18th International Conference on Information Fusion
Washington, DC - July 6-9, 2015

A Computationally Efficient Dynamic Programming based Track-Before-Detect

Jinghe Wang*, Wei Yi*, Mark R. Morelande† and Lingjiang Kong*
*School of Electronic Engineering, University of Electronic Science and Technology of China
Chengdu, P.R. China
E-mail: {kussoyi, lingjiang.kong}@gmail.com
†RMIT University, Australia
E-mail: m.morelande@gmail.com

***Abstract*—Dynamic programming based track before detect (DP-TBD) is a batch processing method. It exploits the space-time correlation among several consecutive frames of measurements and jointly processes all the measurements in these frames. Due to this batch processing manner, DP-TBD suffers heavy computational load since it involves processing a large volume of data. Moreover, in order to track long continuous time target trajectories, sliding window (SW) processing is required which further increases the computational burden. The SW moves forward by one step at each measurement time to include the latest N frames of measurements and then jointly processes these frames and updates target states. This means that N frames have to be processed at every measurement time, and each frame needs to be repeatedly processed N times by N consecutive SWs, which is highly inefficient. To overcome this limitation, we present a computationally efficient DP-TBD algorithm in this paper. Similar to the recursive filter, the proposed method only processes the latest frame at each measurement time, and each frame only needs to be processed once. Our analysis suggests that the proposed method achieves significant reduction of computational cost, especially when N is large. Finally various numerical examples are used to demonstrate the robustness and efficiency of the proposed method.**

I. Introduction

Conventional tracking algorithms [1] declare detections at each measurement time, and use the thresholded measurements to estimate target states. Since the thresholding process only retains the binary results and discards all the rest of information, the tracking performance of conventional algorithms deteriorate inevitably when signal-to-noise ratio (SNR) is low. In contrast, track-before-detect (TBD) [2] is an effective way to detect and track low SNR targets embedded in high noise by directly processing unthresholded measurements.

Various methods have been used to implement TBD, e.g., particle filter [3], [4], Hough transform [5], [6], random finite set [7] and dynamic programming (DP) algorithm [8]. Among these, DP-TBD is a strong competitor. It exploits the space-time correlation among several consecutive frames of measurements and jointly processes all the measurements in these frames. Compared with other TBD procedures, such as particle filter which requires the prior information of targets to initiate tracks, DP-TBD has the natural ability to detect weak targets and initiate tracks [9], meanwhile it can handle the targets with slow maneuvering, and avoid the speed mismatch problem. Early works on DP-TBD [10]–[15] mainly aimed at the optical systems, i.e., detecting dim targets in infrared images. In recent years, its applications have been extended to other areas, such as radar detection [16]–[23], sonar system [24] and cell tracking in biology [25].

The main drawback of DP-TBD is its heavy computational load. This hardly leads to realtime implementable schemes and restrains its further development in practical applications. To overcome this limitation, much efforts have been devoted to improve the computational efficiency of DP-TBD. In [19], [26], to reduce the volume of the processed data, DP-TBD is modified to process thresholded data with significantly lower thresholds than what are used by conventional trackers. In [27], [31], the complexity is reduced by adaptively changing the resolution of the image. These strategies all aim at reducing the computational complexity during the batch processing.

However, due to the batch processing manner of DP-TBD, to track long continuous time target trajectories, sliding window (SW) processing is required which brings extra computational burden and further increases the computational complexity. This can be seen clearly in Fig.1. Every time when a new measurement frame is received, SW moves forward by one step to include the latest N frames of measurements and then completes the batch processing. Therefore N frames have to be processed at every measurement time, and each frame needs to be repeatedly processed N times in N consecutive SWs, which is highly inefficient. The modified strategies referred above can not avoid this repetitive processing in SW processing. In [28] a heuristic approach is proposed to consider the searching efficiency of SW processing based on a plot-lists measurement model. Although it does not provide theoretical analysis, the authors demonstrated by simulation results that huge computation cost may be saved.

This study also focuses on the computational efficiency of DP-TBD, particularly the efficiency of SW processing in DP-TBD, based on the classical trellis diagram model [10]–[14]. It is an extension of our preliminary work [30]. The contribution of this paper is manifold and can be summarized as follows.

1) First, we propose a multidimensional joint (MD-J) filter. The MD-J filter updates the target states upon the information contained by the latest measurement frame.

But different from the classical recursive filter which is implemented by the Chapman-Kolmogorov (C-K) equation and Bayes' rule, and only estimates one state at each recursion [4], the proposed one predicts and updates the target states depending on the multidimensional joint probability distribution function (pdf), and forms the multidimensional joint posterior pdf of several consecutive states. So that this filter is available to the batch processing algorithms, especially, it can avoid the SW processing, since that it only needs to process the latest one frame. The detailed derivation of the MD-J filter is given in the following pages.

2) Next, upon the MD-J filter, we present a computationally efficient TBD method based on DP. The presented method avoids the repetitive processing in SW processing. It only uses the latest frame to update the target state, and each frame only needs to be processed once, which highly improves the computational efficiency of the algorithm, compared with the traditional SW processing DP-TBD (SW-DP-TBD).
3) Finally, various numerical results are provided to compare the proposed algorithm with the traditional SW-DP-TBD and verify the robustness and efficiency of the proposed algorithm.

The rest of this paper is organized in the following manner. In section II, models and notations are introduced; section III describes the traditional SW-DP-TBD and its problems; in section IV, we deduce the proposed algorithm; some numerical results are provided in section V; finally the conclusions are drawn in section VI.

II. Models and Notations

A. Target model

Assume that there is a single point target moving in the two-dimensional (2-D) plane. The state at stage k is described as $\tilde{\mathbf{x}}_k = (p_x, v_x, p_y, v_y)^{\mathrm{T}} \in \mathbb{R}^s$, where $^{\mathrm{T}}$ denotes the matrix transpose, p_x, p_y denote the positions and v_x, v_y denote the velocities in x and y dimension respectively. $\mathbb{R}^s$, $s = 4$ denotes the state space. The the motion model of the target is modeled as a Markov process that

$$\tilde{\mathbf{x}}_k = \mathbf{F}\tilde{\mathbf{x}}_{k-1} + \mathbf{v}_k \tag{1}$$

where $\mathbf{v}_k$ is the process noise which is Gaussian distributed and $\mathbf{F}$ is the state transition matrix given by

$$\mathbf{F} = \mathbf{I}_2 \otimes \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \tag{2}$$

wherein, $\otimes$ is the Kronecker product and $\mathbf{I}_2$ denotes the two dimensional identity matrix. T is the interval of two consecutive frames.

It is worth pointing out that although a single target scenario is assumed here, the scenario with multi-target can also be taken into account by directly increasing the dimension of the state space [17], or using the method introduced in [21].

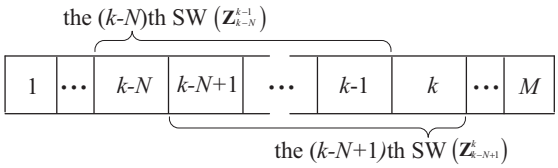


Fig. 1. In traditional SW-DP-TBD, when a new frame of data is received, SW moves forward by one step. N is the length of the SW and $\mathbf{Z}_l^k = (\mathbf{Z}_l, \mathbf{Z}_{l+1}, \cdots, \mathbf{Z}_k)$ denotes the sequence of several consecutive frames.

B. Measurement model

Assume that the size of observation area is $N_x \times N_y \times \Delta_x \times \Delta_y$, where $\Delta_x \times \Delta_y$ is the size of resolution cell and N_x and N_y are the number of resolution cells in x and y dimension respectively. The measurement recorded in the (i,j)th resolution cell at stage k is denoted as $z_k(i,j)$.The measurement model of the system is described as,

$$z_k(i,j) = \begin{cases} w_k(i,j) & \text{there is no target in } \quad (i,j) \\ A(\mathbf{x}_k) + w_k(i,j) & \text{there is a target in } \quad (i,j) \end{cases}$$

where w_k is the measurement noise and $A(\bullet)$ denotes the target amplitude. $\mathbf{Z}_k = \{z_k(i,j), 1 \le i \le N_x, 1 \le j \le N_y\}$ denotes the kth frame of data, e.g., a frame of measurements in radar detection or a frame of image in optical system.

C. Discrete state definition

The DP algorithm considered in this paper is based on the trellis diagram. So that to implement DP-TBD, the continuous state space $\mathbb{R}^s$ must be quantized [10], [13]. Let the position space be divided into cells of size $\Delta_x \times \Delta_y$ and the velocity space be divided into cells of size $\Delta_{vx} \times \Delta_{vy}$. The discrete state at the kth time is denoted as $\mathbf{x}_k = (i, r, j, s)^{\mathrm{T}} \in \bar{\mathbb{R}}^s$, corresponding to the continuous states that

$$\begin{aligned} p_x &\in [i\Delta_x,\ (i+1)\Delta_x) \\ v_x &\in [r\Delta_{vx},\ (r+1)\Delta_{vx}) \\ p_y &\in [j\Delta_y,\ (j+1)\Delta_y) \\ v_y &\in [s\Delta_{vy},\ (s+1)\Delta_{vy}) \end{aligned}$$

where $i \in [1, N_x]$, $r \in [-N_{vx}, N_{vx}]$, $j \in [1, N_y]$, $s \in [-N_{vy}, N_{vy}]$, with N_{vx}, N_{vy} the number of velocity cells.

III. The Traditional DP-TBD

A. DP-TBD

As shown in Fig.1, at each measurement time, SW-DP-TBD jointly processes the latest N frames of data. The algorithm determines the prospective target trajectory based on the maximum a posterior (MAP) estimation. Assume that the current time is k, then the decision can be expressed as,

$$\hat{\mathbf{X}}_{k-N+1}^k = \arg \max_{\mathbf{X}_{k-N+1}^k \in \mathbf{S}_N(k)} \{p(\mathbf{X}_{k-N+1}^k | \mathbf{Z}_{k-N+1}^k)\} \tag{3}$$

where $\mathbf{X}_l^k = (\mathbf{x}_l, \mathbf{x}_{l+1}, \ldots, \mathbf{x}_k)$ denotes the state sequence along the lth time to the kth time and $\mathbf{S}_N(k) \in \bar{\mathbb{R}}^{sN}$ is the set of short trajectories with the length N and terminating at the

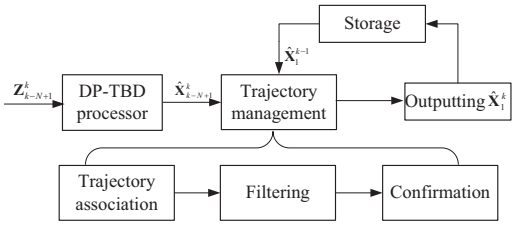


Fig. 2. Processing procedure of SW-DP-TBD at time k. $\hat{\mathbf{X}}_1^k$ denotes the continuous trajectory estimation obtained at time k.

kth time, which is determined by the motion model in section II-A.

Making use of the Bayes' rule, the posterior pdf in (3) can be rewritten as,

$$\begin{aligned} &p(\mathbf{X}_{k-N+1}^k|\mathbf{Z}_{k-N+1}^k) \\ &= \frac{p(\mathbf{Z}_k|\mathbf{x}_k)p(\mathbf{X}_{k-N+1}^k|\mathbf{Z}_{k-N+1}^{k-1})}{p(\mathbf{Z}_k|\mathbf{Z}_{k-N+1}^{k-1})} \\ &\propto p(\mathbf{Z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-N+1}^{k-1}|\mathbf{Z}_{k-N+1}^{k-1}) \end{aligned} \quad (4)$$

Using equation (4) and based on the idea of DP algorithm [8], the jointly N-D optimization (3) can be broken down and solved by a iterative approach

$$I(\mathbf{x}_i) = p(\mathbf{Z}_i|\mathbf{x}_i) \max_{\mathbf{x}_{i-1}\in\tau(\mathbf{x}_i)} \{p(\mathbf{x}_i|\mathbf{x}_{i-1})I(\mathbf{x}_{i-1})\} \quad (5)$$

$$\mathbf{\Psi}(\mathbf{x}_i) = \arg \max_{\mathbf{x}_{i-1}\in\tau(\mathbf{x}_i)} \{p(\mathbf{x}_i|\mathbf{x}_{i-1})I(\mathbf{x}_{i-1})\} \quad (6)$$

for $i = k-N+2, k-N+3, \ldots, k$, where $I(\mathbf{x}_i)$ is the merit function and $\tau(\mathbf{x}_i)$ is the set of $\mathbf{x}_{i-1}$ which can transfer to the given $\mathbf{x}_i$, and $\mathbf{\Psi}(\bullet)$ is used to record the transition relationship between two consecutive stages. Then after the threshold detection, the short trajectory estimation $\hat{\mathbf{X}}_{k-N+1}^k$ can be obtained, as shown in Fig.2.

B. Trajectory management

Also shown in Fig.2, after the DP processing, the output short trajectories $\hat{\mathbf{X}}_{k-N+1}^k$ are further handled in the trajectory management. It consists of several steps, trajectory association, filtering and finally confirming target trajectories [20], [29]. The task of trajectory association is to choose the short trajectories which can be combined with the given continuous trajectory. The task of filtering is to update the continuous target trajectory, in the light of the information provided by the short trajectories, and the task of confirmation is to choose the most possible target trajectory from all the continuous trajectories obtained from filtering.

There are various trajectory management strategies, here we choose a simple one, due to that this paper just focuses on improving the computational efficiency of DP-TBD. Specifically, if $\hat{\mathbf{X}}_{k-N+1}^k$ is within several cells of the continuous trajectory $\hat{\mathbf{X}}_1^{k-1}$ at stage $k-N+1 \sim k-1$, then the final state in $\hat{\mathbf{X}}_{k-N+1}^k$ is linked with $\hat{\mathbf{X}}_1^{k-1}$, forming $\hat{\mathbf{X}}_1^k$. Otherwise a new trajectory is initiated, outputting $\hat{\mathbf{X}}_1^k = (0, 0, \cdots, 0, \hat{\mathbf{X}}_{k-N+1}^k)$ (the number of zero is $k-N+2$).

C. Problems

The SW-DP-TBD is an effective way to detect dim targets, however when the number of observed frames $M \geq N$, it suffers the increase in computational load. This is because the two consecutive SWs repeatedly process the number $N-1$ frames of data shared by them, consequently, each of the received data are processed N times. Assume that the number of possible state transitions (the size of $\tau(\mathbf{x}_k)$) is q. The calculation complexity required by the algorithm in each SW is $\mathcal{O}(|\bar{\mathbb{R}}^s|Nq)$ [21]. To process M ($M > N$) frames of data, the SW-DP-TBD needs number $M-N+1$ of consecutive SWs, with the complexity $\mathcal{O}(|\bar{\mathbb{R}}^s|Nq(M-N+1))$, where $|\bar{\mathbb{R}}^s|$ is the size of the discrete state space, equal to

$$|\bar{\mathbb{R}}^s| = N_x N_y (2N_{vx}+1)(2N_{vy}+1)$$

Compared with other TBD techniques, e.g. particle filter in which only one frame is processed at each stage and processing M frames only needs M times of recursion, SW-DP-TBD is more computationally expensive. The computational inefficiency limits DP algorithm's further development in practical applications.

IV. The Proposed RP-DP-TBD

The main procedure of SW-DP-TBD can be formulated as that at each measurement time k, it needs to calculate the multidimensional joint posterior pdf $p(\mathbf{X}_{k-N+1}^k|\mathbf{Z}_{k-N+1}^k)$ of several consecutive states. The SW processing used to solve this problem causes repetitive processing, and leads to heavy computational burden that is highly inefficient.

The problem in DP-TBD is similar to that in the single state estimation which needs to calculate $p(\mathbf{x}_k|\mathbf{Z}_1^k)$ at each measurement time. The latter one can be implemented by a recursive method that uses the C-K equation and Bayes' rule to predict and update the target state. According to this idea, we put forward a proposition that the multidimensional pdf $p(\mathbf{X}_{k-N+1}^k|\mathbf{Z}_1^k)$ in batch processing can also be solved in a similar recursive way, and then the prospective target trajectory along time $k-N+1$ to time k can be determined by

$$\hat{\mathbf{X}}_{k-N+1}^k = \arg \max_{\mathbf{X}_{k-N+1}^k \in \mathbf{S}_N(k)} \{p(\mathbf{X}_{k-N+1}^k|\mathbf{Z}_1^k)\} \quad (7)$$

Note that this can not be solved by directly applying the classical method, since that the classical one just aims at the single state, and is not available to the multidimensional pdf. This paper solves this problem from two aspects. First, we propose and deduce the MD-J filter that forms multidimensional pdf in a recursive way. Next, upon the MD-J filter, we implement the TBD method based on DP, proposing a recursive processing DP-TBD (RP-DP-TBD).

A. The MD-J filter

The classical recursive filter used to estimate the single state consists of two stages [4]: prediction and update. In this paper we aim to estimate the state sequence with constant length N. To make sure of a constant dimension estimation, at each measurement time when a new state is added to the

multidimensional joint pdf, the first dimension of the state sequence is abandoned. So that another stage (referred as dimensionality reduction) is required in MD-J filter.

Suppose that the N-D pdf $p(\mathbf{X}_{k-N}^{k-1}|\mathbf{Z}_1^{k-1})$ at time $k-1$ ($k > N$)is available. The pdf $p(\mathbf{X}_{k-N+1}^{k}|\mathbf{Z}_1^{k})$ at time k is computed according to the following steps.

1) Dimensionality reduction: To compute the N-D pdf $p(\mathbf{X}_{k-N+1}^{k}|\mathbf{Z}_1^{k})$, the N-1-D pdf $p(\mathbf{X}_{k-N+1}^{k-1}|\mathbf{Z}_1^{k-1})$ is required. So, first we need to estimate $p(\mathbf{X}_{k-N+1}^{k-1}|\mathbf{Z}_1^{k-1})$ in the light of $p(\mathbf{X}_{k-N}^{k-1}|\mathbf{Z}_1^{k-1})$, and this step is used to discard the first dimension of the state sequence $\mathbf{X}_{k-N}^{k-1}$ in $p(\mathbf{X}_{k-N}^{k-1}|\mathbf{Z}_1^{k-1})$.

The conditional probability obeys the following equation that

$$P(A,B|C) = P(A|C)P(B|A,C) \tag{8}$$

According to this equation, the state $\mathbf{x}_{k-N}$ can be separated from the sequence. As

$$\begin{aligned} &p(\mathbf{X}_{k-N}^{k-1}|\mathbf{Z}_1^{k-1}) \\ &= p(\mathbf{X}_{k-N+1}^{k-1}|\mathbf{Z}_1^{k-1})p(\mathbf{x}_{k-N}|\mathbf{X}_{k-N+1}^{k-1},\mathbf{Z}_1^{k-1}) \\ &= p(\mathbf{X}_{k-N+1}^{k-1}|\mathbf{Z}_1^{k-1})p(\mathbf{x}_{k-N}|\mathbf{x}_{k-N+1},\mathbf{Z}_1^{k-1}) \end{aligned} \tag{9}$$

The simplification in the last line of (9) is based on the Markov property mentioned in section II-A. Using the Bayes' rule, (9) can be changed as

$$\begin{aligned} p(\mathbf{X}_{k-N}^{k-1}|\mathbf{Z}_1^{k-1}) = p(\mathbf{X}_{k-N+1}^{k-1}|\mathbf{Z}_1^{k-1})\times \\ \frac{p(\mathbf{x}_{k-N}|\mathbf{x}_{k-N+1})p(\mathbf{Z}_1^{k-1}|\mathbf{X}_{k-N}^{k-N+1})}{p(\mathbf{Z}_1^{k-1}|\mathbf{x}_{k-N+1})} \end{aligned} \tag{10}$$

Upon equation (8), the $p(\mathbf{Z}_1^{k-1}|\mathbf{X}_{k-N}^{k-N+1})$ in (10) can be rewritten as

$$\begin{aligned} &p(\mathbf{Z}_1^{k-1}|\mathbf{X}_{k-N}^{k-N+1}) \\ &= p(\mathbf{Z}_1^{k-N-1}|\mathbf{X}_{k-N}^{k-N+1})p(\mathbf{Z}_{k-N}^{k-1}|\mathbf{Z}_1^{k-N-1},\mathbf{X}_{k-N}^{k-N+1}) \end{aligned} \tag{11}$$

The measurement model in section II-B shows that the measurement $\mathbf{Z}_k$ only depends on the state $\mathbf{x}_k$, so (11) can be further simplified, as

$$p(\mathbf{Z}_1^{k-1}|\mathbf{X}_{k-N}^{k-N+1}) = p(\mathbf{Z}_1^{k-N-1})p(\mathbf{Z}_{k-N}^{k-1}|\mathbf{X}_{k-N}^{k-N+1}) \tag{12}$$

then, using (8) again, and we have

$$\begin{aligned} p(\mathbf{Z}_1^{k-1}|\mathbf{X}_{k-N}^{k-N+1}) &= p(\mathbf{Z}_1^{k-N-1})p(\mathbf{Z}_{k-N}|\mathbf{X}_{k-N}^{k-N+1}) \\ &\quad\times p(\mathbf{Z}_{k-N+1}^{k-1}|\mathbf{Z}_{k-N},\mathbf{X}_{k-N}^{k-N+1}) \\ &= p(\mathbf{Z}_1^{k-N-1})p(\mathbf{Z}_{k-N}|\mathbf{x}_{k-N}) \\ &\quad\times p(\mathbf{Z}_{k-N+1}^{k-1}|\mathbf{x}_{k-N+1}) \end{aligned} \tag{13}$$

Similarly, the $p(\mathbf{Z}_1^{k-1}|\mathbf{x}_{k-N+1})$ in (10) can also be simplified as

$$p(\mathbf{Z}_1^{k-1}|\mathbf{x}_{k-N+1}) = p(\mathbf{Z}_1^{k-N})p(\mathbf{Z}_{k-N+1}^{k-1}|\mathbf{x}_{k-N+1}) \tag{14}$$

Plug (13) and (14) into (10), we get

$$\begin{aligned} p(\mathbf{X}_{k-N}^{k-1}|\mathbf{Z}_1^{k-1}) =& D\times p(\mathbf{Z}_{k-N}|\mathbf{x}_{k-N})p(\mathbf{x}_{k-N}|\mathbf{x}_{k-N+1}) \\ &\times p(\mathbf{X}_{k-N+1}^{k-1}|\mathbf{Z}_1^{k-1}) \end{aligned} \tag{15}$$

where $D = \dfrac{p(\mathbf{Z}_1^{k-N-1})}{p(\mathbf{Z}_1^{k-N})}$.

According to (15), we get the interesting pdf

$$\begin{aligned} &p(\mathbf{X}_{k-N+1}^{k-1}|\mathbf{Z}_1^{k-1}) \\ &= \frac{p(\mathbf{X}_{k-N}^{k-1}|\mathbf{Z}_1^{k-1})}{D\times p(\mathbf{Z}_{k-N}|\mathbf{x}_{k-N})p(\mathbf{x}_{k-N}|\mathbf{x}_{k-N+1})} \end{aligned} \tag{16}$$

where $p(\mathbf{Z}_{k-N}|\mathbf{x}_{k-N})$ is obtained from the measurement model, while $p(\mathbf{x}_{k-N}|\mathbf{x}_{k-N+1})$ is found from $\tilde{\mathbf{x}}_{k-1} = \mathbf{F}^{-1}(\tilde{\mathbf{x}}_k - \mathbf{v}_k)$, with $\mathbf{F}^{-1}$ the inverse matrix of $\mathbf{F}$ given in (1).

2) Prediction: This stage uses the target model (1) to obtain the N-d pdf of the state sequence terminating at time k.

$$p(\mathbf{X}_{k-N+1}^{k}|\mathbf{Z}_1^{k-1}) = p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{X}_{k-N+1}^{k-1}|\mathbf{Z}_1^{k-1}) \tag{17}$$

where $p(\mathbf{X}_{k-N+1}^{k-1}|\mathbf{Z}_1^{k-1})$ is got by step 1). In addition, this equation uses the fact that $p(\mathbf{x}_k|\mathbf{X}_{k-N+1}^{k-1},\mathbf{Z}_1^{k-1}) = p(\mathbf{x}_k|\mathbf{x}_{k-1})$, determined by the Markov property.

3) Update: At measurement time k, the kth measurement frame $\mathbf{Z}_k$ is available. This stage is used to update the N-d pdf in the light of the information contained by $\mathbf{Z}_k$ via Bayes' rule

$$p(\mathbf{X}_{k-N+1}^{k}|\mathbf{Z}_1^{k}) = \frac{p(\mathbf{Z}_k|\mathbf{x}_k)p(\mathbf{X}_{k-N+1}^{k}|\mathbf{Z}_1^{k-1})}{C} \tag{18}$$

where $C = p(\mathbf{Z}_k|\mathbf{Z}_1^{k-1})$ is the normalizing constant.

The above procedure shows the main steps of MD-J filter, and following these steps the $p(\mathbf{X}_{k-N+1}^{k}|\mathbf{Z}_1^{k})$ can be calculated in a recursive way based on $p(\mathbf{X}_{k-N}^{k-1}|\mathbf{Z}_1^{k-1})$ and $\mathbf{Z}_k$. Moreover, upon this filter, we can implement DP-TBD algorithm by the recursive processing which only processes the latest frame at each measurement time.

B. DP implementation with MD-J filter

Now, upon the MD-J filter, we use the DP algorithm to solve the optimum problem in (7), and implement the TBD method.

1) $k = N$: Since the N-D pdf at time $N-1$ does not exist, the pdf $p(\mathbf{X}_1^N|\mathbf{Z}_1^N)$ can not be calculated by the MD-J filter. So that at time $k = N$ we solve the optimum problem (7) following the traditional DP algorithm.

First, for $1 \le i \le N$, calculate (5) and (6). Then the optimum history trajectory of each value of $\mathbf{x}_N$ can be obtained by

$$\mathbf{T}(\mathbf{x}_N)_{i-1} = \mathbf{\Psi}(\mathbf{T}(\mathbf{x}_N)_i) \tag{19}$$

for $i = N, N-1 \ldots 2$, with $\mathbf{T}(\mathbf{x}_N)_N = \mathbf{x}_N$, $\mathbf{x}_N \in \mathbb{R}^s$. $\mathbf{T}(\mathbf{x}_k)$ is the optimum history trajectory of $\mathbf{x}_k$, with length N. $\mathbf{T}(\mathbf{x}_k)_i$ is the ith state of it.

Then the target trajectory $\hat{\mathbf{X}}_1^N$ is determined by

$$\begin{aligned} \hat{\mathbf{X}}_1^N = \underset{\mathbf{T}(\mathbf{x}_N),\mathbf{x}_N\in\mathbb{R}^s}{\arg\max}\ &\{p(\mathbf{T}(\mathbf{x}_N)|\mathbf{Z}_1^N)\} \\ s.t.\ p(\mathbf{T}(\mathbf{x}_N)|\mathbf{Z}_1^N) &> V_{DT} \end{aligned} \tag{20}$$

where $p(\mathbf{T}(\mathbf{x}_N)|\mathbf{Z}_1^N) = I(\mathbf{x}_N)$.

2) $k = N+1$: To determine the target trajectory $\hat{\mathbf{X}}_2^{N+1}$, we need to solve the maximization in (7). Based on the idea of DP algorithm, (7) can be split into a partial maximization on $\mathbf{X}_2^N$ [8], as

$$\begin{aligned}&\max_{\mathbf{X}_2^{N+1}\in\mathbf{S}_N(N+1)}\{p(\mathbf{X}_2^{N+1}|\mathbf{Z}_1^{N+1})\}\\&=\max_{\mathbf{x}_{N+1}\in\bar{\mathbb{R}}^s}\left\{\max_{\mathbf{X}_2^N\in\mathbf{t}(\mathbf{x}_{N+1})}\{p(\mathbf{X}_2^{N+1}|\mathbf{Z}_1^{N+1})\}\right\}\end{aligned}\tag{21}$$

where $\mathbf{t}(\mathbf{x}_k)$ is the set of short trajectories which can transfer to the specified $\mathbf{x}_k$.

Let

$$I(\mathbf{x}_{N+1})=\max_{\mathbf{X}_2^N\in\mathbf{t}(\mathbf{x}_{N+1})}\{p(\mathbf{X}_2^{N+1}|\mathbf{Z}_1^{N+1})\}\tag{22}$$

and it can be further split as

$$I(\mathbf{x}_{N+1})=\max_{\mathbf{x}_N\in\tau(\mathbf{x}_{N+1})}\left\{\max_{\mathbf{X}_2^{N-1}\in\mathbf{t}(\mathbf{x}_N)}\{p(\mathbf{X}_2^{N+1}|\mathbf{Z}_1^{N+1})\}\right\}\tag{23}$$

The maximization

$$\max_{\mathbf{X}_2^{N-1}\in\mathbf{t}(\mathbf{x}_N)}\{p(\mathbf{X}_2^{N+1}|\mathbf{Z}_1^{N+1})\}$$

is used to determined the optimum history trajectory of the state $\mathbf{x}_N$. However the history trajectory $\mathbf{T}(\mathbf{x}_N)$ of $\mathbf{x}_N$ has already been confirmed at time $k=N$, so that here we do not need to solve the (N-2)-D optimum problem. Consequently, the merit function can be simplified as

$$I(\mathbf{x}_{N+1})=\max_{\mathbf{x}_N\in\tau(\mathbf{x}_{N+1})}\left\{p(\mathbf{T}(\mathbf{x}_N)_2^N,\mathbf{x}_{N+1}|\mathbf{Z}_1^{N+1})\right\}\tag{24}$$

where $\mathbf{T}(\mathbf{x}_N)_2^N$ denotes the last $N-1$ states of $\mathbf{T}(\mathbf{x}_N)$.

Next, referring to recursive procedure (17) and (18) in section IV-A, we get

$$\begin{aligned}&p(\mathbf{T}(\mathbf{x}_N)_2^N,\mathbf{x}_{N+1}|\mathbf{Z}_1^{N+1})\\&=\frac{p(\mathbf{Z}_{N+1}|\mathbf{x}_{N+1})p(\mathbf{x}_{N+1}|\mathbf{x}_N)p(\mathbf{T}(\mathbf{x}_N)_2^N|\mathbf{Z}_1^N)}{C}\end{aligned}\tag{25}$$

where upon (16)

$$\begin{aligned}&p(\mathbf{T}(\mathbf{x}_N)_2^N|\mathbf{Z}_1^N)\\&=\frac{p(\mathbf{T}(\mathbf{x}_N)|\mathbf{Z}_1^N)}{D\times p(\mathbf{Z}_1|\mathbf{T}(\mathbf{x}_N)_1)p(\mathbf{T}(\mathbf{x}_N)_1|\mathbf{T}(\mathbf{x}_N)_2)}\\&=\frac{I(\mathbf{x}_N)}{D\times p(\mathbf{Z}_1|\mathbf{T}(\mathbf{x}_N)_1)p(\mathbf{T}(\mathbf{x}_N)_1|\mathbf{T}(\mathbf{x}_N)_2)}\end{aligned}$$

Plug (25) into (24). C and D are constants and need not be carried along, thus, $I(\mathbf{x}_{N+1})$ can be rewritten as

$$\begin{aligned}&I(\mathbf{x}_{N+1})=p(\mathbf{Z}_{N+1}|\mathbf{x}_{N+1})\\&\times\max_{\mathbf{x}_N\in\tau(\mathbf{x}_{N+1})}\{p(\mathbf{x}_{N+1}|\mathbf{x}_N)p(\mathbf{T}(\mathbf{x}_N)_2^N|\mathbf{Z}_1^N)\}\end{aligned}\tag{26}$$

Record that

$$\mathbf{T}(\mathbf{x}_{N+1})_{N-1}=\arg\max_{\mathbf{x}_N\in\tau(\mathbf{x}_{N+1})}\{p(\mathbf{x}_{N+1}|\mathbf{x}_N)p(\mathbf{T}(\mathbf{x}_N)_2^N|\mathbf{Z}_1^N)\}$$

$$\mathbf{T}(\mathbf{x}_{N+1})_1^{N-2}=\mathbf{T}(\mathbf{T}(\mathbf{x}_{N+1})_{N-1})_2^{N-1}$$

Then the target trajectory $\hat{\mathbf{X}}_2^{N+1}$ can be obtained by computing the maximization as the equation (20) does.

3) $k\geq N+2$: The task of the algorithm at time $k\geq N+2$ is also to solve the optimum problem in (7), which is the same as that of time $k=N+1$, so that the target trajectories can be obtained following the procedure introduced when $k=N+1$. The algorithm discards the frame $\mathbf{Z}_{k-N}$, and processes the latest frame $\mathbf{Z}_k$.

The above recursive propagations constitute the proposed RP-DP-TBD algorithm, and the main steps of the algorithm are shown in Algorithm 1. The short trajectories outputted by the DP processer can also be handled by the trajectory management, with the same strategy as SW-DP-TBD.

It is worth pointing that after taking the logarithm, the multiplication in (27) and (28) can be changed to the addition, which is more easily implemented. This simple form is often exploited in the practical applications.

Algorithm 1: RP-DP-TBD algorithm: $k\geq N+1$

1)dimensionality reduction

for $\mathbf{x}_{k-1}\in\bar{\mathbb{R}}^s$ **do**

$$\begin{aligned}&p(\mathbf{T}(\mathbf{x}_{k-1})_2^N|\mathbf{Z}_1^{k-1})\\&=\frac{I(\mathbf{x}_{k-1})}{p(\mathbf{Z}_{k-N}|\mathbf{T}(\mathbf{x}_{k-1})_1)p(\mathbf{T}(\mathbf{x}_{k-1})_1|\mathbf{T}(\mathbf{x}_{k-1})_2)}\end{aligned}\tag{27}$$

end

2)computing merit function

for $\mathbf{x}_k\in\bar{\mathbb{R}}^s$ **do**

$$\begin{aligned}&I(\mathbf{x}_k)=p(\mathbf{Z}_k|\mathbf{x}_k)\times\\&\max_{\mathbf{x}_{k-1}\in\tau(\mathbf{x}_k)}\{p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{T}(\mathbf{x}_{k-1})_2^N|\mathbf{Z}_1^{k-1})\}\end{aligned}\tag{28}$$

$$\mathbf{T}(\mathbf{x}_k)_{N-1}=\arg\max_{\mathbf{x}_{k-1}\in\tau(\mathbf{x}_k)}\{p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{T}(\mathbf{x}_{k-1})_2^N|\mathbf{Z}_1^{k-1})\}$$

$$\mathbf{T}(\mathbf{x}_k)_1^{N-2}=\mathbf{T}(\mathbf{T}(\mathbf{x}_k)_{N-1})_2^{N-1}$$

$$\mathbf{T}(\mathbf{x}_k)_N=\mathbf{x}_k$$

end

3)determining target trajectory

$$\begin{aligned}&\hat{\mathbf{X}}_{k-N+1}^k=\arg\max_{\mathbf{T}(\mathbf{x}_k),\mathbf{x}_k\in\bar{\mathbb{R}}^s}\{I(\mathbf{x}_k)\}\\&s.t.\ I(\mathbf{x}_k)>V_{DT}\end{aligned}\tag{29}$$

C. The types of merit function

In the above discussion, the merit function is the posterior pdf. However, the proposed RP-DP-TBD is also available for other types of merit functions. E.g., in the cases that lack of the prior information, the summation of the measurements is selected as the merit function [14] of traditional DP-TBD.

Similarly, the RP-DP-TBD can use these types too by directly changing the recursive relationship (27) and (28) to

$$\sum_{i=2}^{N} z(\mathbf{T}(\mathbf{x}_{k-1})_i) = I(\mathbf{x}_{k-1}) - z(\mathbf{T}(\mathbf{x}_{k-1})_1) \tag{30}$$

$$I(\mathbf{x}_k) = z(\mathbf{x}_k) + \max_{\mathbf{x}_{k-1}\in\tau(\mathbf{x}_k)} \left\{ \sum_{i=2}^{N} z(\mathbf{T}(\mathbf{x}_{k-1})_i) \right\} \tag{31}$$

where, $z(\mathbf{x}_i)$ denotes the measurement of the state $\mathbf{x}_i$, and for simplicity, the transition probability $p(\mathbf{x}_i|\mathbf{x}_{i-1})$ and $p(\mathbf{x}_{i-1}|\mathbf{x}_i)$ are approximated to a constant $(1/q)$ and removed. This approximation is reasonable, which has already been demonstrated in [11].

D. Complexity analysis

At time $k = N$, the RP-DP-TBD jointly processes N frames of data as the SW-DP-TBD does, with the complexity $\mathcal{O}(|\bar{\mathbb{R}}^s|Nq)$. Then, at time $k \geq N+1$, it proceeds as the Algorithm 1. Since that compared with the optimum problem (28), the complexity of dimensionality reduction (27) is negligible. Thus, the complexity of RP-DP-TBD is ruled by the second loop of the algorithm, which is equal to $\mathcal{O}(|\bar{\mathbb{R}}^s|q)$. Consequently, the complexity required by the algorithm to process M frames of data is

$$\mathcal{O}(|\bar{\mathbb{R}}^s|Nq) + (M-N) \times \mathcal{O}(|\bar{\mathbb{R}}^s|q) = \mathcal{O}(|\bar{\mathbb{R}}^s|Mq)$$

Recall that the complexity of SW-DP-TBD is $\mathcal{O}(|\bar{\mathbb{R}}^s|Nq(M-N+1))$. Compared with the traditional one, the complexity of RP-DP-TBD reduced by

$$\frac{\mathcal{O}(|\bar{\mathbb{R}}^s|Nq(M-N+1))}{\mathcal{O}(|\bar{\mathbb{R}}^s|Mq)} = N - \frac{(N^2-N)}{M} \approx N$$

which approaches to N, when the number of total frames (M) is big enough.

V. Numerical results

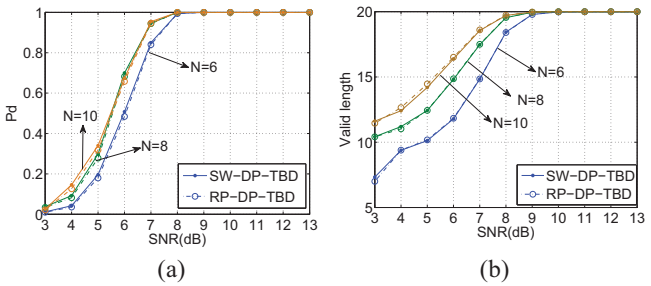
In this part, we consider the scene that a single point target moves in the X-Y plane. For simplicity, the velocity of the target is assumed to be known. The measurement noise of the simulation scene is complex Gaussian distributed. The detection threshold V_{DT} is chosen to achieve a constant probability of false alarm. Except Fig.3 and Fig.4 the length of SW in other simulation scenes is $N = 6$.

Assume that the number of Monte Carlo (MC) runs is C and at the ith ($1 \leq i \leq C$) MC run there are a number $L(i)$ of stages at which the estimated trajectory is within 2 cells of the actual target trajectory. We let $d(i) = 1$ mean that in the ith MC run the target is detected. Define

$$d(i) = \begin{cases} 1 & L(i) \geq 1 \\ 0 & others \end{cases}$$

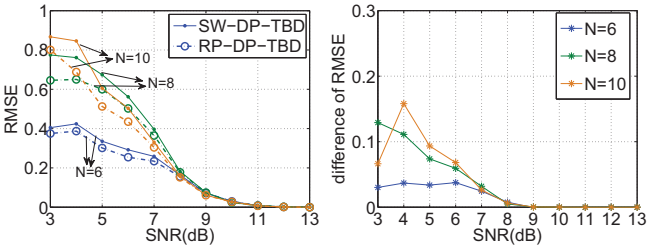
The statistics used in this part to assess the algorithm performance are defined as

$$P_d = \frac{\sum_{i=1}^{C} d(i)}{C} \ , \ V_l = \frac{\sum_{i=1}^{C} L(i)}{C}$$



(a) (b)

Fig. 3. (a) The number of total frames is $M = 20$, and the lengths of SW are $N = 6, 8, 10$, respectively, P_d of RP-DP-TBD and SW-DP-TBD versus SNR. (b) The number of total frames is $M = 20$, and the lengths of SW are $N = 6, 8, 10$, respectively, valid length of RP-DP-TBD and SW-DP-TBD versus SNR.



(a) (b)

Fig. 4. (a) The number of total frames is $M = 20$, and the lengths of SW are $N = 6, 8, 10$, respectively, RMSE of RP-DP-TBD and SW-DP-TBD versus SNR. (b) The difference of the RMSE in (a).

where P_d is the detection probability and V_l is the valid length. RMSE is defined as the root mean square position error.

Fig.3 and Fig.4 show the P_d, V_l and RMSE of RP-DP-TBD and SW-DP-TBD under different lengths of SW. From Fig.3 it is clear that whatever the length of SW is, the detection performance (denoted by P_d and V_l) of RP-DP-TBD is always approximately equivalent to that of SW-DP-TBD, suggesting that the proposed RP-DP-TBD is reasonable. Notice that when the SNR is low the P_d and V_l increase along with the growth of N. This can be explained as that with the increase of the number of processing frames in one batch, the characteristic of the target will be more obvious and the target is more easily to be detected.

It can be seen from Fig.4(a) that the RMSE of the two algorithms both decrease as the SNR increases. However, shown in Fig.4(b), under the low SNR (e.g.,SNR<8dB), the RMSE of RP-DP-TBD is less than that of SW-DP-TBD, implying that the RP-DP-TBD can provide a more accurate trajectory estimation.

Fig.3 shows that compared with SW-DP-TBD, RP-DP-TBD suffers a little detection performance degradation, however, this does not mean that the proposed RP-DP-TBD is a suboptimal algorithm. To further explain this, one run of simulation is conducted here and the results are illustrated in Fig.5 and Table I. Since that the SNR of the given target is not strong

TABLE I
THE MAXIMUM VALUES OF MERIT FUNCTION THAT ARE CALCULATED AT EACH MEASUREMENT TIME BY SW-DP-TBD AND RP-DP-TBD RESPECTIVELY.

Measurement time	6	...	9	10	11	12	13	...	20
SW-DP-TBD	14.3022	...	13.8186	14.1410	14.9264	18.0824	19.2564	...	13.4292
RP-DP-TBD	14.3022	...	13.8186	14.1410	14.8292	18.0824	19.2564	...	13.4292

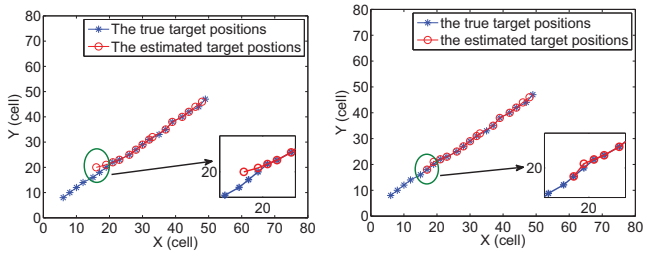


Fig. 5. (a) The target trajectory estimated by the SW-DP-TBD in the case where the number of total frames is $M = 20$, the length of the SW is 6, and the target SNR is 7 dB. (b) The target trajectory estimated by the RP-DP-TBD in the same case with (a).

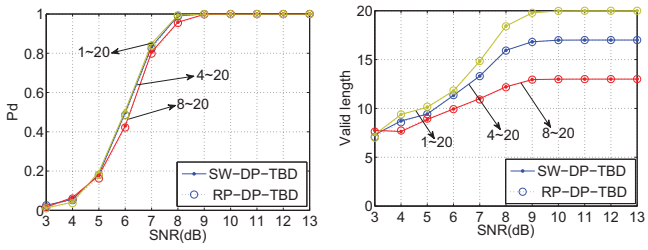


Fig. 6. (a) The number of total frames is $M = 20$, P_d of RP-DP-TBD and SW-DP-TBD for targets which appear at the first, the 4th and the 8th stage, respectively. (b) The number of total frames is $M = 20$, valid length of RP-DP-TBD and SW-DP-TBD for targets which appear at the first, the 4th and the 8th stage, respectively.

enough, the first several states do not be detected and the estimated trajectories of the two algorithms are both declared at the 11th measurement time, and start with the 6th state. In Table I, it is obvious that the merit function at the 11th measurement time calculated by the SW-DP-TBD is bigger than the corresponding one calculated by RP-DP-TBD, but the trajectory estimated by SW-DP-TBD contained a wrong position, shown in Fig.5, circled by the green ellipse. In contrast, RP-DP-TBD provide a more accurate estimation with a smaller merit function. This is because the framework of the two algorithm is different.

At each measurement time, SW-DP-TBD accumulates the measurement frames contained by the current SW afresh. When the number of the accumulated frames is small, the target characteristic is not obvious and the clutter does not be suppressed. So that to pursue a much bigger merit function, SW-DP-TBD may regard the clutter as the prospective target at the first several stages in the SW. Although the bigger merit function can lead to a higher P_d, it forms an inaccurate trajectory estimation.

In other hand, RP-DP-TBD is based on the idea of filtering, it not only makes use of the information provided by the current SW, but also exploits the results of the previous SWs. So that the target characteristic can be maintained along with time, and the algorithm distinguish the clutter and target state more accurately. Although its merit function is not as big as that of SW-DP-TBD, it can provide a more accurate trajectory estimation, as shown in Fig.4.

In section IV, we did not take into account whether the appearance time of target affects the performance of RP-DP-TBD algorithm. Here further simulation is given to value the performance of RP-DP-TBD under this condition. Fig.6

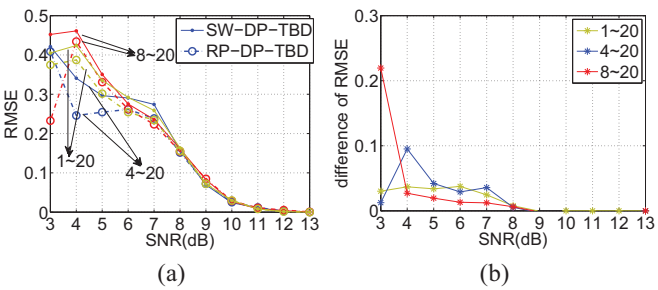


Fig. 7. (a) The number of total frames is $M = 20$, RMSE of RP-DP-TBD and SW-DP-TBD for targets which appear at the first, the 4th and the 8th stage, respectively. (b) The difference of the RMSE in (a).

and Fig.7 provide the numerical results of a particular scene where the target appears at different times. It can be seen that for the same target SNR, whenever the target appears, the detection probability and valid length of RP-DP-TBD are almost equivalent to that of SW-DP-TBD, suggesting that the uncertainty as to the appearance time of targets does not affect the performance of RP-DP-TBD. In addition, shown in Fig.7, the RMSE of RP-DP-TBD is also less than that of the SW-DP-TBD, so that the target positions estimated by RP-DP-TBD are more reliable.

Fig.8 shows the valid length of RP-DP-TBD and SW-DP-TBD in the scenes where the number of total frames is $M = 10, 20, \ldots, 100$ respectively. It can be shown that as the number of total frames increases the valid length of RP-DP-TBD always stays the same level with that of SW-DP-TBD. This suggests that the performance of RP-DP-TBD is as stable as the traditional SW-DP-TBD.

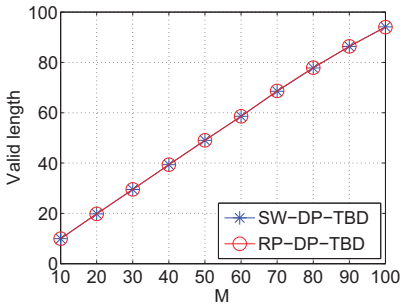


Fig. 8. Target lasts from the first stage to the Mth stage, and its SNR $= 9$ dB, as the number of total frames (M) increases from 10 to 100, the V_l of RP-DP-TBD and SW-DP-TBD.

VI. Conclusion

This paper presents a computationally efficient DP-TBD algorithm (i.e. RP-DP-TBD) based on the proposed MD-J filter. The RP-DP-TBD breaks down the SW framework of traditional SW-DP-TBD, and at each measurement time, unlike SW-DP-TBD which jointly processes N frames of data, RP-DP-TBD only needs to process one frame of data. Consequently, compared with SW-DP-TBD, the complexity of RP-DP-TBD nearly reduces by N times. In addition, due to the use of MD-J filter, RP-DP-TBD exploits the previous results to estimate the current states, using more information implicitly and generating a more accurate trajectory. The simulation results show that the proposed RP-DP-TBD performs robustly. It provides almost the same detection performance as SW-DP-TBD but providing more accurate target positions, meanwhile, it implements a significant computational complexity reduction.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grants 61201276, 61178068, and 61301266, the Fundamental Research Funds of Central Universities under Grants ZYGX2012Z001 and ZYGX2013J012, the Chinese Postdoctoral Science Foundation under Grant 2014M550465, and by the Program for New Century Excellent Talents in University under Grant A1098524023901001063.

References

[1] Y. Bar-Shalom, and W. D. Blair, "*Multitarget-Multisensor Tracking: Applications and Advances,Vol. III.*" Norwood, MA: Artech House, 2000.

[2] S. J. Davey and M. G. Rutten, "A Comparison of Three Algorithms for Tracking Dim Targets," in *Information, Decision and Control, 2007. IDC '07*, 2007, pp. 342-347.

[3] D. J. Salmond, and H. Birch, "A particle filter for track-before-detect," *Proceedings of the American Control Conference*, vol. 5, pp. 3755-3760, 2001.

[4] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, pp. 174-188, 2002.

[5] J. Illingworth and J. Kittler, "The Adaptive Hough Transform," *IEEE Trans. Patt.Anal.Mach.Intell.*, vol. PAMI-9, pp. 690-698, 1987.

[6] L. R. Moyer, J. Spak, and P. Lamanna, "A multi-dimensional Hough transform-based track-before-detect technique for detecting weak targets in strong clutter backgrounds," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 47, pp. 3062-3068, 2011.

[7] V. Ba-Ngu, V. Ba-Tuong, P. Nam-Trung, and D. Suter, "Joint Detection and Estimation of Multiple Objects From Image Observations," IEEE Trans. Signal Process., vol. 58, pp. 5129-5141, Oct. 2010.

[8] R. Larson and J. Peschon, "A dynamic programming approach to trajectory estimation," *IEEE Trans. Autom. Control*, vol. 11, pp. 537-540, 1966.

[9] W. Yi, M. R. Morelande, L.J. Kong, and J.Y. Yang, "Target tracking for an unknown and time-varying number of targets via particle filtering," *17th Int. Conf. Inform. Fusion*, pp. 309-316, 2012.

[10] Y. Barniv, "Dynamic Programming Solution for Detecting Dim Moving Targets," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-21, pp. 144-156, 1985.

[11] Y. Barniv and O. Kella, "Dynamic Programming Solution for Detecting Dim Moving Targets Part II: Analysis," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-23, pp. 776-788, 1987.

[12] J. Arnold, S. W. Shaw, and H. Pasternack, "Efficient target tracking using dynamic programming," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 29, pp. 44-56, 1993.

[13] S. M. Tonissen and R. J. Evans, "Target tracking using dynamic programming: algorithm and performance," *Proc. IEEE Conf. Decision and Control,*, pp. 2741-2746 vol.3, 1995.

[14] S. M. Tonissen and R. J. Evans, "Peformance of dynamic programming techniques for Track-Before-Detect," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 32, pp. 1440-1451, 1996.

[15] L. A. Johnston and V. Krishnamurthy, "Performance analysis of a dynamic programming track before detect algorithm," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, pp. 228-242, 2002.

[16] S. Buzzi, M. Lops, and L. Venturino, "Track-before-detect procedures for early detection of moving target from airborne radars," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 41, pp. 937-954, 2005.

[17] S. Buzzi, M. Lops, L. Venturino, and M. Ferri, "Detection of an Unknown Number of Targets via Track-Before-Detect Procedures," *Proc. IEEE Int. Radar Conf.*, pp. 180-185, 2007.

[18] D. Orlando, L. Venturino, M. Lops, and G. Ricci, "Track-Before-Detect Strategies for STAP Radars," *IEEE Trans. Signal Process.*, vol. 58, pp. 933-938, 2010.

[19] E. Grossi, L. Venturino, and M. Lops, "A two-step multi-frame detection procedure for radar systems," *Inf. Fusion. Int. Conf*, pp. 1196-1201, 2012.

[20] E. Grossi, M. Lops, and L. Venturino, "A Novel Dynamic Programming Algorithm for Track-Before-Detect in Radar Systems," *IEEE Trans. Signal Process.*, vol. 61, pp. 2608-2619, 2013.

[21] W. Yi, M. R. Morelande, L.J. Kong, and J.Y. Yang, "An Efficient Multi-Frame Track-Before-Detect Algorithm for Multi-Target Tracking," *IEEE J. Sel. Top. Signal Process*, vol. 7, pp. 421-434, 2013.

[22] L. Bruno, P. Braca, J. Horstmann, and M. Vespe, "Experimental Evaluation of the Range−Doppler Coupling on HF Surface Wave Radars," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, pp. 850-854, 2013.

[23] B. Errasti-Alcala and P. Braca, "Track before Detect algorithm for tracking extended targets applied to real-world data of X-band marine radar," *Inf. Fusion. Int. Conf*, pp. 1-8, 2014.

[24] D. Orlando, F. Ehlers, and G. Ricci, "Track-before-detect algorithms for bistatic sonars," in *Proc. 2nd Int.Workshop Cognitive Inf. Process.*, pp. 180-185, 2010.

[25] K. Magnusson, J. Jalden, P. Gilbert, and H. Blau, "Global linking of cell tracks using the Viterbi algorithm," emphIEEE Trans. Medical Imaging. vol. pp. 1-1, 2014.

[26] W. Yi, L. J. Kong, and J. Y. Yang, "Thresholding Process Based Dynamic Programming Track-Before-Detect Algorithm," *IEICE Trans. Commun.*, vol. E96.B, pp. 291-300, 2013.

[27] H. Cho and J. Chun, "A new TBD-DP algorithm using multiple IR sensors to locate the target launch point," *Proc. of SPIE*, vol. 8185, p. 81850P, 2011.

[28] E. Grossi, M. Lops, and L. Venturino, "A Heuristic Algorithm for Track-Before-Detect With Thresholded Observations in Radar Systems," *IEEE Signal Process. Lett.*, vol. 20, pp. 811-814, 2013.

[29] R. Liu, W. Yi, L. J. Kong and X. B. Yang, "Recursive Filtering for Target Tracking in Multi-frame Track-Before-Detect," in *Proc. 17th Int. Conf. Inform. Fusion*, 2014

[30] J.H. Wang, W. Yi, L.J. Kong and J.Y. Yang, "A Computationally Efficient Recursive Processing for Multi-Frame Track-Before-Detect," *IEEE Int. Radar Conf.*, 2015, accepted.

[31] Z.C. Fang, W. Yi, G.L. Cui, L.J. Kong, R. Liu and J.Y. Yang, "A Fast Implementation of Dynamic Programming Based Track-Before-Detect for Radar System," *IEEE Int. Radar Conf.*, 2015, accepted.