Particle filter based track-before-detect method in the range-doppler domain

Jihoon Kwon and Nojun Kwak
MIPAL LAB, GSCST
Seoul National University
Republic of Korea
{jihoonkwon, nojunk}@snu.ac.kr

Eunjung Yang and Kwansung Kim
Radar R&D Center
Agency for Defense Development
Republic of Korea
{ejyang, kimgs}@add.re.kr

***Abstract*—In this paper, we discuss the application of the particle filter for the track-before-detect to detect the target in a low SNR situation. We investigate how to find a target in the range-doppler domain without using adaptive threshold level by CFAR. To do this, we propose the modified particle filter for simple calculation of particle weights. We design the radar simulator that includes full radar signal processing for algorithm verification. In spite of a low SNR situation where it is difficult to detect a target, the proposed algorithm can estimate the optimal position of a target. As the process of the particle filter is repeated, the particles converge to the optimum position considered as the estimated position of a target.**

Keywords*—*Particle filter, Track-Before-Detect, Pulsed-Doppler Radar, Radar tracking.

I. Introduction

A conventional pulsed-Doppler radar uses the detect-before-track (DBT) algorithm that is a popular technique to track targets after detection process [1]. This method generally includes CFAR (Constant False Alarm Rate) algorithm and tracking filter. Tacking filter operates after detection by CFAR. Typically, a radar uses CFAR algorithm having a relatively high threshold level in order to maintain high detection probability and low false alarm rate. However, the problem is that small RCS (Radar Cross Section) targets having low Signal to Noise Ratio (SNR) can be removed by CFAR process having high threshold level. In this situation, it is hard that a radar detects and tracks a target effectively. To overcome this problem, a radar designer should consider SNR improvement by increasing RF output power, antenna gain, and pulse integration time. However, in order to improve RF output power and antenna gain, we should see problems related to hardware implementation, physical characteristics, cost, and efficiency. This design considerations may bring about important limitation of implementation. Likely, increasing pulse integration time is considered as one of the solutions but this approach leads to slow operation of search and tracking function.

On the other hand, the track-before-detect (TBD) has been widely researched as an important technology for detecting and tracking weak targets in a low SNR situation where detection is difficult due to environment with heavy clutter, small RCS targets, and stealth targets [1-4]. Unlike DBT, tracking filter in TBD is used to decide success or failure of detection. Detection is confirmed according to the output of tracking filter. The key idea of TBD is associated with eliminating or reducing a threshold level of the detection process. In TBD process, targets in low SNR are no longer removed by the radar signal process. Instead, we should seriously consider unwanted clutter noise that is not removed by CFAR, because these noises may frequently generate false detection and false track.

The Hough transform, which detects the linear motion of the target in heavy clutter environment [5], can be considered as an efficient detector in the TBD environment. This approach is known as one of approaches for solving the initialization problem of tracking [6]. Also, a dynamic programming technique for the track-before-detect has been studied for a long time and various papers have been reported until recently [7]. However, these two methods are known as a batch processing which stores and processes received data within certain period time. This means that several scan times are required for making more accurate result. These approaches also have weak point in detecting and tracking a maneuvering target. As an alternative, the recursive particle filter for TBD has been studied[1-4][8-9]. Kalman filter is known to linearize nonlinear problems through Gaussian approximation. In a situation with good SNR, Kalman filter can provide efficient and optimal estimates. However, if non-linearity of a model and noise is increased, Gaussian approximation of Kalman filter has limitation in accurately expressing state probability distribution. However, it is possible that the particle filter reduces estimation error because nonlinear probability distribution are described relatively accurately through several particles.

TBD researches using the particle filter have been reported in the various papers and many researches described TBD processing in X-Y-Z domain. However, this means CFAR or threshold processing is still used for eliminating unwanted signals. However, in this paper, we investigate the TBD processing in the range-doppler domain to remove CFAR processing. Also, unlike generating raw-data directly on the range-doppler domain for simulation, this research reflects the whole radar signal processing for generating virtual raw data. It means to design the waveform considering the radar system parameters, reflect the loss due to the channel loss and the fluctuation of the target, and apply the matched filter (Pulse compression) and Doppler filtering. Lastly, we also consider quantizing the particles into the range-doppler domain grid for simple processing. Rounding off the value of the decimal point without expressing the detailed values of the particle state enable to provide more efficient and faster processing.

This paper is organized as follows. The section 2 describes the theory of state dynamics and particle filters. The section 3 describes the radar simulator reflecting a general radar signal processing. The section 4 describes our proposed algorithm and we show the result of experiments. Finally, the section 5 presents our conclusions and discussions.

II. Particle Filter [10][11]

To define the problem of tracking, we consider the form of the state $\{\mathbf{x}_k, k \in \mathrm{N}\}$ of a target given by

$$\mathbf{x}_k = \mathrm{f}_k(\mathbf{x}_{k-1}, \mathbf{v}_{k-1}) \tag{1}$$

where $\mathrm{f}_k : \mathrm{R}^{n_x} \times \mathrm{R}^{n_v} \rightarrow \mathrm{R}^{n_x}$ is a possibly nonlinear function of the state $\mathbf{x}_{k-1}$, $\{\mathbf{v}_{k-1}, k \in \mathrm{N}\}$ is an i.i.d. process noise sequence, n_x,n_v are dimensions of the state and process noise vectors, respectively, and N is the set of natural numbers.

The objective of tracking is to recursively estimate $\mathbf{x}_k$ from measurements

$$\mathbf{z}_k = \mathrm{h}_k(\mathbf{x}_k, \mathbf{n}_{k-1}) \tag{2}$$

where $\mathrm{h}_k : \mathrm{R}^{n_x} \times \mathrm{R}^{n_n} \rightarrow \mathrm{R}^{n_z}$ is a possibly nonlinear function, $\{\mathbf{n}_k, k \in \mathrm{N}\}$ is an i.i.d. measurement noise sequence, n_z,n_n are dimensions of the measurement and measurement noise vectors, respectively. In particular, we seek filtered estimates of $\mathbf{x}_k$ based on the set of all available measurements $\mathbf{z}_{1:k} = \{\mathbf{z}_i, i = 1, 2..., k\}$ up to time k

From Bayesian perspective, the tracking problem is to recursively calculate some degree of belief in the state $\mathbf{x}_k$ at time k, taking different values, given the data $\mathbf{z}_{1:k}$ up to time k. Thus, it is required to construct the pdf $p(\mathbf{x}_k|\mathbf{z}_{1:k})$. The pdf $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ may be obtained, recursively, in two stages: prediction and update.

At time step k, a measurement $\mathbf{z}_k$ becomes available, and this may be used to update the prior (update stage) via Bayes' rule

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})} \tag{3}$$

where the normalizing constant

$$p(\mathbf{z}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})d\mathbf{x}_k \tag{4}$$

depends on the likelihood function $p(\mathbf{z}_k|\mathbf{x}_k)$ defined by the measurement model and the known statistics of $\mathbf{n}_k$. In the update stage, the measurement $\mathbf{z}_k$ is used to modify the prior density to obtain the required posterior density of the current state. The recurrence relations (3) and (4) form the basis for the optimal Bayesian solution. This recursive propagation of the posterior density is only a conceptual solution on that in general, it cannot be determined analytically. The particle filters approximate the optimal Bayesian solution.

The posterior filtered density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ of the particle filter can be approximated as the equation (1) and the weight are defined in the equation (2).

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i) \tag{5}$$

$$w_k^i \propto w_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_k^i)}{q(\mathbf{x}_k^i|\mathbf{x}_k^i, \mathbf{z}_k)} \tag{6}$$

It can be shown that as $N_s \rightarrow \infty$, the approximation (1) approaches the true posterior density $p(\mathbf{x}_k|\mathbf{z}_{1:k})$. The sequential importance sampling (SIS) algorithm consists of recursive propagation of the weights and support points as each is measurement received sequentially.

It is known that a common problem with the SIS particle filter is the degeneracy phenomenon. This is related to the phenomenon that, when the SIS is iterated several times, only certain weights are selected repeatedly. To avoid this, we need a suitable measure of degeneracy of the algorithm that is defined as the effective sample size N_{eff} that is shown in the equation (7). However

$$N_{eff} = \frac{N_s}{1 + \mathrm{Var}(\omega_k^{*i})} \tag{7}$$

where, since we can not know the actual weight ω_k^{*i} accurately, we use the estimate $\hat{N}_{eff}$ of N_{eff} as the following equation (8).

$$\hat{N}_{eff} = \frac{1}{\sum_{N_s}^{i=1}(\omega_k^i)^2} \tag{8}$$

where ω_k^i is the normalized weight obtained using (6). A small N_{eff} indicated severe degeneracy. Clearly, the degeneracy problem is an undesirable effect in particle filters. Therefore, we rely on use of re-sampling to overcome this problem. Resampling works when N_{eff} is below some threshold N_T. Resampling eliminates particles having small weight and evenly splits particles having large weight. The new particle set $\{x_k^{i*}\}_{i=1}^{N_s}$ is generated by resampling of the updated $\{x_k^i\}_{i=1}^{N_s}$. In general, after resampling, the total number of particles N_s is not changed and the new weight set $\{w_k^{i*}\}_{i=1}^{N_s}$ of particles is reset to $1/N_s$ uniform division. The generic form of the particle filter is shown in the algorithm 1.

Algorithm 1 Particle Filter

function PF($\{\mathbf{x}_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_s}, \mathbf{z}_k$)
  for $i \leftarrow 1$ to N_s **do**
    prediction: $\mathbf{x}_k^i = \mathrm{f}_k(\mathbf{x}_{k-1}^i, \mathbf{v}_{k-1}^i)$;
    weight update: $w_k^i = \mathrm{weight}(\mathbf{x}_k^i, \mathbf{z}_k)$;
  end for
  $\{w_k^i\}_{i=1}^{N_s} = \mathrm{normalization}(\{w_k^i\}_{i=1}^{N_s})$;
  $\hat{N}_{eff} = 1/\sum_{N_s}^{i=1}(\omega_k^i)^2$
  if $\hat{N}_{eff} < N_{th}$ **then**
    $\{\mathbf{x}_k^{*i}, w_k^{*i}\}_{i=1}^{N_s} = \mathrm{resampling}(\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s})$;
    $\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s} = \{\mathbf{x}_k^{*i}, w_k^{*i}\}_{i=1}^{N_s}$;
  end if
  return $\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s}$
end function

III. Radar Simulator

A. Signal generation and receiving

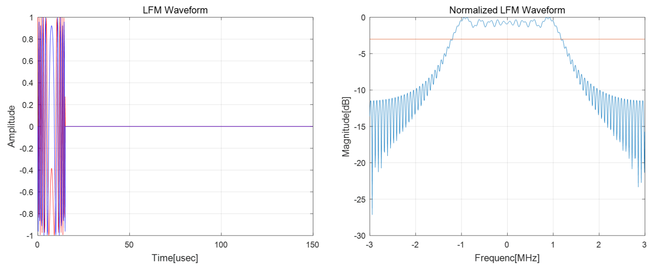


Fig. 1: LFM waveform generated by the transmitter: (Left) the complex signal in the time domain, (Right) LFM spectrum in the frequency domain

The figure 1 shows LFM (Linear frequency modulated) waveform generated. The transmitted signal has attenuation and gain according to the radar equation and it reaches the input of the receiver. In order to reflect target fluctuation, we added Swerling model into the radar equation. Noise signal by the receiver noise figure is added to the reached signal at the receiver input.

B. Pulse Compression and Doppler filtering

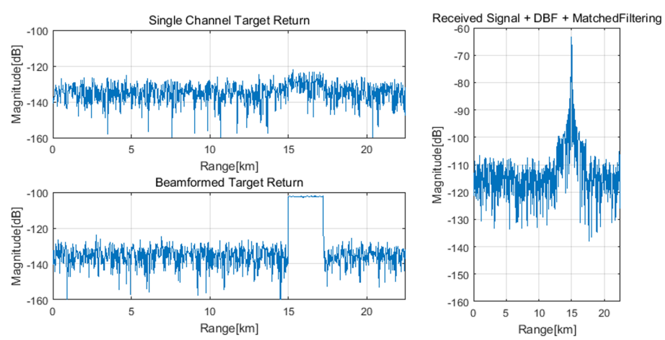
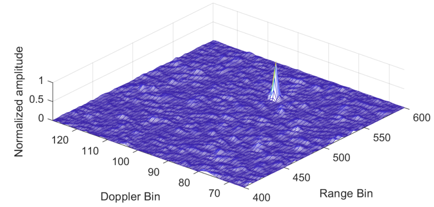
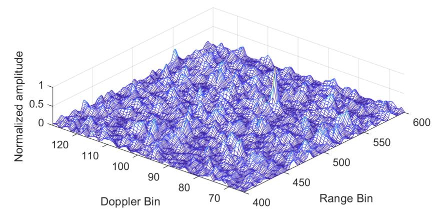


Fig. 2: Pulse compression result: we considered the beamforming of N by N at the receiver. After the beamforming, the beamforming gain was added. The result processed by the pulse compression (matched filtering) after beamforming is shown in the right figure.

Pulse compression is the essential signal processing of a radar to increase the range resolution as well as SNR. This is processed by correlating the received signal and the transmitted signal. The figure 2 shows the output processed by Pulse compression. Doppler filtering improves SNR and helps to distinguish signals on the range-doppler domain. The radar transmits and receives N repeated pulses, and Doppler filtering processes the N sample signals in the each range-bin through Fast Fourier Transform. The figure 3 shows Doppler filtering results of (a) high SNR and (b) low SNR. In the high SNR situation, the desired signal can be easily distinguished from the background noise, and CFAR will detect the desired signal excellently. Conversely, in the low SNR situation, CFAR with high-level threshold may be not effective to detect weak



(a) High SNR situation



(b) Low SNR situation

Fig. 3: The results of Doppler filtering (Normalized amplitude): In the low SNR situation, we can see that it is difficult to clearly find the position of the target

signals. If a threshold value is lowered to solve this problem, many false alarms may occur.

C. CFAR in the low SNR situation

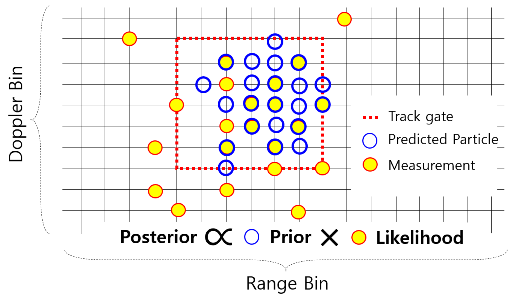
CFAR is the technique for adaptively changing the threshold level according to noise in order to maintain constant false alarm rate. If we use a high CFAR threshold level, a radar cannot detect a small RCS target in a long-range region (low SNR environment). After a target is close enough to a radar, a radar finally detect a target. This means that the maximum detection range of a radar is reduced. On the other hand, in the case of using a lowered threshold level, we can increase a probability to detect the target in a long range region, but we also should regard an increase of a false alarm probability as a significant problem. Therefore, in order to solve this problem, we eliminate the threshold-based CFAR processing and propose the method using the particle filter to detect a small RCS target in the range-doppler domain. This is described in the next section.

IV. Tracking Algorithm in R-D Domain

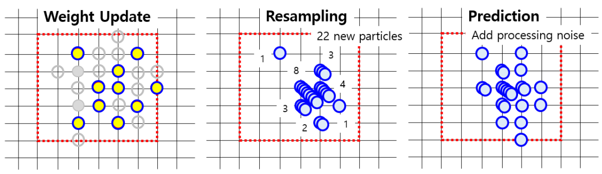
The state vector for the model is defined as

$$\mathbf{x} = [r, v_r]^T \tag{9}$$

where r is the range and $v_r = -\frac{f_d c}{2f_0}$ is the radial direction velocity. The v_r is calculated from $f_d = -\frac{2}{\lambda}\frac{dr}{dt} = -\frac{2v_r}{\lambda}$, where f_d is Doppler frequency of a target, c is the light velocity, and λ is wavelength. The state is quite simple because it has only two components. The equation (10) shows the state transition matrix related to the target dynamics, where Δt is a measurement update time between a previous scan and a current scan.



(a) Approximate the position of the predicated particle to overlap with the measurement value.



(b) Weight update, resampling and new prediction

Fig. 4: The algorithm concept proposed in this paper

Algorithm 2 Modified Particle Filter

function PF($\{\mathbf{x}_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_s}, \mathbf{Z}_k$)
  for $i \leftarrow 1$ to N_s **do**
    $\mathbf{x}_k^i = \mathrm{F}\mathbf{x}_{k-1}^i + \mathbf{v}_k^i$; % Time update
    $r_{ind} = \frac{r}{\Delta r}, f_{ind} = \frac{f_d}{\Delta f}$; % Index calculation
    $\mathbf{x}_{k,ind}^i = \{r_{ind}, f_{ind}\}_k^i$; % Index scale
    $\mathbf{x}_{k,ind}^i \leftarrow round(\mathbf{x}_{k,ind}^i)$; % Quantizing
  end for
  $\mathbf{Z}_k \leftarrow$ Doppler filtering output ;
  for $i \leftarrow 1$ to N_s **do**
    $[r_{ind}, f_{ind}] \leftarrow \mathbf{x}_{k,ind}^i$;
    $w_k^i \leftarrow \mathbf{Z}(r_{ind}, f_{ind})_k$;
  end for
  $\{w_k^i\}_{i=1}^{N_s} = \text{normalization}(\{w_k^i\}_{i=1}^{N_s})$;
  $\hat{N}_{eff} = 1/\sum_{N_s}^{i=1}(\omega_k^i)^2$
  if $\hat{N}_{eff} < N_{th}$ **then**
    $\{\mathbf{x}_{k,ind}^i, w_k^i\}_{i=1}^{N_s} \leftarrow$ Resampling $\{\mathbf{x}_{k,ind}^i, w_k^i\}_{i=1}^{N_s}$;
  end if
  $\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s} \leftarrow \{\mathbf{x}_{k,ind}^i, w_k^i\}_{i=1}^{N_s}$;
  return $\{\mathbf{x}_k^i, w_k^i\}_{i=1}^{N_s}$
end function

$$\mathbf{F} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \tag{10}$$

The equation (1) can be expressed as combination of several particles as shown in the following equation (11). As the particle number increases, it is similar to the actual probability distribution. The $\mathbf{v}_k$ can be understood as the sum of uncorrelated variables as random noise at the current time k. We define $\mathbf{v}_k$ as the equation (12), where X is defined as the uncorrelated noise variables and these are the process noise of x, y, z direction of a target.

$$\{\mathbf{x}_k^i\}_{i=1}^{N_s} = \mathbf{F}\{\mathbf{x}_k^i\}_{i=1}^{N_s} + \{\mathbf{v_k}\}_{i=1}^{N_s} \tag{11}$$

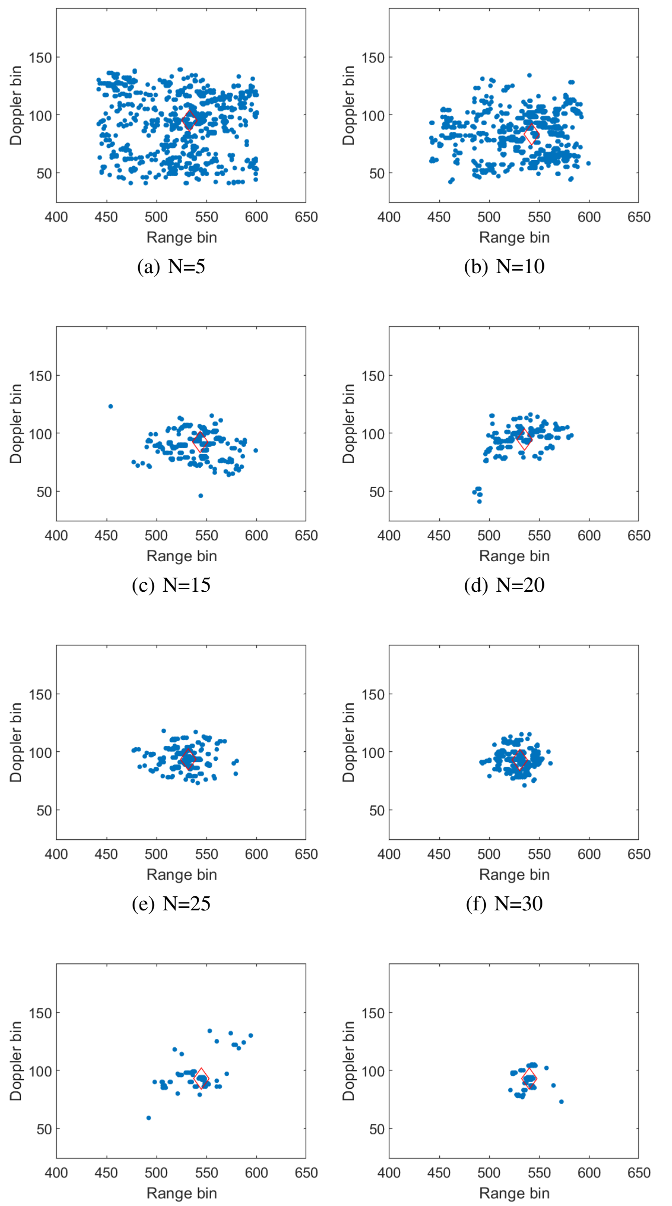
$$\mathbf{v_k} = \sqrt{\sum_{i=1}^{n} \mathrm{Var}(\mathrm{X}_i)} \tag{12}$$

When it comes to the weight update, weights of particles can be updated by using Euclidean or Mahalanobis distance between particles and measurements. These approaches will be valid approach when the number of measurements in the track gate is small. On the other hand, if we have large number of measurements to process, we need to think of a different approach. For example, this problem is related to how we handle complex observation data like the figure 3-(b), which has a lot of measurements or raw-data. In this case, since it is very difficult to designate the wanted signal that is considered to be a target, the nearest neighbor filter is not a good solution. Also, algorithm calculating weights between measurements and particles like the probability data association filter is going to be a complex approach.

In order to process particles in the range-doppler domain, we should first understand the characteristics of the measurement data (likelihood). For example, the likelihood of the measurement shown in the figure 3 looks a continuous appearance, but it is data having actually discontinuous values. If we approximate particles to the closest range-doppler bin of the given measurement data, we can devise the simple weight update method without calculating distance vectors. Our idea is to quantize and approximate the positions of the given particles. The concept is shown in the figure 5.

After the predicted particles are computed and the processing noise are added in the actual position $\mathbf{x}_k^i = \{r, v_r\}_k^i$ (Time update or prediction step), the index number of the range-doppler domain are calculated. And, this index is approximated to have the same index form like the measurement. In this way, we can align the positions of the predicted particles and the current measurement in the same units and grid. The weights of particles are updated at the overlapped parts (Measurement update) and the weights of the remaining parts are set to zero. And the resampling is processed according to this weight information. After the resampling, in the next prediction step, the particles are predicted and approximated to the new positions. The modified algorithm is described in the algorithm 2. The figure 5 shows the processing results of the modified particle filter in the very low SNR. As the number N of processing repetitions increases, the particles converge to the specific position and this position is understood as the optimal value.

Clustering is a useful technique to be able to group related particles together. Density-based spatial clustering of applications with noise (DBSCAN) is known as a data clustering algorithm that does not require the number of clusters. DBSCAN requires two parameters: ϵ (eps) and the minimum number of points (minPts). The figure 6 show the processing result of DBSCAN after the particle filter. Some of the particles are discarded as noise, and the particles that satisfy conditions (eps, minPts) are included in the cluster. The center of these clustered particles is considered as the optimal position. Figure 6-(b) shows the tracking result for a target with constant



(a) N=5 (b) N=10

(c) N=15 (d) N=20

(e) N=25 (f) N=30

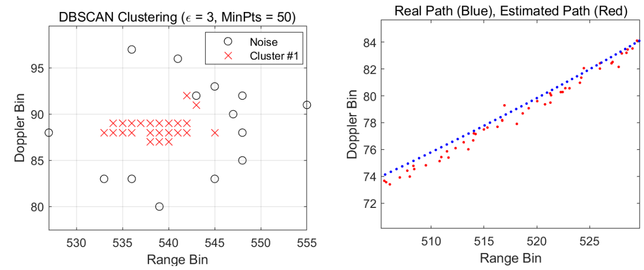
(g) N=35 (h) N=40

Fig. 5: Simulation result of proposed algorithm. As processing is repeated, it can be seen that the particles converge to the specific position.

velocity change.

V. Conclusion

This paper dealt with the application of the particle filter for the track-before-detect to detect a weak target in a low SNR situation. Unlike a conventional approach of extracting plot data through CFAR of a adaptive threshold level and detecting the target in XYZ domain, we investigated how to find the target in the range-doppler domain without CFAR. To do this, we proposed and designed the modified particle filter for simple calculation of particle weights. We also designed the



(a) After DBSCAN (b) Tracking result

Fig. 6: The simulation result processed by DBSCAN after the particle filter (eps=3, minPts=50) and the final tracking result

radar simulator that includes the whole radar signal processing for algorithm verification.

In a low SNR situation where it is difficult to detect a target by CFAR, the proposed algorithm can provide the optimally estimated position of the target. As the processing of the particle filter is repeated, the particles converge to the optimal position to be considered the location of the target. On the other hand, in a high SNR situation, particles converge very quickly to a high measurement signal that is the most reliable information, which is caused by a target.

Acknowledgment

This research was supported by Agency of Defence Development (ADD) that is Korean government institute. Also, the first author was granted a scholarship from Hanwha systems.

References

[1] S.J. Davey, M.G. Rutten, B. Cheung, "A comparison of detection performance for several track-before-detect algorithms." EURASIP Journal on Advances in Signal Processing, vol. 2008, no. 41, 2008.

[2] Rutten, M. G., N. J. Gordon, and S. Maskell, "Recursive track-before-detect with target amplitude fluctuations," IEEE Proceedings on Radar, Sonar and Navigation, vol. 152, no. 5, pp. 345 322, 2005.

[3] Stone, Lawrence D., et al. Bayesian multiple target tracking. Artech House, 2013.

[4] Boers, Yvo, and J. N. Driessen. "Multitarget particle filter track before detect application." IEE Proceedings-Radar, Sonar and Navigation vol. 151, no. 6, pp. 351-357, 2004.

[5] Mukhopadhyay, Priyanka, and Bidyut B. Chaudhuri. "A survey of Hough Transform." Pattern Recognition, vol. 48, no. 3, pp. 993-1010, 2015.

[6] Hu, Zhijian, Henry Leung, and Martin Blanchette. "Statistical performance analysis of track initiation techniques." IEEE Transactions on Signal Processing vol. 45, no. 2, pp. 445-456, 1997.

[7] Johnston, L. A. and V. Krishnamurthy, "Performance analysis of a dynamic programming track before detect algorithm," IEEE Transaction on Aerospace and Electronic Systems, vol. 38, no. 1, pp. 228 242, 2002.

[8] Yin, Shen, and Xiangping Zhu. "Intelligent particle filter and its application to fault detection of nonlinear system." IEEE Transactions on Industrial Electronics, vol. 62, no. 6 pp. 3852-3861, 2015.

[9] Yi, Wei, et al. "A computationally efficient particle filter for multitarget tracking using an independence approximation." IEEE Transactions on Signal Processing, vol. 61, no. 4, pp. 843-856, 2013.

[10] Arulampalam, M. Sanjeev, et al. "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking." IEEE Transactions on signal processing vol.50, no.2, pp. 174-188, 2002 .

[11] Chen, Zhe. "Bayesian filtering: From Kalman filters to particle filters, and beyond." Statistics vol. 182 no. 1, pp. 1-69, 2003.