Article

Adaptive Sample-Size Unscented Particle Filter with Partitioned Sampling for Three-Dimensional High-Maneuvering Target Tracking

Qi Deng $^{1,2}$, Gang Chen $^{1,2,*}$ and Huaxiang Lu $^{1,2,3,4}$

1 High Speed Circuits and Neural Networks Laboratory, Institute of Semiconductors, Chinese Academy of Sciences, Beijing 100083, China; dengqi@semi.ac.cn (Q.D.); luhx@semi.ac.cn (H.L.)
2 University of Chinese Academy of Sciences, Beijing 100089, China
3 Beijing Key Laboratory of Semiconductor Neural Network Intelligent Sensing and Computing Technology, Beijing 100089, China
4 CAS Center for Excellence in Brain Science and Intelligence Technology, Shanghai 200031, China

* Correspondence: chengang08@semi.ac.cn; Tel.: +86-1581-102-7281

Received: 7 August 2019; Accepted: 9 October 2019; Published: 12 October 2019

Abstract: High-maneuvering target tracking is a focused application area in radar positioning and military defense systems, especially in three-dimensional space. However, using a traditional motion model and techniques expanded from general two-dimensional maneuvering target tracking may be inaccurate and impractical in some mission-critical systems. This paper proposes an adaptive sample-size unscented particle filter with partitioned sampling (PS-AUPF), which is used to track a three-dimensional, high-maneuvering target, combined with the CS-jerk model. In PS-AUPF, the partitioned sampling is introduced to improve the resampling and predicting process by decomposing motion space. At the same time, the adaptive sample size strategy is used to adjust the sample size adaptively in the tracking process, according to the initial parameters and the estimated state variance of each time step. Finally, the effectiveness of this method is validated by simulations, in which the sample size of each algorithm is set to the minimum required for the optimal accuracy, thus ensuring the reliability of the tracking results. The results have shown that the proposed PS-AUPF, with higher accuracy and lower computational complexity, performs better than other existing tracking methods in three-dimensional high-maneuvering target tracking scenarios.

Keywords: target tracking; unscented particle filter; partitioned sampling; adaptive sample size

1. Introduction

Maneuvering target tracking is a fundamental and critical task in many practical applications with a wide range of military and civil backgrounds [1,2]. Due to the inconsistency of maneuverability in all directions in three-dimensional space, the target motion cannot be accurately described by the expansion of a general two-dimensional model and method, especially for the high-speed and high-maneuvering targets. Therefore, three-dimensional high-maneuvering target tracking has now become one of the difficulties in this kind of problem [3–6].

For high-maneuvering target tracking, the establishment of a motion model is the primary task. So far, the multiple-model (MM) methods [7–10] are usually adopted because of their fine modeling. However, such methods need to add as many motion models as possible to ensure the completeness of the model set, which will lead to a significant increase in computational complexity and unnecessary competition among similar models.
Besides, the MM methods rely heavily on the priori assumptions, such as the Markov transition matrix. Another method is to assume that the motion model is fixed, and that the maneuver is a disturbance caused by an injection of maneuver noises into our model [11]. Through different assumptions about the statistical characteristics of noises, researchers proposed many different models, among which the Singer model [12], the semi-Markov model [13] and the “current” statistical model [14] are widely used. However, these models only consider the acceleration maneuver of a target. For high-order maneuvers, Kishore proposed the Jerk model [15,16]. This model achieves high accuracy when tracking high-speed and high-maneuvering targets, but it has some steady-state deterministic error in tracking the step acceleration change rate signal. Considering the limitations of the Jerk model, Qiao et al. drew on the idea of the “current” statistical model and proposed a CS-Jerk model that adapts to high-maneuvering targets [17,18]. Compared with the Jerk model, this CS-jerk model has a better tracking performance for high-speed and high-maneuvering targets. Therefore, in our study, the CS-Jerk model is selected as the motion model to solve the three-dimensional high-maneuvering target tracking problem.

In target tracking, the filtering algorithm is the most critical step, which directly affects tracking performance. Owing to the adaptability in strong nonlinear non-Gaussian systems, Particle Filter (PF) is often used [19–21]. However, the PF scheme, of which the complexity depends largely on its sample size, would suffer particle degeneracy. Another new solution is based upon artificial intelligence methods, such as combining a general regression neural network (GRNN) as our tracking algorithm with a multiple-model method [22–24]. This method can deal with non-linear problems well, but its real-time performance is poor [25]. When the target motion changes a lot, the network needs to be retrained in order to adapt to the new motion mode. Therefore, the current mainstream tracking algorithm research is still based on the PF scheme, and on this basis to make improvements. There are usually three ways to improve: Choosing proper proposal distribution, resampling to obtain a high-quality sample set and introducing an effective sample size. Based on the first point, Merwe proposed the unscented particle filter (UPF) [26], which uses the mean and variance of the particle state obtained by an unscented Kalman filter (UKF) to approximate the importance density function. Compared with general PF and UKF, UPF achieves higher accuracy while bringing about higher computational complexity, and it will also be troubled by particle degeneracy when the proposal distribution deviates too far from the real state [27–29].

For further improvement, most studies are focused on improving particle resampling, usually based on traditional resampling mechanisms, such as hierarchical resampling [30], adaptive resampling [31–33], deterministic resampling [34–36], etc. Another way is to increase the diversity of samples by introducing intelligence optimization ideas, such as a genetic algorithm [37,38], firefly algorithm [39], bat algorithm [40] and so on. The above methods could relieve the particle degeneracy to some extent, whereas if they are applied to three-dimensional maneuvering target tracking, the particle distribution would be sparse and hard to cover uniformly, due to an inconsistent maneuvering mode and intensity in different directions, which would intensify particle degeneracy and reduce the diversity of samples. Noting that the target motion could be decomposed into three mutually perpendicular directions in three-dimensional space, the maneuvering form and noises of the target in each direction do not interfere with each other. If these three directions are regarded as orthogonal independent subspaces, the reduction in dimensionality of the maneuver could be considered. The partitioned sampling (PS) method proposed by MacCormick [41] increases the diversity of samples by some decomposition of system dynamics to reduce the sample size required for target tracking when the state space dimension is large [42]. It is mainly used for multi-feature visual target tracking, and is rarely used in other fields. Besides, the premise of using partitioned sampling is that the state space can be decomposed into several independent subspaces, but when there are correlative noises that are difficult to decompose in the system, this method cannot be used directly. Some decoupling methods are needed to remove the correlations in advance [3].
Another improved way that could effectively reduce the redundancy of particles is to adaptively adjust our sample size to avoid computational burden and poor real-time performance caused by fixed sample size.

The adaptive particle filter with KL-sampling proposed by Dieter [43] limits the estimation error through adaptive changes of sample size, but it may bring high computational cost [44,45]. Torma et al. [46] proposed a particle filter based on an adaptive adjustment of likelihood distribution, whereas the weight variance of particles has a great influence on the sample size [47]. In Section 3, we will introduce an adaptive strategy based on the relationship between state variances and sample size, which is derived by introducing theories in Probability & Statistics.

Based on the CS-jerk model, this paper proposes a three-dimensional high-maneuvering target tracking method, by combining the partitioned sampling and the adaptive strategy with UPF. During the tracking process, the target change motion is matched by an adaptive modification of the CS-jerk model, which involves the estimation of the motion state and the correction of the statistical characteristics of maneuver noises, and the UPF is chosen as the basic filtering algorithm to implement this modification. Then, by using adaptive strategy, particles in the UPF are adaptively adjusted to the appropriate amount according to the state variance of target motion for each time step, thereby maintaining an effective sample size and real-time tracking. Furthermore, according to the dynamic decomposability of the target motion, partitioned sampling is used to decompose the motion space so that particles could be independently sampled in each subspace, which reduces the computational complexity and increases the sample diversity, further mitigating the particle degeneracy. Finally, the proposed algorithm is compared to other classic tracking algorithms by experiments. The results verify that the proposed algorithm outperforms other algorithms, in terms of accuracy, and running time, with significant advantages in tracking performance.

2. Model Establishment

This section describes the motion model and measurement models of high-maneuvering target tracking in three-dimensional space. Considering high maneuverability, the CS-jerk model is chosen as the motion model, and the measurement information preprocessing is added in our measurement model to facilitate partitioned sampling. The above two parts will be introduced next separately.

2.1. Target Motion Model

For the motion model, we assume that it is fixed in structure and that maneuver is an interference caused by noise injection into this motion model, so the CS-jerk model can be used to match the target motion by adaptive modification. In this section, we start with a brief introduction of the CS-jerk model.

Suppose the change rate of acceleration of the target is a time-dependent stochastic process with a non-zero mean, which can be expressed as follows [18],

$$\ddot{x}(t) = \ddot{j} + j(t),$$  \hspace{1cm} (1)

where $\ddot{x}(t)$ is the time-dependent rate of change in maneuvering acceleration with non-zero mean, $\ddot{j}$ is the mean of $\ddot{x}(t)$, $j(t)$ is the exponentially-related rate of change of random acceleration with zero mean. Using the Wiener-Kolmogorov whitening algorithm, the random acceleration rate is expressed as result of driving by Gaussian white noises, and the discretized state model is obtained as follows:

$$X_k = FX_{k-1} + U\ddot{j} + w_k,$$  \hspace{1cm} (2)
where \( X_k = [x_k \ \dot{x}_k \ \ddot{x}_k \ \dddot{x}_k]^T \) is the target state vector at time \( k \) which includes the position \( x_k \), velocity \( \dot{x}_k \), acceleration \( \ddot{x}_k \) and the rate of change of acceleration \( \dddot{x}_k \); \( F \) is the state transition matrix and \( U \) is the input control matrix. \( F \) and \( U \) are expressed as follows:

\[
F = \begin{bmatrix}
1 & T & T^2/2 & (2 - 2\omega T + \omega^2 T^2 - 2e^{-\omega T})/(2\omega^3) \\
0 & 1 & T & (e^{-\omega T} - 1 + \omega T)/\omega^2 \\
0 & 0 & 1 & (1 - e^{-\omega T})/\omega \\
0 & 0 & 0 & e^{-\omega T}
\end{bmatrix},
\]

\[
U = \begin{bmatrix}
(2T - \omega T^2 + \omega^2 T^3)/3 - (2 - 2e^{-\omega T})/\omega^2 \\
(2T + \omega T^2)/\omega - (2 - 2e^{-\omega T})/\omega^2 \\
(1 - e^{-\omega T})/\omega \\
1 - e^{-\omega T}
\end{bmatrix},
\]

where \( T \) is the sampling interval, \( \omega \) is the random Jerk maneuver frequency, \( \omega_0 \) is the white noise sequence with zero mean, and its covariance matrix can be expressed as follows:

\[
Q_k = \text{E}[w_k w_k^T] = 2\sigma_w^2 \begin{bmatrix}
q_{11} & q_{12} & q_{13} & q_{14} \\
q_{21} & q_{22} & q_{23} & q_{24} \\
q_{31} & q_{32} & q_{33} & q_{34} \\
q_{41} & q_{42} & q_{43} & q_{44}
\end{bmatrix},
\]

where \( \sigma_w^2 \) is the variance of random jerk. For details of the remaining parameter \( q \), see [14].

During the tracking process, by using a filtering algorithm, the statistical characteristics of maneuver noises are corrected according to the estimated target motion state, and further, the noise covariance matrix \( Q_k \) is updated, so that the CS-jerk model can be adjusted to match the target-changing motion. For brevity, only the basic form of this CS-jerk model is given in this section; details of the CS-jerk model can be found in [17,18].

The above analysis is based on one-dimensional space. In three-dimensional motion space, the motion model needs to be extended. The target state vector will be expanded to \( X_k = [x_k \ \dot{x}_k \ \ddot{x}_k \ \dddot{x}_k \ y_k \ \dot{y}_k \ \ddot{y}_k \ \dddot{y}_k \ z_k \ \dot{z}_k \ \ddot{z}_k \ \dddot{z}_k]^T \), where \( x, y, z \) indicate the state in three directions. The state transition matrix \( F \) and the process noise covariance matrix \( Q_k \) are both expanded into a \( 12 \times 12 \)-dimensional matrix.

2.2. Measurement Model

In practical applications, the measurement information of a maneuvering target is obtained by sensors such as radar, which is mostly based on spherical coordinates, including distance \( r \), azimuth \( b \), and elevation \( e \) [48].

\[
\begin{align*}
    h_r &= r + v_r \\
    h_b &= b + v_b \\
    h_e &= e + v_e
\end{align*}
\]

where \( v_r, v_b, \) and \( v_e \) are measurement noises of distance, azimuth and elevation, respectively; \( h_r, h_b \) and \( h_e \) are noisy measurement information.

Considering the strong nonlinearity in the spherical coordinate system, in order to facilitate partitioned sampling, the Cartesian coordinate system is selected as the reference system to process the measurement information, and the coordinate conversion for data measured by sensors is required. In this paper, we assume that the measurement model in the Cartesian coordinate system is as follows:

\[
Z_k = H_k X_k + v_k,
\]
where $Z_k = [x_k, y_k, z_k]^T$ is the measurement vector at time $k$, $H_k$ is the measurement matrix, $v_k$ is the measurement noise.

Assuming that there is a coordinate transformation between the two coordinate systems $q = h^{-1}$, we can get measurements in the transformed Cartesian coordinate system.

$$Z = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = q(Z_p) = \begin{bmatrix} r \cos b \cos e \\ r \sin b \cos e \\ r \sin e \end{bmatrix}. \quad (8)$$

The actual measurements in the Cartesian coordinate system after conversion can be expressed as follows:

$$\begin{align*}
    h_x &= x + v_x \\
    h_y &= y + v_y \\
    h_z &= z + v_z
\end{align*} \quad (9)$$

where $v_x$, $v_y$, and $v_z$ are the measurement noises after conversion with the variance matrix $R = J(Z_p) \times R_p \times J(Z_p)$, $R_p$ is the original noises variance matrix and $J(Z_p)$ is the Jacobian determinant of measurements. After conversion process, the noise variances of various parties are coupled to each other, so the decoupling process needs to be performed later. The detail of variance decoupling of measurement noises is shown in [3].

3. Unscented Particle Filter with Adaptive Sample Size

In this section, an adaptive sample-size UPF (AUPF) is proposed to reduce the computational cost of invalid and redundant samples. We will start with a brief review of the unscented Kalman filter for establishing the importance sampling density. Then, an adaptive sample-size strategy based on probability will be introduced. Finally, the iteration step of AUPF will be given at the end of this section.

3.1. Unscented Kalman Filter

In the UPF, the important density function is generated by the nonlinear filtering algorithm UKF, so that its overlap with the true posterior probability density is larger. The idea of UT transformation is mainly used to approximate a Gaussian distribution by a fixed number of parameter branches. Assuming that the posterior probability distribution of a system state approximates a Gaussian distribution, UKF directly estimates the probability density function of this system state according to the model of a nonlinear system. At time $k$, the state distribution $N(\hat{X}_{k-1}, \hat{P}_{k-1})$ at time $k - 1$ is approximated by a set of samples, where $\hat{X}_{k-1}$ and $\hat{P}_{k-1}$ are estimates of the mean and variance at time $k - 1$, respectively. These samples are then propagated to obtain the predicted mean and variance of state through the nonlinear state model of the system. Next, the weights of particles are given by the measurement model to update the estimated state. Finally, estimated $\hat{X}_k$ and $\hat{P}_k$ at time $k$ are obtained by a particles weighted summation. The detail of the UKF is shown in [49].

3.2. Adaptive Sample-Size Strategy

For the algorithms based on PF with fixed sample size, the invariant sample size directly affects the computational complexity, making the real-time and accuracy of these algorithms worse [5]. To find out the relationship between accuracy and the number of samples, $\chi$ is defined as the state space, and the target motion model shown in Equation (2) is rewritten as the general form of the nonlinear system state equation [50],

$$X_k = f(X_k|X_{k-1}, w_k), \quad (10)$$
where \( f(\cdot) \in \chi \) is the state transition function, which includes the current and previous state. It is assumed that \( N_k \), the number of samples at time \( k \), is known, the probability density function of samples in state space \( \chi \) is expressed as \( p(X) \), and the expectation of \( X_k \) can be expressed as follows,

\[
\bar{X}_k = \int_{\chi} f(X_k|X_{k-1})p(X_{k-1})dX_{k-1}.
\]  

(11)

The integral in Equation (11), which is hard to solve in reality, is usually expressed by sum of samples. Let \( \{X^{(i)}_{k-1}\}^{N_k}_{i=1} \sim p \) be the sample set randomly sampled from the probability density function \( p \), then the empirical sample mean of state \( X_k \) is as follows,

\[
\hat{X}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} X^{(i)}_k,
\]  

(12)

where \( \hat{X}^{(i)}_k = f(X_k|X^{(i)}_{k-1}) \) is the updated sample propagated by the state transition function.

Assuming the variance of state \( X_k \) is defined as \( P_k \), i.e., the propagated samples come from the state population \( N(\bar{X}_k, P_k) \), then we have a Gaussian density with mean \( \bar{X}_k \) and variance \( P_k \). Once the mean and variance of the samples are obtained, a t-distribution could be obtained according to the definition of t-distribution [51]. Actually, the sample mean \( \hat{X}_k \) is obtained by Equation (12), and the sample variance \( P_k \) can be calculated as follows,

\[
P_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \left[ f(X_k|X^{(i)}_{k-1}) - \hat{X}_k \right] \left[ f(X_k|X^{(i)}_{k-1}) - \hat{X}_k \right]^T.
\]  

(13)

However, due to the presence of process noises and measurement noises, the sample variance directly calculated by (13) is inaccurate and needs to be corrected by the amendment algorithm, such as a Kalman filter [52]. Then, with corrected sample mean and variance, there is

\[
\frac{\hat{X}_k - \bar{X}_k}{\sqrt{P_k/N_k}} \sim t(N_k - 1),
\]  

(14)

where \( t(N_k - 1) \) is the t-distribution with \( N_k - 1 \) degrees of freedom. Based on the confidence interval estimate of the t-distribution [53], given the sample size \( N_k \) and the probability value \( 1 - \alpha \), with the quantile \( t_{\alpha/2}(N_k - 1) \), we can get the following results,

\[
\Pr \left\{ -t_{\alpha/2}(N_k - 1) < \frac{\hat{X}_k - \bar{X}_k}{\sqrt{P_k/N_k}} < t_{\alpha/2}(N_k - 1) \right\} = 1 - \alpha,
\]  

(15)

i.e.,

\[
\Pr \left\{ |\hat{X}_k - \bar{X}_k| < t_{\alpha/2}(N_k - 1) \sqrt{\frac{P_k}{N_k}} \right\} = 1 - \alpha,
\]  

(16)

where \( t_{\alpha/2}(N_k - 1) \) is the bilateral \( \alpha \) quantile of \( t(N_k - 1) \). The confidence interval of \( \bar{X}_k \) with a confidence level of \( 1 - \alpha \) is as follows,

\[
\left[ \bar{X}_k \pm t_{\alpha/2}(N_k - 1) \sqrt{\frac{P_k}{N_k}} \right].
\]  

(17)

For brevity, it is recommended to reorganize the interval shown in (17) by using \( L \) to represent the length of interval, where

\[
L = 2t_{\alpha/2}(N_k - 1) \sqrt{\frac{P_k}{N_k}}.
\]  

(18)
The interval in Equation (17), which is expressed as \((\bar{X}_k - 0.5L, \bar{X}_k + 0.5L)\) now, is a random interval with the probability of \(1 - \alpha\) containing the empirical sample mean \(\bar{X}_k\).

As the degree of freedom increases, t-distribution will infinitely approach the standard normal distribution [53]. In this case, \(t_{\alpha/2}(N_k - 1)\) can be replaced by \(z_{\alpha/2}\), where \(z_{\alpha/2}\) is the bilateral \(\alpha\) quantile of a standard normal distribution, and the length of interval \(L\) shown in (18) can be expressed as

\[
L = 2z_{\alpha/2}\sqrt{\hat{P}_k/N_k}.
\]

(19)

With certain variance \(\hat{P}_k\) and confidence coefficient \(\alpha\), as sample size \(N_k\) increases, the length of confidence interval \(L\) decreases. Once the confidence coefficient, variance and interval length are determined, the lower bound of the sample size \(N_k\) is required to meet the following equation,

\[
N_k = \frac{(L/2z_{\alpha/2})^2}{\hat{P}_k},
\]

(20)

where \(N_k\) is proportional to the state variance \(\hat{P}_k\) at time \(k\) and inversely proportional to \((L/2z_{\alpha/2})^2\), i.e., given the parameters \(L\) and \(\alpha\), and the state variance of each time step, the lower bound of the sample size can be determined.

To verify the effectiveness of adaptive sample-size strategy, it is combined with the PF, and a simple experiment is carried out by the adaptive sample-size particle filter (APF). The detail of general PF is shown in [5].

It is assumed that \(\vec{X}_k = [X^{(i)}_k, w^{(i)}_k]_{i=1}^{N_{k-1}}\) is the sample set after importance sampling and weight update at time \(k\), \(N_{k-1}\) is the sample size estimated at time \(k-1\), \(\vec{P}_k\) is the uncorrected estimated variance of the system state at time \(k\). Next, \(\vec{P}_k\) is corrected by introducing the Kalman gain which is calculated as follows,

\[
K_k = P_{XZ}P_{ZZ}^{-1},
\]

(21)

where \(K_k\) is the Kalman gain at time \(k\), \(P_{XZ}\) is the cross-covariance between the sample measurements and the sample states, \(P_{ZZ}\) is the auto-covariance of the sample measurements. The detail of the calculation of \(P_{XZ}\) and \(P_{ZZ}\) is shown in [26]. Based on \(K_k\), the estimated variance is corrected as follows,

\[
\hat{P}_k = \hat{P}_k + K_kP_{ZZ}K^{-1}_k.
\]

(22)

With the corrected estimated variance \(\hat{P}_k\), the new sample size \(N_k\) is obtained by (20), and further the sample set can be resampled to \(X_k = [X^{(i)}_k, w^{(i)}_k]_{i=1}^{N_k}\). Based upon the above discussion, the structural diagram of the adaptive sample-size strategy is shown in Figure 1.

\[
\begin{align*}
\text{Initialization} & : \quad N_0, \quad \text{Prior} \quad \text{Probability} \quad P_0 \quad \text{Sampling} \\
& \quad \{X^{(i)}_0\}_{i=1}^{N_0} \sim P_0 \\
& \quad \alpha, L \\
\text{Prediction} & : \quad \text{Updated Sample Set} \quad \{\vec{X}^{(0)}_k, \vec{w}^{(0)}_k\}_{i=1}^{N_k} \\
& \quad \text{Filtering Algorithm} \\
& \quad \text{Correcting} \quad \hat{P}_k \quad \text{Corrected Variance} \\
& \quad \text{Sample Size Estimation} \quad \hat{P}_k \\
& \quad \text{Resampling} \quad \text{New Sample Set} \quad \{X^{(0)}_k, w^{(0)}_k\}_{i=1}^{N_k} \\
& \quad k = k + 1
\end{align*}
\]

Figure 1. Structural diagram of the adaptive sample-size strategy.
To compare the performance of APF and general PF, the univariate non-static growth (UNGM) model [5], a common verification model for filtering algorithms, is chosen as the experimental model, where the state transition model is:

\[
x_{k+1} = 0.5x_k + \frac{25x_k}{1 + x_k^2} + 8 \cos(1.2x_k) + w_k,
\]

and the measurement model is:

\[
z_k = \frac{x_k^2}{20} + v_k,
\]

where \(x_k\) is a one-dimensional state variable with no practical physical meaning, which replaces the target state vector \(X_k\) to preliminarily verify the effectiveness of the adaptive sample-size strategy. The variance of process noises \(w_k\) and measurement noises \(v_k\) are set to 20 and 1, respectively, and the simulation duration is 100 s. Both of these two algorithms repeat 100 Monte Carlo simulations, and the algorithm accuracy is defined by the root mean square error (RMSE) [35], which reflects the deviation between the estimated state \(\hat{x}\) and the real value \(x\).

\[
RMSE = \left[ \frac{1}{T} \sum_{k=1}^{T} (x_k - \hat{x}_k)^2 \right]^{1/2}.
\]

The experimental results are shown in Table 1. For further verification, the experimental results of the KL-sampling particle filter (KL-PF), known as a classical adaptive algorithm, is added for comparison. It is found that the estimation accuracy of the PF could be improved by increasing the sample size, but this improvement is limited. When sample size is increased from 500 to 1000, there is only a small improvement in accuracy, and the running time is positively correlated with sample size. For KL-PF and APF, their estimation accuracy and running time are both stable with different initial sample sizes. In terms of accuracy, both of them are close to the PF with 1000 samples. However, in terms of running time, under the same accuracy, APF is about 76% of PF, while KL-PF is higher than PF, and is about 1.5 times of APF. Since KL-PF requires us to calculate the information distance between samples, the computational complexity is increased to some extent. On the contrary, APF adaptively adjusts the sample size according to the state variance at each time step, therefore as initial sample size changes, the estimation accuracy could be maintained, while the running time is reduced. Figure 2 shows changes of particle number caused by adaptive sample size strategy.

### Table 1. Comparison of simulation results in the univariate non-static growth (UNGM) model.

| Algorithm | Initial Sample Size | RMSE    | Running Time (s) |
|-----------|---------------------|---------|------------------|
| PF        | 100                 | 4.0561  | 0.2152           |
|           | 500                 | 3.7749  | 1.0601           |
|           | 1000                | 3.7458  | 2.4824           |
| KL-PF     | 100                 | 3.7436  | 2.7982           |
|           | 500                 | 3.7428  | 2.8630           |
|           | 1000                | 3.7441  | 2.8258           |
| APF       | 100                 | 3.7447  | 1.8856           |
|           | 500                 | 3.7419  | 1.8919           |
|           | 1000                | 3.7423  | 1.9136           |
Figure 2. Changes of sample size with different initial sample size. (a) The initial sample size is 100. (b) The initial sample size is 500. (c) The initial sample size is 1000.

3.3. Adaptive Sample-Size Unscented Particle Filter

The adaptive sample-size UPF aims to obtain a proposal distribution through UKF for importance sampling, and before particle resampling of each iteration, use the current estimated variance of samples to calculate the lower bound of the sample size of the next time step by introducing the adaptive method, so that the sample size can be adaptively adjusted as the estimated variance of the samples changes. In order to prevent particle shortage caused by too-few samples at a certain moment, we will set a lower limit of sample size $N_{\text{min}}$ at the time of initialization. When the estimated sample size is lower than the lower limit, $N_{\text{min}}$ is taken as the required number of particles of the next iteration. Based on the above ideas, the detail of the AUPF is as shown in Appendix A.

4. Adaptive Sample-Size Unscented Particle Filter with Partitioned Sampling

4.1. Partitioned Sampling

In this section, we concentrate on the combination of partitioned sampling and high-maneuvering target tracking algorithm in three-dimensional motion space. In simple terms, partitioned sampling divides the system state space into several independent subspaces, and for each one, to apply the transition function and to perform a weighted resampling operation. The particles are independently sampled in each subspace with their respective best strategies, then we can further predict the optimal sub-states.
The final output is the weighted sum of all predicted sub-states. It is assumed that the state space \( \chi \) can be partitioned as \( \chi = \chi^1 \times \cdots \times \chi^M \), while \( g(\cdot) : \chi \rightarrow \mathbb{R} \) is a weighting function used to weight the sub-states in weighted resampling, and the particles are propagated by the proposal function \( f_k(\cdot) : \chi \rightarrow \chi \) which can be decomposed as follows [41],

\[
f_k(X_{k-1}, n^X_k) = f^M_k \circ f^{M-1}_k \circ \cdots \circ f^2_k \circ f^1_k(X_{k-1}),
\]

where \( X_k \) is the state vector at time \( k \), \( n^X_k \) variable correlated with state \( X \) and time, \( \circ \) is the usual function composition operator. By definition, \( f^i_k \) can propagate particles over subspace \( \chi^i \times \cdots \times \chi^M \) in practice. Exploiting the features of weighted resampling, partitioned sampling achieves the same result by substituting the \( f_k \) propagation by a sequence of applications of the \( f^i_k \) followed by weighted resampling, as shown in Figure 3. In this figure, operations “\( *f^i_k \)” refer to propagations of particles using proposition function \( f^i_k \) as defined above, and operations “\( \sim g^i_k \)” refer to weighted resampling w.r.t. the importance function \( g^i_k \).

![Figure 3. Partitioned sampling condensation diagram.](image)

It is noted that in the Cartesian coordinate system, the sub-states of the target in each direction of motion space are independent of each other, and are not affected by maneuvering motion in other directions, i.e., the motion space is three-dimensionally orthogonal. Based on the idea of dynamic decomposition and synthesis, the overall motion state of the target can be expressed as a superposition of motions in all directions. Figure 4 shows the structural diagram of three-dimensional space partitioned sampling.

![Figure 4. Partitioned sampling structural diagram in three-dimensional motion space.](image)

By using partitioned sampling, the three-dimensional motion space could be decomposed into three independent subspaces. Suppose that particles are sampled in each one-dimensional subspace, \( N_x, N_y \) and \( N_z \) indicate the particles number of each direction respectively, then it is easily find the total number of particles is \( N = N_x + N_y + N_z \), and the number of particle species is \( N_x \times N_y \times N_z \). If particles
are sampled directly in three-dimensional space, the number of particle species is $N_x + N_y + N_z$, with the same sample size as the former, and obviously, the spatial coverage of particles is lower than the former. Next, the algorithm complexity is compared in both cases. According to the analysis in Section 2 of this paper, the state vector dimension is 12, and the measurement vector dimension is 3. It is assumed that the state one-step prediction time in filtering is $T_f$, and the measurement one-step prediction time is $T_h$. Using partitioned sampling, the state space is decomposed into three independent subspaces, the state one-time prediction time is reduced to $T_f/9$, and the measurement one-step prediction time is reduced to $T_h/3$. With the same model, the complexity of the traditional method is $N \times (T_f + T_h)$, whereas the partitioned sampling one is $N \times (T_f + 3T_h)/9$, which has the same order of complexity but less computational time than the former. When $T_f \gg T_h$, the algorithm complexity is dominated by the state one-step prediction time, as the complexity of the state model increases, and the operation time of the algorithm with partitioned sampling can be reduced to a minimum of 12% of the original method.

In fact, partitioned sampling can also be used for general two-dimensional target tracking. Similar to the above analysis for three-dimensional high-maneuvering targets, the state vector of a general two-dimensional maneuvering target is 6-dimensional, and the two-dimensional motion space could be decomposed into two independent subspaces. At this time, the number of particle species is $N_x \times N_y$ rather than $N_x + N_y$, and when $T_f \gg T_h$, as the complexity of state model increases, the operation time of the algorithm with partitioned sampling can be reduced to a minimum of 25% of the original method. Obviously, this method still maintains its advantage for the general two-dimensional target, but with the decrease of state dimension, this advantage will not be as significant as in three-dimensional high-maneuvering target tracking. Therefore, partitioned sampling can be interpreted as a method applicable for any high-dimensional problems with a high dimensional decomposable space model; as the space dimension is higher and the decomposable subspace is smaller, its advantage will be more significant.

4.2. AUPF with Partitioned Sampling

Combining partitioned sampling with the AUPF introduced in Section 3, the adaptive sample-size UPF with partitioned sampling (PS-AUPF) is proposed in this section. In PS-AUPF, the dimension of target maneuver is reduced by decomposing the three-dimensional motion space into three independent one-dimensional subspaces according to the Cartesian coordinate system, so that particles could be sampled in each subspace by using the optimal sampling strategy respectively, and further be propagated by the AUPF, thus reducing the computational complexity and increasing the sample diversity. Then, the sub-states corresponding to these subspaces are predicted according to the measurements, and finally are synthesized into the output results. Based on the above ideas, the summary of PS-AUPF is presented in Appendix B.

5. Results

In this section, we evaluate the proposed PS-AUPF in the high-maneuvering target tracking problem through simulations and compare the performance of PS-AUPF to that of PF, PF with partitioned sampling (PS-PF), UKF, UPF and UPF with partitioned sampling (PS-UPF). In addition, a modern tracking method of artificial intelligence, the multiple-model neural filter (MMNF) [24], is also selected for comparison. All of the algorithms are implemented using MATLAB R2015a on computer with the following specification: CPU, Intel Core i5-3230M 2.6 GHz; Memory, 8 GB. The target motion model and measurement model given in Section 2 of this paper are used in our experiments.

5.1. Initial Setting

To validate the effectiveness of the algorithm, a high-maneuvering target motion process with variable acceleration in three-dimensional state space is simulated for experiments. The maneuvering parameters are set to the maneuvering frequency $f = 1/6$, and the maximum acceleration rate of change $j_{\text{max}} = 5 \, \text{m/s}^3$. The initial position of the given target in the Cartesian coordinate system is assumed as
The initial position of the given target in the Cartesian coordinate system is $[-50,000 \text{ m}, 20,000 \text{ m}, 30,000 \text{ m}]$, and the initial velocity corresponding to each direction is $[300 \text{ m/s}, 150 \text{ m/s}, 200 \text{ m/s}]$, and the initial acceleration and rate of change of acceleration in each direction are $0 \text{ m/s}^2$ and $0 \text{ m/s}^3$, respectively. In summary, the initial state vector is expressed as $X_0 = [-50,000 \text{ m}, 300 \text{ m/s}, 0 \text{ m/s}^2, 0 \text{ m/s}^3, 20,000 \text{ m}, 150 \text{ m/s}, 0 \text{ m/s}^2, 0 \text{ m/s}^3, 30,000 \text{ m}, 200 \text{ m/s}, 0 \text{ m/s}^2, 0 \text{ m/s}^3]$. The sampling interval is $T = 1 \text{ s}$, and the total number of time steps for the duration of tracking is $L = 200$. From 0 to 60 s, the target moves with constant velocity. From 60 to 100 s, it moves with constant acceleration $[-20 \text{ m/s}^2, 30 \text{ m/s}^2, 15 \text{ m/s}^2]$. From 100 to 120 s, it moves with the rate of change of acceleration $[3 \text{ m/s}^3, 1 \text{ m/s}^3, 0 \text{ m/s}^3]$. From 120 to 140 s, its rate of change of acceleration jumps to $[-1 \text{ m/s}^3, -3 \text{ m/s}^3, -2 \text{ m/s}^3]$. From 140 to 160 s, its rate of change of acceleration jumps to $[-2 \text{ m/s}^3, 0 \text{ m/s}^3, -1 \text{ m/s}^3]$. From 160 to 180 s, it holds constant acceleration in this 20 s. Finally, the target keeps moving with constant velocity from 180 s to the end. Figure 5 shows the real target trajectory.

![Figure 5. Real trajectory of a high-maneuvering target.](image)

5.2. Experimental Results and Analysis

5.2.1. Sample Size Selection

For algorithms based on the PF scheme with fixed sample size, expanding the sample size will improve tracking accuracy. However, once the sample size reaches a certain degree, due to the limitations in algorithms and application scenario, the tracking performance will gradually stabilize, i.e., the expansion of sample size will not lead to significant improvement in accuracy as before, and the tracking error will approach the minimum error threshold. In this section, assuming that both the process noises and measurement noises obey the Gaussian distribution with the mean of 0, the standard deviation of measurement noises is set to 100 m for experiments to find the required sample size of PF, PS-PF, UPF and PS-UPF with the optimal accuracy. Furthermore, the simulation of PS-AUPF is also carried out in this section. By changing the initial sample size, the adaptive strategy in Section 3 is further validated.

Figure 6 shows the relationship between the sample size and tracking performance of algorithms, where the tracking performance is defined by the root mean square error (RMSE) of the target position, reflecting the deviation between the estimated position and the real state.

For general PF, when the sample size is less than 1000, increasing particles can effectively reduce the tracking error. After 1000, the trend of improvement slowed down, and after 3000, it approached the minimum error threshold. For PS-PF, the tracking error can be effectively reduced by increasing particles when the sample size is not more than 500, and the minimum error threshold can be approached as long as the number of particles exceeds 1000. Clearly, the accuracy of PS-PF is improved faster.
by increasing samples compared to PF, and only needing fewer samples will it achieve the optimal accuracy. Besides, the minimum error threshold of PS-PF is lower than PF. Both UPF and PS-UPF can obtain more accurate results than PF, because the importance sampling density established by UKF is closer to the real posterior probability distribution, which can reduce the required sample size to some extent. Therefore, expanding sample size could not significantly improve the performance of UPF and PS-UPF. Among these five algorithms, the tracking error of PS-AUPF is the lowest and most stable. Since its sample size is adaptively adjusted according to the estimated variance at each time step, the initial sample size has little effect on the accuracy of PS-AUPF, so the optimal accuracy can be obtained under a different initial sample size. In subsequent experiments, the sample size of PF, PS-PF, UPF and PS-UPF are set to 3000, 1000, 30, and 30, respectively, where all of these algorithms could achieve their optimal accuracy, ensuring the reliability of experimental results. The initial sample size of PS-AUPF is also set to 30 to hold the same initial condition as UPF.

5.2.2. Comparison of Tracking MAE Distribution

Assuming that both the process noises and measurement noises obey the Gaussian distribution with mean of 0, Figure 7 shows the simulation results of target tracking from one Monte Carlo run when the standard deviation of measurement noises is 100 m, and where the tracking mean absolute error (MAE) distribution over time can be found. Since the non-determinism of the CS-jerk model requires constant modification of the maneuvering noises to match real the motion state, whether in the initial uniform motion phase or the later variable acceleration phase, there is an equal magnitude of tracking errors.

Among these algorithms, PF has the largest randomness of errors, while the error distribution of UPF is approximately close to that of MMNF, which shows the effectiveness of using UKF to establish the proposal distribution of PF. PS-PF, both in the error randomness and in the occurrences of error spikes, is lower than PF and UKF, and is close to UPF and MMNF. Relatively speaking, the MAE distributions of PS-UPF and PS-AUPF are more concentrated than that of others, with lower fluctuation range, randomness of error and occurrences of large error spikes, and the tracking accuracy of these two are also close. For further verification, Figure 8 shows the average distribution of tracking MAE in each numerical interval during 100 Monte Carlo runs. From PF to PS-AUPF, the MAE distribution appears to be concentrated to the left, and its randomness tends to decrease. For PS-UPF and PS-AUPF, the medians of MAE are reduced by about one interval compared with general PF, and the overall MAE distributions are more concentrated than UPF and MMNF. Especially for PS-AUPF, its 80% MAEs are less than 200, reflecting the improvement of accuracy.

Figure 6. Relationship between sample size and tracking Root Mean Squared Error (RMSE).

5.2.2. Comparison of Tracking MAE Distribution

Assuming that both the process noises and measurement noises obey the Gaussian distribution with mean of 0, Figure 7 shows the simulation results of target tracking from one Monte Carlo run when the standard deviation of measurement noises is 100 m, and where the tracking mean absolute error (MAE) distribution over time can be found. Since the non-determinism of the CS-jerk model requires constant modification of the maneuvering noises to match real the motion state, whether in the initial uniform motion phase or the later variable acceleration phase, there is an equal magnitude of tracking errors.

Among these algorithms, PF has the largest randomness of errors, while the error distribution of UPF is approximately close to that of MMNF, which shows the effectiveness of using UKF to establish the proposal distribution of PF. PS-PF, both in the error randomness and in the occurrences of error spikes, is lower than PF and UKF, and is close to UPF and MMNF. Relatively speaking, the MAE distributions of PS-UPF and PS-AUPF are more concentrated than that of others, with lower fluctuation range, randomness of error and occurrences of large error spikes, and the tracking accuracy of these two are also close. For further verification, Figure 8 shows the average distribution of tracking MAE in each numerical interval during 100 Monte Carlo runs. From PF to PS-AUPF, the MAE distribution appears to be concentrated to the left, and its randomness tends to decrease. For PS-UPF and PS-AUPF, the medians of MAE are reduced by about one interval compared with general PF, and the overall MAE distributions are more concentrated than UPF and MMNF. Especially for PS-AUPF, its 80% MAEs are less than 200, reflecting the improvement of accuracy.
The simulation results are shown in Tables 2–4, where partitioned sampling improves the performance of a multiple-model method, while its tracking accuracy is significantly higher than that of traditional algorithms. MMNF combines GRNN with UKF is over PF. Obviously, UKF is more susceptible to noise than PF. The tracking accuracy of PS-PF over 100 times.

During the tracking process, so the results are still somewhat random even if simulations are repeated. Nevertheless, the process noises and the measurement noises are randomly generated. Among these algorithms, PF has the largest randomness of errors, while the error distribution of UPF is approximately close to that of MMNF, which shows the effectiveness of using UKF to establish the proposal distribution of PF. PS-PF, both in the error randomness and in the occurrences of error spikes, is lower than PF and UKF, and is close to UPF and MMNF. Relatively speaking, the 80% MAEs of PS-UPF and PS-AUPF are less than 200, reflecting the improvement of accuracy.

Figure 7. The mean absolute error (MAE) distribution of the estimated target position.

Figure 8. Statistical histogram of tracking MAE distribution for 100 Monte Carlo runs. I1 to I8 represents the numerical interval [0, 50), [50, 100), [100, 150), [150, 200), [200, 250), [250, 300), [300, 350), [350, ∞), respectively.

5.2.3. Discussion on Tracking Accuracy

Change the distribution of noises, we perform over 100 Monte Carlo simulations for each algorithm. The simulation results are shown in Tables 2–4, where σ_G is the standard deviation of Gaussian noise, σ_R is the scale parameter of Rayleigh noise, and the tracking performance is defined by the root mean square error (RMSE) of the target position, reflecting the deviation between the estimated position and the real state. It should be noted that those results are close to the optimal accuracy of these algorithms. Nevertheless, the process noises and the measurement noises are randomly generated during the tracking process, so the results are still somewhat random even if simulations are repeated over 100 times.

For the noises with Gaussian distribution, when the standard deviation of measurement noises σ_G is 50 or 100, the RMSE of PF are the largest. However, as σ_G is increased to 150, the RMSE of UKF is over PF. Obviously, UKF is more susceptible to noise than PF. The tracking accuracy of PS-PF and PS-UPF are higher than that of PF and UPF, respectively. This shows that the introduction of partitioned sampling improves the performance of different algorithms to varying degrees, which is related to the limitations of the algorithm and its corresponding system. MMNF combines GRNN with a multiple-model method, while its tracking accuracy is significantly higher than that of traditional
methods such as PF and UKF, and even slightly higher than that of UPF. By introducing partitioned sampling, the tracking accuracy of PS-UPF is significantly improved. For the proposed PS-AUPF, its tracking error is reduced by 6.06%, 6.43% and 6.60%, compared to UPF, when \( \sigma_G \) is 50, 100 and 150, respectively. It performs better than the traditional algorithms and MMNF, and could achieve closer accuracy of PS-UPF with a small extent of improvement. As we can see, the accuracy of the improved algorithm with adaptive sample-size strategy is close to and even higher than that of the general algorithm, but this type of improvement in accuracy is not significant.

**Table 2.** Tracking RMSE of each algorithm with Gaussian noise.

| Overall RMSE (m) | \( \sigma_G = 50 \) | \( \sigma_G = 100 \) | \( \sigma_G = 150 \) |
|------------------|---------------------|---------------------|---------------------|
| PF              | 115.7570            | 182.6600            | 247.5047            |
| PS-PF           | 88.1420             | 156.7254            | 227.3936            |
| UKF             | 89.9564             | 167.2354            | 264.3609            |
| UPF             | 81.8241             | 150.7658            | 215.7149            |
| MMNF            | 80.4735             | 148.3845            | 214.0359            |
| PS-UPF          | 77.8543             | 142.9482            | 202.3238            |
| PS-AUPF         | 75.9206             | 139.1967            | 199.8989            |

Changing the noise distribution as a Uniform distribution or Rayleigh distribution, we can find that the performance change of the algorithm is similar to that with Gaussian noise. With the increase of noise randomness, the accuracy improvement of partitioned sampling and adaptive strategy becomes more and more significant. Overall, regardless of the noise distribution, the tracking accuracy of PS-AUPF is higher than that of traditional algorithms and MMNF, and it can be well applied to non-linear, non-Gaussian systems.

**Table 3.** Tracking RMSE of each algorithm with Uniform noise.

| Overall RMSE (m) | \( \pm 50 \text{ m Range of Noises} \) | \( \pm 100 \text{ m Range of Noises} \) | \( \pm 150 \text{ m Range of Noises} \) |
|------------------|--------------------------------------|--------------------------------------|--------------------------------------|
| PF              | 89.8381                              | 121.3732                             | 156.4902                             |
| PS-PF           | 56.8432                              | 94.8275                              | 135.7030                             |
| UKF             | 59.2855                              | 101.2696                             | 154.4144                             |
| UPF             | 51.7421                              | 89.4955                              | 129.6116                             |
| MMNF            | 50.5371                              | 88.3553                              | 128.9389                             |
| PS-UPF          | 48.4721                              | 86.1945                              | 124.6514                             |
| PS-AUPF         | 46.4825                              | 84.4288                              | 123.3828                             |

**Table 4.** Tracking RMSE of each algorithm with Rayleigh noise.

| Overall RMSE (m) | \( \sigma_R = 50 \) | \( \sigma_R = 100 \) | \( \sigma_R = 150 \) |
|------------------|---------------------|---------------------|---------------------|
| PF              | 143.3577            | 256.4032            | 376.2221            |
| PS-PF           | 125.8578            | 239.6098            | 359.9481            |
| UKF             | 127.4355            | 252.4281            | 374.7265            |
| UPF             | 123.8671            | 238.5487            | 356.6859            |
| MMNF            | 122.0391            | 236.6279            | 354.3415            |
| PS-UPF          | 120.0796            | 230.2823            | 349.8173            |
| PS-AUPF         | 119.1287            | 227.8623            | 347.2862            |

Changing the noise distribution as a Uniform distribution or Rayleigh distribution, we can find that the performance change of the algorithm is similar to that with Gaussian noise. With the increase of noise randomness, the accuracy improvement of partitioned sampling and adaptive strategy becomes more and more significant. Overall, regardless of the noise distribution, the tracking accuracy of PS-AUPF is higher than that of traditional algorithms and MMNF, and it can be well applied to non-linear, non-Gaussian systems.

**5.2.4. Discussion on Running Time**

Table 5 shows the average running time obtained after 100 Monte Carlo experiments of each algorithm. Due to the absence of sampling and propagation of particles, UKF took the shortest running
time. The MMNF also has no particle propagation, but unlike UKF, as a multiple-model method based on GRNN, MMNF needs to train multiple models simultaneously to adapt to different motion patterns of the target. Although those models can be implemented in parallel, it still takes a lot of time in the process of parameter training. For the traditional PF, it needs to generate 3000 particles during operation, which means that it spent the longest time. The sample size of PS-PF is reduced to 1000 by partitioned sampling, and its running time is reduced to 11.35% of PF with higher accuracy. For UPF, its accuracy is already greatly improved by using UKF to establish the importance density. Therefore, only 30 particles needed could allow UPF to achieve lower tracking errors with less running time than PF, even though the extra computation of UKF increased the computational complexity. After introducing partitioned sampling, the running time of PS-UPF is reduced to 24.66% of UPF with the same sample size. In addition, by adaptively adjusting our sample size, with the same initial sample size, the time complexity of PS-AUPF is further reduced by nearly half of PS-UPF. Figure 9 shows changes of the total number of particles in PS-AUPF during a tracking process with $N_{min} = 9$.

### Table 5. Running time of each algorithm.

| Algorithm   | Running Time (s) |
|-------------|------------------|
| PF          | 29.0031          |
| PS-PF       | 3.2933           |
| UKF         | 0.2361           |
| UPF         | 7.4856           |
| MMNF        | 13.1387          |
| PS-UPF      | 1.8459           |
| PS-AUPF     | 1.0191           |

![Figure 9. Changes of sample size in adaptive sample-size unscented particle filter with partitioned sampling (PS-AUPF) during a tracking process.](image)

#### 5.2.5. Summary of Overall Performance

In general, as for traditional PF, the high complexity of the motion model in high-maneuvering target tracking requires a large number of particles to achieve the optimal accuracy, i.e., the accuracy and complexity of the algorithm, both of which must sacrifice one party to satisfy the other. UPF improves the accuracy to reduce the required sample size by establishing the importance sampling density through UKF. However, the extra computation of UKF increases the iteration time of each sample. The MMNF chooses GRNN as the basic tracking algorithm, and uses a set of models to match different motion patterns of the target. Its accuracy is slightly better than UPF, but the real-time performance is poor. The proposed PS-AUPF combines the adaptive sample-size strategy to adaptively adjust sample size according to the state variance at each time step, together with the partitioned sampling to reduce the dimension of the maneuver. Finally, with only about 13.6% of the running
time of UPF, PS-AUPF improved accuracy by about 6% over UPF, bringing about better real-time and effectiveness of tracking performance. In summary, the PS-AUPF proposed in this paper has better overall performance.

6. Conclusions

In this paper, we have proposed a novel adaptive sample-size unscented particle filter with partitioned sampling to track a high-maneuvering target in three-dimensional motion space. Firstly, aiming at the high maneuver of a target, the CS-jerk model was introduced as the motion model to match the target-changing motion by its self-adaptive modification, and to facilitate subsequent processing, the measurement model with coordinate transformation was further introduced to convert the coordinate of measurements obtained by sensors. Next, aiming at the tracking algorithm, a novel adaptive sample-size UPF was proposed, in which the number of samples could be self-adaptively adjusted in real time according to the estimated state variance at each time step, thus effectively satisfying the accuracy and avoiding computational redundancy. Then, the partitioned sampling method was combined with the AUPF presented in Section 3 to reduce the dimension of the maneuver by decomposing the motion space and to take the optimal sample strategy in each subspace, further reducing the computational complexity and increasing the sample diversity. Finally, the simulations in Section 5 verified that the proposed PS-AUPF improves tracking performance in high-maneuvering target tracking scenarios, with at least 6% reduction in terms of RMSE of tracking errors and 86% reduction in terms of running time compared to UPF. In our future work, the target motion model will be further improved to describe the target motion more accurately, which involves attempts to combine multiple-model methods.

Also, based on an improved motion model, the proposed algorithm would be compared with other new artificial intelligence methods. We also consider to apply the PS-AUPF to the multi-target tracking problem and some other computer vision applications.

Author Contributions: G.C. and H.L. provided the direction for the algorithm. Q.D. derived the algorithms, led the experiments and wrote the paper.

Funding: This research was funded in part by the Chinese Academy of Sciences National Defense Innovation Foundation, grant number CXJJ-M215; in part by the Chinese Academy of Sciences Strategic Pilot Science and Technology Project, grant number XDA18040400; and in part by the National Natural Science Foundation of China, grant number 61701473.

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

Algorithm A1. Adaptive Sample-Size UPF

Input: Initial sample size $N_0 \in \mathbb{N}_+$, lower limit of sample size $N_{\text{min}}$, the prior probability $p_0$ of $X$, the confidence interval $L$, the confidence level $\alpha$, the system model given in Section 2.

Output: State estimation results $\hat{X}_k$ at time step $k$.

Initialize: $N = N_0$, for $i = 1, 2, \ldots, N$, draw particle $X_0^{(i)}$ from the prior $p_0$.

For time step $k = 1, 2, \ldots$ do

Importance Sampling:

For $i = 1, 2, \ldots, N$ do

Update particle $X_{k-1}^{(i)}$ with UKF to obtain the preliminary estimations at time $k$: state mean $\bar{X}_k^{(i)}$ and variance $\bar{P}_k^{(i)}$.

Build an approximate Gaussian distribution $\mathcal{N}(\bar{X}_k^{(i)}, \bar{P}_k^{(i)})$.

Use $q(X_k^{(i)} | X_{k-1}^{(i)}, Z_k) = \mathcal{N}(\bar{X}_k^{(i)}, \bar{P}_k^{(i)})$ as the importance density function of particle sampling to draw new particles: $\tilde{X}_k^{(i)} \sim \mathcal{N}(\bar{X}_k^{(i)}, \bar{P}_k^{(i)})$.

End

Weight update:

For $i = 1, 2, \ldots, N$ do

Given observation $Z_k$, calculate the likelihood $p(Z_k | \tilde{X}_k^{(i)})$ of particle $\tilde{X}_k^{(i)}$.

Calculate the particle weights using following equation:

$$w_k^{(i)} = w_{k-1}^{(i)} \frac{p(Z_k | \bar{X}_k^{(i)}) p(\bar{X}_k^{(i)} | \bar{X}_{k-1}^{(i)})}{q(\bar{X}_k^{(i)} | \bar{X}_{k-1}^{(i)}, Z_k)}$$

Normalize the weights:

$$\tilde{w}_k^{(i)} = w_k^{(i)} / \sum_{i=1}^{N} w_k^{(i)}$$

End

State mean and variance prediction:

$$\hat{X}_k = \sum_{i=1}^{N_k} w_k^{(i)} \bar{X}_k^{(i)} \quad \hat{P}_k = \sum_{i=1}^{N_k} w_k^{(i)} (\bar{X}_k^{(i)} - \hat{X}_k)(\bar{X}_k^{(i)} - \hat{X}_k)^T$$

Correct the estimated variance $\hat{P}_k$ using (21) and (22);

Estimate the sample size $N_k$ at time $k$ using (20), When $N_k < N_{\text{min}}$, set $N_k = N_{\text{min}}$;

Particles resampling:

Remove the particles with small weight and multiply the particles with large weight from sample set $\{\tilde{X}_k^{(i)}, w_k^{(i)}\}_{i=1}^{N_k}$.

Reset weight $w_k^{(i)} = 1/N_k$ of each particle to get new sample set $\{X_k^{(i)}, w_k^{(i)}\}_{i=1}^{N_k}$.

End
Appendix B

Algorithm A2. AUPF with Partitioned Sampling

Input: Initial sample size \( N_0 \in N_+, \) lower limit of sample size \( N_{\text{min}} \), the prior probability \( p_0 \) of \( X \), the confidence interval \( L \), the confidence level \( \alpha \), the system model given in Section 2.

Output: State estimation results \( \hat{X}_k \) at time step \( k \).

Initialize: \( N = N_0 \), for \( i = 1, 2, \ldots, N \), draw particle \( X_0^{(i)} \) from the prior \( p_0 \).

Decompose state space into 3 one-dimensional subspace according to orthogonal independence, allot particles into subspace \( k \), respectively for each subspace, where \( N_x + N_y + N_z = N_0 \);

For subspace \( d = x, y, z \) do

for \( i = 1, 2, \ldots, N \), draw particles from the prior \( p_0 \) to obtain particle set \( \{X_0^{d(i)}, w_0^{d(i)}\}_{i=1}^{N} \);

End

For time step \( k = 1, 2, \ldots \) do

For subspace \( d = x, y, z \) do

Importance Sampling: Update particle \( X_0^{d(i)} \) with UKF, build approximate Gaussian distribution to draw new particles: \( \hat{X}_k^{d(i)} \sim N(X_k, \, \hat{\alpha}_k^{d(i)}, \hat{\gamma}_k^{d(i)}) \);

End

Calculate and normalize the particle weights;

Predict the sub-state mean \( \hat{X}_k^{d} \) and correct the sub-state variance \( \hat{P}_k^{d} \);

Estimate the sample size \( \hat{N}_d \) at time \( k \) using (20), when \( N_k < N_{\text{min}} \), set \( N_k = N_{\text{min}} \);

Remove the particles with small weight and multiply the particles with large weight from sample set \( \{X_k^{d(i)}, w_k^{d(i)}\}_{i=1}^{N} \);

Reset \( N_d = \hat{N}_d \);

End

Composite output: \( \hat{X}_k = \hat{X}_k^x \otimes \hat{X}_k^y \otimes \hat{X}_k^z \), where \( \otimes \) is the state composition operator.

End

References

1. Yu, Y. Distributed Multimodel Bernoulli Filters for Maneuvering Target Tracking. IEEE Sens. J. 2018, 18, 5885–5896. [CrossRef]

2. Wang, J.; Zhang, T.; Xu, X.; Li, Y. A Variational Bayesian Based Strong Tracking Interpolatory Cubature Kalman Filter for Maneuvering Target Tracking. IEEE Access 2018, 6, 52544–52560. [CrossRef]

3. Zhang, A.; Lin, H. Parallel Filtering Algorithm in Three Dimensions for IMM Maneuvering Target Tracking. Fire Control Command Control. 2017, 42, 6–9. [CrossRef]

4. Bi, X.; Wang, W.; Gao, J.; Du, J. The improved IMM tracking algorithm for high-speed maneuvering target. In Proceedings of the IET International Radar Conference 2015, Hangzhou, China, 14–16 October 2015; pp. 1–4.

5. Zhu, Z. Application of particle filter algorithm in maneuvering target tracking. In Particle Filter and Its Application, 1st ed.; Science Press: Beijing, China, 2010; pp. 139–150.

6. Wang, G.; Wu, J.; Li, X.; Mo, R.; Zhang, M. A real-time tracking prediction for maneuvering target. In Proceedings of the 2018 Chinese Control and Decision Conference (CCDC), Shenyang, China, 9–11 June 2018; pp. 3562–3566.

7. Ma, C.; San, Y.; Zhu, Y. Multiple Model Truncated Particle Filter for maneuvering target tracking. In Proceedings of the 32nd Chinese Control Conference, Xi’an, China, 26–28 July 2013; pp. 4773–4777.

8. Wang, X.; Xu, M.; Wang, H.; Wu, Y.; Shi, H. Combination of Interacting Multiple Models with the Particle Filter for Three-Dimensional Target Tracking in Underwater Wireless Sensor Networks. Math. Probl. Eng. 2012, 2012. [CrossRef]

9. Visina, R.; Bar-Shalom, Y.; Willett, P. Multiple-Model Estimators for Tracking Sharply Maneuvering Ground Targets. IEEE Trans. Aerosp. Electron. Syst. 2018, 54, 1404–1414. [CrossRef]
10. Han, B.; Huang, H.; Lei, L.; Huang, C.; Zhang, Z. An Improved IMM Algorithm Based on STSRCKF for Maneuvering Target Tracking. *IEEE Access* 2019, 7, 57795–57804. [CrossRef]

11. Li, X.R.; Jilkov, V.P. Survey of maneuvering target tracking. Part I. Dynamic models. *IEEE Trans. Aeroesp. Electron. Syst.* 2003, 39, 1333–1364. [CrossRef]

12. Jia, S.; Zhang, Y.; Wang, G. Highly maneuvering target tracking using multi-parameter fusion Singer model. *J. Syst. Eng. Electron.* 2017, 28, 841–850. [CrossRef]

13. Barbu, V.S.; Karagrigoriou, A.; Makridis, A. On Semi-Markov Modelling and Inference for Multi-state Systems. In Proceedings of the 2016 Second International Symposium on Stochastic Models in Reliability Engineering, Life Science and Operations Management (SMRLO), Beer Sheva, Israel, 15–18 February 2016; pp. 56–60.

14. Bao, S.; Cheng, S.; Xu, C. Strong Tracking Current Statistical Model Based on Mixed Bayesian-Fisher Model. In Proceedings of the 2017 2nd International Conference on Cybernetics, Robotics and Control (CRC), Chengdu, China, 21–23 July 2017; pp. 75–79.

15. Meng, Q.; Hou, B.; Dong, L.; He, Z.; Wang, J. Performance Analysis and Comparison for High Maneuver Target Track Based on Different Jerk Models. *J. Control Sci. Eng.* 2018, 2018. [CrossRef]

16. Mehrotra, K.; Mahapatra, P.R. A jerk model for tracking highly maneuvering targets. *IEEE Trans. Aeroesp. Electron. Syst.* 1997, 33, 1094–1105. [CrossRef]

17. Qiao, X.; Wang, B. A CS-Jerk Model for Tracking Highly Maneuvering Targets. *Syst. Eng. Electron.* 2002, 24, 53–56.

18. Dai, S.W.; Fang, J.; Dai, H.D.; Zhou, J.; Wang, Y.T. A Highly Maneuvering Target Tracking Algorithm Based on Improved CS-Jerk Model. *Electron. Opt. Control.* 2016, 23, 11–15.

19. Wang, F.; Lu, M.; Zhao, Q. Particle filter algorithm. *Chin. J. Comput.* 2014, 37, 1679–1694.

20. Wang, F.; Zhang, J.; Lin, B.; Li, X. Two Stage Particle Filter for Nonlinear Bayesian Estimation. *IEEE Access* 2018, 6, 13803–13809. [CrossRef]

21. Xie, W.; Wang, L.; Bai, B.; Peng, B.; Feng, Z. An Improved Algorithm Based on Particle Filter for 3D UAV Target Tracking. In Proceedings of the ICC 2019—2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–6.

22. Kazimierski, W.; Łubczonek, J. Verification of marine multiple model neural tracking filter for the needs of shore radar stations. In Proceedings of the 2012 13th International Radar Symposium, Warsaw, Poland, 23–25 May 2012; pp. 554–559.

23. Kazimierski, W.; Stateczny, A. Optimization of multiple model neural tracking filter for marine targets. In Proceedings of the 2012 13th International Radar Symposium, Warsaw, Poland, 23–25 May 2012; pp. 543–548.

24. Kazimierski, W.; Zaniewicz, G.; Stateczny, A. Verification of multiple model neural tracking filter with ship’s radar. In Proceedings of the 2012 13th International Radar Symposium, Warsaw, Poland, 23–25 May 2012; pp. 549–553.

25. Sun, C.; Zhu, S.; Liu, J. Fusing Kalman filter with TLD algorithm for target tracking. In Proceedings of the 2015 34th Chinese Control Conference (CCC), Hangzhou, China, 28–30 July 2015; pp. 3736–3741.

26. Merwe, R.V.D.; Doucet, A.; Freitas, N.D.; Wan, E. The Unscented Particle Filter. In Proceedings of the 13th International Conference on Neural Information Processing Systems, Denver, CO, USA, 13 January 2001; pp. 584–590.

27. Gao, Y.; Gao, Y.; Gao, S. Adaptive sample-size unscented particle filter based on likelihood distribution. *J. Chin. Inert. Technol.* 2015, 23, 648–652.

28. Yin, F.; He, B.; Feng, C.; Guo, S.; Zhang, X.; Liu, H.; Mu, X.; Lv, P.; Yan, T. Application of AUV Navigation Based on SVD Unscented Particle Filter. In Proceedings of the 2019 IEEE Underwater Technology (UT), Kaohsiung, Taiwan, 16–19 April 2019; pp. 1–4.

29. Wu, X.; Song, S. Robust information unscented particle filter based on M-estimate. *IET Signal Process.* 2019, 13, 14–20. [CrossRef]

30. Kouritzin, M.; Azen, S. Residual and stratified branching particle filters. *Comput. Stat. Data Anal.* 2017, 111, 145–165. [CrossRef]

31. Teng, F.; Xue, L.; Li, X. Self-adaptive resampling particle filter based on student’s t distribution. *Control Decis.* 2018, 33, 361–365. [CrossRef]
32. Aunsri, N. Improved Dual Tone Multi Frequency identification Using Adaptive Resampling Particle Filter. In Proceedings of the 2018 21st International Symposium on Wireless Personal Multimedia Communications (WPMC), Chiang Rai, Thailand, 25–28 November 2018; pp. 173–176.

33. Zhang, X.; Gu, F. Adaptive Particle Sampling and Resampling in Parallel/Distributed Particle Filters. In Proceedings of the 2019 Spring Simulation Conference (SpringSim), Tucson, AZ, USA, 29 April–2 May 2019; pp. 1–12.

34. Yang, W.; Yuan, S.; Chen, J. Application of deterministic resampling particle filter to fatigue prognosis. *J. Vibroeng.* 2017, 19, 5978–5991.

35. Fei, X.; Shen, Y.; Yan, T. Application of AUV Navigation Based on Deterministic Particle Filter Algorithm. In Proceedings of the OCEANS 2018 MTS/IEEE Charleston, Charleston, SC, USA, 22–25 October 2018; pp. 1–4.

36. Yang, F.; Zheng, L.; Luo, Y. A Novel Particle Filter Based on Hybrid Deterministic and Random Sampling. *IEEE Access* 2018, 6, 67536–67542. [CrossRef]

37. Wang, W.; Tan, Q.K.; Chen, J.; Ren, Z. Particle filter based on improved genetic algorithm resampling. In Proceedings of the 2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC), Nanjing, China, 12–14 August 2016; pp. 346–350.

38. Zhang, S. Mobile Robot Location Algorithm Based on Improved Particle Filtering. In Proceedings of the 2018 IEEE 18th International Conference on Communication Technology (ICCT), Chongqing, China, 8–11 October 2018; pp. 1417–1421.

39. Tian, M.C.; Yu-Ming, B.O.; Pan-Long, W.U.; Chen, Z.M.; Yue, C.; Wang, H.; Automation, S.O. Novel target tracking method based on firefly algorithm optimized particle filter. *Control Decis.* 2017, 32, 1758–1766.

40. Chen, Z.; Wu, P.; Bo, Y.; Tian, M.; Yue, C. Adaptive control bat algorithm intelligent optimization particle for maneuvering target tracking. *Acta Electron. Sin.* 2018, 46, 886–894.

41. Maccormick, J.; Isard, M. Partitioned Sampling, Articulated Objects, and Interface-Quality Hand Tracking. In Proceedings of the European Conference on Computer Vision, Dublin, Ireland, 26 June–1 July 2000; pp. 3–19.

42. Gonzales, C.; Dubuisson, S.; Xuan, S.N.G. Simultaneous Partitioned Sampling for Articulated Object Tracking. In Proceedings of the Advances Concepts for Intelligent Vision Systems-International Conference, Lecce, Italy, 24–27 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 150–161.

43. Fox, D. KLD-Sampling: Adaptive Particle Filters. *Adv. Neural Inf. Process. Syst.* 2002, 14, 713–720.

44. Dihua, S.; Hao, Q.; Min, Z.; Senlin, C.; Liangyi, Y. Adaptive KLD sampling based Monte Carlo localization. In Proceedings of the 2018 Chinese Control and Decision Conference (CCDC), Shenyang, China, 9–11 June 2018; pp. 4154–4159.

45. Li, A.W.; Bastos, G.S. A hybrid self-adaptive particle filter through KLD-sampling and SAMCL. In Proceedings of the 2017 18th International Conference on Advanced Robotics (ICAR), Hong Kong, China, 10–12 July 2017; pp. 106–111.

46. Torma, P.; Szepesvari, C. On using likelihood-adjusted proposals in particle filtering: Local importance sampling. In Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, Zagreb, Croatia, 15–17 September 2005; pp. 58–63.

47. Liang-qun, L.; Chao, H.; Guo-liang, Z. A novel particle filtering algorithm for the noncooperative target tracking in general aviation. In Proceedings of the 2014 International Conference on Audio, Language and Image Processing, Shanghai, China, 7–9 July 2014; pp. 561–564.

48. Li, X.; Jilkov, V. A survey of maneuvering target tracking—Part III: Measurement models. *IEEE Trans. Aerosp. Electron. Syst.* 2001, 39, 1333–1364.

49. Kandepu, R.; Foss, B.; Imsland, L. Applying the unscented Kalman filter for nonlinear state estimation. *J. Process Control.* 2008, 18, 753–768. [CrossRef]

50. Doucet, A.; Johansen, A.M. A tutorial on particle filtering and smoothing: Fifteen years later. In *Proceedings of the Oxford Handbook of Nonlinear Filtering*; Crisan, E.D., Rozovskiy, B., Eds.; Oxford University Press: Oxford, UK, 2011; pp. 656–704.

51. Sheng, Z.; Xie, S.; Pan, C. Parameter Estimation. In *Probability Theory and Mathematical Statistics*, 4th ed.; Higher Education Press: Beijing, China, 2008; pp. 178–217.
52. Chen, S.Y. Kalman Filter for Robot Vision: A Survey. *IEEE Trans. Ind. Electron.* 2012, 59, 4409–4420. [CrossRef]
53. Sheng, Z.; Xie, S.; Pan, C. Sample & Sampling distribution. In *Probability Theory and Mathematical Statistics*, 4th ed.; Higher Education Press: Beijing, China, 2008; pp. 178–217.

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).