symmetry

Article

Parallel Computing Based Dynamic Programming Algorithm of Track-before-Detect

Qiang Guo [1,2], Zhenwu Li [2,*], Wenming Son [1] and Wenyu Fu [2]

1 Institute of Aeronautical Radio Electronics of China, Shanghai 201100, China; guoqiang@hrbeu.edu.cn (Q.G.); windmill_latest@163.com (W.S.)
2 College of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China; l20122005@gmail.com
* Correspondence: lizhenwu@hrbeu.edu.cn

Received: 19 November 2018; Accepted: 11 December 2018; Published: 31 December 2018

Abstract: The conventional dynamic programming-based track-before-detect (DP-TBD) methods are usually intractable in multi-target scenarios. The adjacent targets may interfere with each other, and the computational complexity is increased with the number of targets. In this paper, a DP-TBD method using parallel computing (PC-DP-TBD) is proposed to solve the above problems. The search region of the proposed PC-DP-TBD is divided into several parts according to the possible target movement direction. The energy integration is carried out independently and parallel in each part. This contributes to reducing the computational complexity in each part, since the divided search region is smaller than the whole one. In addition, the target energy can only be integrated adequately in the part in which the search direction matches the target movement. This is beneficial to improve the ability to detect the targets with various movement directions in different parts with different search directions. The solution to the problem of the adjacent targets interfering with each other is discussed. The procedure of the parallel computing in the proposed PC-DP-TBD is presented in detail. Simulations are conducted to verify the superiority of the proposed PC-DP-TBD in terms of detection probability and computational complexity.

Keywords: parallel computing; track-before-detect; dynamic programming; weak target; computational complexity

1. Introduction

Weak target detection and tracking is one of the difficult problems in modern radar systems. A practical sensor usually provides a data image, wherein each pixel corresponds to the received power in a particular spatial location (e.g., range bins and azimuth beams). The common approaches apply a threshold to the data, and to treat those cells that exceed the threshold as point measurements, interpolation methods are used to improve accuracy generally. This is acceptable when the signal-to-noise ratio (SNR) is high. However, military targets like aircraft and warships usually have lower SNR due to their low radar-cross-section (RCS) and complex noise environments. For low SNR targets, the target echo intensity is too low to exceed the threshold in a procedure of single-frame detection, which has a miserable influence on the target tracking since the conventional tracking procedures are usually based on the detected points. The threshold must be low enough to obtain a sufficient probability of target detection. However, a low threshold also results in a high rate of false detections, which leads the tracker to form false tracks. To implement weak target detection and tracking, a more robust technique is proposed, i.e., the track-before-detect (TBD) technique [1]. It can achieve a superior detection performance for the low SNR targets by jointly processing several consecutive frames of raw data, which is unthresholded. TBD avoids the requirement for the direct measurement-to-track association; it combines the target detection and estimation processes, which are usually sequentially applied to sensor data in a conventional system, and the association procedure is embedded in the TBD framework.

Existing TBD techniques include dynamic programming-based TBD (DP-TBD) [2–4], particle filtering-based TBD (PF-TBD) [5–7], Hough transform-based TBD (HT-TBD) [8,9], and so on. The HT-TBD technique extracts the characteristic parameters of the target signal in the parameter space by applying the Hough transform. It is naturally not suitable for non-linear motion targets, so few research works have been devoted to HT-TBD on radar systems, especially multi-target detection and tracking. In recent years, PF-TBD and DP-TBD have become some of the hot research directions, but they both face the problem of large computational complexity. Particle filtering is a method for finding the suboptimal solutions of Bayesian estimation. Unlike the DP-TBD method, PF-TBD conducts multi-frame data integration in the recursive filtering process. DP-TBD treats target detection and tracking as a multi-stage decision problem, finds the optimal decision by the merit function, and obtains the target state and track simultaneously. Although DP-TBD usually has a poor detection performance for the maneuvering targets, it is easier to implement in practical applications and more capable in multi-target scenarios than PF-TBD. Therefore, we mainly focus on the DP-TBD technique in this paper.

Buzzi proposed a dynamic programming-based TBD algorithm [10] for the multi-target problem firstly. This method achieved a valid detection and tracking when targets were well separated. A detailed study was carried out in [11,12] to solve the interference problem when targets are in proximity. The proposed generalized detection procedure achieved a computational complexity that was almost linear to the number of targets and more efficient than the exponential computational complexity resulting from the high-dimensional problem. However, the existing methods still suffer a heavy computational cost. In this paper, a parallel computing-based dynamic programming algorithm of track-before-detect referred to as PC-DP-TBD is studied. The purpose of PC-DP-TBD is to reduce the computational cost and detect proximity targets effectively in the meantime. Two contributions are made towards this problem. Firstly, by utilizing multi-core processor's parallel computing capabilities, the computational burden is divided into several cores. Second, as the process of energy integration is directional, we use different CPU cores to perform the procedures of the technique proposed in [11] along different directions and several searching spaces, respectively. Simulation results show that the proposed PC-DP-TBD algorithm effectively improves the performance on target detection probability and computational expense.

The rest of this paper is organized as follows. Firstly, mathematical models and a brief description of the DP-TBD technique are introduced in Section 2. Then, we discuss multi-target TBD problems and give the PC-DP-TBD method in Section 3. In Section 4, simulation experiments and result analysis for the proposed PC-DP-TBD method are presented. Finally, some conclusions are given in the last section.

2. Models and Method Statement

2.1. Target Dynamic Model and Measurement Model

Different from the point measurement model of conventional tracking algorithms, TBD requires a statistical model of the echo signal from each sensor pixel. The measurement at each radar scan is an image of an arbitrary dimensionality with $N_x \times N_y$ pixels indexed by $i = 1...N_x$ in the X axis and the $j = 1...N_y$ in Y axis, respectively. Recent TBD research [13,14] on radar considers extended targets with Gaussian or non-Gaussian noise, such as Rayleigh noise [15] and KA-distributed noise [16,17]. Due to the complexity of the multi-target tracking problem, we assume the point target of Constant Velocity (CV) model with Gaussian noise for simplicity in this paper. TBD is a batch-processing technique performed in a discrete state space consisting of K measurement frames from K consecutive scans, and the target state at the k^{th} measurement frame is given by:

$$\mathbf{x}_k = [x_k, \dot{x}_k, y_k, \dot{y}_k]', \tag{1}$$

where x_k and y_k denote the position of the discrete target state at the k^{th} radar scan on the X-Y plane, while $\dot{x}_k$ and $\dot{y}_k$ denote the velocity towards the X and Y axis, respectively. Then, the motion of an individual target can be modeled in a state space $\mathbb{R}^4$ [11]. The state evolution can be written as:

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{u}_k\,, \tag{2}$$

where $\mathbf{F}$ is the transition matrix and $\mathbf{u}_k$ is a Gaussian distributed process noise with covariance $\mathbf{Q}$, and $\mathbf{F}$ and $\mathbf{Q}$ are presented respectively by:

$$\mathbf{F} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

$$\mathbf{Q} = q_1 \begin{bmatrix} T^3/3 & T^2/2 & 0 & 0 \\ T^2/2 & T & 0 & 0 \\ 0 & 0 & T^3/3 & T^2/2 \\ 0 & 0 & T^2/2 & T \end{bmatrix}, \tag{4}$$

where T denotes the fixed frame period of the radar and q_1 represents the intensity of the process noise. Then, the target state transition is assumed to be a first-order Markov model and depicted by:

$$\mathbf{x}_k | \mathbf{x}_{k-1} \sim N(\cdot; \mathbf{F}\mathbf{x}_{k-1}, \mathbf{Q})\,, \tag{5}$$

where $N(x; \mu, \mathbf{\Omega})$ denotes the Gaussian probability density evaluated at x with mean μ and covariance matrix $\mathbf{\Omega}$. When the radar is scanning K frames, the estimated target track among a processing batch of K frames can be written as:

$$\mathbf{X}_{1:K} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K) \tag{6}$$

As previously demonstrated, the surveillance region depicted by a two-dimensional (2D) plane is divided into a grid of $N_x \times N_y$ resolution cells [1]. Then, the k^{th} measurement frame defined as $\mathbf{z_k}$ is an $N_x \times N_y$ array whose $(i,j)^{\text{th}}$ element $z_k(i,j)$ denotes the measurement intensity of the $(i,j)^{\text{th}}$ cell at time k, given by:

$$z_k(i,j) = \begin{cases} w_k(i,j) & \text{no target in cell } (i,j) \\ A_k + w_k(i,j) & \text{target exist in } (i,j) \end{cases}, \tag{7}$$

where $i \in (1, 2, \cdots, N_x)$ and $j \in (1, 2, \cdots, N_y)$. Target signal A_k is a complex random variable whose amplitude is assumed to be constant for simplicity [18], i.e., $A = |A_k|$ undergoes an unknown phase assumed to be uniformly distributed over $0 \sim 2\pi$. $w_k(i,j)$ is assumed to be n independent identically-distributed (IID) complex Gaussian white noise with the mean value of zero. K measurement frames can be expressed as:

$$\mathbf{Z}_{1:K} = (\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_K) \tag{8}$$

2.2. Basic Dynamic Programming Track-before-Detect (TBD) Algorithm

According to the method proposed in [2], DP-TBD is implemented using the measurement data $\mathbf{Z}_{1:K}$ of the first K frames. The dynamic programming algorithm is an equivalent implementation of the exhaustive search method. DP-TBD adopts a merit function to search a dim target and the optimal estimation of the target trajectory in the discrete state spaces. Assuming the merit function of former $k-1$ frames is $I(\mathbf{x}_{k-1})$, then the integration merit function of the state $\mathbf{x}_k$ corresponding to the k^{th} frame can be derived by:

$$I(\mathbf{x}_k) = z_k(\mathbf{x}_k) + \max_{\mathbf{x}_{k-1} \in \Gamma(\mathbf{x}_k)} I(\mathbf{x}_{k-1}) \tag{9}$$

where $\Gamma(\mathbf{x}_k)$ denotes the state transition set consisting of possible states at time $k-1$, which is determined by the position and velocity of the target. The merit function of the initial state is $I(\mathbf{x}_1)$. The target state transition process is illustrated in Figure 1, and the state transition set of the target is well described in [18,19] and will not be repeated here. For all $\mathbf{x}_k \in \mathbb{R}^4$, the state transition relationship between frames is given by:

$$\mathbf{\Phi}_{\mathbf{x}_k}(k) = \arg \max_{\mathbf{x}_{k-1} \in \Gamma(\mathbf{x}_k)} I(\mathbf{x}_k) \tag{10}$$

where $\mathbf{\Phi}_{\mathbf{x}_k}(k)$ is used to store the state transition relation of the integrated merit function between frames and the initial state $\mathbf{\Phi}_{\mathbf{x}_k}(1) = 0$.

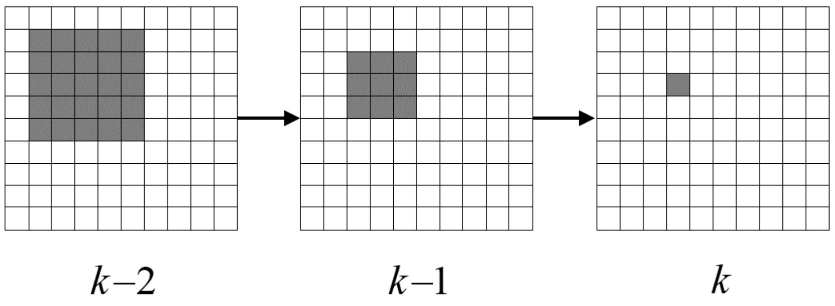

Figure 1. Target state transition process illustrated by admissible search regions among successive scans. The target velocity is assumed to be one cell/frame, and its position in the k^{th} frame is given. The corresponding transition areas in the $k-1^{\text{th}}$ and $k-2^{\text{th}}$ frames are represented by the shadow cells.

3. Multi-Target Dynamic Programming for Track-before-Detect

3.1. Target Cancellation

The multi-target tracking problem is well studied in [11], and a multi-target track-before-detect algorithm (MT-DP-TBD) was proposed. Unlike the multi-target tracking model (MTT) proposed in [20], which is a concatenation of N individual target states, i.e., $\mathbf{x}_k = [\mathbf{x}_k^1, \mathbf{x}_k^2, ..., \mathbf{x}_k^N]$, MT-DP-TBD considers the tracking of an unknown number of targets with the single-target model defined in (1), thereby avoiding the hypothesis testing of N. The key procedure is known as target cancellation. Its main idea is selecting the maximum value of $\{I(\mathbf{x}_K), \mathbf{x}_K \in \mathbb{R}^4\}$ that exceeds the threshold V_t; the position of this value in the K^{th} integration plane represents where the target is most possible. After determining the trajectory $\hat{\mathbf{X}}_{1:K}^i$ of the most possible target i, detach the measurement information related to $\hat{\mathbf{X}}_{1:K}^i$ from the original measurement data $Z_{1:K}$, then re-run DP procedures to search repeatedly for the maximum measurement information until the merit function does not exceed the threshold. This method turns the multi-target dynamic programming problem into the multiple single-target dynamic programming problem, making the computational dimension lower. At the same time, eliminating the relevant information of the optimal target solves the merit function interference problem caused by the adjacent and cross-way targets.

3.2. Parallel Computing-Based DP-TBD

From the analysis of the MT-DP-TBD algorithm [11], it can be concluded that in the process of cyclical target cancellation, the detection of multiple targets becomes a serial single-target search process, meaning the execution time of the algorithm is prolonged. Therefore, we propose the PC-DP-TBD method to detect and track the targets with different motion orientations at the same time, which achieves superior performance and reduces the time consumption for multi-target tracking through parallel computing.

3.2.1. Partition of the Target State Transition Set

Generally, the target states between radar scans have an obvious correlation, which is directly related to the orientation of the target state transition. Since the target state of the previous frame can partially determine the target state of the next frame, the target state transition set $\Gamma(\mathbf{x}_k)$ can directly affect the result of DP integration. That is, if $\Gamma(\mathbf{x}_k)$ used to perform DP search matches the target motion orientation, the integration result will be the maximum extent, making it easier to detect the target.

However, due to the fact that the target motion orientation is unknown, as shown in Figure 2, $\Gamma(\mathbf{x}_k)$ is usually assumed to be a discretized state space of the circle area $\mathbf{S}$ with the target position as the center. $\mathbf{S}$ is known as the potential target-hidden area of the $k-1^{\text{th}}$ measurement frame, so the radius of the circle is determined by a predefined maximum allowable target velocity V^{max}. In the rasterized X-Y plane given by $\mathbf{z}_k$, we consider that $\Gamma(\mathbf{x}_k)$ consists of $q \in \{1, 9, 25, ...\}$ possible states and the corresponding $V^{max}(cell/frame) \in \{0, 1, 2, \ldots\}$ [18], e.g., $q = 9$ and $V^{max} = 1$ in Figure 1.

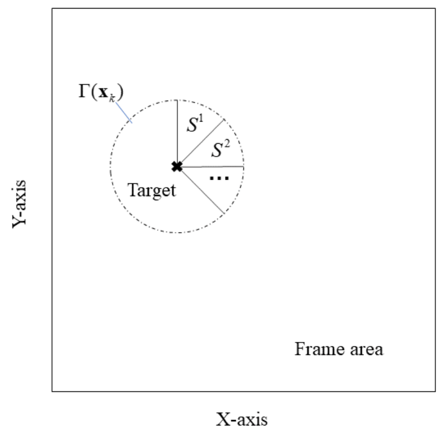

Figure 2. Target state transition area and the corresponding partition adopted in parallel computing dynamic programming-based track-before-detect (PC-DP-TBD). MT, multi-target.

As shown in Figure 2, the PC-DP-TBD method adopts a partition of the state transition area, which is given by:

$$\begin{aligned} \mathbf{S} &= \bigcup_{i=1}^{N_c} S^i \\ s.t. \quad &\{|S^i| = |S^j|; i, j \in [1, N_c]\}, \end{aligned} \tag{11}$$

where each S^i corresponds to a target state transition subset $\Gamma^i(\mathbf{x}_k)$. Each $\Gamma^i(\mathbf{x}_k)$ is used by an individual CPU core to perform DP integration, which means N_c implementations towards N_c target motion orientations. Thereby, target energy can be well integrated in the DP procedures performed by a typical implementation with S^i that matches target motion orientation.

In order to facilitate the implementation of PC-DP-TBD, the partition of $\mathbf{S}$ is even and symmetrical, as defined in (11). Therefore, the obtained target state transition subsets contain an identical number of cells, and the neighboring subsets share common boundary cells, which means a target moving along the boundary of the subset may be detected by two computing cores. Therefore, we adopt a track fusion procedure as the last step of the PC-DP-TBD method.

3.2.2. Implementation Steps Based on Parallel Computing

Figure 3 shows the schematic diagram of the parallel computing dynamic programming algorithm.

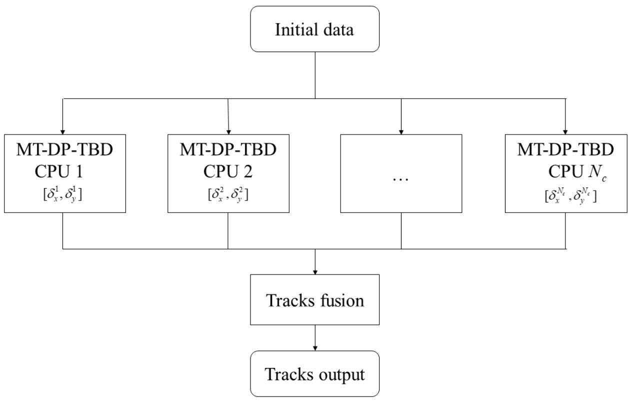
Initial data
MT-DP-TBD CPU 1 $[\delta_x^1,\delta_y^1]$
MT-DP-TBD CPU 2 $[\delta_x^2,\delta_y^2]$
…
MT-DP-TBD CPU N_c $[\delta_x^{N_c},\delta_y^{N_c}]$
Tracks fusion
Tracks output

Figure 3. Schematic diagram of PC-DP-TBD.

By partitioning the transition space $\Gamma(\mathbf{x}_k)$ into N_c subsets as previously proposed, N_c implementations of MT-DP-TBD are performed in N_c computing cores, respectively. Each CPU core results in an integration data map and backtracks the detected target tracks. Then, the estimated trajectories of all targets are obtained through the tracks' fusion procedure. The complete procedures of PC-DP-TBD are given as follows:

Step 1: According to (11), partition the state transition set $\Gamma(\mathbf{x}_k)$ by:

$$\begin{aligned} &\Gamma(\mathbf{x}_k) = \bigcup_{i=1}^{N_c} \Gamma^i(\mathbf{x}_k) \\ &\Gamma^i(\mathbf{x}_k) = \{\mathbf{x}_{k-1} = [x_{k-1}, \dot{x}_{k-1}, y_{k-1}, \dot{y}_{k-1}] \big| x_{k-1} \in [x_k, x_k + \delta_x^i], y_{k-1} \in [y_k, y_k + \delta_y^i]\} \end{aligned} \tag{12}$$

where δ_x^i and δ_y^i are the numbers of cells contained in $\Gamma^i(\mathbf{x}_k)$ towards the X-axis and Y-axis, respectively, which are determined by V^{max}.

Step 2: Implement MT-DP-TBD in N_c computing cores with N_c transition subset $\Gamma^i(\mathbf{x}_k)$, respectively. For $1 \le i \le N_c$:

Step 2.1: DP integration: For $1 \le k \le K$ and all $\mathbf{x}_k \in \mathbb{R}^4$:

$$I^i(\mathbf{x}_k) = z_k(\mathbf{x}_k) + \max_{\mathbf{x}_{k-1} \in \Gamma^i(\mathbf{x}_k)} I^i(\mathbf{x}_{k-1}) \tag{13}$$

This procedure is a recursive process of DP-TBD demonstrated in Section 2.2

Step 2.2: Obtain a candidate target state $\mathbf{x}_K^m$ at the K^{th} scan:

$$\begin{aligned} \mathbf{x}_K^m &= \arg\max_{\mathbf{x}_K \in \mathbb{R}^4} I^i(\mathbf{x}_K) \\ s.t. \quad & I^i(\mathbf{x}_K) > V_{DT} \end{aligned} \tag{14}$$

Step 2.3: Target cancellation: Backtrack the trajectory $\hat{\mathbf{X}}_{1:K}^m$ by $\mathbf{\Phi}_{\mathbf{x}_K^m}$ defined in (10), and detach the measurement information related to $\hat{\mathbf{X}}_{1:K}^m$ from the original measurement data $Z_{1:K}$, then go to Step 2.1. Step 2 is a recursive process and ends when $I^i(\mathbf{x}_K) < V_{DT}$. Then, each computing core gets a track collection $\hat{\mathbf{X}}^i = [(\hat{\mathbf{X}}_{1:K}^1)^i, (\hat{\mathbf{X}}_{1:K}^2)^i, \ldots, (\hat{\mathbf{X}}_{1:K}^m)^i]$.

Step 3: Merge the N_c track collections. If the coincidence of every two tracks exceeds 50%, they will be merged and considered to belong to one target. Then, the final estimation of all tracks is given as $\hat{\mathbf{X}} = [\hat{\mathbf{X}}_{1:K}^1, \hat{\mathbf{X}}_{1:K}^2, \ldots, \hat{\mathbf{X}}_{1:K}^M]$. This method avoids the trajectories' repetition caused by the operation of different transition sets and eliminates the false trajectories generated due to the spread of target energy.

4. Simulations and Analysis

In this section, simulations are performed to verify the tracking performance of the PC-DP-TBD algorithm previously proposed in different scenarios. Comparisons towards MT-DP-TBD are also carried out in some typical scenarios: two parallel targets and two cross targets.

4.1. Interference of Cross Targets

As shown in Figure 4, here, we assume a scenario with two cross targets in a surveillance region divided into a 50×50 grid of cells, 50 cells in the X- and Y-axis, respectively.

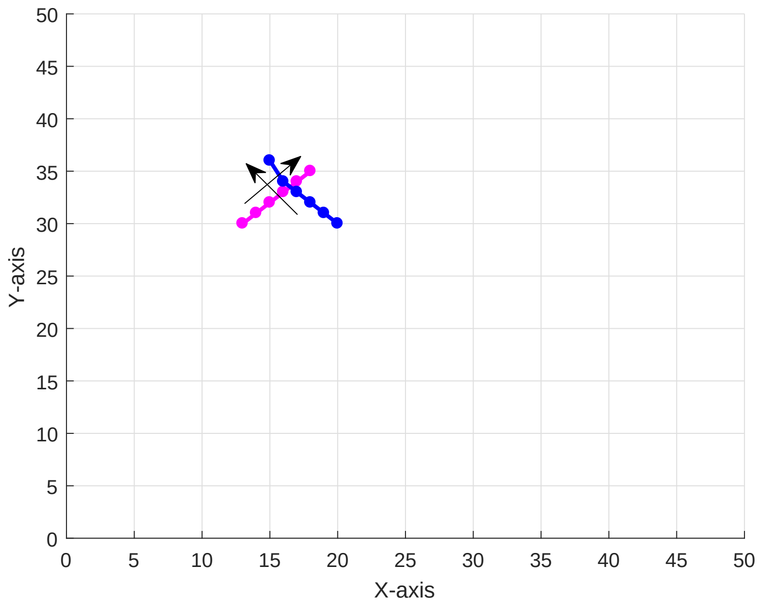

Figure 4. Two target tracks with a target SNR of 10 dB; the solid circles represent the position of targets among $K = 6$ frames of measurement.

According to (7) and the measurement model demonstrated in Section 2.1, the mean value of Gaussian white noise in the simulated surveillance is zero, and we assume the variance $\sigma^2 = 1$ and the signal rate $A^2 = 10$. The signal to noise ratio (SNR) is denoted as:

$$\text{SNR (dB)} = 10 \log \frac{A^2}{\sigma^2} \tag{15}$$

The state updating of the target is determined by (5); the time period T between measurements in the target state transition matrix and the process noise matrix is 1 s, and we use six frames of measurements for one batch processing. The initial states of two cross targets are $\mathbf{x}_1^1$ = [30.8 0.9 13.1 1.0] and $\mathbf{x}_1^2$ = [30.8 1.0 20.1 -1.0], respectively.

The data plane of DP integration result is shown in Figure 5. We can see apparently the energy of two cross targets clustered together, meaning it is hard to distinguish. MT-DP-TBD is developed to deal with this case by performing single-target DP search circularly and wiping out the measurement data of the target once declared to be detected until the integration result does not exceed the detection threshold. However, MT-DP-TBD is computationally prohibitive due to the repeated procedures of DP search.

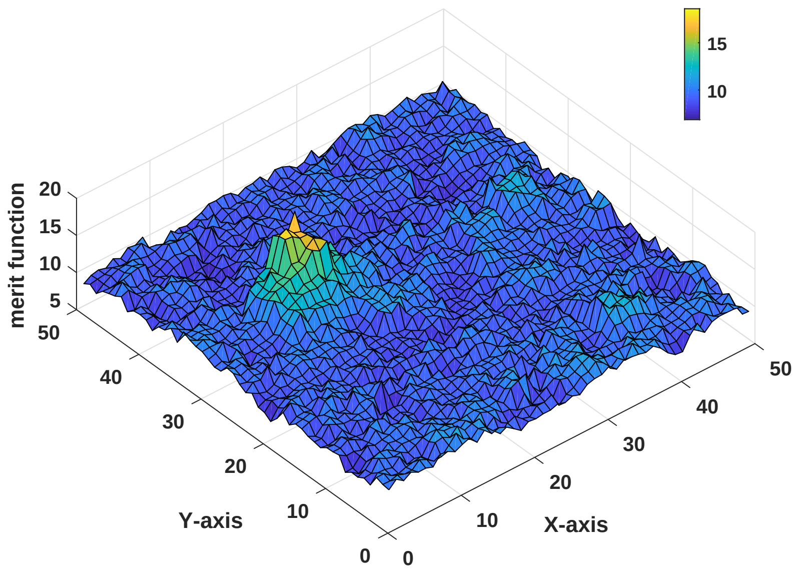

Figure 5. Merit function integration result of two cross targets.

4.2. Simulation of PC-DP-TBD

As shown in Figure 6, we set up a scenario wherein six numbered targets are arbitrarily located. The arrows around each track denote target motions. Target 1 and Target 2 are considered a scenario wherein two adjacent cross targets, Target 4 and Target 5, are close and parallel moving, while Target 3 and Target 6 are mutually isolated and away from each other. The simulations were carried out with the Parallel Computing Toolbox of MATLAB. The hardware platform was a quad-core computer, which means $N_C = 4$ in our implementations.

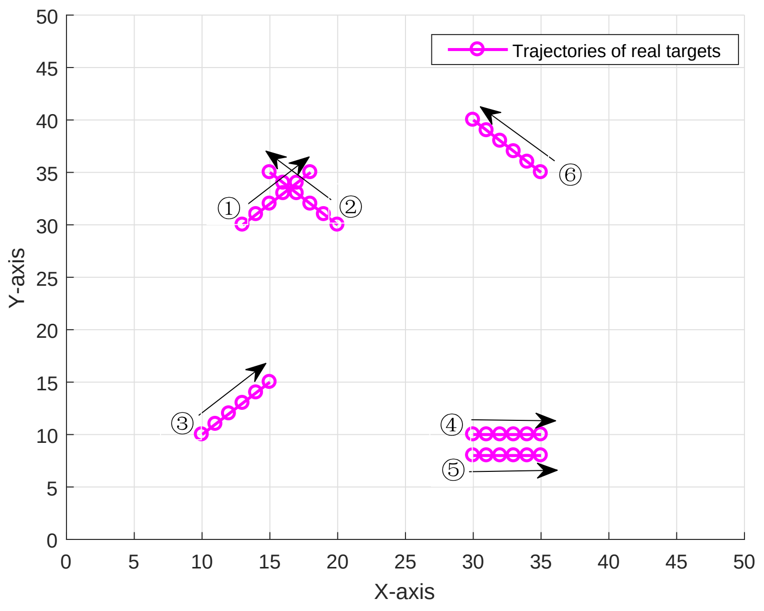

Figure 6. Target trajectories and directions in simulation.

Similar to the previous simulation, the state transition of targets is defined in (5). The time period T for the transition matrix and process noise matrix was 1 s. The targets were always present in one batch process of measurement frames $K = 6$ and $SNR = 8$ dB. The initial target states were $\mathbf{x}_1^1$ = [13.1 1.0 30.8 0.9], $\mathbf{x}_1^2$ = [20.1 -1.0 30.8 1.0], $\mathbf{x}_1^3$ = [10.1 1.0 10.8 0.9], $\mathbf{x}_1^4$ = [30.1 1.0 10.8 0], $\mathbf{x}_1^5$ = [30.1 1.0 8.2 0], $\mathbf{x}_1^6$ = [35.1 -0.9 35.8 1.0], respectively. We partitioned the state space of the potential target-hidden area in the $k-1$ frame into four partitions according to (12): $\delta_x^1 \in [0, M]$, $\delta_y^1 \in [-M, 0]$, $\delta_x^2 \in [-M, 0]$, $\delta_y^2 \in [-M, 0]$, $\delta_x^3 \in [-M, 0]$, $\delta_y^3 \in [0, M]$, $\delta_x^4 \in [0, M]$, $\delta_y^4 \in [0, M]$, and $M = 1$.

Figure 7 shows the merit function integration result of one processing batch of the PC-DP-TBD method, including four data plane sketch of the merit function value obtained from four parallel implementations of DP integration. It is obvious that the two cross targets are well separated by two independent CPUs' processing. The integration value map shown in Figure 7b resulted from the implementation with state space towards $\delta_x^2 \in [-M, 0]$, $\delta_y^2 \in [-M, 0]$ having no apparent merit peak and the others having at least one peak value, respectively. In addition, Figure 7 indicates that there are peak values in some same positions, because four transition state spaces are not isolated from each other absolutely; the common boundary may lead to identical tracks, as mentioned in Section 3.2.1.

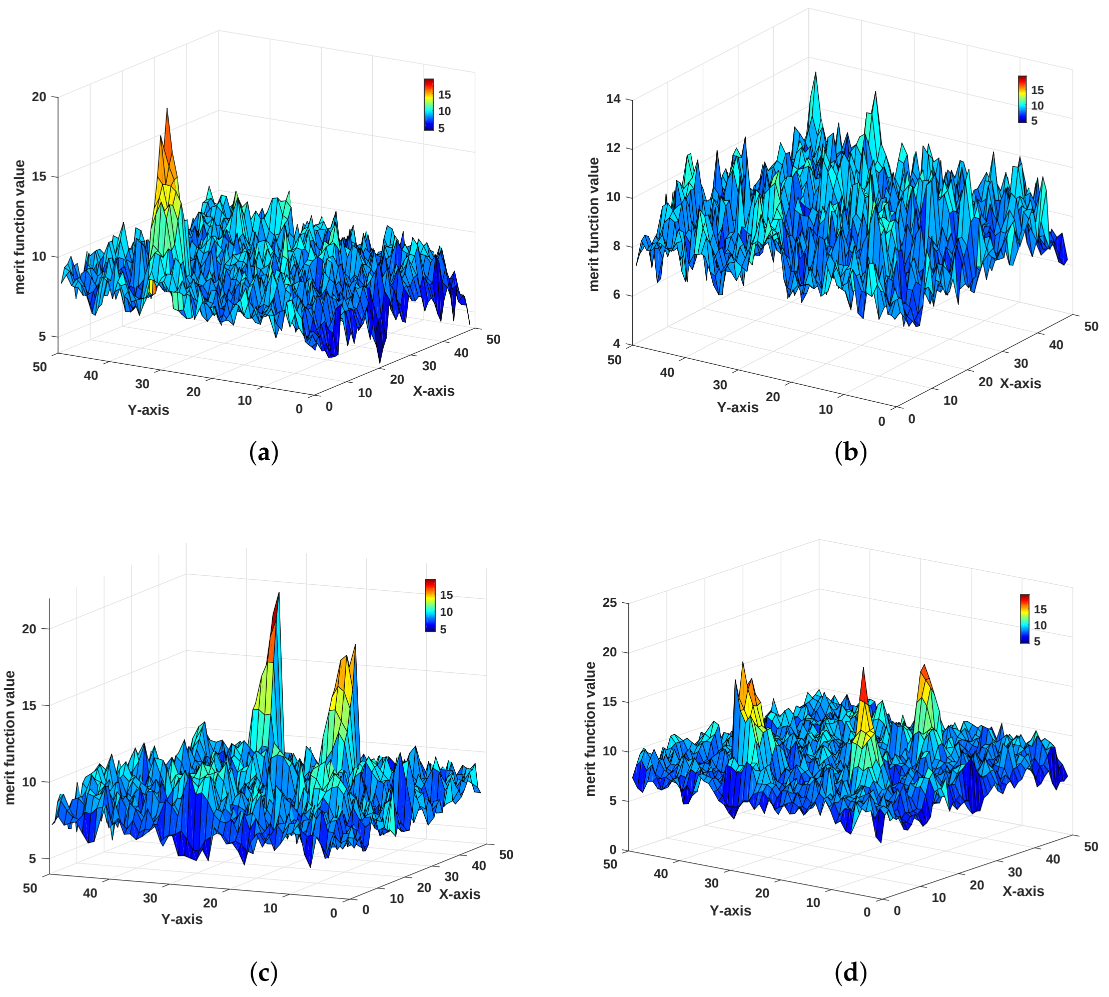

Figure 7. This figure shows the DP integration result of one PC-DP-TBD batch process from four processors respectively. (**a**) δ_x^1, δ_y^1; (**b**) δ_x^2, δ_y^2; (**c**) δ_x^3, δ_y^3; (**d**) δ_x^4, δ_y^4.

After performing track recovering through the repeatedly searching and target cancellation procedures of MT-DP-TBD, each processing of the CPU cores obtained tracks towards the direction decided by the corresponding δ_x and δ_y. To deduce the whole tracks of all targets, as previously demonstrated, track fusion was adopted to merge the potential identical tracks from several tracks. Figure 8 shows the result of the tracks' fusion procedures.

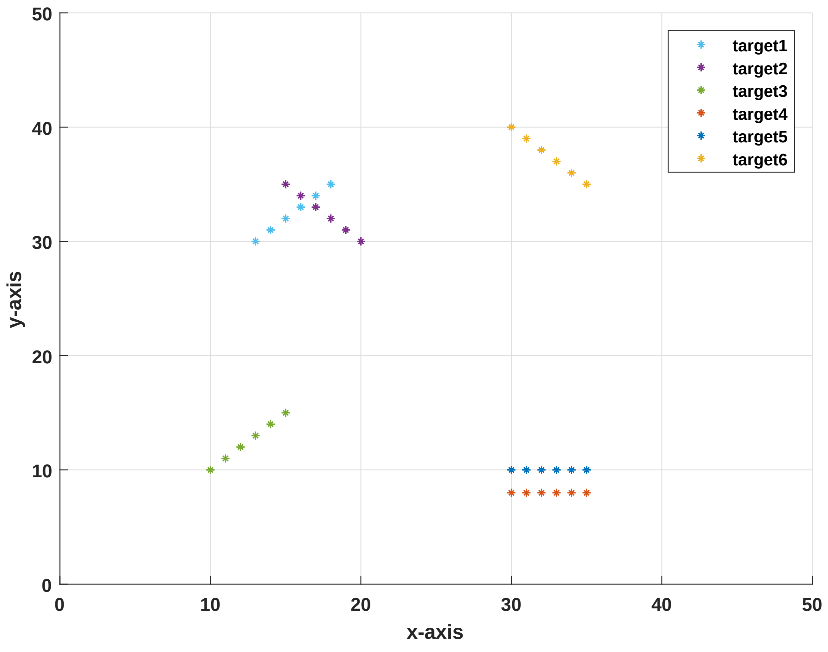

Figure 8. Estimated tracks after track fusion.

4.3. Performance Analysis

The performances of the previously-proposed PC-DP-TBD and MT-DP-TBD [11] was investigated using the following performance measurements: (1). the probability of target detection P_d: the probability that the last stage positions of the estimated state are within two cells of the actual position, as given in [11], derived by the Monte Carlo method. (2); the root mean squared position error ($RMSE$) of the valid tracks: the average position difference between the estimated trajectory and the discretized trajectory of the real target.

PC-DP-TBD is proposed for valid and efficient detection of multi-target scenarios with cross dim targets, so we performed simulations of the PC-DP-TBD method in the scenario depicted in the last subsection to deduce P_d. In contrast, we performed simulations of the MT-DP-TBD method in the same way. The P_d and $RMSE$ of each target in the scenario evaluated over 1000 Monte Carlo runs with an SNR range of 4 dB–15 dB for $P_fa = 10^{-1}$ are plotted in Figures 9 and 10, respectively. The legend shows the particular marks of each target to distinguish them from each other.

It can be seen that the two methods can achieve valid detection when SNR was high, and P_d of the PC-DP-TBD method was larger than the MT-DP-TBD generally. When SNR ranged from 5 dB–9 dB, the RMSE of the target position from PC-DP-TBD simulations was much smaller than MT-DP-TBD, and PC-DP-TBD had a better performance on P_d obviously, especially Target 1 and Target 2. As demonstrated in the preceding section, Target 1 and Target 2 are in proximity and move in intersecting directions. Therefore, the MT-DP-TBD method cannot integrate target energy individually; as for PC-DP-TBD method, it runs DP integration in multiple CPU cores with partitioned target state transition sets towards particular motion orientations, making the merit function of Target 1 and Target 2 integrate along tracks closer to the actual tracks.

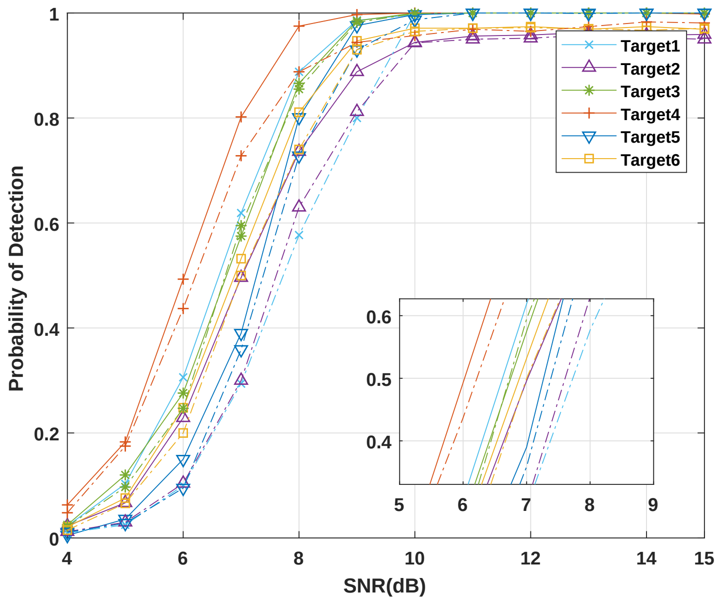

Figure 9. The detection probability P_d of all six targets against SNR from 4 dB–15 dB for $P_f a = 10^{-1}$. The P_d of PC-DP-TBD method is indicated by the solid lines, and the dotted lines indicate the P_d of MT-DP-TBD for comparison.

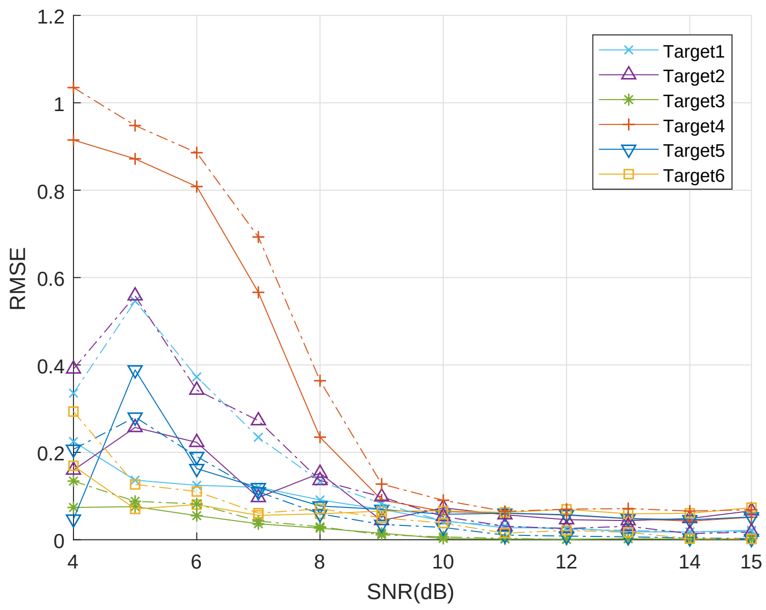

Figure 10. The RMSE of all six targets against SNR from 4 dB–15 dB for $P_f a = 10^{-1}$. The RMSE of PC-DP-TBD method is indicated by the solid lines, and the dotted lines indicate the RMSE of MT-DP-TBD for comparison.

4.4. Computational Expense

Generally, the computational expense of PC-DP-TBD is spent on the procedures embed in the N_c implementations of MT-DP-TBD: the recursive DP integration. Consider M_t targets moving in the surveillance; the computational cost of MT-DP-TBD is about $\mathcal{O}(M_t K |\mathbb{R}^4(\Gamma)|)$ [11]; $\mathbb{R}^4(\Gamma)$ denotes the target state space determined by the rasterized measurements data plane and discretized state transition set $\Gamma(\mathbf{x}_k)$. The number of $\mathbb{R}^4(\Gamma)$ is calculated as:

$$|\mathbb{R}^4(\Gamma)| = N_x \times N_y \times (2V_x^{max} + 1) \times (2V_y^{max} + 1) \tag{16}$$

By utilizing the parallel computing capability of the multi-core processor, PC-DP-TBD performs DP integration and target cancellation towards the N_c subsets of the $\Gamma(\mathbf{x}_k)$ defined in (12). Besides, the implementations of different computing cores may result in a different number of detected targets, as mentioned in Section 3.2.2. Consequently, the computational expense of PC-DP-TBD is about $\mathcal{O}(\boldsymbol{\theta}(i, M_t)K|\mathbb{R}^4(\Gamma^i)|)$. $\boldsymbol{\theta}(i, M_t) \in [0, M_t]$ denotes the target count detected by the i^{th} computing core. The number of $\mathbb{R}^4(\Gamma^i)$ is calculated as:

$$|\mathbb{R}^4(\Gamma^i)| = \frac{|\mathbb{R}^4(\Gamma)|}{N_c} \tag{17}$$

Therefore, the computational expense of PC-DP-TBD compared to MT-DP-TBD is:

$$\frac{\mathcal{O}(\boldsymbol{\theta}(i, M_t)K|\mathbb{R}^4(\Gamma^i)|)}{\mathcal{O}(M_t K|\mathbb{R}^4(\Gamma)|)} = \frac{\boldsymbol{\theta}(i, M_t)}{M_t N_c} \tag{18}$$

It can be seen that the proposed PC-DP-TBD achieves at least N_c-times reduction of the computational time expense. Besides, the computational expense of both MT-DP-TBD and PC-DP-TBD become more conspicuous as the increase of K, V^{max}, M_t, and the surveillance area size, i.e., $N_x \times N_y$.

5. Conclusions

Although the MT-DP-TBD algorithm effectively solves the problem of approximate targets' tracking and detection, the computational expense is still large. In this paper, to detect approximately cross targets and improve the computational efficiency, we proposed the PC-DP-TBD method. The main advantage is that PC-DP-TBD makes full use of the feature that different state transition sets result in different integration values of the merit function. Moreover, with the utilization of parallel computing, PC-DP-TBD achieves an improvement of operation efficiency because the computational expense is shared by several CPU cores.

The simulation results show that the PC-DP-TBD method does have a good performance of multi-target TBD and less operation time burden, especially better than the MT-DP-TBD method in conditions like approximate and cross targets.

Author Contributions: Conceptualization, Q.G. and Z.L.; methodology, Q.G. and Z.L.; software, Z.L.; validation, W.S. and Z.L.; writing, original draft, Z.L.; writing, review are editing, Z.L. and W.F.

Funding: This research was funded by Aeronautical Science Foundation of China (2017ZC15002), Aeronautical Science Foundation of China (2016ZC15013), the Application Technology Research and Development of Heilongjiang Science and Technology Agency (No. GC13A307), and the Fundamental Research Funds for the Central Universities (No. HEUCF1808).

Acknowledgments: The authors would like to thank the Editors and the reviewers for their comments on a draft of this article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Davey, S.J.; Rutten, M.G.; Cheung, B. A comparison of detection performance for several track-before-detect algorithms. *Eurasip J. Adv. Signal Process.* **2007**, 22–27, doi:10.1155/2008/428036. [CrossRef]
2. Larson, R.; Peschon, J. A dynamic programming approach to trajectory estimation. *IEEE Trans. Autom. Control* **1966**, *11*, 537–540, doi:10.1109/TAC.1966.1098348. [CrossRef]

3. Huang, X.; Zhao, Y.; Hao, Y. Dynamic Programming Algorithm for Track-Before-Detect Technology. *Command Inf. Syst. Technol.* **2015**, 53–71, doi:10.15908/j.cnki.cist.2015.03.015. [CrossRef]
4. Tonissen, S.; Evans, R. Peformance of dynamic programming techniques for Track-Before-Detect. *IEEE Trans. Aerosp. Electron. Syst.* **1996**, *32*, 1440–1451, doi:10.1109/7.543865. [CrossRef]
5. Cai, F.; Fan, H.; Fu, Q. Dual-Channel Particle Filter Based Track-Before-Detect for Monopulse Radar. *Math. Probl. Eng.* **2014**, *2014*, 750279, doi:10.1155/2014/750279. [CrossRef]
6. Rutten, M.G.; Ristic, B.; Gordon, N.J. A comparison of particle filters for recursive track-before-detect. In Proceedings of the 2005 7th International Conference on Information Fusion, Philadelphia, PA, USA, 25–28 July 2005; Volume 1, p. 7, doi:10.1109/ICIF.2005.1591851. [CrossRef]
7. Jing, C.; Lin, Z.; Li, J. Detection and tracking of an underwater target using the combination of a particle filter and track-before-detect. In Proceedings of the OCEANS 2016-Shanghai, Shanghai, China, 10–13 April 2016; pp. 1–5, doi:10.1109/OCEANSAP.2016.7485684. [CrossRef]
8. Bi, X.; Du, J.; Zhang, Q.; Wang, W. Improved multi-target radar TBD algorithm. *J. Syst. Eng. Electron.* **2015**, *26*, 1229–1235, doi:10.1109/JSEE.2015.00135. [CrossRef]
9. Carlson, B.D.; Evans, E.D.; Wilson, S.L. Search radar detection and track with the Hough transform. I. system concept. *IEEE Trans. Aerosp. Electron. Syst.* **1994**, *30*, 102–108, doi:10.1109/7.250410. [CrossRef]
10. Buzzi, S.; Lops, M.; Venturino, L.; Ferri, M. Detection of an Unknown Number of Targets via Track-Before-Detect Procedures. In Proceedings of the 2007 IEEE Radar Conference, Boston, MA, USA, 17–20 April 2007; pp. 180–185.
11. Yi, W.; Morelande, M.R.; Kong, L.; Yang, J. An Efficient Multi-Frame Track-Before-Detect Algorithm for Multi-Target Tracking. *IEEE J. Sel. Top. Signal Process.* **2013**, *7*, 421–434, doi:10.1109/JSTSP.2013.2256415. [CrossRef]
12. Grossi, E.; Lops, M.; Venturino, L. A Track-Before-Detect Algorithm With Thresholded Observations and Closely-Spaced Targets. *IEEE Signal Process. Lett.* **2013**, *20*, 1171–1174, doi:10.1109/LSP.2013.2283586. [CrossRef]
13. Bruno, M.G.S.; Moura, J.M.F. Multiframe detector/tracker: optimal performance. *IEEE Trans. Aerosp. Electron. Syst.* **2001**, *37*, 925–945, doi:10.1109/7.953247. [CrossRef]
14. Yan, B. Track-before-detect algorithm based on dynamic programming for multi-extended-targets detection. *IET Signal Process.* **2017**, *11*, 674–686, doi:10.1049/iet spr.2016.0582. [CrossRef]
15. McDonald, M.; Balaji, B. Impact of Measurement Model Mismatch on Nonlinear Track-Before-Detect Performance for Maritime RADAR Surveillance. *IEEE J. Ocean. Eng.* **2011**, *36*, 602–614, doi:10.1109/JOE.2011.2165369. [CrossRef]
16. Jiang, H.; Yi, W.; Cui, G.; Kong, L.; Yang, X. Knowledge-Based Track-Before-Detect Strategies for Fluctuating Targets in *K*-Distributed Clutter. *IEEE Sens. J.* **2016**, *16*, 7124–7132, doi:10.1109/JSEN.2016.2597320. [CrossRef]
17. Moyer, L.R.; Spak, J.; Lamanna, P. A Multi-Dimensional Hough Transform-Based Track-Before-Detect Technique for Detecting Weak Targets in Strong Clutter Backgrounds. *IEEE Trans. Aerosp. Electron. Syst.* **2011**, *47*, 3062–3068, doi:10.1109/TAES.2011.6034689. [CrossRef]
18. Johnston, L.A.; Krishnamurthy, V. Performance analysis of a track before detect dynamic programming algorithm. In Proceedings of the 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing, Istanbul, Turkey, 5–9 June 2000; Proceedings (Cat. No.00CH37100); Volume 1, pp. 49–52, doi:10.1109/ICASSP.2000.861860. [CrossRef]
19. Johnston, L.A.; Krishnamurthy, V. Performance analysis of a dynamic programming track before detect algorithm. *IEEE Trans. Aerosp. Electron. Syst.* **2002**, *38*, 228–242, doi:10.1109/7.993242. [CrossRef]
20. Orton, M.; Fitzgerald, W. A Bayesian approach to tracking multiple targets using sensor arrays and particle filters. *IEEE Trans. Signal Process.* **2002**, *50*, 216–223, doi:10.1109/78.978377. [CrossRef]