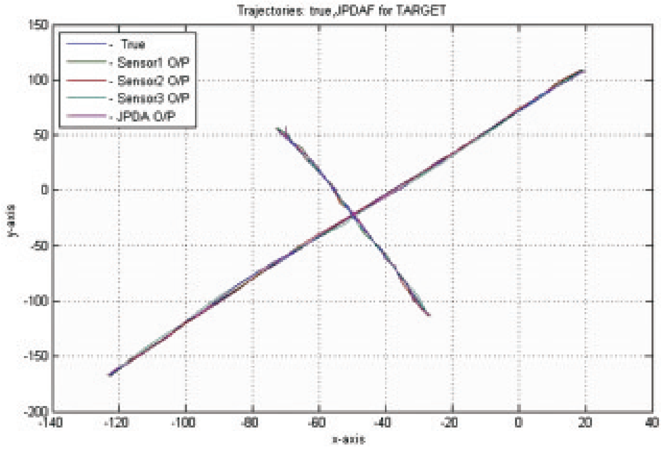

Figure 8. Two crossing targets plots for JPDAF.

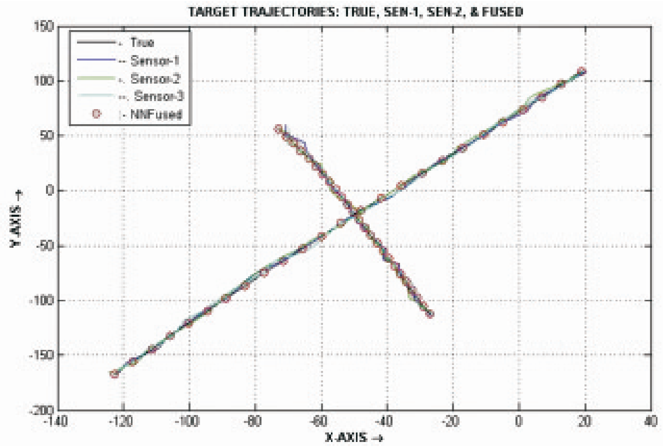

Figure 9. Two crossing targets plots for neural network fusion.

the same scenarios with NNF are shown in Figs 13 and 14 and the corresponding position errors are plotted in Figs 15 and 16. The NN-fused trajectory of each of the three targets closely follows the respective true trajectory as confirmed by the X and Y position errors plots. Thus, NNF is most accurate tracking technique and gives better results than JPDAF. In real life the sensors are dispersed and spatial alignment of the target data needs to be done before any track to track fusion algorithm is executed to get the fused trajectories. A single target being tracked by three dispersed sensors, before spatial alignment, is plotted in Fig. 17. Initial settings for Sensors tracking the target are

(a) Location of Sensor-1: [10 10 10]
(b) Location of Sensor-2: [-20 -20 -20]
(c) Location of Sensor-3: [50 50 50]
(d) Target Initial State Vector: [250 5 100 15]

The trajectory of the single target, as tracked by three dispersed sensors after spatial alignment along with the true target trajectory, is also plotted in Fig.17. This validates the data alignment transformations for each sensor. The NNF algorithm is applied to the data of a single target being tracked by two sensors in three dimensions[23], and the results are plotted in Fig.18. The position errors for all three dimensions for the two sensors and the NN fused track are plotted in Fig.19. As evident from these figures, the NN fused data give a much better estimate than the estimate from any individual sensor. This shows that the NNF technique can be successfully used for tracking targets in three dimensions as well.

7. CONCLUSION

The Kalman filter performs very well when tracking non-maneuvering, slow moving, and fixed-site targets. The Kalman filter does not handle maneuvering targets satisfactorily. The IMM is the most consistent tracker of the evaluation set for maneuvering targets. The IMM also performs fairly well with non-maneuvering targets and slow-moving targets. Results show that the fused trajectory is always closer to the true target trajectory as compared to any of the sensor measurements of that target. It is also observed that the fusion results of NNF are most accurate and follow the true trajectory very closely as compared to the JPDAF/SF trajectory for the same scenario. Also the computation time for the NNF algorithm is almost half of that taken by the JPDAF algorithm for the same scenario. The present work has been designed with great flexibility to allow for

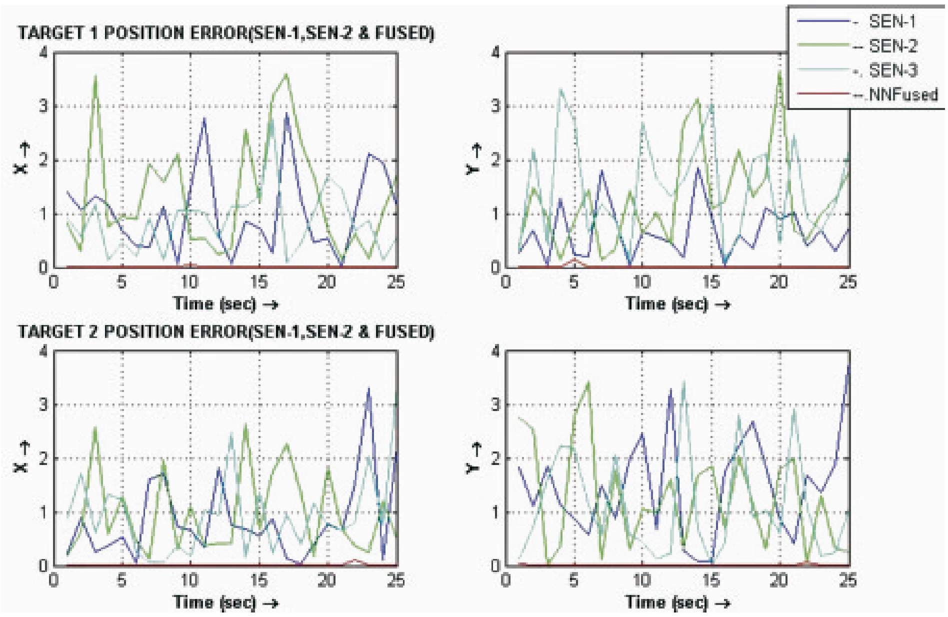

Figure 10. Two crossing targets position error plots for neural network fusion.

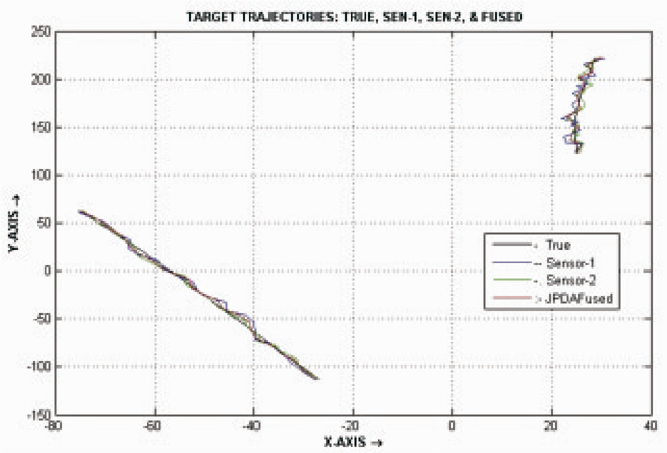

Figure 11. Target trajectory plots for JPDAF.

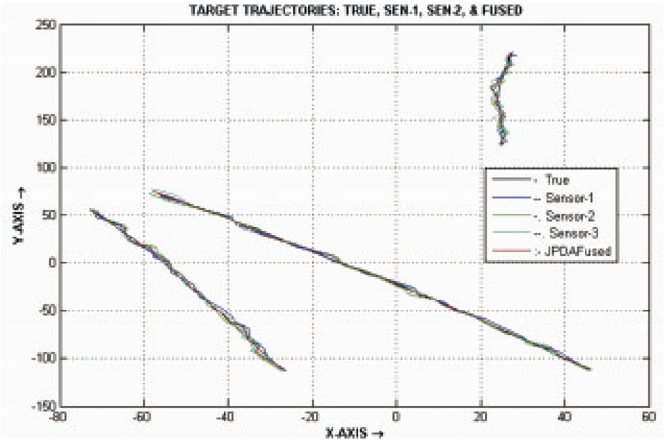

Figure 12. Three targets trajectory plots with JPDAF

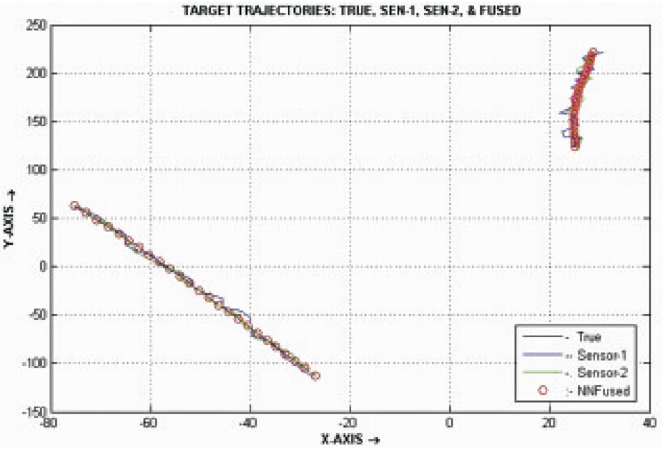

Figure 13. Target trajectory plots for NN Fusion.

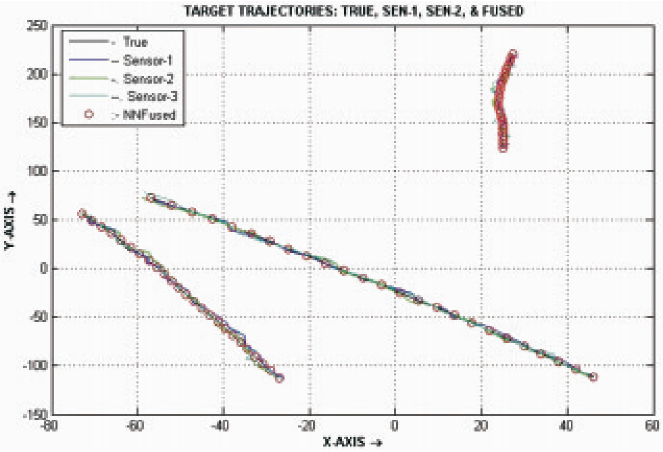

Figure 14. Three targets trajectory plots with neural network fusion.

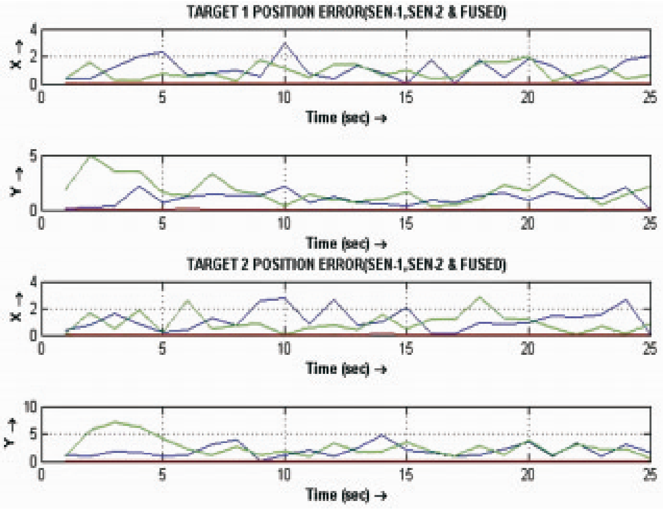

Figure 15. Position error plots (2 targets & 2 sensors) for NN Fusion.

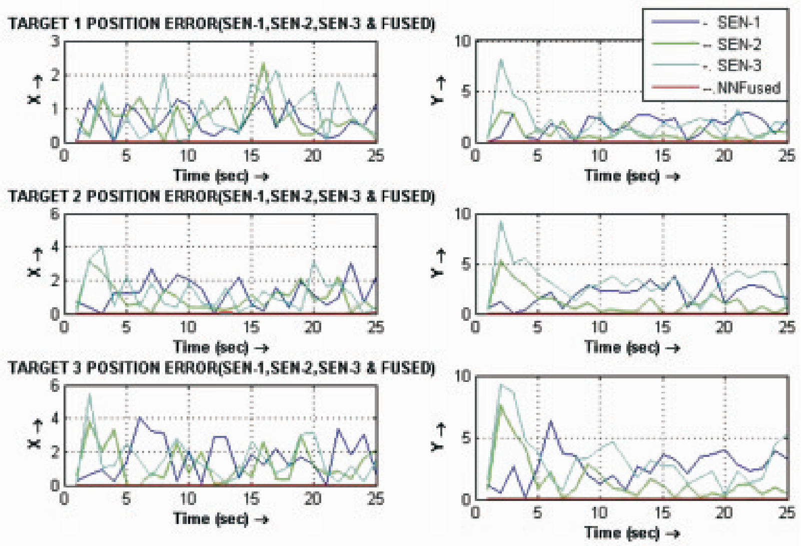

Figure 16. Three targets position error plots with neural network fusion.

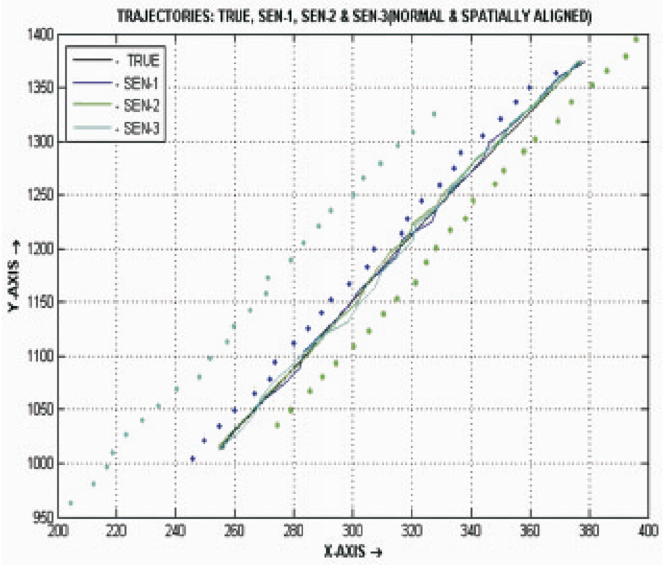

Figure 17. Trajectory plots of three dispersed sensors (before & after spatial alignment).

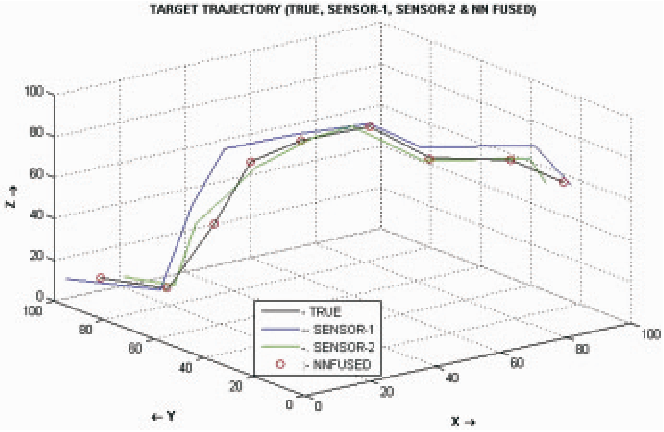

Figure 18. Target trajectory plots in 3-D (single target & 2 sensors) for NN fusion.

future applications and upgrades. To change parameters, the user can alter the MATLAB code within the applicable files to produce the desired output. This program can be used to compare tracking algorithms in a multiple sensor/multiple target environment, view the effects of using dispersed sensors, and also evaluate the effects of data association algorithms.

ACKNOWLADGEMENTS

Authors are thankful to Prof G C Pant, Chairman, Department of Aerospace Engineering, Defence Institute of Advanced Technology(DIAT), Girinagar, Pune, for the encouragement and also to VC, DIAT, for granting permission to publish this work.

REFERENCES

1. Waltz, E. & Llinas, J. Hand book of multi-sensor data fusion. Artech House, 2000.
2. Hall, D. Mathematical techniques in multisensor data fusion. Artech House, 2004.
3. White, Franklin E. & Llinas, J. Data fusion:

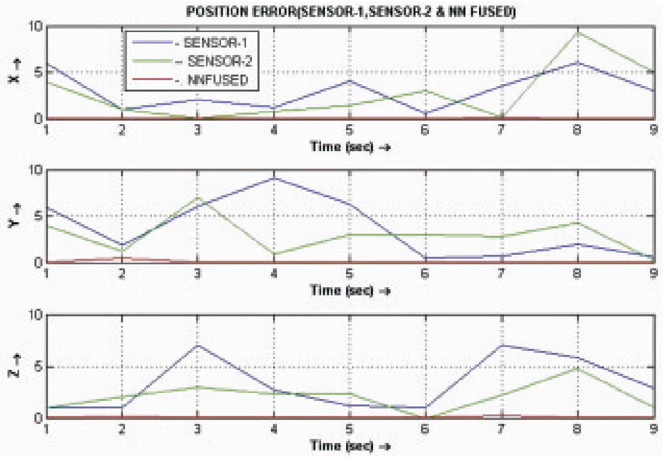

Figure 19. Error plots in 3-d (single target & two sensors) for NN fusion.

The Process of C^{3}I, technology update. *Def. Electron.*, June 1990, 77-83.

4. Waltz, E. & Llinas, J. Multi-sensor data fusion, Artech House, 1990.
5. Karlson, Bjorn; Jarrhed, Jan-Ove. & Peter Wide. A fusion toolbox for sensor data fusion in industrial recycling. *IEEE Trans,Instr. Measure.*, 2002, **51**(1), 144-49.
6. Zachary, J. M. & Iyengar, S.S. Three-dimensional data fusion for biomedical surface reconstruction.*In* Proceedings of the second international conference on information fusion, Sunnyvale, California, USA, July 6-8,1999, **1**, pp. 39-45.
7. Ramac, L.C.& Varshney, P.K. Application of image fusion to wireless image transmission. *In* Proceedingsof the Second International Conference on Information Fusion, Sunnyvale, California, USA, July 6-8,1999,.**1**, pp.207-12.
8. Wuyi,Yu; Xiaoping Qi & Zou L. Evaluation of multi sensor remote sensing data applied for oil and gas exploration. *In* Twenty-second Asian Conference on Remote Sensing, Singapore, 5-9 Nov 2001.
9. Abuhadrous, I. 3-D land vehicle localisation: a real-time multi-sensor data fusion approach using RTMAPS. Robotics Center-Ecole des Mines de Paris, 2003.
10. Hill, C., Multi-sensor data fusion for collision avoidance using neuro-fuzzy estimators. *In* Fusion 98, International Conference on Multi-Source Multi-Sensor Information Fusion, Nevada, USA, July 6-9, 1998.
11. Bar-Shalom, Y., & Li, X-R. Multi-target- multi-sensor tracking: Principles and techniques. Academic Press, 1995.
12. Bar-Shalom, Y. & Fortmann, T. E. Tracking and data association, Academic Press, 1988.
13. Bar-Shalom, Y. Multi-target multi-sensor tracking: advanced applications, Artech House, 1990.
14. Bar-Shalom, Y. & Li, X. R. Estimation and tracking: principles, techniques, and software. Artech House, 1993.
15. Bar-Shalom, Y. Multi -target multi-sensor tracking: applications and advances. **II**, Artech House, 1992.
16. Tugnait, J.K. Tracking of multiple manoeuvring targets in clutter using multiple sensors, IMM, and JPDA Coupled Filtering. In *IEEE Trans. AES*, 2004, **40**(1), 320-30.
17. Kirubarajan, T. & Bar-Shalom, Y. Probabilistic data association techniques for target tracking in clutter. *IEEE Proceeding.*, 2004, **92**(3), 536-56.
18. Bar-Shalom, Y. On the track-to-track correlation problem. *IEEE Trans. Autom. Cont.*, 1981, **AC-26**(2), 571-72.
19. Bar-Shalom,Y. The Effect of the common process noise on the two sensor fused track covariance. *IEEE Trans. Aerosp. Electron. Sys.*, 1986, **AES-22**(2), 803-05.
20. Singer, R.A. & Kanyuck A. J., Correlation of multiple-site track data. *IEEE Trans. Aerosp. Electron. Sys.*, 1970, **AES-6**(2), 180-87.
21. Bar-Shalom Y. & Chang, K.C. On optimal track to track fusion. *IEEE Trans. Aerosp. Electr. Sys.*, 1997, **AES-33**(4), 1271-275.
22. Blackman, S.S. Multiple-target tracking with radar applications, Artech House, 1993.
23. Girija,G.; Raol, J.R., Raj, R. Appavu. & Kashyap, S. Tracking filter and multi-sensor data fusion. *Sadhna* 2000, **25**(2),159-67.

Contributor

Dr A.K. Singh obtained his PhD from University of Rajasthan, Jaipur. He joined DRDO in 1991. Presently, he is working in the Department of Aerospace Engineering at the Defence Institute of Advanced Technology (DIAT), Deemed University, Girinagar, Pune. He has done many sponsored projects related to sensors and nanotechnology. His areas of research include: piezoelectric sensors, thermal sensors, PC based instrumentations, smart materials, nano material synthesis and characterisation, etc. He has published more than 50 papers in various national and international journals.