Design and Development of Probabilistic Data Association Filters for Complete Object Refinement

Shyam Srinivasan[1], Gangadhar N. D. [2], Hariharan Ramasangu[3]
1 – M.Sc. (Engg) Student, 2 – Professor and Head, Department of Computer Engineering,
3 – Head and Head, Department of Electronics and Electrical Engineering,
M. S. Ramaiah School of Advanced Studies, Bangalore 560 058

Abstract

Over the recent past, significant attention has been focused on the use of multiple sensors for target tracking over a large geographic area, as using a single sensor with a very large range is highly impractical. In addition, data from multiple sensors can be combined to provide improved estimation. Such Multi Sensor Data Fusion (MSDF) is similar to how humans and animals use multiple senses for perception. Probabilistic Data Association algorithms for Multiple Target Tracking (MTT), such as PDA and JPDA, use the statistical knowledge of the target dynamics and sensor measurement to obtain improved tracking.

In this work, Probabilistic Data Association filters (i.e., both PDA and JPDA filters) for Complete Object Refinement have been designed, developed and studied. The designed PDA algorithms have been developed using a Track Oriented approach. For JPDA, association matrices are determined at each scan based on the current targets and measurements; from this the set of event matrices are formed which is observed to be the most computationally intensive part of the JPDA. A radar simulator for generating synthetic 3D target measurements along with false measurements is developed. The system has been developed using an Object Oriented Development scheme and implemented as a plug-in architecture in which new features can be added.

The developed PDA and JPDA filters are extensively tested using synthetic track measurement data produced using the radar simulator. It was observed from the test results that PDA has lesser computation load in comparison to JPDA, but fails in the case of multiple interfering targets in a dense clutter. JPDA is successful in tracking such interfering targets due its joint association between targets and measurements. Due to JPDA's computation load only about 10 targets could be tracked, hence, clustering has to be used to decrease the computational load. PDA on the other hand is able to track more than 150 targets per scan. The work carried out can be used as a foundation for further development of target tracking systems with complex tracking needs.

Key Words: JPDA, MTT, MSDF, Multiple Target Tracking, Multi Sensor Data Fusion

1. INTRODUCTION

The processes of tracking targets for either military applications or for surveillance was a manual process of estimation which required human intelligence and intervention until the advent of sensors and computing methods for keeping track of targets. The emergence of sensors and advanced processing techniques using computers gave impetus to automated target tracking. Over the past few decades, sophisticated algorithms and techniques have been developed by independent researchers and organizations notably the Department of Defense (DoD) to automate target tracking. These techniques are particularly useful for military applications such as tracking targets either over the air or on the ground or under water.

The development of technology for target tracking was brought into prominence during the cold war era, with the Soviets and Americans looking to develop better and more accurate ways of pinpoint tracking of objects like cruise missiles, heavy artillery, and warships and so on. The Kalman Filter algorithm and its variants first addressed the real time target tracking issue. The use of Kalman Filter in trajectory estimation was started during the 1960's with the Apollo program. This was the first full implementation of the Kalman filter for a complex real time process. The ability of the Kalman Filter to give precise estimates for a well-defined system despite inordinately high measurement noise made it indispensable to the present day target tracking systems. The computational efficiency of the Kalman Filter, along with its ease of implementation, especially in computer programming has made it an ideal tracking filter for target tracking applications that merge image processing and filtering/estimation.

With the advent of powerful computing hardware, networking and sophisticated sensors with the ability to sense a change in the sensing environment within a few micro seconds, "target tracking" has become a highly sought after domain for researchers and organizations alike. Over the past few decades, significant attention has been focused on multiple sensors for keeping track over a larger geographic area since a single sensor with a very large range is highly impractical. Also, multiple sensors give more information for improved situation assessment, since one sensor may be blind to some parameters.

Multiple Target Tracking (MTT) refers to target tracking where multiple targets are of interest and hence, all of them have to be simultaneously tracked. MTT Filter is

based on the basic Kalman Filter, which is extended to keep track of multiple targets using stochastic or probabilistic techniques such as Multiple Hypotheses or Probabilistic Data Association (PDA). Joint Probabilistic Data Association (JPDA), which is an extension to PDA, is used for tracking multiple objects of interest for Complete Object Refinement of the Joint Directors of Laboratories (JDL) Data Fusion Working Group, which was established in 1986 as an effort to codify the terminology related to data fusion. JDL is a process model for data fusion.

2. PROBLEM DEFINITION

This work is motivated by an interest to develop and implement a Multiple Target Tracker using the Probabilistic Data Association Filters for Complete Object Refinement. The work done addresses MTT using MSDF i.e., multiple target tracking using fusion of the information received from multiple sensors as already introduced. The work is done to mainly review the many flavours of PDA and point out the best among them. The developed system is also tested for validity.

3. DESIGN

In this section the *high level design* of the system is discussed. The components of the system and the Software Architecture are discussed using a context diagram. The block diagram of the system and also the design alternatives and the choice of design are discussed.

3.1 Overview of the System

The main entities in the MTT System are the sensors, fusion processor, state estimation filters, data association algorithms and the user interface needed to show the output and the input to the system. The overview of the system shows the entry point and exit point of the software being developed. This gives the main idea behind the design and development of the software.

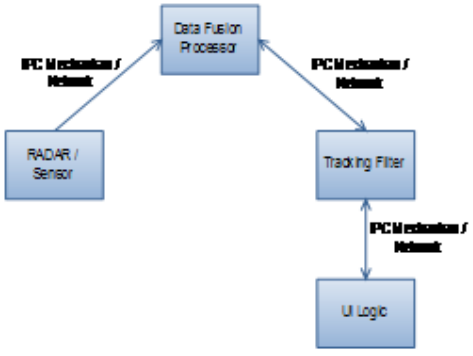

Fig. 1 MTT system overview

Fig. 1 shows the overview of the system to be designed. The overview shows the interacting components in the system. In an MTT System a sensor/RADAR is needed to point out the correct position of the target to be captured. An important consideration to be made here is that the sensor is not perfect and hence, detects a lot of noise. Hence, the sensor measurements are normally made up of the actual coordinates of a target as well as noise. These measurements are sent to a fusion processor, which in turn maintains the data in a data store. The tracking filter accesses these stored data using a chosen communication mechanism and estimates the state of the system. The processed data is then shown as a part of the UI Logic. Taking this overview a context of the system is derived.

3.2 Block Diagram

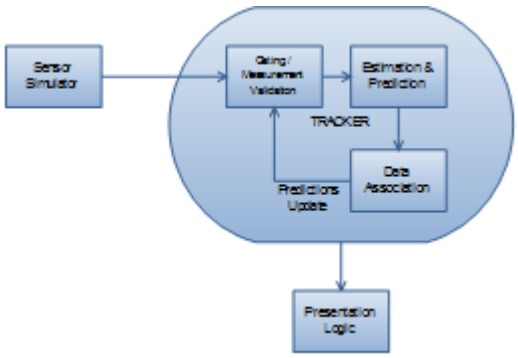

Fig. 2 Block diagram of MTT system

Fig. 2 shows the block diagram of the MTT System. As it can be seen the main block is the "Tracker" which houses the PDAF or the corresponding filter. In this paper the implementation details of both PDA as well as JPDA filters would be discussed along with their comparison. The "Tracker" block consists of all necessary sub–functionalities such as the measurement validation block, estimation, predication block and the data association block needed to keep track of a set of targets. The "Tracker" block gets its input from the "sensor simulator" block and gives the output to the "presentation logic" block with in turn gives measurements and shows the output correspondingly.

3.3 Class Diagram of the MTT System Software

The class diagram shows the necessary classes that have been identified as a part of Object Oriented Analysis of a system. The MTT System is designed as classes and objects.

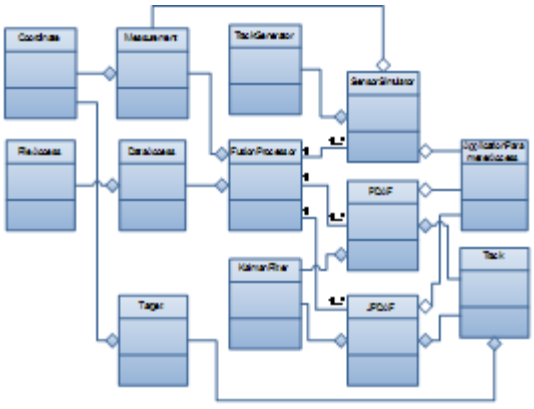

Fig. 3 Class diagram of MTT system

Fig. 3 shows the class diagram of the MTT System. As it can be seen even the sensor simulation has been taken to

be a part of the MTT System. Both JPDA and PDA algorithms will be developed so as to understand the working of an MTT System and also to be able to compare the two algorithms under consideration. The class diagram has been directly derived from the requirements.

Every sensor produces measurements for each scan, which can be either false alarms or the actual target coordinates. In this case the sensor is not a physical entity where as a simulated entity and hence, the name "SensorSimulator". The simulated "SensorSimulator" is dependent on the "TrackGenerator" for generating the actual target track that is to be tracked. Each "Measurement" consists of a set of "Coordinates". Every sensor uses the services of the "FusionProcessor" to store their measurements which, in turn uses the services of the "DataAccess" class to store the data which, in this case is the measurement. The "DataAccess" class encapsulates the file access mechanisms and functionalities of the "FileAccess" class. This is done so that changes in the file access methods do not affect the interfaces used by the "FusionProcessor". The target tracking filter classes i.e., "PDAF" and "JPDAF" use the functionalities of the "FusionProcessor" to access the scan details. The target tracking filter classes use the state estimation filter, i.e., "KalmanFilter" for predicting the next state of the system. "ApplicationParameterAccess" class consists of static methods which in turn provide the necessary parameters to the other classes such as number of simulation runs and so on. "Target" is the class which maintains the current state of a target. "Track" is the class, which consists of all the previous known state of the targets. It is a set of "Target" objects.

3.4 Sensor Simulator

Since, there is no physical sensor present for giving input to the target tracker the data provided by the sensor it has been simulated. The "SensorSimulator" class consists of the functionalities necessary for giving sensor output. It can be assumed that a sensor gives the coordinates of the entities that have been detected by it. Every sensor has a Signal to Noise Ratio (SNR) due to which no sensor is ever perfect. Hence, it can be established that every sensor makes false measurements. Research has shown that the probability of false alarm (P_{FA}) is assumed to be Uniformly Distributed in the measurement volume. Hence, along with the actual target coordinates, Uniformly Distributed random measurements are added. The actual target coordinates themselves are simulated by the use of a track generator. Similarly with a good quality sensor the detection probability (P_D) can be assumed to be 0.99 i.e., the target detection happens with a surety of 99%. Also the P_{FA} can be assumed to be 0.01.

Fig. 4 shows the required flowchart for the sensor simulation. As it can be seen the simulation is carried out for a set number of runs and for each simulation run both the target coordinates and random uniformly distributed noise are summed up as measurements and are sent to the fusion processor which in turn stores the scan details

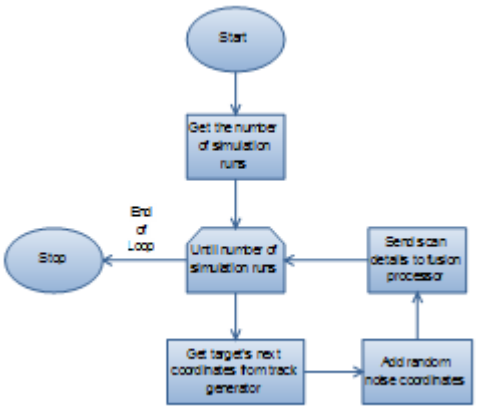

Fig. 4 Flowchart of sensor simulator

3.5 Performance Evaluation of Tracker

Performance of a target tracking application is seen as the ability of tracker to predict as closely as possible the next state of the target's future position and attributes. The performance evaluation method finds out the deviation of the predicted state of a target from the perfect state of a target and hence, tries to establish the performance of the tracker. Using this inference about the tracking ability of a particular tracker can be made.

3.6 Algorithm Test Performance

```
;Input – None
;Output – Dumps the measured performance
;                metrics into a text file.
Start
        getScanDetails for first scan from scanDump.txt
        getOutputDetails of tracking filter; JPDA/PDA
        deviation <- original – predicted
        store into file <- deviation
end
```

The algorithm given above can be modified to include more parameters such as difference between the "starting time" and the "ending time" of processing and hence, finding out the time efficiency. In this way the performance can be evaluated.

3.7 General Steps for Developing a Tracker

The normal steps of operation for target tracking are:

Start

Step 1: Get first scan details

Step 2: Perform state estimation with initial conditions

Step 3: Get next scan details

Step 4: Perform gating and measurement validation

Step 5: Perform data association using appropriate data association filter

Step 6: Update the state estimation equations

Step 7: Go to Step 3 if there are more scan details available

Step 8: Display the tracked target's set of coordinates

Step 9: End

Stop

3.8 State Estimation

The MTT System is a *linear discrete time controlled dynamic system*. Hence, a method is necessary to predict/estimate the next state of the system. The literature review suggests the use of the Kalman Filter for this purpose. The necessary background for the Kalman Filter has been described by Chen (2003). In this section the design of the Kalman Filter in the form of a flowchart has been described.

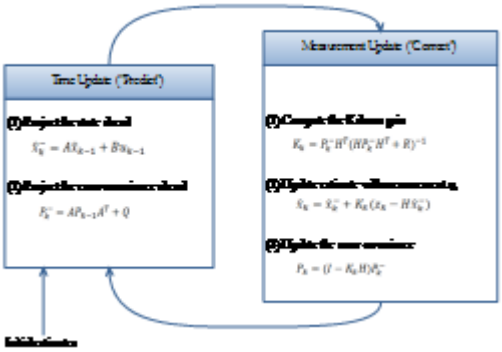

Fig. 5 Kalman filter flowchart

Fig. 5 shows the Kalman filter's flowchart. Figure has the actual steps and equations involved in the prediction and correction process of the Kalman Filter.

3.9 Measurement Validation

Measurement validations can be done using gating itself. The measurements which fall inside the gates are valid measurements and the ones which fall outside are invalid. Invalid measurements or False Alarms (FA) are not considered for data association and hence, make the system more efficient. The FAs are assumed to occur with a probability of P_{FA}. P_{FA} is lesser for higher SNR since there is less noise in the system.

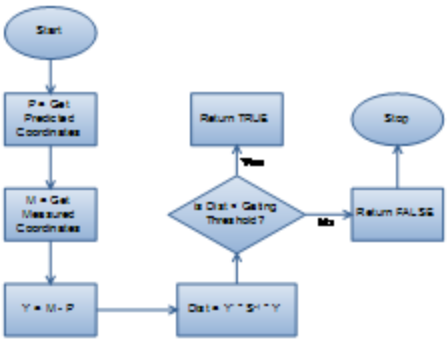

Fig. 6 Flowchart for measurement validation

Fig. 6 shows the flowchart necessary for measurement validation. As it can be seen the distance between the measured value and the predicted value is considered and if the difference falls outside a threshold (Gating Threshold) it is assumed to be invalid. The Gating Threshold has been found to be a constant value of 11.34 for 3D (Table 4.2, Blackman: 1986) in most cases and hence, it is the assumed value for development (i.e., $P_G = 11.34$).

3.10 Data Association

After estimation using the Kalman filter, the data has to be associated to the corresponding track. The literature review done indicates that for this particular purpose the PDA or the JPDA is used as the data association algorithm. PDA is normally used for single target tracking case but can be extended to multiple targets tracking case by having multiple PDA filters running in the same application. JPDA on the other hand is used for multiple targets tracking case. In this section the design details of the two algorithms has been discussed. The problem formulation for PDA has been given by Kirubarajan and Bar-Shalom (2004) and JPDA algorithm has been given by Fortmann, Bar-Shalom and Scheffe (1983).

3.11 PDA

The PDA algorithm is based on the Kalman filter as the core for state estimation. Hence, in this design the Kalman filter is considered as the state estimation filter. The PDA algorithm needs 50% more computational resources than the Kalman filter algorithm (Bar-Shalom, Daum, Huang 2009). The PDA algorithm performs data association and also corrects the Kalman filter with better feedback due to which the Kalman filter predicts correctly.

For every scan measurements are validated and then the Kalman filter error covariance matrix i.e., 'P' and the state i.e., 'x' are updated based on the equations from Kirubarajan and Bar-Shalom (2004). The data association probability is used to modify the Kalman filter parameters. The measured data is not actually associated, only the Kalman filter state update matrix is modified, so as to converge to the actual target coordinates which happens over a period of few scans.

3.12 JPDA

JPDA is used for tracking either single or multiple targets. JPDA is able to handle clutter better than PDA and it also addresses the problem of multiple interfering targets. The basic JPDA algorithm cannot handle target initiation and target deletion.

JPDA can handle multiple interfering targets and hence, every track needs to be initiated. For every scan measurements are validated and then the corresponding Kalman filter error covariance matrix i.e., 'P' and the state i.e., 'x' are updated based on the equations discussed in Fortmann, Bar-Shalom and Scheffe (1983). The data association probability is used to modify the Kalman filter parameters. Like in the PDA algorithm the measured data is not actually associated, only the Kalman filter state update matrix is modified, so as to converge to the actual

target coordinates which happens over a period of few scans

3.13 Data Structures

Various data structures are needed for storing and processing the information. In this section the data structures needed for implementing the MTT System is considered and is discussed about. From the requirements and from the design of the blocks of the MTT System the data structures have been identified:

3.13.1 Coordinate is a data structure which consists of a three floating point numbers which represent the position of anything that has been detected by a sensor.

3.13.2 Measurement is a data structure which consists of a set of coordinates. A measurement consists of the coordinates detected along with the total number of coordinates that were detected.

3.13.3 Target is a data structure which consists of a coordinate to which the target corresponds. It can also consist of the Kalman filter corresponding to the predicted coordinates which makes it easier to correlate the target to a track.

3.13.4 Track is a data structure which consists of a set of Target measurements. Hence, it can be maintained as a vector of the Target data structure. The corresponding Kalman filter can also be maintained as a part of the Track data structure along with the current scan number to maintain the state of the system.

3.13.5 False Alarm is a data structure which holds all the invalid coordinates from the measurement data. This can be used for further processing.

Figure 9 shows the data structures of the MTT System. As it can be seen Track is a set of Target data structures and Measurement is a set of Coordinate data structures

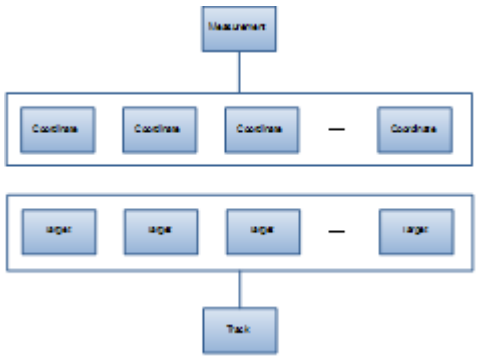

Fig. 7 Data Structures of MTT system

Also data structures specific to the target tracking filters have also been identified:

3.13.6 Association Matrix is a data structure which shows the data to track association.

The matrix shown in Fig. 10 shows only one association per track but a track can have multiple associations until the actual data association decision is taken.

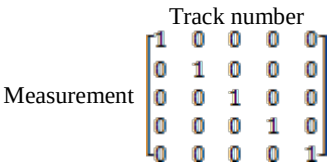

Fig. 8 JPDA association matrix

3.13.7 Permutation Matrix is a vector of association matrices which maintains all the possible association events.

4. IMPLEMENTATION

The implementation is based on the design that has been discussed. JAVA was used to implement the whole system and Eclipse IDE was used for development. The design was realised as a part of implementation using the JAVA platform. All assumptions made for designing and implementation of the system has been discussed in the appropriate section.

5. RESULTS AND DISCUSSION

The implemented system was tested for validity and functionality. The observed results are then used to draw inferences about the behaviour of the system. For instance the performance of the system is studied based on the load applied on the system. All the tests were done on a personal computer having 2 Giga Bytes of RAM, with an Intel Atom dual core processor running at 1.8 Giga Hertz clock speed, running Ubuntu 10.10 – 32Bit Operating System. The secondary memory is not a cause of concern for this implementation.

5.1 Simulation Assumptions and Parameters

The following assumptions have been made for the simulations:

- Process and measurement noise variance are assumed to be 1
- Gate G value is 11.34 as M value is 3 because sensor is designed for three dimensional tracking referring to table 4-2 of (Blackman 1986)
- Probability of observation falling within gate – P_G is 0.99
- Probability of detection P_D is 0.99
- Probability of false alarms P_FA is 0.1
- Measurement volume element V_c is 1
- The RADAR radius or sensor's sensing radius is assumed to be 2000 units

5.2 Tracking Target in Straight Line Motion

Targets moving in straight line are those that do not change their direction of motion while moving, though they can have changing velocities. In this case, a constant velocity model is chosen for conducting the test. The tracking has been done for a three dimensional space. The

test has been carried out for 50 scans of the sensor simulator. The test was performed for both the PDAF and JPDAF and the results were found to be similar. This is because the PDAF and JPDAF have the same state estimation core i.e., the Kalman Filter for prediction of the future path.

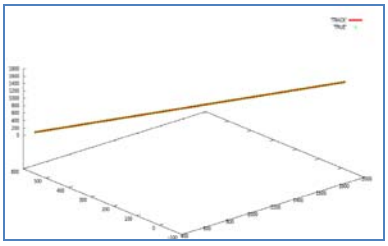

Fig. 9 Tracking single target in straight line motion

From Fig. 11 it can be seen that the MTT System is able to keep track of a target in straight line motion. The true coordinates of the target at each time interval are shown with "green hollow dots" and the track that is tracked by the tracking filter is shown using the "red line". This is would be followed in all cases unless otherwise specified. As it can be seen the tracking filter tracks the target motion correctly when it is moving in a straight line. It can be seen that this test is successful.

5.3 Tracking Manoeuvring Target

Manoeuvring targets are targets whose angular acceleration is constant. Due to this the target traces a curved path. Targets with such manoeuvring capability are simulated using the sensor simulator and the system is tested to see if it can track the target. The tracking has been done for a three dimensional state space.

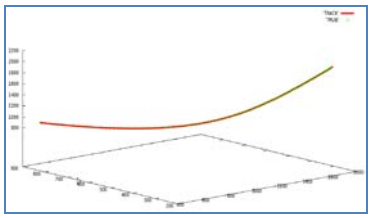

Fig. 10 Tracking single manoeuvring target

Fig. 12 shows that the tracker can track manoeuvring targets.

5.4 Tracking Target Having Non-Linear Motion

Targets having non-linear motion are those that change their direction of motion randomly. This causes the system model itself to have non-linearity. The sensor simulator is made to simulate such targets having non-linear motion and the results are observed.

As it can be seen from Fig. 13, the tracking filter is able to track targets having a mild amount of non-linearity. When the level of non-linearity increases though the tracking filter is unable to keep up and hence, as seen in Fig. 14 the tracking fails. This is because the state estimation filter i.e., Kalman Filter makes instantaneous predictions with feedbacks from the Kalman gain block, when the data correlation is incorrect the state estimation fails.

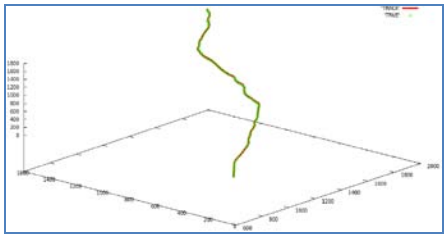

Fig. 11 Tracking single target having non-linear motion

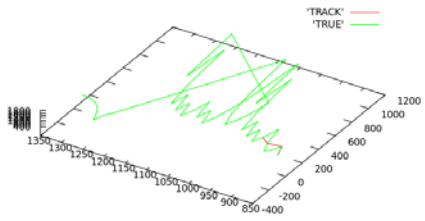

Fig. 12 Failure in tracking single target having highly non-linear motion

5.5 Tracking Multiple Targets

The ability of the tracking filter to track multiple targets of interest is tested. For tracking multiple targets using PDA, a PDA filter is instantiated and used for each target of interest. For JPDA though only one instance of the JPDA filter is used.

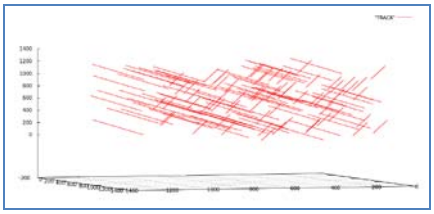

Fig. 13 PDA tracking 150 targets

Fig. 15 shows that the multiple PDA filters track 150 targets of interest (each moving in straight line motion) without a hitch. Later tests show that the amount of memory needed for PDA is very less i.e., PDA is computationally very less intensive and hence, such operations of tracking many targets is performed even by small desktop computers. This is one of the positive aspects of the PDA approach. A non-linear target dynamics model is not chosen in the PDA case as it causes some interference between the targets which may cause the PDA tracking to fail.

The main reason for evolution from PDA to JPDA is this particular problem i.e., multiple interfering targets in dense clutter. JPDA gives weightage to the other targets instead of treating them as false alarms. Due to this a more correct association probability can be calculated which in turn makes predication better.

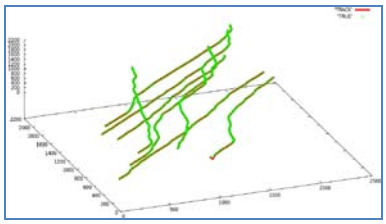

Fig. 14 JPDA tracking 8 targets

From Fig. 16, it can be seen that JPDA is able to keep track of 8 targets of interest. A non-linear model is chosen here to show that the filter can track multiple targets having non-linear motion.

When the number of targets crosses about 10 though, JPDA fails due to lack of memory, going on to show that 'optimal JPDA' is computationally very expensive. This is because the calculation of association probabilities of the JPDA algorithm is a NP-Hard problem as described by Collins and Uhlmann (1992). Many sub-optimal methods have been described to deal with this problem; they have been described in Projection based JPDA (Van Wyk et al. 2004), Fast JPDA (Fisher and Casasent 1989), DFS based JPDA (Zhou & Bose 1993), Goosen, B.J. van Wyk and M.A. van Wyk (n.d.), Pao and O'Neil (n.d.). Optimal JPDA has been considered for implementation in this work. Computational load can be further addressed by 'Clustering' as described by Konstantinova, Nikolov, Semerdjiev (2004).

5.6 Tracking Closely Spaced Interfering Targets

PDA fails in tracking closely spaced interfering targets since, it treats other targets as a part of the clutter. On the other hand JPDA succeeds as it gives a joint association probability i.e., a weighted association probability which in turn helps in correct state estimation. Tests are performed on PDA and JPDA for verifying this particular scenario.

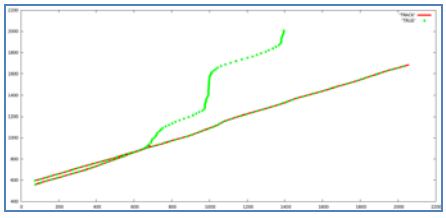

Fig. 15 PDA failure in tracking 2 interfering targets

From Fig. 17 and Fig. 18 it can be concluded that JPDA is successful in tracking interfering targets. PDA does not fail in all scenarios where targets crisscross but only when the targets are very closely placed as in Fig. 17.

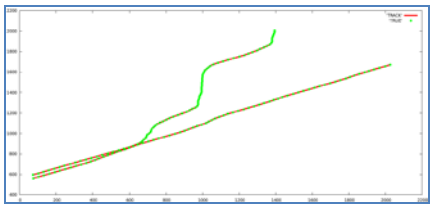

Fig. 16 JPDA success in tracking 2 interfering targets

5.6.1 Stress and Performance Tests

In this section the performance of the MTT System has been discussed. Comparison of the PDA and JPDA algorithms are shown based on time taken for each scan, increasing number of targets and so on.

5.6.2 Tests Based on Increasing Number of FAs

MTT System is tested with PDA to find out the performance of the MTT System when the number of false alarms is increased keeping the number of scans constant. The performance metric considered here is 'time'. The tabulation was made and when it was converted to a graph a linear curve was traced going to show that the system is a predictable system. The tests were carried out a set of 8 targets for 50 scans each.

5.6.3 Tests Based on Increasing Number of Targets

In this section the MTT System is tested with PDA and JPDA to find out the performance of the MTT System when the number of targets is increased keeping the number of scans constant. The performance metrics considered here are 'time' and 'memory'. Table 1 shows the details of the tests performed. The tests were carried out for 50 scans. It is to be noted that the time is measured in terms of 'milliseconds' and memory is measured in terms of 'megabytes'. The observations are tabulated for PDAF and JPDAF for both time taken and the memory used.

When more than eight to ten targets are to be tracked JPDA fails in terms of memory efficiency. This is because the space complexity for calculating the association probabilities is in the order of 2^{m+n} where, m and n are the number of 1s in the rows and columns in the association matrix respectively. The main reason for JPDA to take a lot of time is also because the JVM takes time to allocate memory for the association matrices and maintain them.

Test Scenario	PDAF Mem.	PDAF Time (ms)	JPDAF Mem.	JPDAF Time (ms)
8 Targets	28.3	1011	250.4	88823
30 Targets	91	4456	> 520	-
60 Targets	118.4	10795	-	-
90 Targets	127.1	20656	-	-
120 Targets	180.6	33545	-	-
150 Targets	192.2	49762	-	-

Table 1. Number of targets vs. memory and time

5.7 Time Comparison of PDA and JPDA

PDA takes about 15ms at an average per scan whereas JPDA takes about 1500ms at an average per scan. This is because the number of computations done by JPDA is much more than the computations done by PDA.

6. CONCLUSIONS

From this study we conclude that both PDA and JPDA can track targets with mild manoeuvring capabilities. Targets with highly non-linear motion are tracked only for few scans after which both the Tracking Filter fails to estimate the target's motion- for such cases UKF needs to be used.

PDA suffers in the case of multiple interfering targets in closely spaced targets scenario i.e., in a highly cluttered and dense targets scenario as it considers other targets as false alarms. When there is only linear motion, PDA does not suffer from interfering targets. The main advantage of the JPDA is that it succeeds in tracking multiple interfering targets in a cluttered scenario since it gives a weighted measurement to track association probability instead of treating other targets as a part of the clutter.

PDA has very predictable performance in terms of memory and time. PDA uses the least memory resource in comparison to most other MTT algorithms. The time required for tracking is also less due to fewer computations. JPDA has a lot more computational load in comparison to PDA. The time required for tracking is more than that of PDA (this is mainly because of space allocation needed for association matrices used by JPDA).

When the clutter density is uniform both PDA and JPDA give very good tracking performance. When the clutter density is non-uniform or Gaussian distributed both PDA and JPDA have lesser success rates, but in comparison JPDA has higher success rates than PDA. In 3D, it was noticed that interference was very less even with many targets and PDA was able to track targets correctly as the interference was found to be less.

7. REFERENCES

[1] Bar-Shalom, Y., Daum, F., and Huang, J. (2009) 'The Probabilistic Data Association Filter - Estimation in the Presence of Measurement Origin Uncertainty'. *IEEE Control Systems Magazine*, 29(6), 82-100.

[2] Bazzani, L., Bloisi, D., and Murino, V. (2009) 'A Comparison of Multi Hypothesis Kalman Filter and Particle Filter for multi-target Tracking'. *Performance Evaluation of Tracking and Surveillance workshop at CVPR*, 47-54.

[3] Blackman, S. S. (1986) *MTT with Radar Applications.* Norwood: Artech House Inc.

[4] Blackman, S. S., and Popoli, R. (1999) *Design and Analysis of Modern Tracking Systems.* Norwood: Artech House Inc.

[5] Blackman, S. S. (2004) 'Multiple hypothesis Tracking For Multiple Target Tracking'. *IEEE Aerospace and Electronic Systems Magazine*, 19(1), 5-18.

[6] Bojilov, V. L., Alexiev, M. K., and Konstantinova, D. P. (2003) 'An Algorithm Unifying IMM and JPDA Approaches'. *Comptes Rendus-Academie Bulgare Des Sciences* 56(12), 53-60.

[7] Chen, Z. (2003) *Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond.* Unpublished manuscript. Hamilton, Ontario, Canada: Communications Research Laboratory, McMaster University.

[8] Collins, J. B., and Uhlmann, J. K. (1992) 'Efficient gating in data association with multivariate Gaussian distributed states'. *IEEE Transactions on Aerospace and Electronic Systems*, 28(3), 909-916.

[9] Fisher, J.L., and Casasent, D. P. (1989) 'Fast JPDA multitarget tracking algorithm'. *Applied Optics*, 28(2), 371-376.

[10] Fortmann, T. E., Senior Member, IEEE, Bar-Shalom, Y., Senior Member, IEEE and Scheffe, M., Member, IEEE. (1983) 'Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association'. *IEEE Journal of Oceanic Engineering*, 8(3), 173-184.

[11] Goosen, M.J., van Wyk, B.J., and van Wyk, M.A. (n.d.) *Survey of JPDA algorithms for possible Real-Time implementation.* Unpublished booklet, Rand Afrikaans University, Johannesburg, South Africa.

[12] Hall, D. L., Senior Member, IEEE and Llinas J. (1997) 'An Introduction to Multi Sensor Data Fusion'. *Proceedings of the IEEE*, 85(1), 6-23.

[13] Kalman, R. E. (1960) 'A New Approach to Linear Filtering and Prediction Problems'. *Transactions of the ASME–Journal of Basic Engineering*, 82 (Series D), 35-45.

[14] Khaleghi, B., Saiedeh, N. R., Khamis, A., Fakhreddine, O. K., and Kamel, M., University of Waterloo. (2009) *Multisensor Data Fusion: Antecedents and Directions.* '2009 International Conference on Signals, Circuits and Systems', 1-6.

[15] Kirubarajan, T., Senior Member, IEEE, and Bar-Shalom, Y., Fellow, IEEE. (2004) 'Probabilistic Data Association Techniques for Target Tracking in Clutter'. *Proceedings of the IEEE*, 92(3), 536-557.

[16] Onstantinova, P., Nikolov, M., and Semerdjiev, T. (2004) *A Study of Clustering Applied to Multiple Target Tracking Algorithm.* International Conference on Computer Systems and Technologies - CompSysTech'04.

[17] Lucy, Y. P., and Sean, D. O'Neil. (1993) 'Multisensor Fusion Algorithms for Tracking'. *American Control Conference*, 859-863.

[18] Roecker, J.A., (1994) 'A class of near optimal JPDA algorithms'. IEEE Transactions on Aerospace and Electronic Systems, 30(2), 504 –510.

[19] Van Wyk, B. J., van Wyk, M. A., Noel, G. (2004) 'A Projection-Based Joint Probabilistic Data Association Algorithm'. *AFRICON, 2004. 7th AFRICON Conference in Africa*, 1, 313-317.

[20] Welch, G., and Bishop, G. (1995) *An Introduction to the Kalman Filter*. Technical Report, Chapel Hill: Department of Computer Science, University of North Carolina.

[21] Zhou, B., and Bose, N. K. (1993) 'Multitarget tracking in clutter: fast algorithms for data association'. *IEEE Transactions on Aerospace and Electronic Systems*, 29(2), 352-363.