Target Track Initiation Comparison and Optimization*

Darko Mušicki
Dept. of Electrical and Electronic Engeneering
University of Melbourne
VIC
Australia
d.musicki@ee.mu.oz.au

Sofia Suvorova
Electronic Warfare and Radar Division
DSTO
SA
Australia
sofia.suvorova@dsto.defence.gov.au

Abstract – *This paper proposes a procedure for comparison of target tracking filters in cluttered environment, using simulated experiments. Important simulation issues, including track generation and retention across simulation runs and true track definition are discussed. False track discrimination is identified as an important target tracking filter performance measure. This measure has two components; confirmed true track statistics and confirmed false track statistics. In order to meaningfully compare false track discrimination capabilities of target tracking filters, we suggest optimizing true track statistics, while using false track statistics as the optimization constrains. The optimization procedure is described, and the importance of correct parameter selection is illustrated.*

Keywords: target tracking, false track discrimination, optimization, steepest descent, quasi-Newton, Nelder-Mead simplex

1 Introduction

Data association algorithms deal with situations where there are measurements of uncertain origin. In many radar and sonar applications, measurements (detections) originate from both targets and non-targets, i.e from various objects such as terrain, clouds etc and from thermal noise. Unwanted measurements are usually referred to as clutter. In addition target measurements are present at each measurement scan with only a certain probability of detection. In a multi-target situation, the measurements may have originated from one of several targets and the number of such measurements is generally unknown. The number of targets and their trajectories are generally a priori unknown.

Under such conditions automatic track initiation creates true tracks (from target measurements) and false tracks (from clutter). In the course of track evolution and maintenance, true tracks may become false due to lack of detections, clutter measurements, target maneuvers which the track filter could not follow, etc. Equivalently, false tracks may become true tracks if target measurements are used to update. For practical purposes, target tracking requires a mechanism to distinguish between true and false tracks, without a-priori knowledge of target existence and trajectory. When a track is considered to be a true track, it is confirmed and presented to the operator or next processing level. When a track is considered to be a false track, it is terminated. We call this process "False Track Discrimination".

Target tracking algorithms are usually evaluated and compared using a simulation study. There are a number of issues to consider when designing a target tracking algorithm, however the more usual ones, listed in the usual order of importance are:

- False track discrimination,
- Track maintenance (long term track retention),
- Reliability/Robustness of tracking algorithm,
- Track estimate/prediction errors, etc.

Of course, these priorities may differ for different tracking application. When presenting new algorithms for target tracking in clutter, authors usually concentrate on the false track discrimination and track estimate errors. This paper presents issues raised when several algorithms have to be compared with each other, specifically comparison of their performance regarding the false track discrimination. It also presents an optimization procedure to improve false track discrimination.

There is a variety of algorithms for target tracking in clutter. Algorithms can address single or multiple target tracking. They can be single scan algorithms, if they compress measurement history into one track state estimate, or multi scan algorithms, if they keep measurement history separate over a number of scans. The issues and procedure presented below are valid for PDA based single-scan target tracking algorithms, however, we believe, they are applicable for above classes of target tracking algorithms. For reasons of simplicity and clarity, in this paper we will use single-target single-scan target tracking algorithms, and illustrate it using Integrated Probabilistic Data Association (IPDA) algorithm [1].

The rest of the paper is organized as follows. False track discrimination is defined in Section 2. Simulation and issues with false track discrimination comparison of target tracking algorithms are presented in Section 3, followed by the optimization of target tracking algorithms in Section 4.

[1]This research has been supported by the Centre of Expertise in Networked Decision and Sensor Systems and funded by the Defence Science and Technology Organisation Australia

2 False Track Discrimination

Target tracking filtering is, in a large number of cases, a Bayesian recursion. Bayesian recursions assume existence of initial probabilities / probability density functions (pdfs). Therefore, the details of the initialization process are usually not part of tracking filter presentations, with some exceptions as in [2, 3]. The usual procedure is to initiate tracks using two "near enough" measurements in two consecutive scans, using a "two-point differencing" procedure [2, 3].

To distinguish between true and false tracks, practical target tracking in clutter requires a track quality measure to be updated using available measurements. Different tracking filters use different names for the track quality measures. MHT [4] uses a track score function, IPDA [1] uses the probability of target existence, IMM-PDA [5] uses the probability of target detectability, etc. The usual procedure (with some variations) is to declare a track to be "true" when the track quality measure rises above a certain predefined value, which we call in this paper the confirmation threshold, and to declare a track to be "false" when the track quality measure falls below another predefined value, which we call in this paper the termination threshold. In the simplest case these thresholds are constant, although a better result [6] can be obtained when the thresholds vary with track age. When tracking filters are compared, for reasons of simplicity usually constant thresholds are used. The tacit (and non-rigorous) assumption is that the relation "better tracking filter" under constant thresholds regime would also hold under variable thresholds regime. In this paper we consider only the constant thresholds case.

An initial track state estimate pdf is calculated using the measurement coordinates. The initial track quality depends on a single parameter. This parameter can be used directly as the initial value of track quality, or it may be used together with an a priori value of the clutter measurement density to calculate the initial value of track quality [3]. In the text below we call this parameter the initial track quality parameter.

Thus, in the simple case of constant thresholds, there are three parameters which control false track discrimination process: the confirmation threshold, the termination threshold and the initial track quality parameter.

3 Target Track Filters Simulation and Comparison

A simulation study is usually used to (initially) evaluate and compare target tracking filters. A simulation is an imitation of reality, there is a tradeoff between computational resources required, complexity of the model of reality and the degree of confidence in statistical simulation results. The number of parameters that the tracking process depends on is large, making it virtually impossible to exhaustively test and compare target tracking filters. This can sometimes lead to erroneous conclusions. As shown in Fig. 1.a of [11], different algorithms can be declared better in different combinations of environments/target trajectories. Such exhaustive simulations are often not practical in research institutions and the reported/published work is usually based on single or a small number of environment/target trajectories. Therefore, these results should be taken only as indicative and further comprehensive evaluation is advised before implementing the tracking filters in practice.

Some of the parameters and the usual simplifications are briefly outlined below.

Sensors provide measurements to the tracking filters. Real sensors are non-linear, have finite resolution (will merge measurements within resolution "cells") which also depends on the environment, the probability of target detection P_D and measurement noise R are non-linear functions of target and environment properties etc. The usual simplification is to assume infinite sensor resolution and known and constant P_D and R. The non-linearity of the sensor is also sometimes ignored, specially when comparing target tracking filters.

Initial tracking filter comparison usually simulates a single target case with well defined trajectory, with constant and known P_D and either a straight line trajectory or a well defined maneuvering trajectory when the data association is combined with maneuvering target tracking.

The choice of clutter measurement distribution can influence the target tracking filter comparison results. Usually uniform clutter density is assumed and simulated. In the real world, however, clutter density is usually non-uniform. Target tracking filter equations use clutter density estimates in data association and track quality calculations. If the clutter is uniform, the target tracking filters with a large selection gate have better estimate of clutter density; the opposite is the case when the clutter is non-uniform. As non-uniform clutter happens more often in the real world, more realistic simulations would have non-uniform clutter, at least along the target trajectory.

Each algorithm is compared using simulations. The simulations usually consist of a number of simulation runs, where targets re-appear at the beginning of each simulation run, and the performance statistics are accumulated. The number of simulation runs should be large enough to obtain performance statistics with a high degree of confidence, usually 200 or more. One should be careful to ensure that each simulation experiment uses the same random data [6] including target trajectories and detection sequences, measurement noise and clutter measurements. All simulation experiments should use the same track initiation procedure unless, of course, it is the track initiation procedure that is being evaluated. Usually, each simulation run starts with a new track initiation, i.e all tracks in the previous run are discarded. This is correct for true tracks and undesirable for false tracks. In the real world, target tracking filters are used continuously and there is a steady-state distribution of the number of false tracks. To achieve this during simulation, false tracks should not be discarded at the end of a simulation run (only at the end of a simulation experiment), instead the false tracks should be terminated only during the normal false track discrimination process, or when process logistics dictates so, i.e when tracks are outside the surveillance area or merged with another track.

Target track filter performance statistics requires each track to be classified as either a true or a false track. The true track definition requires careful consideration. If a true track is declared false by mistake, it is likely to be confirmed and will substantially alter the false track discrimination statistics. If a false track is declared true by mistake, it is likely to alter substantially the state estimation error statistics. The definition of the true track being a track initialized by target measurements will fail to recognize the event of a false track becoming a true track. The definition of the true track being a track which selects target measurements will falsely declare as a true track a false track with a large selection gate crossing the target trajectory. Also, this definition does not recognize the event that a true track becomes false in an environment of moderate to low P_D. Instead, we use the following statistics:

$$\begin{aligned} \chi_f &= \left(\hat{x}_k - x_k\right)^T \hat{P}_k^{-1} \left(\hat{x}_k - x_k\right) \\ \chi_t &= \left(H\left(\hat{x}_k - x_k\right)\right)^T \left(HPH^T\right)^{-1} \left(H\left(\hat{x}_k - x_k\right)\right) \end{aligned} \tag{1}$$

where k denotes time instant, $\hat{x}_k$ and $\hat{P}_k$ are the track state estimate mean and covariance respectively, x_k is the target state and H is the measurement matrix. When $\chi_f < \chi_{ft}$ for a false track, the false track becomes a true track. When the statistics $\chi_t > \chi_{tf}$ for a true track, the true track becomes false. True-to-false threshold χ_{tf} should be at least as big as the gating threshold.

False track discrimination has two measures: true and false track statistics. The true track statistics may be the peak probability of true track confirmation; however for optimization purpose, we believe it is better to use

$$\tau = \sum_{r \in runs} \sum_{s \in scans} T(r, s)$$

where $T(r, s)$ equals one if there was a true track confirmed in simulation scan s of run r, and zero otherwise. In this paper this will be referred to as the true track statistics. In a similar style, a reasonable false track statistic is defined as

$$\sigma = \sum_{r \in runs} \sum_{s \in scans} F(r, s)$$

where $F(r, s)$ is the number of confirmed false tracks in simulation scan s of run r. In this paper this will be referred to as the false track statistics.

When evaluating and comparing target tracking algorithms, it is important to use both true and false track statistics. Unfortunately, authors often use fixed false track discrimination parameters for all target tracking filters being compared, ignore (or do not mention) the false track statistics and use only true track statistics. Also, false track discrimination results for a given target tracking algorithm will depend on the parameter choice. A couple of options are possible:

- Choose random values and manipulate the termination thresholds until all filters deliver the same false track statistics. As shown below it may lead to dubious conclusions.
- Evaluate algorithms for the whole range of parameters and then combine the results using some averaging procedure. This is clearly not practical, although it will let us reach conclusions regarding the robustness of the algorithms.
- For each algorithm find the optimum set of parameters for the environment. This will reduce the randomness of the first choice and also show the limits of each algorithm in the simulated environment. This is our preferred option.

We choose the following procedure:

- determine the acceptable value of the false track statistics.
- for each tracking filter chosen, use a separate optimization procedure to maximize true track statistics, while using the chosen value of false track statistics as an optimization constraint.

In this fashion all algorithms compared will have (almost) the same false track statistics and the true track statistics will be a meaningful tool to compare target tracking filters, at least as far as the simulation environment is considered. This procedure also removes subjectivity from the parameter choice and the possibility to randomly (or subconsciously) pick parameters which favor one filter over the other. While this still leaves open numerous questions, such as target tracking filter reliability and robustness, it is a systematic way to compare target tracking filters with limited computational resources.

A short note on the track estimate errors comparison: The track estimate errors statistics are most often performed on confirmed true tracks only. During simulation runs, due to different effectiveness of false track discrimination performance of target tracking filters, different target tracking algorithms will have confirmed true tracks in different runs/scans. For reasons of objectivity, track estimate error statistics should be gathered only in runs/scans in which all tracking algorithms have confirmed true tracks. Otherwise, the algorithms with more efficient false track discrimination properties will get penalized as they will confirm true tracks in more difficult situations with corresponding higher estimation errors.

As each simulation experiment takes considerable resources, often of the order of hour(s), it is important to carefully choose optimization algorithms to minimize the number of simulation experiments necessary for optimization.

4 Optimization Algorithms

Denoting true track statistics by τ, false track statistics by σ, the initial track quality parameter by ψ_0, track confirmation threshold by η, track termination threshold by ζ, we write the problem as follows

$$\{\psi_0, \eta, \zeta\}^\star = \arg\max_{\psi_0, \eta, \zeta} \tau \tag{2}$$

subject to

$$\sigma \leq \sigma_0, \tag{3}$$

where σ_0 is the chosen value of false track statistics. A non-linear programming algorithm, such as the method of penalizing functions [7] can be used to solve this problem. We reformulate the problem as follows:

$$\{\psi_0, \eta, \zeta\}^\star = \arg \max_{\psi_0,\eta,\zeta} \left(\tau - A_s \Phi^2\right), \tag{4}$$

where

$$A_s > 0, s = 1, 2, \cdots, \quad \lim_{s\to\infty} A_s = +\infty \tag{5}$$

$$\Phi = \begin{cases} 0, & \text{if } \sigma \leq \sigma_0, \\ \sigma - \sigma_0, & \text{if } \sigma > \sigma_0. \end{cases} \tag{6}$$

Having transformed the non-linear programming problem into an un-constrained optimization problem, we then use an unconstrained optimization method to tune the parameters of the target tracking filters. Since analytical forms for τ and σ are not easily deductible, a numerical optimization method is needed. Although a wide spectrum of methods exists [7, 8, 9, 10] for unconstrained optimization, they can be broadly categorized in terms of the derivative information that is, or is not, used. Search methods that use only function evaluations are most suitable for problems that are very nonlinear or have a number of discontinuities. Gradient methods are generally more efficient when the function to be minimized is continuous in its first derivative. Higher order methods, such as Newton's method, are only really suitable when the second order information is readily and easily calculated, because calculation of second order information, using numerical differentiation, is computationally expensive.

Gradient methods use information about the slope of the function to dictate a direction of search where the minimum is thought to lie. The simplest of these is the method of steepest descent in which a search is performed in a direction $-\nabla f(\mathbf{x})$ where $\nabla f(\mathbf{x})$ is the gradient of the objective function. Of the methods that use gradient information, the most favored are the quasi-Newton methods. These methods build up curvature information at each iteration to formulate a quadratic model problem of the form

$$f(\mathbf{x}) = \frac{\mathbf{1}}{\mathbf{2}}\mathbf{x}^\mathbf{T}\mathbf{H}\mathbf{x} + \mathbf{c}^\mathbf{T}\mathbf{x} + \mathbf{b} \tag{7}$$

where the Hessian matrix, H, is a positive definite symmetric matrix, c is a constant vector, and b is a constant. The optimal solution for this problem occurs when the partial derivatives of $\mathbf{x}$ go to zero. Newton-type methods (as opposed to quasi-Newton methods) calculate H directly and proceed in a direction of descent to locate the minimum after a number of iterations. Calculating H numerically involves a large amount of computation. Quasi-Newton methods avoid this by using the observed behavior of $f(\mathbf{x})$ to build up curvature information to make an approximation to H using an appropriate updating technique [7, 8, 9]. The gradient information is either supplied through analytically calculated gradients, or derived by partial derivatives using a numerical differentiation method via finite differences. At each major iteration, a line search is performed

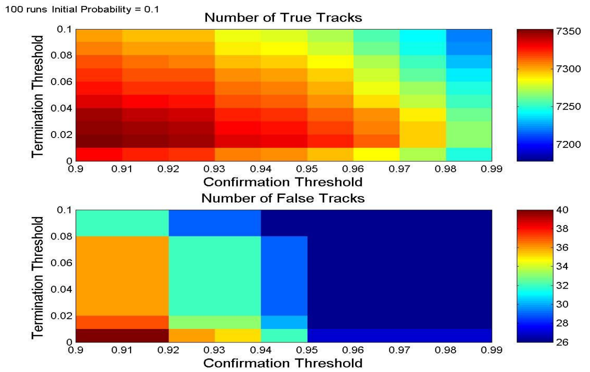

Fig. 1: True and False Track Statistics for $\psi_0 = 0.1$

in the direction $-H^{-1}\nabla f(x)$. When analytic derivatives are not available, evaluation of finite difference gradients is computationally expensive. Therefore, another interpolation/extrapolation method is needed so that gradients are not evaluated at every iteration. The approach in these circumstances, when gradients are not readily available, uses a quadratic or cubic interpolation method [7, 8, 9].

The Nelder-Mead simplex method [10] uses the simplex search method. This is a direct search method that does not use numerical or analytic gradients. If n is the length of $\mathbf{x}$, a simplex in n-dimensional space is characterized by the $n + 1$ distinct vectors that are its vertices. At each step of the search, a new point in or near the current simplex is generated. The function value at the new point is compared with the its values at the vertices of the simplex and one of the vertices is replaced by the new point, giving a new simplex. This step is repeated until the diameter of the simplex is less than the specified tolerance.

This method is generally less efficient than gradient methods. However, when the problem is highly discontinuous, the Nelder-Mead simplex method might be more robust.

Fig. 1 shows true and false track statistics τ and σ for fixed value of initial probability of track existence ψ_0 and varying values of termination ζ and confirmation η thresholds.

In our simulations we used all tree optimization algorithms: Steepest Descent, Quasi-Newton and Nelder-Mead Simplex Method. Latter with apparent superiority in speed of convergence to the gradient methods; in some cases by an order of magnitude improvement in the number of required simulation experiments needed to reach convergence.

5 Importance of Parameter Selection

In this section we give an example to illustrate how the choice of parameters can affect the overall performance of the target tracking filter. Two target tracking algorithms are compared; IPDA [1] and IMM-PDA [12], in an environment of non- homogenous clutter.

Each simulation experiment consists of 1000 runs, each run has 50 scans. A 2-dimensional surveillance situation was considered. The area under surveillance was $1000m$ long and $400m$ wide. The false measurements satisfied a

Poisson distribution with density $2.0 \cdot 10^{-5}/scan/m^2$ over the area except for two patches with five times this clutter density. The high clutter density patches are rectangular with corner coordinates $(x_{min}, x_{max}, y_{min}, y_{max})$ of $(330, 490, 203, 303)m$ and $(715, 840, 100, 200)m$. A single target follow a uniform motion trajectory past the edges of the heavy clutter patches.

A two dimensional radar system was modelled and measurements of track position were provided. The measurement noise was simulated with standard deviation of 5 m in range and 1 mrad in bearing at the range 5 km. The probability of target detection was chosen to be $P_D = 0.7$. The statistics of interest was the total number of confirmed true tracks over time of target appearance. Track initiation and the true and false track termination were as described in section 3. Confirmation thresholds for each algorithm were chosen to have the false track statistic equal to 50.

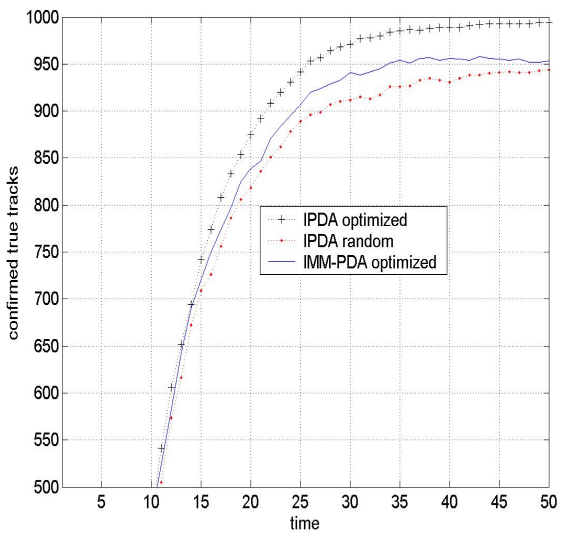

Fig. 2: Confirmed true tracks

The true target statistics are shown in Figure 2 for the following cases:

- IPDA with parameters optimized for best false track discrimination - “IPDA optimized”,
- IMM-PDA with parameters optimized for best false track discrimination - “IMM-PDA optimized”, and
- IPDA with random choice of initial probability of target existence and termination threshold - “IPDA random”.

Table 1: Target tracking parameters

	IPDA optimized	IMM optimized	IPDA random
Initial prob.	0.1347	0.3331	0.5
Termination Thr.	0.0066	0.1367	0.2
Confirmation Thr.	0.8654	0.9928	0.9865

The parameters for each case are shown in Table 1. As presented in Figure 2, using optimized parameters we reach the conclusion that in this environment IPDA may have advantage over IMM-PDA. Using the “IMM-PDA optimized” and “IPDA random” curves, we reach the opposite conclusion.

6 Conclusion

A correct comparison of target tracking filters is important both when a filter choice needs to be made for a particular application, and when presenting a new or improved target tracking filter in a publication. The usual procedure of ignoring false track statistics and using the same, somewhat randomly chosen, set of parameters for all filters compared is not the best approach. This paper proposes a rigorous and systematic procedure for comparison of target tracking filters in a cluttered environment. The importance of the ability of filter to discriminate false tracks is outlined. A basis for comparison is firmly established by optimizing true track statistics while maintaining similar false track statistics, for each tracking filter separately.

References

[1] D. Mušicki, R. J. Evans, and S. Stanković. Integrated probabilistic data association. *IEEE Transactions on Automatic Control*, 39(6):1237–1241, 1994.

[2] Y. Bar-Shalom. Tracking methods in a multitarget environment. *IEEE Trans. Automatic Control*, 23(4):618–626, Aug 1978.

[3] D. Mušicki and R. Evans. Clutter map information for data association and track initalization. *IEEE Trans. Aerospace Electronic Systems*, 40, Accepted for publication 2004.

[4] S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems*. Artech House, MA, 1999.

[5] Y. Bar-Shalom and K. C. Chang. Joint probabilistic data association for multitarget tracking with possibly unresolved measurements and maneuvers. *IEEE Trans. Automatic Control*, 29:585–594, Jul 1984.

[6] Y. Bar-Shalom and X. R. Li. *Estimation and Tracking: Principles, Techniques and Software*. Artech House, MA, 1993.

[7] W. Murray, P. E. Gill and M. H. Wright. *Practical Optimization*. London, Academic Press, 1981.

[8] S. P. Han. A globally convergent method for nonlinear programming. *Journal of Optimization Theory and Applications*, 22:297, 1977.

[9] M. J. D Powell. *Nonlinear Programming 3*. Academic Press, 1978.

[10] J. A. Reeds, M. H. Wright, J. C. Lagarias and P. E. Wright. Convergence properties of the nelder-mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9(1):112–147, 1998.

[11] H. A. P. Blom and E. A. Bloem. Probabilistic data association avoiding track coalescence. *IEEE Trans. Automatic Control*, 45:247–259, 2000.

[12] Y. Bar-Shalom, K. C. Chang, and H. A. P. Blom. *Multitarget Multisensor Tracking, vol.I*, chapter Automatic track formation in clutter with a recursive algorithm, pages 25–42. Artech House, Inc., Norwood MA, 1990.