Abstract: The paper presents a simulation study on the generation of a random scenario for the performance of track splitting algorithm on a digital signal processor. Much of the previous work [1] was done on specific (deterministic) scenarios. One of the reasons for considering the specific scenarios, which were normally crossing targets, was to test the efficiency of the track splitting algorithm for different situations. However this approach only gives a measure of performance for a specific, possibly unrealistic, scenario and it was felt appropriate to develop procedures that would enable a more general performance assessment. Therefore, a random target motion scenario is adopted. Its implementation in particular for testing the track splitting algorithm using Kalman filters is used and a couple of tracking performance parameters are computed to investigate such random scenarios.

Keywords: Kalman filter, Target tracking, State estimation, Track splitting algorithm.

1. Background

Tracking of a single target, in the ideal situation where one noisy measurement is obtained at each radar scan, can be done using standard Kalman filter techniques. In the multi-target case, an unknown number of measurements are received at each radar scan and, assuming no false measurements, each measurement has to be associated with an existing or new target tracking filter. When the targets are well apart from each other then forming a measurement prediction ellipse around a track to associate the correct measurement with that track is a standard technique [2]. When targets are near to each other, then more than one measurement may fall within the prediction ellipse of a filter and prediction ellipses of different filters may interact. The number of measurements accepted by a filter will therefore be quite sensitive in this situation to the accuracy of the prediction ellipse. Several approaches may be used for this situation [3][4][5], one of that is called the track splitting algorithm. In this algorithm, if n measurements occur inside a prediction ellipse, then the filter branches or splits into n tracking filters. This situation, which results in an increased number of filters, makes the algorithm not only computationally expensive but also the memory requirements also grow exponentially. Some mechanism for restricting the excess tracks that originated from track splitting is required, since eventually this process may result in more than one filter tracking the same target. The first criterion is the support function which uses the likelihood function of a track as the pruning criterion and the second the similarity criterion which uses a distance
threshold to prune similar filters tracking the same target [1].

2. Motion Model Consideration

The motion of a target being tracked is assumed to be approximately linear and modeled by the equations

\[ x_{n+1} = \Phi x_n + \Gamma w_n \]  
\[ z_{n+1} = H x_{n+1} + \nu_{n+1} \]

Where the state vector

\[ x_{n+1} = (x \ x' \ y \ y')_{n+1} \]

is a four-dimensional vector, \( w_n \) the two-dimensional disturbance vector, \( z_{n+1} \) the two dimensional measurement vector and \( \nu_{n+1} \) is the two-dimensional measurement error vector. Also \( \Phi \) is the assumed (4x4) state transition matrix, \( \Gamma \) (4x2) is the excitation matrix and \( H \) (2x4) is the measurement matrix and they are defined respectively,

\[ \Phi = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \]  
\[ \Gamma = \begin{bmatrix} \Delta t^2 / 2 & 0 \\ \Delta t & 0 \\ 0 & \Delta t^2 / 2 \\ 0 & \Delta t \end{bmatrix} \]  
\[ H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \]

Here \( \Delta t \) is the sampling interval and corresponds to the time interval (scan interval) assumed constant, at which radar measurement data is received.

The system noise sequence \( w_n \) is a two dimensional Gaussian white sequence for which

\[ E(w_n) = 0 \]

where \( E \) is the expectation operator. The covariance of \( w_n \) is

\[ E(w_n w_n^T) = Q_n \delta_{nm} \]

where \( Q_n \) is a positive semi-definite (2x2) diagonal matrix and \( \delta_{nm} \) is the Kronecker delta defined as

\[ \delta_{nm} = \begin{cases} 0 & n \neq m \\ 1 & n = m \end{cases} \]

The measurement noise sequence \( \nu_n \) is a two-dimensional zero mean Gaussian white sequence with a covariance of

\[ E(\nu_n \ \nu_n^T) = R_n \delta_{nm} \]

where \( R_n \) is a positive semi-definite symmetric (2x2) matrix given by

\[ R_n = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{bmatrix} \]

\( \sigma_x^2 \) and \( \sigma_y^2 \) are the variances in the errors of the \( x, y \) position measurements, and \( \sigma_{xy} \) is the covariance between the \( x \) and \( y \) measurement errors. It is assumed that the measurement noise sequence and the system noise sequence are independent of each other, that is

\[ E(\nu_n w_n^T) = 0 \]

The initial state \( x_0 \) is also assumed independent of the \( w_n \) and \( \nu_n \) sequences that is

\[ E(x_0 \ w_n^T) = 0 \]
\[ E(\mathbf{x}_0, v_n^T) = 0 \] (13)

\[ \mathbf{x}_0 \] is a four dimensional random vector with mean \( E(\mathbf{x}_0) = \bar{\mathbf{x}}_{0/0} \) and a \((4 \times 4)\) positive semi-definite covariance matrix defined by

\[ P_0 = E[(\mathbf{x}_0 - \bar{\mathbf{x}}_0)(\mathbf{x}_0 - \bar{\mathbf{x}}_0)^T] \] (14)

where \( \bar{\mathbf{x}}_0 \) is the mean of the initial state \( \mathbf{x}_0 \). The Kalman filter is an optimal filter as it minimizes the mean squared error between the estimated state and the true (actual) state provided the target dynamics are correctly modeled.

The standard Kalman filter equations for estimating the position and velocity of the target motion described by equations [1] & [2] are;

\[ \hat{\mathbf{x}}_{n+1|n} = \Phi \hat{\mathbf{x}}_n \] (15)

\[ \hat{\mathbf{x}}_{n+1} = \hat{\mathbf{x}}_{n+1|n} + K_{n+1} \mathbf{v}_{n+1} \] (16)

\[ K_{n+1} = P_{n+1|n} H^T \Sigma^{-1} \] (17)

\[ P_{n+1|n} = \Phi P_{n+1|n} \Phi^T + \Gamma Q_n \Gamma^T \] (18)

\[ B_{n+1} = R_{n+1} + HP_{n+1|n} H^T \] (19)

\[ P_{n+1} = (I - K_{n+1} H)P_{n+1|n} \] (20)

\[ \mathbf{v}_{n+1} = \mathbf{z}_{n+1} - H \hat{\mathbf{x}}_{n+1|n} \] (21)

Where \( \hat{\mathbf{x}}_{n+1|n}, \hat{\mathbf{x}}_{n+1}, K_{n+1}, P_{n+1|n}, B_{n+1} \)

and \( P_{n+1} \) are the predicted state, estimated state, the Kalman gain matrix, the prediction covariance matrix, the covariance matrix of innovation, and the covariance matrix of estimation respectively. \( Q_n^F \) is the covariance of the measurement noise assumed by the filter which is normally taken equal to \( Q_n \). In a practical situation, however, the value of \( Q_n \) is not known so the choice of \( Q_n^F \) should be such that the filter can adequately track any possible motion of the target. To start the computation an initial value is chosen for \( P_0 \). Even if this is a diagonal matrix, then clearly from the above equations the covariance matrices \( B_{n+1}, P_{n+1}, P_{n+1|n} \) for a given \( n \) do not remain diagonal when \( R_n \) is not diagonal.

3. Random Scenario Generation

The targets move in the X-Y plane, and the positions of the targets are considered with respect to the tracker located at the fixed origin of coordinates. The initial target positions are randomly selected in a predefined global space such that they are uniformly distributed inside the global space. The directions of the targets are also randomly selected between \( 0 - 2\pi \). The initial velocity of the targets is taken from a random distribution by specifying a mean and standard deviation for the velocity. The targets follow a random velocity path given by equation 1. A detection window is defined inside the global space so that only those targets are tracked which fall inside this window. The data for different target scenarios is generated by specifying the target density, the mean value and variance for the initial velocity and the probability of detection. Other parameters such as the radar resolution, acceleration noise in the target model and measurement noise can also be changed according to the requirements of the scenario. The density of the targets for a complete run remains constant by replacing those targets which leave the tracking window by other targets whose initial positions, velocity and heading are again selected randomly as described earlier.

4. Performance Measure

A single parameter for the performance evaluation of a multiple target tracking algorithm is difficult to obtain. The target tracking problem is statistical in nature and many factors enters performance assessment. For example, one tracking algorithm may be computationally efficient but lose true tracks for a significant time.
On the other hand another algorithm may perform better in tracking accuracy and rarely lose the true tracks but require more computation time. A practical approach for the assessment of a multiple target tracking algorithm is to use simulation studies, typically analytical methods are somewhat complicated. We are investigating three parameters which seem logical for the described situation:

- **Terror**: The average tracking error that is the difference between true target positions and estimated positions.
- **Rtm**: The ratio of the number of branches to the number of true targets.
- **Tt**: The average time for which a particular track is maintained, that is from initialization to track loss.

As it is said earlier we are using Kalman filters, although a less expensive $\alpha-\beta$ algorithm in terms of space and computation is more attractive but simulations have shown that the trade-off in using Kalman filter is; better measurement prediction ellipse and support function assessment, which are important factors when multiple target exist \[6\][7][8][9]. The average tracking error for the \( x \) coordinate is given by;

\[
T_x(\mathcal{V}) = \sqrt{\frac{1}{k} \sum_{i=1}^{k} \frac{1}{b} \sum_{j=1}^{b} (X_i(\mathcal{V})^j - \hat{X}_i(j))^2} \quad (22)
\]

\[
T_y(\mathcal{V}) = \sqrt{\frac{1}{k} \sum_{i=1}^{k} \frac{1}{b} \sum_{j=1}^{b} (Y_i(\mathcal{V})^j - \hat{Y}_i(j))^2} \quad (23)
\]

where \( k \) is the number of scans, \( b \) is the number of branches belonging to the tree of track \( \mathcal{V} \), \( X_i(\mathcal{V})^j \) is the true position of track \( \mathcal{V} \) (noise free measurement known from the measurement generation program) at scan \( i \) and \( \hat{X}_i(j)^j \) is the track estimate of branch \( j \) at scan \( i \). The global average tracking error is then defined as;

\[
T_{error} = \sqrt{\frac{1}{T} \sum_{i=1}^{T} \frac{1}{2} [T_x(i) + T_y(i)]^2} \quad (24)
\]

Even when all measurements are correctly taken by a filter from one target an average tracking error will exist due to the statistical nature of the problem. In a multiple target tracking environment, however, since incorrect measurements may also be taken by a specific filter this also affect the tracking accuracy.

\( R_{tm} \) is another important parameter which tells actually what kind of activity present in the tracking area. A value of unity for the ratio of the number of branches to the number of true targets \( (R_{tm}) \) indicates that a system employing the track splitting algorithm is working perfectly. In a multiple target tracking environment with crossing targets, maneuvering targets and false measurements, a unit value for \( R_{tm} \) will not be possible. However, it is generally advisable to have more tracks rather than lose them, that is \( R_{tm} > 1.0 \). Another parameter we selected provides information about sustained tracking period. The tracking time \( T_t \) for any target should be equal to the number of scans for which it has been detected. In a multiple target tracking scenario a correct target motion model is not enough to ensure successful track maintenance due to the presence of multiple measurements, since the acceptance of a measurement from any neighboring target may result in termination of the true track through similarity. Because of the track splitting process the lost target may be absorbed by the neighboring targets, therefore the algorithm may require a few scans to reinitiate the same target which results in a discrepancy (gap) in the track filter life for that particular target.

5. Implementation & Simulation

To simulate the output of a radar, a data generator routine was written in C to run on a TMS320C6713 DSK board and the
Parameters describing the simulation can be entered interactively by the user or defined as default prior to compiling. The trajectories for the targets are generated using the kinematics described in equation 1, namely a constant velocity motion with acceleration noise. Simulation results for three random scenarios with target densities giving 10, 20 and 30 targets respectively have been obtained. The normalized performance evaluation parameters for these scenarios are given in Table 1. The tracking window space is 100 by 100 and the results are for a single run of 100 scan intervals. From Table 1 it can also be seen that the performance evaluation parameters do not vary a lot if the density of targets in the tracking window is reasonable, meaning as long as the targets are spread over the whole space.

<table>
<thead>
<tr>
<th>Performance Parameters</th>
<th>10 Targets</th>
<th>20 Targets</th>
<th>30 Targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{tm}^{n}$</td>
<td>0.00</td>
<td>0.001</td>
<td>0.0025</td>
</tr>
<tr>
<td>$T_{t}^{n}$</td>
<td>1.00</td>
<td>0.999</td>
<td>0.995</td>
</tr>
<tr>
<td>$T_{error}^{n}$</td>
<td>0.09</td>
<td>0.14</td>
<td>0.20</td>
</tr>
</tbody>
</table>

Table 1: Tracking Window (100 x 100)

Figure 1 shows expanded tracking window with 20 targets, although some places targets are very close to each other and giving us the illusion of crossing or moving parallel to each other. In fact the tracking algorithm only perform badly if the crossing or parallel movement is at the same scan. However, this is not the case in this particular scenario, therefore, the tracking algorithm is able to track it very efficiently and the track paths (filtered) are shown in Figure 2. Here, we would like to point out that as the target position, target heading are all randomly selected so depending upon the seed they can appear anywhere in the 100 x 100 window.

In our second simulated scenario with 30 targets appearing in 20 x 20 windows, the performance is degraded as one might expect. The probable reason being targets are in close proximity at the same scan number. Table 2 gives the normalized performance parameters for this scenario and it can be seen as the number is increased only of the order of 10 targets each time the performance parameters are showing an exponential degradation.

<table>
<thead>
<tr>
<th>Performance Parameters</th>
<th>10 Targets</th>
<th>20 Targets</th>
<th>30 Targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{tm}^{n}$</td>
<td>0.00</td>
<td>0.0035</td>
<td>0.005</td>
</tr>
<tr>
<td>$T_{t}^{n}$</td>
<td>1.00</td>
<td>0.983</td>
<td>0.951</td>
</tr>
<tr>
<td>$T_{error}^{n}$</td>
<td>0.098</td>
<td>0.23</td>
<td>0.41</td>
</tr>
</tbody>
</table>

Table 2: Tracking Window (20 x 20)

Figure 3 shows the targets observation paths and it can be seen that not only targets are close to each other but also they are re-appearing at various positions randomly at
the same scan time. Figure 4 shows the track paths for these targets the joining lines are kept to show which target belong to which once they disappear/re-appear.

6. Conclusions

In this paper the performance of a track splitting algorithm, using a random scenario has been studied. The track splitting approach requires a large number of tracking filters so not only less accurate tracking is observed but also memory and computational requirements grow exponentially if multiple targets exist in the same vicinity for a longer period of time. As expected the study here has found that when the tracking window becomes denser all the performance parameters deteriorate exponentially. For scenarios having more number of targets in the same vicinity some time may represent an unrealistic situation. However, obtaining empirical values for various performance parameters provide a more in depth vision to understand the situation. The parameters values can help in the design and development of a tracking system. This study has used a simple simulated approach instead of more complicated analytical which may be the best approach.

7. References


