applied sciences

Article

Moving-Target Position Estimation Using GPU-Based Particle Filter for IoT Sensing Applications

Seongseop Kim, Jeonghun Cho and Daejin Park *

School of Electronics Engineering, Kyungpook National University, Daegu 41566, Korea; kss92318@gmail.com (S.K.); jcho.knu@gmail.com (J.C.)
* Correspondence: boltanut@knu.ac.kr; Tel.: +82-053-950-5548

Received: 18 September 2017 ; Accepted: 7 November 2017 ; Published: 9 November 2017

Abstract: A particle filter (PF) has been introduced for effective position estimation of moving targets for non-Gaussian and nonlinear systems. The time difference of arrival (TDOA) method using acoustic sensor array has normally been used to for estimation by concealing the location of a moving target, especially underwater. In this paper, we propose a GPU -based acceleration of target position estimation using a PF and propose an efficient system and software architecture. The proposed graphic processing unit (GPU)-based algorithm has more advantages in applying PF signal processing to a target system, which consists of large-scale Internet of Things (IoT)-driven sensors because of the parallelization which is scalable. For the TDOA measurement from the acoustic sensor array, we use the generalized cross correlation phase transform (GCC-PHAT) method to obtain the correlation coefficient of the signal using Fast Fourier Transform (FFT), and we try to accelerate the calculations of GCC-PHAT based TDOA measurements using FFT with GPU compute unified device architecture (CUDA). The proposed approach utilizes a parallelization method in the target position estimation algorithm using GPU-based PF processing. In addition, it could efficiently estimate sudden movement change of the target using GPU-based parallel computing which also can be used for multiple target tracking. It also provides scalability in extending the detection algorithm according to the increase of the number of sensors. Therefore, the proposed architecture can be applied in IoT sensing applications with a large number of sensors. The target estimation algorithm was verified using MATLAB and implemented using GPU CUDA. We implemented the proposed signal processing acceleration system using target GPU to analyze in terms of execution time. The execution time of the algorithm is reduced by 55% from to the CPU standalone operation in target embedded board, NVIDIA Jetson TX1. Also, to apply large-scaled IoT sensing applications, we use NVIDIA Tesla K40c as target GPU. The execution time of the proposed multi-state-space model-based algorithm is similar to the one-state-space model algorithm because of GPU-based parallel computing. Experimental results show that the proposed architecture is a feasible solution in terms of high-performance and area-efficient architecture.

Keywords: GPU-based acceleration; acoustic sensor; time of difference arrival (TDOA); generalized cross correlation phase transform (GCC-PHAT); garticle filter (PF); Internet of Things (IoT)

1. Introduction

In this paper, we propose an accelerated target position tracking system using a GPU-based acoustic sensor and a particle filter (PF) for effective tracking of moving targets. We focus on using parallel processing of GPU to track sudden change of target movement by using multiple system state equations in the existing PF. The proposed parallel processing is scalable for number of sensors and for tracking multiple target. So, proposed architecture can be used in systems such as Internet of Things (IoT) applications. We analyzed the execution time of the algorithm for actual operation on the GPU.

Through this, we searched for the proper design elements of the memory buffer and the algorithm that computes the signal processing.

Position estimation of moving targets using an acoustic signal is a method that can be used not only in air but also in water due to its special environment [1]. The sensor array receives the acoustic signal from the target and estimates the position by measuring three or more time difference of arrival (TDOA) values in the three-dimensional space [2]. Currently, there is a generalized cross correlation phase transform (GCC-PHAT) method for measuring TDOA in the frequency domain not only in the time domain. In particular, GCC-PHAT is a method of obtaining a correlation coefficient by converting a signal into a frequency domain, and has strength in a real-time system because a relatively small amount of calculation is required compared to a method of obtaining a general correlation coefficient. In this paper, we propose an algorithm using parallel Fast Fourier Transform (FFT) in GPU to increase the processing speed of TDOA measurement for each acoustic sensor in three-dimensional space.

In general, various filters such as the kalman filter (KF), extended KF, unscented KF and PF are used to estimate the state of the target [3]. Additionally, there is a problem that it is difficult to estimate the target state due to the non-linearity between the system state and the measured value in estimating the state of the target using TDOA measurement. Therefore, in this paper, we propose an accelerated system that estimates the target position using the PF, which has an advantage in non-linear systems and non-Gaussian systems.

The PF is a sequential monte carlo (SMC) method that estimates the state of a system by observing an error. If the number of particles is sufficient, an optimal estimate can be obtained. However, if the number of particles is not sufficient, the estimation may be problematic. That is, increasing the number of particles to obtain an optimal estimate means that the amount of computation in the system increases, which inevitably affects the operation speed of the system.

Therefore, in this paper, we propose accelerating the processing speed by simultaneously processing the state update process and weight calculation process of each particle using GPU. Additionally, we propose the PF system for multiple state equation to accurately track even when the sudden movement of a target deviates from the system state equation used in the process in the Markov chain based PF.

We described the target system with multiple state-space models, which are processed by the parallel processing algorithm on GPU. In the experiment, we show an implemented result using the proposed algorithm on the GPU and analyzed the execution time of the algorithm required for the target position estimation, including the partial execution time of the proposed algorithm.

2. Related Works

PF using TDOA measurements have been widely studied, but the processing of noise in the algorithm and noise environment have been emphasized rather than high speed processing [4,5]. Also, a target estimation method using TDOA underwater was studied [6]. In this paper, we focus on providing algorithm implementation and verification by approaching GPU from acceleration using parallel processing and estimating the sudden movement of the target using multiple system state equations.

As a method for measuring TDOA, various methods using the GCC algorithm have been introduced [7,8]. They have relative advantage for real-time applications in which performance is more important. The significant approach based steered response power-phase transform (SRP-PHAT) introduces an effective approach for the robust signal processing in sound localization [9], and GPU-based acceleration is also proposed in TDOA measurement using SRP-PHAT [10]. Our paper is based on initial approach [11], which using the GCC-PHAT-based TDOA measurement, specially presenting our experience in implementing GPU-based acceleration approach to guarantee real-time performance. In addition, the GPU-based acceleration for TDOA measurement value in specific number of sensors [12] has been presented case study.

Various studies on the GPU-based PF [13,14] and parallelization of multi-target tracking have been presented. The scheduling algorithm and processor selection in paralleling PF processing is introduced [15]. A significant approach [16] also introduce the parallelization approach of the memetic algorithm and PF for tracking multi objects. Compare to these approaches, we additionally consider the uncertain model of target system state, in case of suddenly abnormal target movement. So, we adopted multiple state-space models which are selectively applied to the PF processing, to determine the good enough estimation result. This requires the more computation resource or increase the calculation time, so that we had to accelerate the computation in PF processing by allocating them into independent calculation unit block in GPU.

There are many studies of SMC methods in using uncertain state-space models [17]. Some approaches, which adopt multiple state-space models to overcome weakness in PF processing for the uncertain model, are introduced [18,19]. These studies consider the PF processing using multiple state-space models in single system. Our approach is slightly comparable because we try to allocate the individual kernels for PF processing into multiple processing unit in GPU. This enables to apply the specific state-space model to the PF processing kernel independently, so that we could get scalability using the proposed architecture in case of tracking large-scaled moving target, which is more important in IoT-driven applications. Some target localization studies for IoT-driven applications using radio-frequency identification (RFID) tags including PF processing have already been studied [20–22]. Therefore, this study aims at implementation of scalable GPU-based algorithm which can be used in IoT sensing applications.

3. Proposed Architecture

The proposed overall architecture of this paper is as follows. This paper focuses on accelerating the position estimation of a sound source based on a PF using an acoustic signal sensor array in a GPU. The TDOA measurement is required for the sound source localization based on the proposed GPU, and the location of the current target can be determined using three TDOA measurements in the three-dimensional space. In order to obtain three TDOA measurements, four acoustic sensors are required including one reference acoustic sensor. In this study, we used GCC-PHAT to obtain TDOA measurements. To obtain TDOA measurements at high speed, the GCC-PHAT process of the acoustic signal input by multiple sensors is performed in parallel using GPU.

The proposed method uses a PF to estimate the current state of a target using TDOA measurements, and we accelerate it. The PF process used in the study is as follows. Using the computed TDOA measurements and the state equations of the system, we predict the current state from the previous state of the system and obtain the observed value from the state of the updated system. Then, the weight of each particle is obtained by comparing with the actual measurement. In this process, we propose accelerating the state update and weight calculation process of the particle mentioned above by parallel operation of each particle in the kernel using GPU. Also, PF operation is customized using multiple system state equations based on GPU for tracking sudden movements of moving targets which also can be used for tracking multiple target. The proposed architecture is shown in Figure 1. Moreover, this proposed architecture can be used in IoT applications as shown in Figure 2, that is consisted with many sensors. Due to the amount of signal data from many sensors, it is reasonable using GPU in terms of high performance computing to accelerate signal processing for detecting target more accurately by increasing the number of sensors.

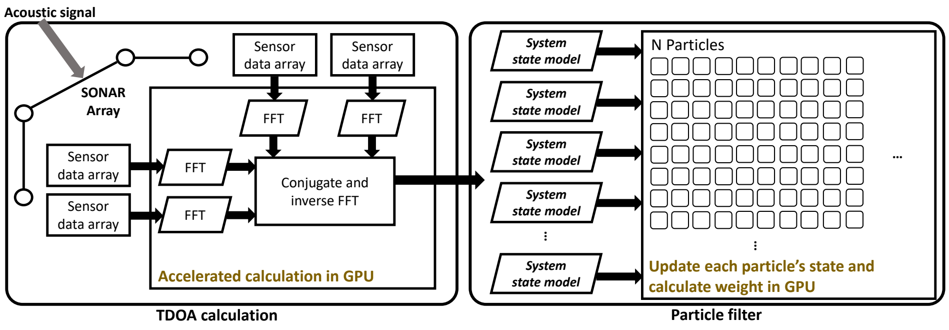

Figure 1. Proposed GPU based accelerated algorithm.

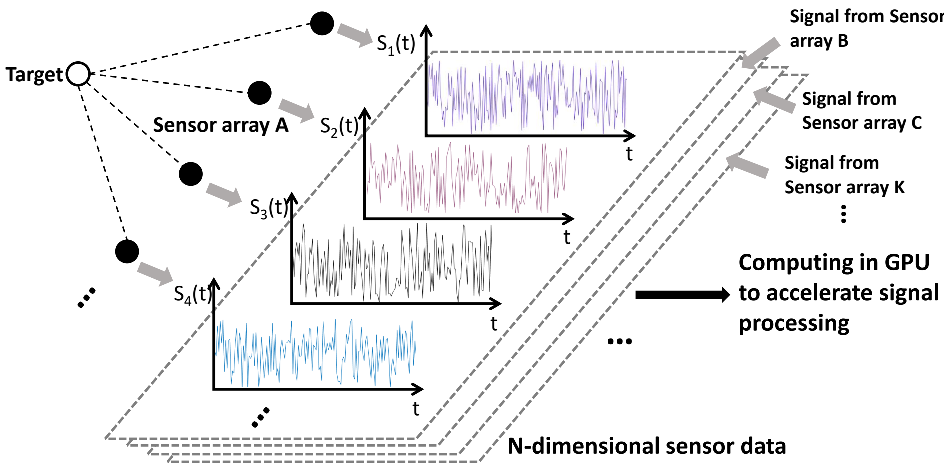
Target
Sensor array A
S1(t)
S2(t)
S3(t)
S4(t)
t
Signal from Sensor array B
Signal from Sensor array C
Signal from Sensor array K
Computing in GPU to accelerate signal processing
N-dimensional sensor data

Figure 2. Accelerated signal processing for large amounts of data using GPU.

3.1. Particle Filter (PF) with Time Difference of Arrival (TDOA) Measurement

To estimate the movement of the acoustic signal source, we use the TDOA measurement obtained from the acoustic sensors. Four acoustic sensors including one reference sensor are used to estimate the position of the target in a three-dimensional space. The proposed PF is based target position estimation using TDOA measurements obtained in this structure. The PF is a method of probabilistically estimating the state of a target using N particles, also called the Monte Carlo method. Unlike the extended KF and unscented KF, the PF has strength in non-Gaussian and non-linear systems. The proposed algorithm is scalable to the number of sensors, and it is applicable even if the number of sensors is increased as shown in Figure 3. The proposed architecture uses four sensors which is minimum number of sensors required to detect a target in three-dimensional space or can use more than minimum number of sensors. The PF processing can operated on the GPU using each sensor node set, and the proposed architecture can detect multiple targets in parallel using acoustic signal.

$$s_k = As_{k-1} + p_k \tag{1}$$

$$z_k = h(s_k) + v_k \tag{2}$$

$$\begin{aligned} r_a(s_k) &= \mid u_k - s_a \mid, \\ r_{ab}(s_k) &= cT_{ab}(s_k) = r_a(s_k) - r_b(s_k) \end{aligned} \tag{3}$$

$$T_{ab}(s_k) = \frac{1}{c}(r_a(s_k) - r_b(s_k)) \tag{4}$$

$$h_{ab}(s_k) = \frac{1}{c} r_{ab}(s_k) \tag{5}$$

$$H(s_k) = [h_{ab}(s_k)\ h_{ac}(s_k)\ ...\ h_{aX}(s_k)]^T \tag{6}$$

First, the state of the system used at PF processing is defined as follows. The state of the moving target is defined as a uniform speed motion. In this case, the system state equation for estimating the current state s_k in a previous state s_{k-1} is expressed by Equation (1), where s_k denotes a position vector of the three-dimensional space at time k. A denotes a state transition equation matrix, and p_k denotes process noise. The observation model, which represents the relationship between the state of the system and the measured value, is defined by Equation (2), where z_k denotes measured value and v_k denotes measurement noise.

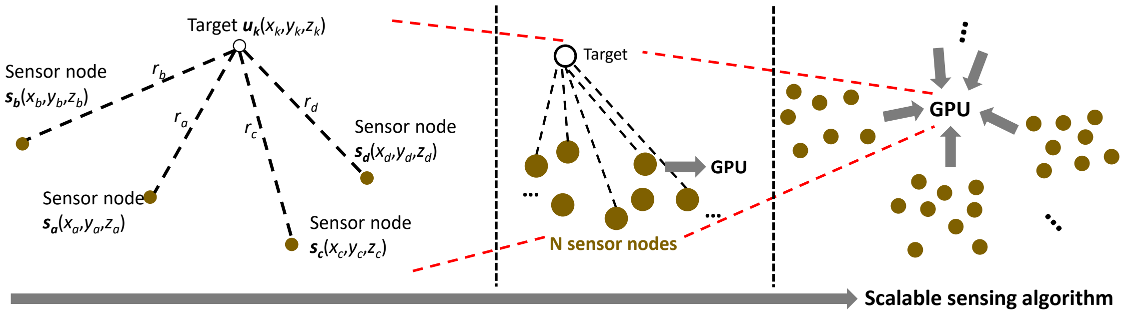
Target $u_k(x_k,y_k,z_k)$
Sensor node $s_b(x_b,y_b,z_b)$
r_b
r_a
r_c
r_d
Sensor node $s_d(x_d,y_d,z_d)$
Sensor node $s_a(x_a,y_a,z_a)$
Sensor node $s_c(x_c,y_c,z_c)$
Target
GPU
...
N sensor nodes
GPU
Scalable sensing algorithm

Figure 3. Scalable target position estimation using GPU.

In this study, we propose accelerating the position estimation of the target using TDOA measurements, and the measured value z_k is a TDOA measurement obtained from the signal input to each acoustic sensor. In Equation (2), the relation between the system state and the TDOA measurements is represented by h, and it can be obtained from the following procedure. The distance between the target and the sensor is defined by $r_a(s_k)$ in Equation (3). Also the reference sensor for obtaining the TDOA measurement is defined as s_a, where u_k is the position vector of the target and s_a is the position vector of the ath sensor. As shown in second line of Equation (3) , r_{ab} is displacement value between distance to s_a and s_b from target, which can be represented with the velocity of source wave 'c' multiplied by the TDOA value between sensor s_a and s_b.

Figure 3's left one shows the reference sensor and the other sensor. TDOA measurements used for the PF are obtained between reference sensor s_a and the other sensor. Equation (4) is used to obtain the TDOA measurement value by using the distance from each reference sensor s_a and sensor s_b to the target. The velocity c is the speed of the sound wave in the water, which is 1500 m/s.

Therefore, the target location estimation by applying three TDOA measurement values is described with the following equation; Equation (5) defines a relation h_{ab} between the system state and the TDOA measurement. The state from three or more TDOA measurement values are represented in Equation (6) by a matrix H, where index 'a' represents reference sensor and index 'X' represents relative sensors.

The operation of the PF for estimating the position of the moving target using the relation between the system and the measurement is shown in Algorithm 1. The overall algorithm of the PF using the TDOA measurement is as follows. First, the state of the system for each particle is updated. Then, each particle predicts the observed value using the relational expression between the system state and the measurement values from each updated state. Additionally, the weight of each particle is calculated according to the probability distribution of the actual observed TDOA measurement value and the predicted measurement value. After that, the effective sample size (ESS) is calculated, then it is compared with threshold N_{th} [23,24]. The N_{th} is defined as a predefined threshold for resampling. In our implementation, the threshold N_{th} is defined as 0.8 times of number of particles. Therefore, if the ESS is below a predetermined number of samples, which is denoted as threshold N_{th}, the weight of each particle is resampled. The resampling process removes low weighted particles and selects particles with high weight to reduce errors due to probability distributions. The PF estimates the position of the moving target more accurately using the resampling process.

Algorithm 1 PF

```
1: Set initial state $s_0$
2: Generate particles $s_{0(i)}$ $(i = 1, 2, ..., N)$
3: Set initial weights $W_{0(i)} = 1/N$ $(i = 1, 2, ..., N)$
4: for $k = 1, 2, 3, ...$
5:     $s_{k(i)} = As_{k-1(i)} + p_k(i = 1, 2, ...N)$
6:     $z = TDOA\ measurement\ values$
7:     Reweight $w_{k(i)} = W_{k-1(i)} p(z \mid s_{k(i)})$
8:         where $p(z \mid s_{k(i)}) \sim \mathcal{N}(H(s_{k(i)}), Q_v)$
9:     Normalize $W_{k(i)} = w_{k(i)} / \sum_{i=1}^{N} w_{k(i)}$
10:    $ESS = 1/\sum_{i=1}^{N} (w_{k(i)})^2$
11:    if $ESS \le N_{th}$
12:        Resampling
13:    end if
14:    $s_k = \sum_{i=1}^{N} W_{k(i)} s_{k(i)}$
15: end for
```

3.2. TDOA Measurement Using Generalized Cross Correlation Phase Transform (GCC-PHAT)

To estimate the location of sound sources from TDOA observations, we use the GCC-PHAT method. As mentioned before, the GCC-PHAT is a method of obtaining the correlation coefficient of signals using phase transform, and operates in the frequency domain as that requires less calculation that the time domain. Algorithm 2 shows the algorithm for obtaining the TDOA measurement using acoustic signal input from two acoustic sensors. The TDOA measurements can be obtained by using the Fourier transform process and inverse Fourier transform process of two acoustic signals as in the algorithm. The position of the sound source in the three-dimensional space can be obtained from these measurements. In this paper, we propose performing the calculation process in parallel and accelerating the operation speed of the entire algorithm.

Algorithm 2 GCC-PHAT

```
1: for two signal $x_a[n]$ and $x_b[n]$
2:     $X_a(f) = FFT(x_a[n])$
3:     $X_b(f) = FFT(x_b[n])$
4:     $G_{PHAT}(f) = \frac{X_a(f)[X_b(f)]^*}{|X_a(f)[X_b(f)]^*|}$
5:     $I_{GPHAT}(p) = InverseFFT(G_{PHAT}(f))$
6:     TDOA measurement $T_{ab} = argmax_p(I_{GPHAT}(p))/sampling\ rate$
7: end for
```

3.3. Accelerated Algorithm Based on GPU

Our approach accelerates the proposed position estimation of a moving target using GPU. The GPU is an architecture that has strength in parallel computing, and is used for high-speed computation of a large amount of data. The proposed GPU-based moving target position estimation can be divided into the operation of obtaining the TDOA measurement value from the sensor, the operation of PF processing, and the tracking of the moving target using multiple system state equations.

The calculation of the TDOA measurements in the frequency domain uses GCC-PHAT. Therefore, we use the cuFFT library supported by CUDA for accelerating the computation of TDOA measurements using Fast fourier transform. To accelerate Fourier transform on the data using the same coefficients, the Fourier transform operations for multiple sensors are processed in parallel as shown in Figure 4 by using the batch function of cuFFT. At this time, when the number of sensors is K and the amount of data obtained from each sensor is N, the amount of data copied to the GPU device is K*N. Because the position estimation of the moving target using TDOA measurement is more accurate with the use of many sensors, the proposed accelerated TDOA measurement calculation has an advantage as the data amount increases. As mentioned before, since the proposed algorithm is scalable for number of sensors, the proposed parallel Fast fourier transform is also scalable in the number of sensors.

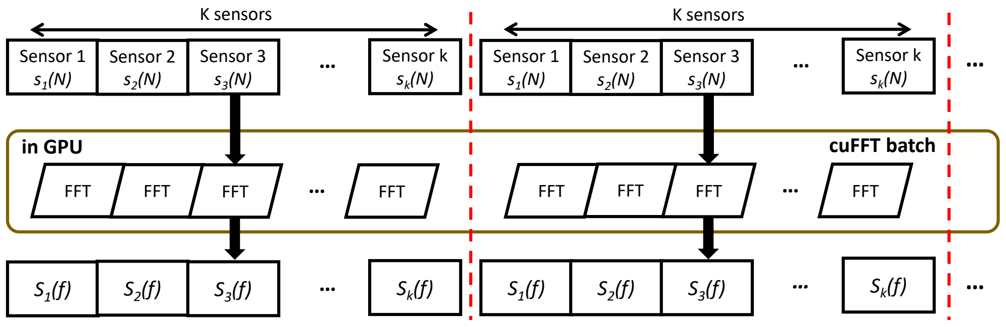

Figure 4. Parallel fast fourier transform using GPU.

Also, we propose the parallel computing of the PF using GPU as shown in Figure 5. Because the PF processing depends on the number of particles, the entire process can be accelerated by paralleling and accelerating the operation of each particle in GPU. In particular, the process of estimating the position of moving targets required the acceleration in performing the weight calculation process of each particle in parallel through the process of updating the state of each particle and calculating the measured value from the state.

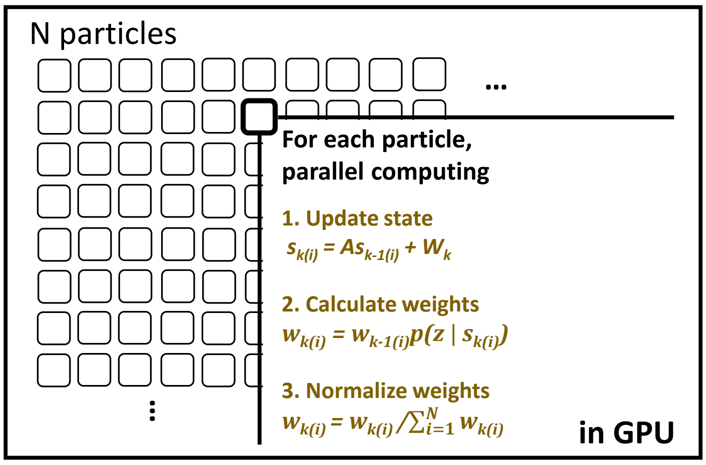
N particles
For each particle, parallel computing
1. Update state
$s_{k(i)} = As_{k-1(i)} + W_k$
2. Calculate weights
$w_{k(i)} = w_{k-1(i)} p(z \mid s_{k(i)})$
3. Normalize weights
$w_{k(i)} = w_{k(i)} / \sum_{i=1}^{N} w_{k(i)}$
in GPU

Figure 5. Particle filter (PF) acceleration in GPU.

3.4. Target Tracking Using Multiple State-Space Models

The KF and PF, which can be used for tracking the position of a target have system state equations based on Markov chains. In other words, if the target moves differently from the predefined system state equation, the estimation result in inaccurate. Therefore, in this paper, we define several other system state equations that are expected and go through them like the current system state equations. This proposed structure also can be used for multiple target tracking using multiple system models. From each multiple state update model, we obtain a measured value. Then the measured values are compared with actual measured values, and particles that have measured values with the smallest differences are selected for the remaining PF process. Using proposed method, tracking of sudden movements of a target can be easily enabled using multiple system state equations based on GPU.

The operation of the GPU-based PF using the proposed system state equations is shown in Figure 6. The process of updating the state of each particle by multiple system state equations is performed in parallel using the kernel function in the GPU. For all particles obtained from the multiple state-space model, the state update process, weight calculation, and normalized process are performed in parallel. To obtain the particles of the state-space model suitable for the current system state, the predicted measured values are compared with actual measured values by finding the smallest difference. Then the particles with the smallest differences are selected and resampled. In our implementation, we use systematic resampling algorithm [25,26] in which the random number generation is required in resampling process. There is a study [27] to accelerate resampling in GPU-based PF. In our study, for reducing the additional overhead running random number calculation during the resampling process, we utilized thread-based running method, which is provided by host CPU, in the process of copying data to the GPU.

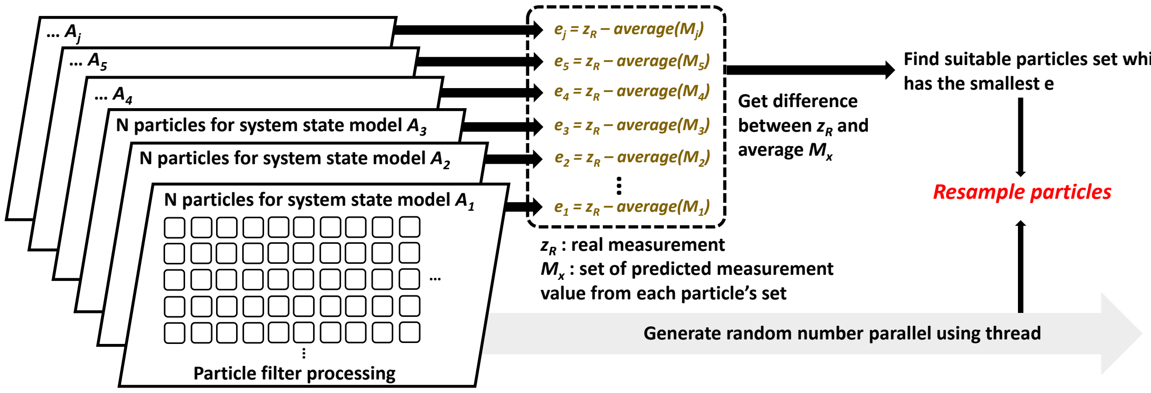
... A_j
... A_5
... A_4
N particles for system state model A_3
N particles for system state model A_2
N particles for system state model A_1
Particle filter processing
$e_j = z_R - average(M_j)$
$e_5 = z_R - average(M_5)$
$e_4 = z_R - average(M_4)$
$e_3 = z_R - average(M_3)$
$e_2 = z_R - average(M_2)$
$e_1 = z_R - average(M_1)$
Get difference between z_R and average M_x
Find suitable particles set which has the smallest e
Resample particles
z_R : real measurement
M_x : set of predicted measurement value from each particle's set
Generate random number parallel using thread

Figure 6. PF using multiple state-space models in GPU.

Algorithm 3 describes the PF processing using multiple state-space models in GPU. The transition matrix A_m is used to formulate multiple state-space models. The PF processing for the selected state-space model is performed in the individual block unit of GPU. This block runs the threads on which the corresponding particles are allocated. Through the synchronization between threads, the average measurement result can be estimated from the updated state, so that we could determine the appropriate state-space model, then continue to perform the remained PF processing incrementally.

Algorithm 3 PF using multiple state-space models in GPU

```
1: $Set\ initial\ state\ s_0^m\ (m = 1, 2, ..., L)$
2: $Generate\ particles\ s_{0(i)}^m\ (i = 1, 2, ..., N), (m = 1, 2, ..., L)$
3: $Set\ initial\ weights\ W_{0(i)}^m = 1/N\ (i = 1, 2, ..., N), (m = 1, 2, ..., L)$
4: for $k = 1, 2, 3, ...,\quad i \leftarrow\ calculated\ by\ thread\ index,\quad m \leftarrow\ calculated\ by\ block\ index$
5:     $s_{k(i)}^m = A_m s_{k-1(i)}^m + p_k^m (i = 1, 2, ...N)$
6:     $z = TDOA\ measurement\ values$
7:     $\_\_syncthreads();$
8:     $Calculate\ difference\ e^m = z - (\sum_{i=1}^N |H_k(s_{k(i)}^m)|)/N$
9:     $\_\_syncthreads();$
10:    if $e^m$ $is\ minimum\ value$
11:        $Reweight\ \ w_{k(i)}^m = W_{k-1(i)}^m p(z \mid s_{k(i)}^m)$
12:          $where\ \ p(z \mid s_{k(i)}^m) \sim \mathcal{N}(H(s_{k(i)}^m), Q_v)$
13:        $Normalize\ \ W_{k(i)}^m = w_{k(i)}^m / \sum_{i=1}^N w_{k(i)}^m$
14:        $ESS = 1/\sum_{i=1}^N (w_{k(i)}^m)^2$
15:        if $ESS \le N_{th}$
16:            $Resampling$
17:        end if
18:        $s_k^m = \sum_{i=1}^N W_{k(i)}^m s_{k(i)}^m$
19:    end if
20: end for
```

4. Experimental Results

Figure 7 shows each experimental flow of MATLAB and GPU. We implemented the proposed GPU-based target position estimation algorithm and analyzed it by operating on a target board. The NVIDIA Jetson TX1 and Tesla K40c (NVIDIA Corporation, Santa Clara, CA, USA) were used as the target GPU as shown in Table 1. We verified the implemented algorithm and analyzed the execution time. The target position estimation algorithm using the PF and the TDOA measurement were implemented using MATLAB (Mathworks, Natick, MA, USA) and its operation was verified as shown in Figure 8. The black dots indicate each position of the acoustic sensor, the blue line indicates the actual moving path of the target, and the red line is the result of estimating the target position using the algorithm. Figure 8's right graph shows the result of target position estimation according to the state change of the target using the multiple state-space models proposed in this study. Even though the state equations used in PF was changed due to the target's sudden movement after a certain time, it was confirmed that the target was accurately estimated using the proposed PF technique.

We compared and analyzed the algorithm execution time used in the present position estimation of the proposed PF with a CPU-based algorithm and GPU-based algorithm in embedded system. Target movement was measured from the signal input to the sensors at 1 s intervals. We used NVIDIA Jetson TX1 as shown in Figure 7 and the fastest fourier transform in the west (FFTW3) library was used to compute FFT in CPU-based algorithm. For the result of when 2000 particles, the execution time of the GPU based algorithm was 6.19 ms, and the CPU based algorithm's execution time was 15.06 ms. We could estimate the target position more quickly when we use the proposed GPU-based algorithm in embedded system. The algorithm execution time is shown in Figure 9a according to the number of particles. In this graph, the horizontal axis is the number of particles and the vertical axis is the execution time of the algorithm. As shown in the graph, when we used the GPU-based algorithm, the execution time of the algorithm to estimate the current state of the target was reduced by about 55% on average. As a result, we find out that proposed GPU-based architecture is feasible for embedded system.

Table 1. Target devices.

Target Device	Model
GPU 1	Jetson TX1 256 cores
CPU 1	64-bit ARM A57 CPUs
GPU 2	Tesla K40c 2880 cores

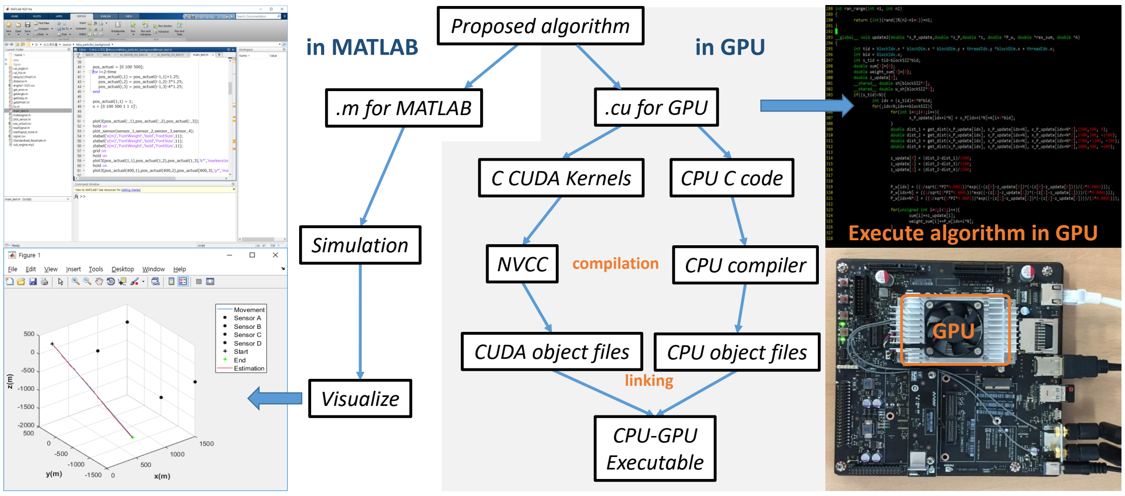

Figure 7. Experimental flow and target embedded board(NVIDIA Jetson TX1).

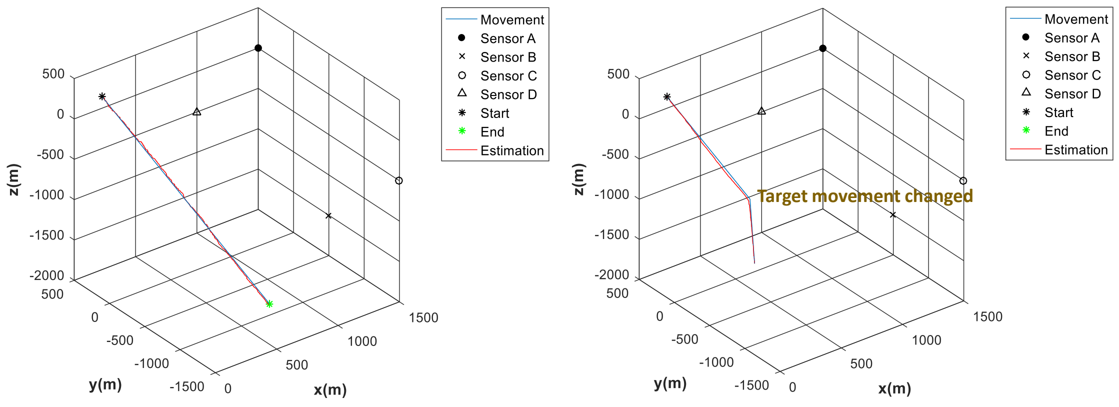

Figure 8. Verification of PF using MATLAB.

Figure 9b shows the partial execution time of the algorithm in TDOA measurement calculation, PF processing, and copying data to GPU in the proposed multiple state-space model based algorithm. As with the result in Figure 9a, the total execution time of the algorithm is increased as the number of particles increased. Also, we found out that the total execution time is shorter than that of the CPU based algorithm, even though the time required to copy data for executing the algorithm in GPU is included.

We compared the execution time of the proposed GPU-based algorithm and the CPU-based algorithm according to the sampling rate of input data when using 2000 particles. The GPU algorithm used for this experiment is a simple GPU-based PF algorithm, not using a multiple state-space model based GPU PF algorithm for accurate comparison with the CPU-based algorithm. Table 2 shows the execution time of the algorithm measured by changing the sampling rate. As the sampling rate increases, the time required to execute the entire algorithm increases. That is, the amount of data input during the same time increases, so that more time is taken to calculate the TDOA measurement value. When using the sampled signal at a sampling rate of 44 kHz, the CPU-based algorithm took about

62.60 ms and the GPU-based algorithm took about 18.24 ms. The reducing rate of GPU-based algorithm for CPU-based algorithm has increased as sampling rate increased. As a result, we found that the GPU-based algorithm is less affected by the execution time relative to the sampling rate.

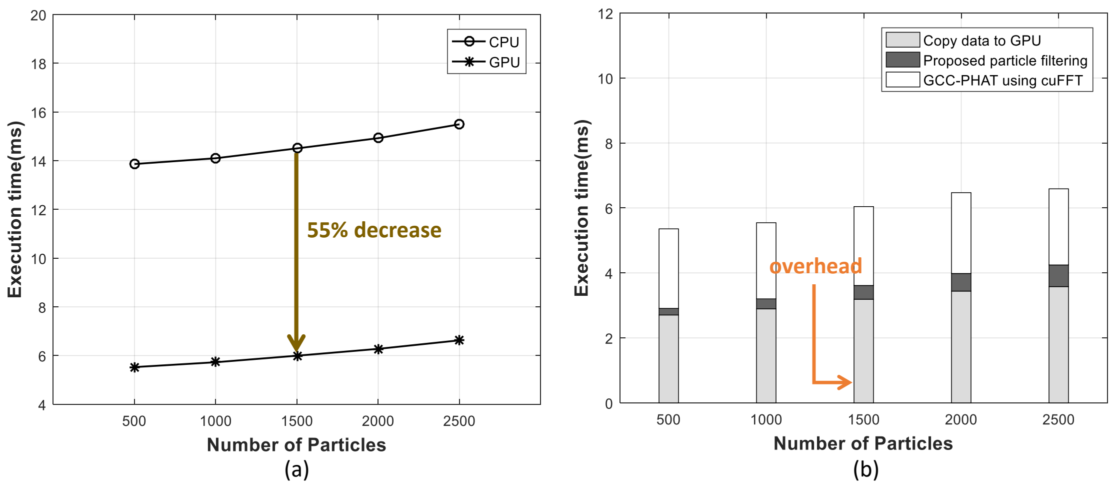

Figure 9. (**a**) Execution time of algorithm according to the number of particles (**b**) Partial execution time of algorithm according to the number of particles.

Table 2. Execution time of algorithm according to the sampling rate.

Sampling Rate (kz)	GPU Based (ms)	Time Increment (ms)	CPU Based (ms)	Time Increment (ms)	Time Reduction Rate (%)
11	6.19	-	15.06	-	58.87
22	10.96	4.77	27.16	12.10	59.64
33	14.23	3.26	46.27	19.10	69.25
44	18.24	4.01	62.60	16.32	70.86

To estimate sudden state changes of the system, we implemented the proposed GPU-based PF using a multiple state-space model that can be predicted. The multiple state-space model based PF using GPU with added parallel processing has similar performance to the PF that just uses the GPU. The results are as follows. Figure 10 shows the execution time according to the number of state-space models with the proposed GPU-based multiple system model using a PF. In this experiment, we use NVIDIA Tesla K40c to see GPU parallelism by multiple state-space model. In the GPU kernel, we implemented block-thread kernel architecture, especially divided the particles in each block according to the state-space model and each block has 1024 threads in two dimensions. Therefore, the implemented algorithm uses the same number of GPU blocks as the number of state-space models. Because of GPU's parallelism, for each number of particles, 1000 to 5000, the execution time is only different by less than 0.16 ms according to the number of state-space models. We found that parallel execution in the GPU does not significantly affect execution time even if the number of particles increases by the number of system models. For example, if the number of state-space models is X, the number of particles is X*N. Table 3 shows the number of blocks and threads, execution time for PF processing, and execution time for copying particles to GPU according to the number of state-space models when the number of particles is 5000. When the number of state-space models is 1 and 100, it shows just a 0.16 ms time difference for processing the PF, while there is a 10.99 ms time difference for copying data to GPU. It shows that a lot of execution time is used in the process of copying data to the GPU. It also shows that the PF processing time does not change significantly when the number of state-space models increases. Using this proposed GPU-based algorithm, we found that the state-space model suitable for sudden movement of the target was found through parallel computing as verified using MATLAB. With this parallel architecture, we find out that this architecture can also be applied to multiple target tracking using different state-space models, as implemented in this study in IoT sensing applications.

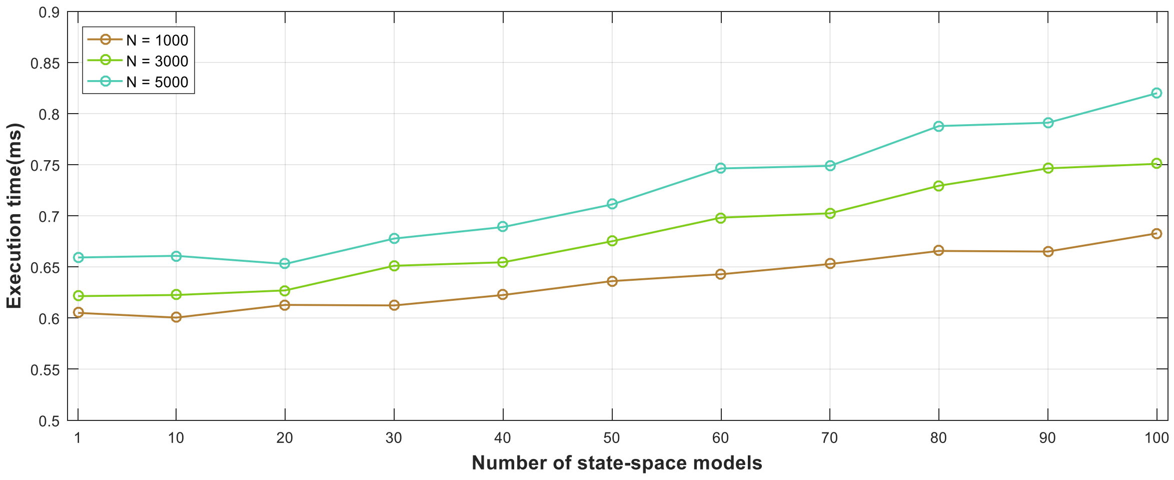

Figure 10. Execution time of algorithm according to the number of particles and state-space models.

Table 3. Number of threads and blocks and partial execution time according to number of state-space models.

Number of Model	Block	Thread	Time for PF Process (ms)	Time for Copy Particles (ms)
1	1	1024	0.66	0.59
10	10	10,240	0.66	1.83
20	20	20,480	0.65	3.24
30	30	30,720	0.68	3.94
40	40	40,960	0.69	5.01
50	50	51,200	0.71	6.36
60	60	61,440	0.75	7.32
70	70	71,680	0.75	7.79
80	80	81,920	0.79	9.59
90	90	92,160	0.79	11.00
100	100	102,400	0.82	11.59

5. Conclusions

In this paper, we implemented an accelerated GPU-based algorithm to estimate moving target position by using multiple state-space models which can be applied to IoT sensing applications. The PF was used to estimate the position of the target, and we parallelized the calculation process of each particle in the GPU kernel. Also, we accelerated TDOA calculation through a parallel FFT process using GPU. Additionally, we extended the PF processing algorithm using a GPU-based multiple state-space model to estimate sudden movement change of the target. The proposed algorithm was initially simulated using MATLAB, and then, the proposed GPU-based algorithm was verified on target GPU. As a result, the execution time of the proposed algorithm using GPU was reduced by about 55% as compared with the CPU-based algorithm in target embedded board, NVIDIA Jetson TX1. The multiple state-space model based PF with parallel processing has a similar execution time in target GPU NVIDIA Tesla K40c with a difference of less than 0.16 ms when estimating the sudden change of movement of the target. Based on this result, the proposed architecture is more effective in terms of high performance and detecting sudden movement changes of the target with lots of sensor data in large-scaled sensing applications. Therefore, the proposed architecture can be effectively applied in these IoT applications due to scalability and parallelism. In a future study, we plan to extend our study on the GPU-based high speed processing algorithm considering the perspective of real-time processing in the big data sensing applications.

Acknowledgments: This study was supported by the BK21 Plus project funded by the Ministry of Education, Korea (21A20131600011), the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2014R1A6A3A04059410) and Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Korea government (MSIT) (2017M3C4A7064179).

Author Contributions: Seongseop Kim designed the entire core architecture and performed the hardware/software implementation and experiments; Jeonghun Cho performed the validation using real embedded board; Daejin Park is a principle investigator for this project, who has responsibility as a corresponding author.

Conflicts of Interest: The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

PF	Particle Filter
KF	Kalman Filter
SMC	Sequential Monte Carlo
ESS	Effective Sample Size
TDOA	Time Difference of Arrival
GCC-PHAT	Generalized Cross Correlation Phase Transform

References

1. Isik, M.T.; Akan, O.B. A three dimensional localization algorithm for underwater acoustic sensor networks. *IEEE Trans. Wirel. Commun.* **2009**, *8*, 4457–4463.
2. Poursheikhali, S.; Zamiri-Jafarian, H. TDOA based target localization in inhomogenous underwater wireless sensor network. In Proceedings of the 2015 5th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, Iran, 29–29 October 2015; pp. 1–6.
3. Won, S.-h.P.; Melek, W.W.; Golnaraghi, F. A Kalman/Particle Filter-Based Position and Orientation Estimation Method Using a Position Sensor/Inertial Measurement Unit Hybrid System. *IEEE Trans. Ind. Electron.* **2010**, *57*, 1787–1798.
4. Vermaak, J.; Blake, A. Nonlinear filtering for speaker tracking in noisy and reverberant environments. In Proceedings of the 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing, Salt Lake City, UT, USA, 7–11 May 2001; Volume 5, pp. 3021–3024.
5. Gustafsson, F.; Gunnarsson, F. Positioning using time-difference of arrival measurements. In Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, Hong Kong, China, 6–10 April 2003.
6. Xu, Y.; Dandan, W.; Hua, F. Underwater acoustic source localization method based on TDOA with particle filtering. In Proceedings of the 26th Chinese Control and Decision Conference (2014 CCDC), Changsha, China, 31 May–2 June 2014; pp. 4634–4637.
7. Broeck, B.V.D.; Bertrand, A.; Karsmakers, P.; Vanrumste, B.; Van Hamme, H.; Moonen, M. Time-domain generalized cross correlation phase transform sound source localization for small microphone arrays. In Proceedings of the 2012 5th European DSP Education and Research Conference (EDERC), Amsterdam, The Netherlands, 13–14 September 2012; pp. 76–80.
8. Qin, B.; Zhang, H.; Fu, Q.; Yan, Y. Subsample time delay estimation via improved GCC PHAT algorithm. In Proceedings of the 2008 9th International Conference on Signal Processing, Beijing, China, 26–29 October 2008; pp. 2579–2582.
9. Belloch, J.A.; Gonzalez, A.; Vidal, A.M.; Cobos, M. On the performance of multi-GPU-based expert systems for acoustic localization involving massive microphone arrays. *Exp. Syst. Appl.* **2015**, *42*, 5607–5620.
10. Minotto, V.P.; Jung, C.R.; Luiz Gonzaga da Silveira, J.; Lee, B. GPU-based approaches for real-time sound source localization using the SRP-PHAT algorithm. *Int. J. High Perform. Comput. Appl.* **2013**, *27*, 291–306.

11. Kim, S.; Cho, J.; Park, D. GPU-based Acceleration of Particle Filter Signal Processing for Efficient Moving-Target Position Estimation. *IEMEK J. Embed. Syst. Appl.* **2017**, *12*.
12. Liang, Y.; Cui, Z.; Zhao, S.; Rupnow, K.; Zhang, Y.; Jones, D.L.; Chen, D. Real-time implementation and performance optimization of 3D sound localization on GPUs. In Proceedings of the 2012 Design, Automation Test in Europe Conference Exhibition (DATE), Dresden, Germany, 12–16 March 2012; pp. 832–835.
13. Choi, C.; Christensen, H.I. RGB-D object tracking: A particle filter approach on GPU. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 1084–1091.
14. Hendeby, G.; Hol, J.D.; Karlsson, R.; Gustafsson, F. A graphics processing unit implementation of the particle filter. In Proceedings of the 2007 15th European Signal Processing Conference, Poznan, Poland, 3–7 September 2007; pp. 1639–1643.
15. Sutharsan, S.; Kirubarajan, T.; Lang, T.; Mcdonald, M. An Optimization-Based Parallel Particle Filter for Multitarget Tracking. *IEEE Trans. Aerosp. Electron. Syst.* **2012**, *48*, 1601–1618.
16. Cabido, R.; Montemayor, A.S.; Pantrigo, J.J. High performance memetic algorithm particle filter for multiple object tracking on modern GPUs. *Soft Comput.* **2012**, *16*, 217–230.
17. Urteaga, I.; Bugallo, M.F.; Djuri, P.M. Sequential Monte Carlo methods under model uncertainty. In Proceedings of the 2016 IEEE Statistical Signal Processing Workshop (SSP), Palma de Mallorca, Spain, 26–29 June 2016; pp. 1–5.
18. Martino, L.; Read, J.; Elvira, V.; Louzada, F. Cooperative parallel particle filters for online model selection and applications to urban mobility. *Digit. Signal Process.* **2017**, *60*, 172–185.
19. Drovandi, C.C.; McGree, J.M.; Pettitt, A.N. A Sequential Monte Carlo Algorithm to Incorporate Model Uncertainty in Bayesian Sequential Design. *J. Comput. Graph. Stat.* **2014**, *23*, 3–24.
20. Yang, P.; Wu, W. Efficient Particle Filter Localization Algorithm in Dense Passive RFID Tag Environment. *IEEE Trans. Ind. Electron.* **2014**, *61*, 5641–5651.
21. Yang, P. PRLS-INVES: A General Experimental Investigation Strategy for High Accuracy and Precision in Passive RFID Location Systems. *IEEE Int. Things J.* **2015**, *2*, 159–167.
22. Yang, P.; Wu, W.; Moniri, M.; Chibelushi, C.C. Efficient Object Localization Using Sparsely Distributed Passive RFID Tags. *IEEE Trans. Ind. Electron.* **2013**, *60*, 5914–5924.
23. Kong, A. *A Note an Importance Sampling Using Standardized Weight*; Technical Report; Department of Statistics, University of Chicago: Chicago, IL, USA, 1992.
24. Martino, L.; Elvira, V.; Louzada, F. Effective sample size for importance sampling based on discrepancy measures. *Signal Process.* **2017**, *131*, 386–401.
25. Douc, R.; Cappe, O. Comparison of resampling schemes for particle filtering. In Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis (ISPA 2005), Zagreb, Croatia, 15–17 September 2005; pp. 64–69.
26. Li, T.; Bolic, M.; Djuric, P.M. Resampling Methods for Particle Filtering: Classification, implementation, and strategies. *IEEE Signal Process. Mag.* **2015**, *32*, 70–86.
27. Murray, L.M.; Lee, A.; Jacob, P.E. Parallel Resampling in the Particle Filter. *J. Comput. Graph. Stat.* **2016**, *25*, 789–805.