

Article

An Improved Particle Filtering Algorithm for Indoor Robot Localization Using a Prior Map

Chongyang Hu ^{1,*}, Qingxuan Gao ¹ and Ruiping Ji ²

¹ College of Electrical and Control Engineering, Shaanxi University of Science and Technology, Xi'an 710021, China; 240612041@sust.edu.cn

² College of Electrical and Control Engineering, Xi'an University of Science and Technology, Xi'an 710054, China; jiruiping@xust.edu.cn

* Correspondence: hcyang@sust.edu.cn

Abstract

This paper focuses on the indoor robot localization problem in the presence of accumulated errors from odometry and IMU. Considering that the motion of the robot is constrained by the prior map, employing the map as a position reference is beneficial for reducing accumulated errors. Therefore, a particle filtering method with map constraints is proposed to improve indoor robot localization accuracy. First, a LiDAR measurement model is constructed to provide the distance from the robot to map obstacles. Since each particle represents a possible state of the robot, the residual between its virtual measurement and the actual LiDAR measurement is used to update the corresponding particle weight. Additionally, an iterative movement strategy is designed to guide the sampling particles toward the high-probability region for enhancing the effectiveness of the particles. Finally, the experimental results show that the proposed method achieves higher localization accuracy than traditional particle filter methods.

Keywords: indoor localization; particle filter; map constraint; movement strategy

1. Introduction

1.1. Motivation

Accurate localization is essential for indoor mobile robots to perform autonomous navigation, path planning, obstacle avoidance, and collaborative tasks [1–4]. However, Global Navigation Satellite System (GNSS) signals are often unavailable or unreliable in indoor environments because of signal blockage and multipath effects [5]. Therefore, indoor robots usually rely on onboard sensors, such as odometers, inertial measurement units (IMUs), LiDAR, and cameras, to acquire motion information and perceive their surroundings for localization [6–8]. Although odometers and IMUs can provide continuous translational and heading measurements, measurement uncertainties and environmental disturbances will cause cumulative errors [7]. In many structured indoor environments, a prior map is usually available. It contains rich environmental structural information and can provide a fixed environmental reference for localization [9,10]. While a map constrains the robot's range of motion, it also provides information about the spatially feasible region. Therefore, there is a strong need to develop state estimation methods with a prior map for robot location systems.



Academic Editor: Ashwin Ashok

Received: 30 June 2026

Revised: 19 July 2026

Accepted: 30 July 2026

Published: 1 August 2026

Copyright: © 2026 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.

1.2. Related Works

Generally, existing study is classified into two categories based on whether map information is fused. The first category, which does not incorporate map constraints, primarily consists of Kalman-filter (KF)-based variants and numerical approximation methods. KF-based methods have been widely used because of their relatively low computational cost [11,12]. Yin et al. [13] propose a robust adaptive extended Kalman filter for GNSS/inertial navigation system (INS) vehicle navigation, where a Huber-based adaptive factor is constructed from standardized residuals to detect abnormal measurements and reduce their influence through covariance adjustment. Yuan et al. [14] develop an improved strong-tracking UKF-SLAM method using three-position ultrasonic detection. It combines measurement screening, arc-shaped modeling, and multiple fading factors to enhance localization and mapping accuracy. Durodié et al. [15] propose a UWB-ego-motion PF for indoor pose estimation by using a moving horizon hypothesis. The method fuses ego-motion with UWB measurements to correct heading and position drift. It also separates bias effects from the model equations to reduce the computational load. Xia et al. [16] propose an improved Monte Carlo localization method based on off-line feature matching and improved particle swarm optimization. The method identifies possible robot poses through feature matching and improves localization performance under unknown initial poses and robot kidnapping conditions. These methods improve localization accuracy to some extent, but do not fully exploit the additional constraint information provided by prior maps. A prior map contains rich spatial information, which provides a stable environmental reference for robot localization and helps reduce cumulative sensor errors. Therefore, an increasing number of studies incorporate prior maps into indoor robot localization methods. However, existing methods generally rely on map information obtained from LiDAR detections for localization. When the LiDAR point cloud is excessively sparse, the detection information becomes highly uncertain, which is unfavorable for robot localization.

An alternative category is particle filter (PF)-based methods incorporating map information. Zhu et al. [17] introduce collision detection into Adaptive Monte Carlo Localization (AMCL) and remove particles that are inconsistent with the mapped environment. Yang et al. [9] integrate multi-sensor observations with semantic map constraints to improve the robustness of PF-based indoor localization. Harder et al. [18] propose a real-time map-matching method based on a backtracking particle filter and geospatial analysis. The method uses floor-plan elements as spatial constraints to identify and remove particles whose positions or motion trajectories are inconsistent with the map. Li et al. [19] construct an ambiguity grid map to improve localization reliability in environments with repetitive structures. Adurthi [20] proposes a scan-matching-based particle filter for localization in a known global map. The method uses scan-to-map matching to improve particle resampling and employs a precomputed likelihood grid to accelerate likelihood evaluation. Sasaki et al. [21] develop a mobile robot self-localization method using an existing layout map. Stable wall point clouds extracted from 3D LiDAR data are matched with mapped wall information and then used for particle-filter-based pose estimation. Cheng and Manduchi [22] use plane constraints and certain empty-space constraints to support PF initialization. Elhousni et al. [23] propose a LiDAR-OSM (Open Street Map) vehicle localization method based on a constrained particle filter. The method matches LiDAR point clouds with road and building information extracted from OSM and employs constrained resampling to keep the estimated vehicle position within the feasible road region. The above methods incorporate prior map information as a fixed environmental reference to improve localization accuracy and robustness.

In fact, LiDAR detections can accurately reflect the spatial relationship between the robot and the map. Existing methods generally define an effective LiDAR measurement range to

exclude measurements beyond the reliable range. Only measurements within this range are used for map matching and state updating. However, when no useful detection information is obtained within the effective measurement range, the corresponding observations are usually ignored. Consequently, the corresponding free-space information is not used to update particle weights. Therefore, it is necessary to develop a method that exploits non-detection information. The particle weights can be updated by comparing the actual measurements with the virtual measurements of the particles, thereby improving localization accuracy.

1.3. Results and Organization

To reduce accumulated sensor errors, this paper proposes a map-constrained particle filtering (MC-PF) method for indoor localization. The main contributions of this paper are as follows: (1) A map-based LiDAR measurement model is constructed to establish a correspondence between the actual LiDAR measurements and the virtual measurements of each particle along the same beam directions. (2) Based on the measurement model, a weight update method is proposed to penalize particles whose virtual measurements are inconsistent with the actual measurements by using the corresponding measurement residuals. (3) An iterative movement strategy is designed to improve the effectiveness of particles, in which the particle velocities and positions are iteratively adjusted according to the measurement likelihood, thereby guiding the sampled particles toward high-probability regions.

The rest of the paper is summarized as follows. Section 2 introduces the robot motion model and presents the investigated problem. Section 3 proposes the particle weight updating algorithm with the map information. Section 4 presents the implementation and analysis of the proposed algorithm. Section 5 verifies the effectiveness of the proposed algorithm through a simulation study. Section 6 draws the conclusion.

2. Problem Formulation

A dynamic system with map information is constructed for robot localization. The robot state at time k is defined as x_k , which includes the position vector $[\xi, \psi]^T$ and heading angle ϕ . The state evolution model is:

$$x_k = f(x_{k-1}) + w_k, \quad (1)$$

where

$$f(x_{k-1}) = \begin{bmatrix} \xi_{k-1} + \Delta s \cos \phi_{k-1} \\ \psi_{k-1} + \Delta s \sin \phi_{k-1} \\ \phi_{k-1} + \Delta \phi \end{bmatrix}, \quad w_k \sim \mathcal{N}(0, Q);$$

Δs and $\Delta \phi$ denote the translational increment and heading angle increment measured by the odometer and the IMU, respectively. w_k is the Gaussian process noise with zero mean and covariance matrix Q . Accordingly, the corresponding state transition probability density is denoted by $p(x_k | x_{k-1})$.

LiDAR has been widely adopted in practical measurement tasks because of its high measurement accuracy and long detection range. Therefore, LiDAR is used as the measurement sensor in this paper.

$$z_k^{\text{LiDAR}} = h(x_k, M) + v_k, \quad (2)$$

where M represents the map containing the position information of infeasible regions, $h(\cdot)$ is a nonlinear function and v_k is the measurement noise. The actual measurement made by the LiDAR at time k is recorded as $z_k = \{z_k^{(1)}, z_k^{(2)}, \dots, z_k^{(m)}\}$, where $m = 120$ denotes the number of LiDAR beams. The LiDAR has a 180° front-facing field of view and an angular resolution $\Delta \theta = 1.5^\circ$. The angular direction of each beam $\theta_k^{(j)}$ is denoted by

$\theta_k^{(j)} = \phi_k - 90^\circ + (j-1) \times \Delta\theta$. $z_k^{(j)}$ represents the measured distance value of the j -th laser beam [24], the measurements of each laser beam are independent, and the joint likelihood of the overall laser measurement can be written as:

$$p(z_k^{\text{LiDAR}} | x_k, M) = \prod_{j=1}^m p(z_k^{(j)} | x_k, M). \quad (3)$$

Remark 1. The prior map not only describes the robot's traversable paths or feasible areas but also includes infeasible regions. For robot localization, the infeasible area is also valuable information and is worth being integrated into the particle filter localization algorithm.

For the dynamic system (1)–(2), the goal of this paper is to find the state posterior probability density function (PDF) $p(x_k | z_{1:k}, M)$ by using measurements $z_{1:k} = \{z_1, z_2, \dots, z_k\}$ and map information M . The Bayesian estimation with map information includes two stages: prediction and update. At the prediction stage, the predicted PDF can be written as:

$$\begin{aligned} p(x_k | z_{1:k-1}, M) &= \int p(x_k, x_{k-1} | z_{1:k-1}, M) dx_{k-1} \\ &= \int p(x_k | x_{k-1}, z_{1:k-1}, M) p(x_{k-1} | z_{1:k-1}, M) dx_{k-1} \\ &= \int p(x_k | x_{k-1}, M) p(x_{k-1} | z_{1:k-1}, M) dx_{k-1}. \end{aligned} \quad (4)$$

At the update stage, the posterior PDF is expressed as:

$$\begin{aligned} p(x_k | z_{1:k}, M) &= \frac{p(x_k, z_k, z_{1:k-1}, M)}{p(z_k, z_{1:k-1}, M)} \\ &= \frac{p(z_k | x_k, z_{1:k-1}, M) p(x_k | z_{1:k-1}, M)}{p(z_k | z_{1:k-1}, M)} \\ &= \frac{p(z_k | x_k, M) p(x_k | z_{1:k-1}, M)}{p(z_k | z_{1:k-1}, M)}. \end{aligned} \quad (5)$$

For nonlinear and non-Gaussian systems, the above posterior PDF is usually difficult to solve analytically, so the particle filter is used to approximate it numerically. To characterize the posterior PDF, a set of particles with corresponding weights is utilized, where N is the number of particles. Then, the posterior PDF fused with map information at time k can be expressed as:

$$p(x_k | z_{1:k}, M) \approx \sum_{i=1}^N \omega_k^{(i)} \delta(x_k - x_k^{(i)}),$$

where $\delta(\cdot)$ is the Dirac delta function and the particle weights satisfy $\sum_{i=1}^N \omega_k^{(i)} = 1$. Particles are drawn from the importance density function (IDF) fused with map information. Here, $p(x_k^{(i)} | x_{k-1}^{(i)}, M)$ denotes the state transition PDF that integrates map information, and $q(x_k^{(i)} | x_{k-1}^{(i)}, z_k, M)$ denotes the corresponding IDF. The i -th particle unnormalized weight is $\omega_k^{(i)} = \omega_{k-1}^{(i)} \frac{p(z_k | x_k^{(i)}, M) p(x_k^{(i)} | x_{k-1}^{(i)}, M)}{q(x_k^{(i)} | x_{k-1}^{(i)}, z_k, M)}$.

How to incorporate map constraints into particle filtering is a key issue. Existing methods usually use map constraints at the sampling step by truncation or projection [25,26]. The truncation method directly discards particles in infeasible regions and keeps particles in feasible regions. This reduces the effective sample size (ESS) and affects the approximation accuracy of the posterior PDF, as shown in Figure 1a. The projection method corrects particles

by projecting the particles that fall into infeasible areas to the boundary of the feasible area. However, this process distorts the original PDF and affects the true representation of the posterior PDF. Some particles may move away from the true state after projection and gather near the boundary of the infeasible region, as shown in Figure 1b. As mentioned above, existing methods force the sampled particles to satisfy the constraints. However, in practical localization scenarios, robots do not move along the boundary, meaning that there is a threshold distance between the particles and the constraint boundary. Therefore, by incorporating this threshold distance, we formulate the prior map as a measurement model and propose a map-constrained particle filtering method.

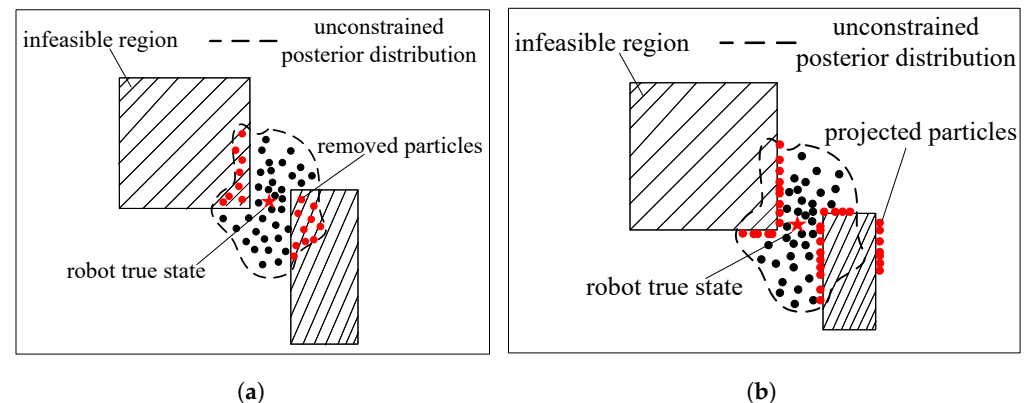


Figure 1. Existing constrained particle filtering algorithms: (a) Truncated PDF. (b) Projected particles in the infeasible region.

3. Particle Weight Updating with Map Constraints

3.1. Map Constraint Modelling

In the indoor localization process of mobile robots, the prior map M not only describes the spatial distribution of infeasible regions in the environment, but also constrains the possible positions of the robot. Here, we focus on the two-dimensional motion space of the robot, denoted by $\Omega \subset \mathbb{R}^2$, and the map environment can be divided into the feasible region Ω_{free} and the infeasible region Ω_{occ} satisfying:

$$\begin{aligned}\Omega_{\text{occ}} \cap \Omega_{\text{free}} &= \emptyset, \\ \Omega_{\text{occ}} \cup \Omega_{\text{free}} &= \Omega.\end{aligned}$$

In fact, the essence of a map is that the robot position at any time must be within the feasible region. Since the map constraints mainly act on the geometric position of the robot on the plane, the position vector of the robot is denoted as $p_k = [\xi, \psi]^T$, which can be expressed as $p_k \in \Omega_{\text{free}}$. In actual environments, it is insufficient to consider only whether the robot centroid lies in the free region. The robot body has a certain size, and a safety distance is also needed to avoid collision with obstacles. The robot is modeled as a cylinder, and the constraint radius is $r_c = r_b + r_s$, where r_b is the cylinder radius and r_s is the safety distance. Therefore, the new infeasible region can be written as:

$$\Omega'_{\text{occ}} = \{b \mid \text{dist}(b, \Omega_{\text{occ}}) \leq r_c\}, \quad (6)$$

where b denotes an arbitrary point in the map, and $\text{dist}(b, \Omega_{\text{occ}})$ represents the minimum Euclidean distance from point b to the boundary of the infeasible area Ω_{occ} . The effective feasible region after considering the robot size and safety distance can be expressed as $\Omega'_{\text{free}} = \Omega \setminus \Omega'_{\text{occ}}$. The condition that the robot satisfies map constraints can be further written as $p_k \in \Omega'_{\text{free}}$. Therefore, the map constraint discriminant function can be defined as:

$$C(x_k) = \begin{cases} 0, & p_k \notin \Omega'_{\text{free}} \\ 1, & p_k \in \Omega'_{\text{free}} \end{cases} \quad (7)$$

$C(x_k) = 1$ indicates that the current position of the robot satisfies the map constraints, and $C(x_k) = 0$ indicates that the current position violates the map constraints. To measure the distance between the current position of the robot and the boundary of the infeasible area, the map distance function is introduced:

$$D(p_k) = \min_{q \in \Omega_{\text{occ}}} \|p_k - q\|_2 - r_c, \quad (8)$$

where q denotes an arbitrary point in the region Ω_{occ} , and $\|\cdot\|_2$ is used to measure the geometric distance between the current position of the robot p_k and the boundary of the infeasible area. Compared with other statistical distances, the Euclidean distance is well suited for geometric distance calculation because of its simple and straightforward computation. Therefore, it is used to measure the distance relationship between the robot position and the infeasible region. When $D(p_k) > 0$, it indicates that the robot is within the safe and feasible region. When $D(p_k) = 0$, it indicates that the robot is on the constraint boundary. In addition, the robot movement is a continuous process, so the position at a single moment should satisfy the map feasibility requirements, and the motion trajectory should also avoid crossing the infeasible area. Therefore, the discriminant function $C(x_k)$ and the distance function $D(p_k)$ are not only geometric descriptions of the map constraint, but are also used in the subsequent particle filtering process. Specifically, $C(x_k)$ is used to determine whether each particle satisfies the map feasibility constraint, while $D(p_k)$ provides the geometric basis for constructing the map-matching error in the weight update stage.

3.2. Update Particle Weights Based on Map Constraint

After optimizing the sampled particles, some particles may remain in the infeasible region. Therefore, a map constraint model is introduced to deal with these particles. The particle weight is updated using the current measurement as $\omega_k^{(i)} \propto \omega_{k-1}^{(i)} p(z_k | x_k^{(i)}, M)$. The measurement likelihood function is composed of the LiDAR measurement likelihood $p(z_k^{\text{LiDAR}} | x_k, M)$ and the map constraint observation likelihood $p(z_k^{\text{map}} | x_k, M)$. The unconstrained particle distribution is shown in Figure 2a, and the constrained particle distribution is shown in Figure 2b. When a particle falls into an infeasible area, it is eliminated; otherwise, it is retained.

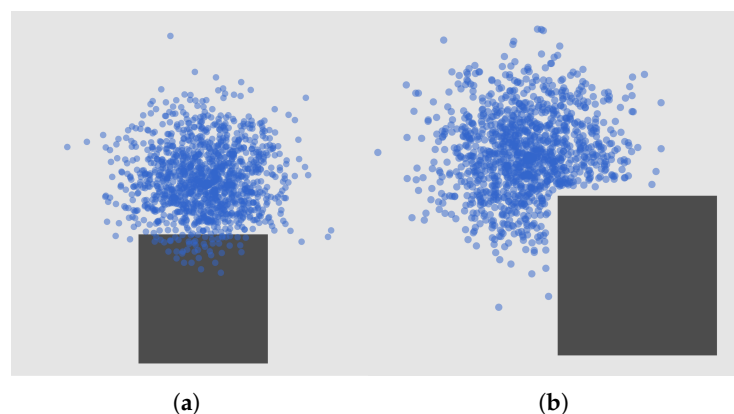


Figure 2. Schematic diagram of particle distribution: (a) Unconstrained particle distribution. (b) Constrained particle distribution.

$C(\cdot)$ can be used not only to determine whether the robot's current position satisfies the map constraints, but also to evaluate each particle, so as to eliminate invalid particles that fall into infeasible regions. We define the map-constraint measurement as $z_k^{\text{map}} = 1$ and the corresponding measurement model as:

$$p(z_k^{\text{map}} = 1 \mid x_k^{(i)}, M) = C(x_k^{(i)}). \quad (9)$$

The joint measurement likelihood can be expressed as:

$$p(z_k \mid x_k, M) = p(z_k^{\text{LiDAR}} \mid x_k, M) p(z_k^{\text{map}} \mid x_k, M). \quad (10)$$

Generally, the LiDAR has a maximum effective measurement distance d_{max} , within which obstacles can be reliably detected and their distances measured, $z_k^{(j)}$ denotes the actual range measurement, and $\hat{z}_k^{(i,j)}$ denotes the range obtained by ray casting from the i -th particle. To explicitly characterize the agreement between the measured and map-predicted ranges, the beam-wise residual is defined as a piecewise function covering obstacle matches, measurement–map mismatches, and consistent free-space observations as follows:

$$e_k^{(i,j)} = \begin{cases} |z_k^{(j)} - \hat{z}_k^{(i,j)}|, & z_k^{(j)} \leq d_{\text{max}}, \hat{z}_k^{(i,j)} \leq d_{\text{max}}, \\ d_{\text{max}} - \hat{z}_k^{(i,j)}, & z_k^{(j)} > d_{\text{max}}, \hat{z}_k^{(i,j)} \leq d_{\text{max}}, \\ 0, & z_k^{(j)} > d_{\text{max}}, \hat{z}_k^{(i,j)} > d_{\text{max}}. \end{cases} \quad (11)$$

When an obstacle is detected, this measurement is regarded as a valid measurement, as shown in Figure 3a. If a measured distance exceeds the maximum effective measurement distance, it indicates that no obstacle is detected within the effective detection range of the sensor. Such a situation is usually regarded as invalid and ignored in the measurement update process, as shown in Figure 3b. In fact, this information indicates that there should be no obstacle within d_{max} along the robot's measurement direction. Figure 3 illustrates the weight correction process. Since each particle represents a different position and heading hypothesis, its virtual measurement direction may differ from the robot's actual measurement direction.

Effective use of map information enables a more accurate evaluation of each particle's importance weight, thereby providing a closer approximation to the true PDF. A criterion function $\delta(E_k^{(i)})$ is constructed to modify the particle weights:

$$\delta(E_k^{(i)}) = \exp(-\lambda E_k^{(i)}), \quad (12)$$

where $\lambda > 0$ represents the penalty coefficient, which is used to adjust the strength of weight decay. $E_k^{(i)}$ represents the matching error. The penalty coefficient λ controls the decay rate of particle weights with respect to the map-matching error. A larger λ strongly penalizes inconsistent particles but may rapidly reduce the ESS, whereas a smaller λ weakens the discriminative effect of the map constraint. When the LiDAR detects an obstacle (valid measurement), its matching error is:

$$E_k^{1,(i)} = \frac{1}{m} \sum_{j=1}^m e_k^{(i,j)}, \quad (13)$$

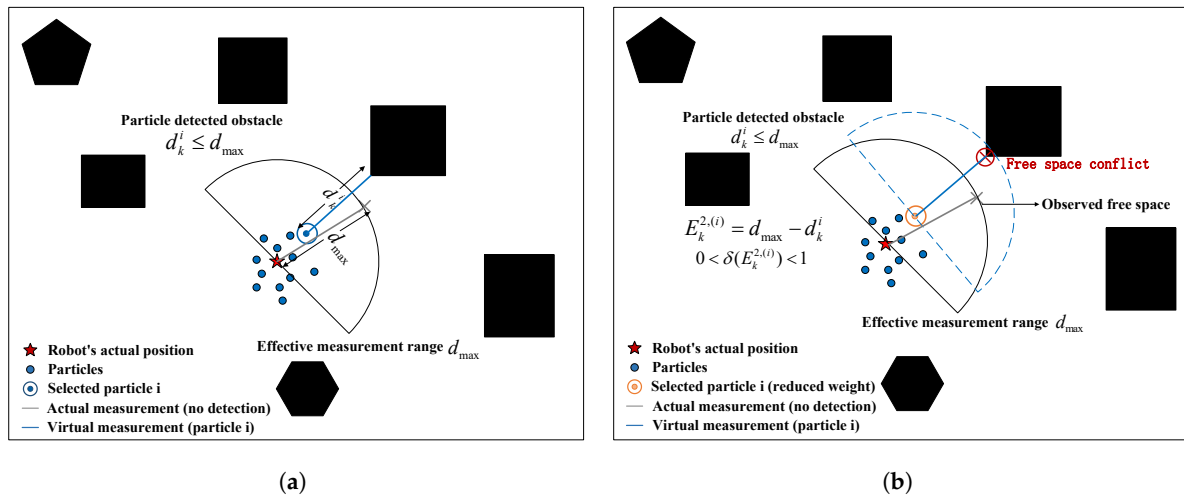


Figure 3. Schematic diagram of particle weight modification based on map constraints: (a) Conventional detection-only update. (b) Proposed free-space-aided update.

The particle weights are updated using the obtained joint observation likelihood as follows:

$$\omega_k^{1,(i)} = \omega_{k-1}^{(i)} p(z_k | x_k, M) \delta(E_k^{1,(i)}), \quad (14)$$

where $\omega_k^{1,(i)}$ denotes the weight obtained from obstacle-measurement matching using Equations (13) and (14). When the measurement result shows that no obstacle is detected within the valid sensing range, map constraint is used to correct the particle weights. For each particle, the distance d_k^i denotes the minimum distance from the particle to the nearest obstacle, i.e., the minimum value of the $z_k^{(i,j)}$. If the particle predicts that there is no obstacle within the valid measurement range, its weight remains unchanged, and this can also be regarded as a valid measurement. Otherwise, if the particle predicts that there is an obstacle within the valid measurement range, this contradicts the actual measurement result. At this time, the matching error $E_k^{(i)}$ can be expressed as:

$$E_k^{2,(i)} = \max(0, d_{\max} - d_k^i), \quad (15)$$

The correction form of the particle weights can be expressed as:

$$\omega_k^{2,(i)} = \omega_{k-1}^{(i)} p(z_k | x_k, M) \delta(E_k^{2,(i)}), \quad (16)$$

where $\omega_k^{2,(i)}$ denotes the corrected weight for the measurement–map mismatch case using Equations (15) and (16). When the environment around a particle matches the real environment well, $E_k^{(i)}$ is small, the penalty term is close to 1, and it has little effect on the original weight. When there is a significant difference between the surrounding environment of the particle and the map structure, its weight will be significantly weakened. In summary, the weight is updated according to the following formula:

$$\omega_k^{(i)} = \begin{cases} \omega_k^{1,(i)}, & z_k \leq d_{\max}, \\ \omega_k^{2,(i)}, & z_k > d_{\max} \text{ and } d_k^i \leq d_{\max}, \\ \omega_{k-1}^{(i)}, & z_k > d_{\max} \text{ and } d_k^i > d_{\max}. \end{cases} \quad (17)$$

The method for updating particle weights based on map constraint is described in Algorithm 1. For the i -th particle $x_k^{(i)}$, the map constraint function $C(x_k^{(i)})$ is first used to determine its feasibility. If $C(x_k^{(i)}) = 0$, the particle is discarded. Otherwise, when the LiDAR measurement z_k is consistent with the particle virtual measurement, the matching error $E_{k_1}^{(i)}$ is used to update the particle weight. When it is inconsistent with the actual measurement, the matching error $E_{k_2}^{(i)}$ is used to penalize the particle weight.

Algorithm 1 Particle weight updating with map constraints

```

1: Input:  $\{(x_k^{(i)}, \omega_{k-1}^{(i)})\}_{i=1}^N, z_k, M$ 
2: Output:  $\{(x_k^{(i)}, \omega_k^{(i)})\}_{i=1}^N$ 
3: for  $i = 1$  to  $N$  do
4:    $\delta(E_k^{(i)}) = \exp(-\lambda E_k^{(i)})$ ;
5:    $p(z_k | x_k, M) = p(z_k^{\text{LiDAR}} | x_k, M)p(z_k^{\text{map}} | x_k, M)$ ;
6:   if  $x_k^{(i)}$  falls into an infeasible region then
7:     Discard particle  $x_k^{(i)}$ ;
8:   else if  $z_k \leq d_{\max}$  then
9:      $E_k^{1,(i)} = \frac{1}{m} \sum_{j=1}^m e_k^{(i,j)}$ ;
10:     $\omega_k^{1,(i)} = \omega_{k-1}^{(i)} p(z_k | x_k, M) \delta(E_k^{1,(i)})$ ;
11:     $\omega_k^{(i)} = \omega_k^{1,(i)}$ ;
12:   else if  $z_k > d_{\max}$  and  $d_k^{(i)} \leq d_{\max}$  then
13:      $E_k^{2,(i)} = \max(0, d_{\max} - d_k^{(i)})$ ;
14:      $\omega_k^{2,(i)} = \omega_{k-1}^{(i)} p(z_k | x_k, M) \delta(E_k^{2,(i)})$ ;
15:      $\omega_k^{(i)} = \omega_k^{2,(i)}$ ;
16:   else
17:     Retain the original weight of the particle  $\omega_{k-1}^{(i)}$ ;
18:      $\omega_k^{(i)} = \omega_{k-1}^{(i)}$ ;
19:   end if
20: end for
  
```

3.3. Iterative Movement for Sampling Particles

Since the true IDF is unknown, the state transition density is commonly used for particle sampling. However, it does not incorporate the current measurement, which may generate particles far from high-probability regions and reduce particle effectiveness. To improve sample effectiveness, we focus on the sampling method and propose an iterative movement strategy.

The particle set at time $k - 1$ is $\{x_{k-1}^{(i)}, \omega_{k-1}^{(i)}\}_{i=1}^N$, where $x_{k-1}^{(i)}$ and $\omega_{k-1}^{(i)}$ represent the state and weight of the i -th particle, respectively. The predicted PDF $p(x_k | x_{k-1})$ is used as the IDF to obtain particles.

$$x_k^{(i)} \sim p(x_k | x_{k-1}^{(i)}), \quad i = 1, 2, \dots, N. \quad (18)$$

To guide particles toward high-likelihood regions, an iterative movement strategy is designed to adjust the sampled particles, while a threshold distance r_{th} is introduced to prevent them from entering infeasible regions. By constructing a fitness function, the search process guides each particle toward a higher-probability region, thereby enabling

the i -th particle to move to higher-probability regions. The velocity and position of the i -th particle are iterated by the following equations:

$$v_{k,n+1}^{(i)} = \omega v_{k,n}^{(i)} + c_1 r_1 (P_{best}^{(i)} - x_{k,n}^{(i)}) + c_2 r_2 (G_{best} - x_{k,n}^{(i)}), \quad (19)$$

$$x_{k,n+1}^{(i)} = x_{k,n}^{(i)} + v_{k,n+1}^{(i)}, \quad (20)$$

where the inertia weight ω balances the global exploration and local refinement abilities of the particles. The variables r_1 and r_2 are independent random numbers uniformly distributed in the interval $[0, 1]$. The learning factors c_1 and c_2 control the influence of the personal-best and global-best positions, respectively. Moreover, $P_{best(i)}$ denotes the personal-best position previously found by the i -th particle, while G_{best} denotes the global-best position found by the entire particle swarm.

The fitness function based on the measurement likelihood is defined as:

$$\varepsilon^{(i)} = \exp \left\{ -\sqrt{(z_k - z_{pred}^{(i)}) R_k^{-1} (z_k - z_{pred}^{(i)})^T} \right\}, \quad (21)$$

where z_k represents the current measurement, $z_{pred}^{(i)} = z_k^{(i)} = h(x_k, M)$ represents the virtual measurement of the particle and R_k denotes the measurement-noise covariance matrix. The same fitness function is used to evaluate the current particle position, its $P_{best}^{(i)}$ and the G_{best} . At the beginning of the iterative movement procedure, the $P_{best}^{(i)}$ of each particle is initialized as its initial sampled position. The G_{best} is initialized as the $P_{best}^{(i)}$ having the highest fitness. At the n -th internal iteration, a candidate position is generated using Equations (19) and (20), and its fitness is calculated using Equation (21). The candidate position is accepted only if it satisfies the map constraint and its fitness $\varepsilon_{new}^{(i)}$ is not lower than the fitness $\varepsilon_{old}^{(i)}$ of current position; otherwise, the current particle position is retained. Repeated search iterations may cause the particles to become overly concentrated, preventing them from accurately representing the true state distribution. Therefore, we set a small maximum number of iterations N_t and compare the average particle fitness $\bar{\varepsilon}_n$ with the threshold ε_{th} to avoid particles gathering too much in a local optimum area.

4. Implementation and Analysis of Map-Constrained Particle Filtering

4.1. Map-Constrained Particle Filter Implementation

The proposed MC-PF is described in Algorithm 2. We obtain particles from the unconstrained IDF, and design an iterative movement strategy to guide the sampling particles toward the high-probability region. We set a maximum number of iterations and a fitness threshold as stopping criteria to prevent the particles from becoming overly concentrated. At the updating stage, the particle weights are updated by combining the LiDAR observation likelihood with the map constraint information. For particles that fall into infeasible regions, their weights are set to 0. The weights of the remaining particles are updated using the criterion function. At the resampling stage, standard resampling is performed when the ESS falls below the predefined threshold. The resampled particle set is then used for state estimation and propagated to the next time step.

Algorithm 2 : The MC-PF

```

1: Input:  $\{x_{k-1}^{(i)}, \omega_{k-1}^{(i)}\}_{i=1}^N, M, n, c_1, c_2, N_t, \varepsilon_{th}, d_{max}$ 
2: Output:  $\{x_k^{(i)}, \omega_k^{(i)}\}_{i=1}^N, \hat{x}_k$ 
3: Draw particles from the unconstrained IDF;
4: Initialize  $P_{best}^{(i)}$  and  $G_{best}$ ;
5: for  $n = 1$  to  $N_t$  do
6:   for  $i = 1$  to  $N$  do
7:     Update particle velocity and position by Equations (19) and (20);
8:     Accept the candidate if it satisfies the map constraints and  $\varepsilon_{new}^{(i)} \geq \varepsilon_{old}^{(i)}$ ;
9:     Update  $P_{best}^{(i)}$  if its fitness improves;
10:   end for
11:   Update  $G_{best}$  using the best  $P_{best}^{(i)}$ ;
12:    $\bar{\varepsilon}_n = \frac{1}{N} \sum_{i=1}^N \varepsilon_n^{(i)}$ ;
13:   if  $\bar{\varepsilon}_n \geq \varepsilon_{th}$  then
14:     break;
15:   end if
16: end for
17: for  $i = 1$  to  $N$  do
18:   Construct the criterion function  $\delta(E_k^{(i)}) = \exp(-\lambda E_k^{(i)})$ ;
19:   The joint observation likelihood is constructed as shown in Equation (10);
20:   if  $x_k^{(i)}$  falls into the infeasible region then
21:      $\omega_k^{(i)} = 0$ ;
22:   else if  $z_k \leq d_{max}$  then
23:     Update the particle weight  $\omega_k^{1,(i)}$  by Equations (13) and (14);
24:   else if  $z_k > d_{max}$  and  $d_k^i > d_{max}$  then
25:     Update the particle weight  $\omega_k^{2,(i)}$  by Equations (15) and (16);
26:   else
27:     Retain the original weight of the particle  $\omega_{k-1}^{(i)}$ ;
28:   end if
29: end for
30: Normalize weights:  $\bar{\omega}_k^{(i)} = \frac{\omega_k^{(i)}}{\sum_{j=1}^N \omega_k^{(j)}}$ ;
31: State estimate:  $\hat{x}_k = \sum_{i=1}^N \bar{\omega}_k^{(i)} x_k^{(i)}$ ;
32: Compute the ESS;
33: if  $N_{eff} < N_{th}N$  then
34:   Apply standard resampling;
35: end if
36: Form the new particle set  $\{x_k^{(i)}, \omega_k^{(i)}\}_{i=1}^N$ ;

```

The block diagram of the proposed method is shown in Figure 4. At each time step, particles are first sampled from the unconstrained importance density function and then adjusted using the iterative movement strategy. Subsequently, the particle weights are updated by jointly using the LiDAR measurements and the prior map, including both obstacle-detection and obstacle non-detection information. After weight normalization, resampling is performed according to the ESS, and the robot state is estimated from the weighted particle set.

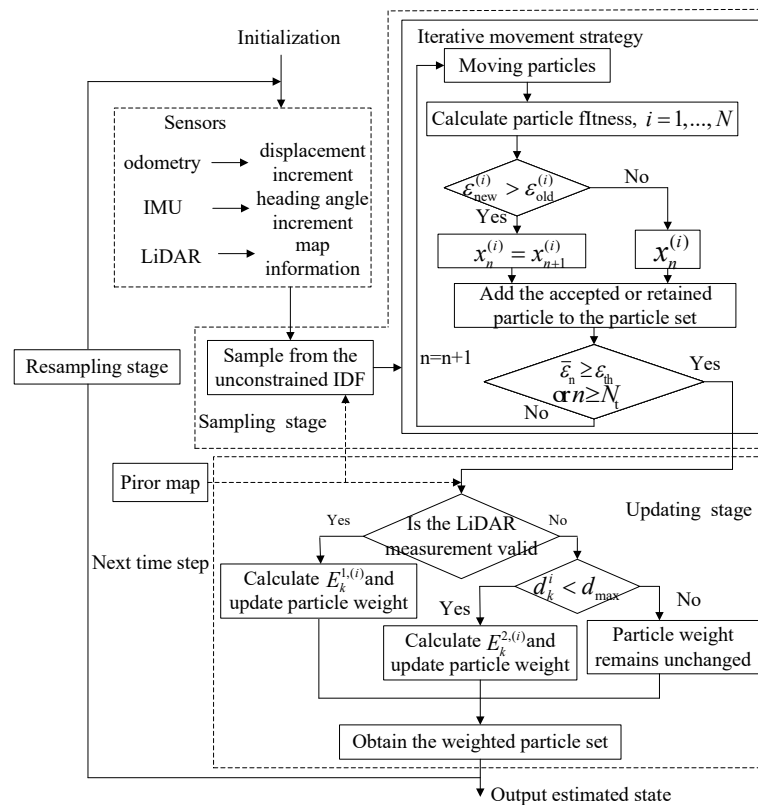


Figure 4. Block diagram of MC-PF.

4.2. MC-PF vs Conventional Constrained Particle Filters

As shown in Figure 5a,b, the particle filter mainly consists of four stages: initialization, sampling, update, and resampling. Initialization sets the initial particle set. Sampling generates particles. Update calculates and normalizes the particle weights. Resampling is used to prevent particle degeneracy.

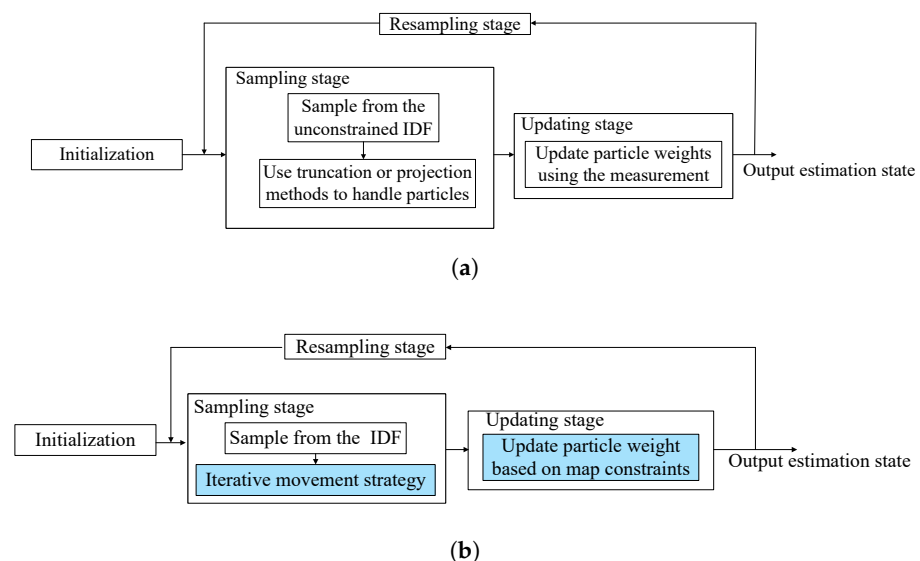


Figure 5. Constrained particle filters: (a) Conventional constrained particle filter [18,19]. (b) Map-constrained particle filtering.

As shown in Figure 5a, at the sampling stage, the conventional constrained particle filter usually draws particles from the unconstrained IDF, and then uses map constraints together with truncation or projection methods to correct particles that do not satisfy the constraints, thereby preventing particles from falling into infeasible regions of the map. Although such methods can improve particle feasibility to some extent by using prior map information, the map is mainly used as a hard constraint at the sampling stage. The proposed method uses the prior map in both the sampling and update stages, as shown in Figure 5b. At the sampling stage, the iterative movement strategy is designed to guide particles toward regions with high measurement likelihood. It uses the measurement residual between the actual LiDAR observation and the particle virtual measurement to construct the fitness function and determine whether each particle should be updated based on its fitness value. In this way, the sampled particles are optimized before weight update, which improves particle effectiveness.

$$\begin{aligned} \text{Conventional PF: } \mathbf{x}_{k,\text{PF}}^{(i)} &\sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}) \xrightarrow{\text{truncation or projection}} \mathbf{x}_{k,\text{C-PF}}^{(i)} \\ \text{MC-PF: } \mathbf{x}_{k,0}^{(i)} &\sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(i)}) \xrightarrow{\text{Equations (18)–(20)}} \mathbf{x}_{k,\text{MC-PF}}^{(i)} \end{aligned} \quad (22)$$

where $\mathbf{x}_{k,\text{C-PF}}^{(i)}$ denotes the constrained particle obtained through truncation or projection, and $\mathbf{x}_{k,0}^{(i)}$ denotes the initial particle before iterative movement.

During the update stage, the conventional PF updates particle weights solely based on the LiDAR measurement likelihood. Building on this update, MC-PF further incorporates map information and the constructed criterion function to correct the particle weights.

$$\begin{aligned} \omega_{k,\text{PF}}^{(i)} &= \omega_{k-1}^{(i)} p(z_k^{\text{LiDAR}} | \mathbf{x}_k^{(i)}, M), \\ \omega_{k,\text{MC-PF}}^{(i)} &= \underbrace{\omega_{k-1}^{(i)} p(z_k^{\text{LiDAR}} | \mathbf{x}_k^{(i)}, M)}_{\text{conventional PF update}} \underbrace{p(z_k^{\text{map}} | \mathbf{x}_k^{(i)}, M) \delta(E_k^{(i)})}_{\text{proposed map-based correction}}, \end{aligned} \quad (23)$$

As shown in the update stage of Figure 5a, after calculating the observation likelihood, conventional methods typically update particle weights according to the consistency between the particles and the obstacles represented in the map. They mainly compare the actual sensor measurements with the virtual measurements of the particles to evaluate particle–environment consistency. However, when no obstacle is detected within the effective sensing range, the corresponding free-space information is generally not fully exploited. In contrast, the proposed MC-PF further utilizes non-detection information during the particle weight update stage, as shown in Figure 5b. For valid measurements in which an obstacle is detected, the particle weight is updated using the matching error between the actual measurement and the particle’s virtual measurement. When no obstacle is detected, if a particle’s virtual measurement predicts an obstacle, it contradicts the current observation and the particle weight is therefore reduced. This strategy enables a more reasonable allocation of particle weights, thereby improving particle effectiveness and localization accuracy.

4.3. Complexity Analysis

The computational complexity of MC-PF is analyzed for one filtering time step. It mainly depends on the particle number N , the number of LiDAR beams m , the number of iterations N_t , and the state dimension Λ . In the prediction stage, sampling N particles from the motion model requires $O(N\Lambda)$ operations. During sampling optimization, each particle updates its velocity and position, while its fitness is evaluated by matching the measured

LiDAR scan with the particle virtual scan. Since this evaluation involves m LiDAR beams and is repeated for N_t iterations, the corresponding complexity is $O(NN_t(m + \Lambda))$.

In the weight-update stage, the LiDAR likelihood, map constraint, and non-detection measurement constraint are evaluated for each particle. With a precomputed occupancy map and distance map, each map query can be performed in $O(1)$ time. Therefore, the beam-wise LiDAR–map matching dominates this stage, resulting in a complexity of $O(Nm)$. The subsequent weight normalization and systematic resampling both require $O(N)$ operations.

Therefore, the complete per-step computational complexity of MC-PF can be written as $O(N\Lambda + NN_t(m + \Lambda) + Nm + N)$. Because the state dimension Λ , the iteration number N_t , and the number of LiDAR beams m are fixed in the experiments, the running time mainly increases with the particle number N . Thus, the per-step runtime is approximately linear with respect to N under the fixed experimental settings. A detailed comparison of the algorithm complexity is shown in Table 1.

Table 1. Comparison of algorithm computational complexity.

Algorithm	Computational Complexity
AMCL [19]	$O(N_{AMCL}m)$
EER-PF [27]	$O(Nm + N\Lambda^2 + \Lambda^3)$
PSO-PF [28]	$O(NN_t(m + \Lambda) + Nm)$
MC-PF (Algorithm 2)	$O(NN_t(m + \Lambda) + Nm)$

5. Simulation and Analysis

In this section, numerical simulations are conducted in MATLAB R2020b to verify the effectiveness of the proposed algorithm for indoor robot localization.

5.1. Simulation Parameters

Numerical random-walk simulation experiments are conducted in MATLAB R2020b. The experiment is conducted on a computer platform configured with Windows 11, featuring a 4-core i5 CPU and 32 GB of memory. The computation time is recorded to evaluate the computational cost of the proposed algorithm under different parameter settings. The robot is set to move at a constant speed in a two-dimensional planar environment of $50\text{ m} \times 50\text{ m}$ in size. Its initial position is set at (10, 10), and the initial heading angle is 0° . During the experiment, the robot moves continuously for 50 steps, and its movement trajectory and state evolution are recorded in real time for the subsequent performance evaluation of the algorithm. The relevant simulation parameters and experimental configurations are shown in Table 2.

Table 2. Experimental parameter settings.

Parameter	Value
Process noise variance	0.1 m^2
Angle noise variance	3°
Measurement noise variance	0.5 m^2
Number of particles	3000
Resampling threshold N_{th}	0.4
Effective measurement distance	10 m
LiDAR field of view	180°
Angular resolution $\Delta\theta$	1.5°
Number of LiDAR beams m	120
Penalty coefficient λ	0.5
Number of iterations N_t	3

5.2. Simulation in Map-Constrained Scenarios

To verify the effectiveness of the proposed MC-PF algorithm (Algorithm 2), a simplified experimental scenario including static obstacles is constructed, as shown in Figure 6a. AMCL, a particle filter based on error-elliptic resampling (EER-PF) [27] and a particle filter combined with the PSO method (PSO-PF) [28], are selected as comparison methods. The robot performs a random walk in the scene, and its motion trajectory is shown in Figure 6b. Starting from the initial position, it moves for 50 steps to generate the true trajectory. In the initialization stage, each algorithm generates the same number of particles randomly. Then, state prediction and update are performed at each time step to obtain the corresponding estimated trajectory. By calculating the Euclidean distance error between the estimated position and the true position, and drawing the error curve over time, the localization accuracy of different algorithms is compared and analyzed.

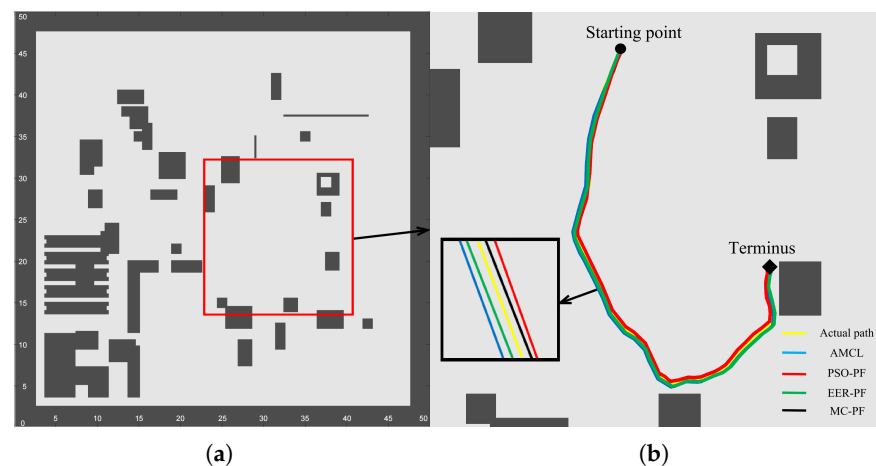


Figure 6. (a) Simulated scene structure 1. (b) Robot trajectory.

Figure 7 shows the error comparison results of different localization algorithms in the random-walking experiment. Compared with traditional AMCL, both PSO-PF and EER-PF significantly reduce localization errors, indicating that error ellipses or the prior map can effectively improve localization performance. The MC-PF algorithm (Algorithm 2) proposed in this paper maintains the lowest and most stable localization errors during the whole motion process, and its overall accuracy is better than that of the comparison methods. The average localization errors of the four algorithms are 0.1181 m, 0.0528 m, 0.0533 m and 0.0273 m, respectively, verifying the effectiveness of the proposed method.

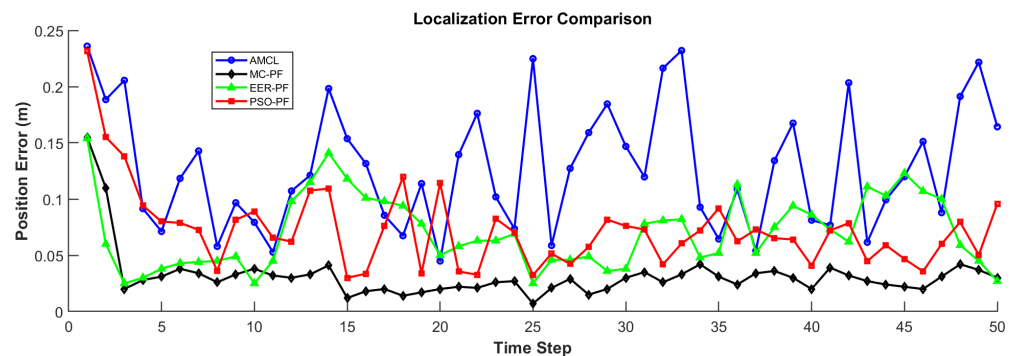


Figure 7. Comparison of localization errors among the four algorithms in Scenario 1.

To further verify the advantage of MC-PF in maintaining effective particles, the ESS values of different algorithms in the random walking experiment are counted. The results are shown in Figure 8. The ESS of AMCL remains at a relatively low level throughout most

of the motion process, with an average value of 1014. The corresponding average ESS values for EER-PF and PSO-PF are 1238 and 1267, respectively. In contrast, MC-PF maintains a higher ESS throughout the entire motion process, with an average value of 1628. This result indicates that MC-PF better preserves particle diversity and alleviates particle degeneracy.

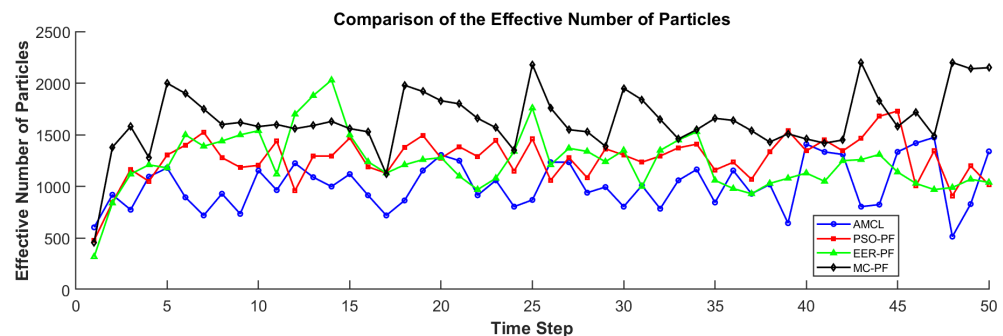


Figure 8. Comparison of ESS values among the four algorithms in Scenario 1.

To evaluate the robustness of the localization algorithms under challenging measurement conditions, a non-Gaussian noise scenario is constructed using a Gaussian mixture model combined with random missed detections. AMCL produces the largest average localization error of 0.1465 m and exhibits significant fluctuations, indicating its high sensitivity to non-Gaussian measurement noise. EER-PF and PSO-PF achieve lower average localization errors of 0.0826 m and 0.0757 m, respectively, but remain affected by occasional abnormal measurements. In contrast, MC-PF achieves the lowest average localization error of 0.0548 m and maintains relatively stable performance. In terms of effective sample size, the average ESS values of AMCL, EER-PF, PSO-PF, and MC-PF are 943, 1224, 1273, and 1488, respectively. MC-PF maintains the largest number of effective particles, indicating that the proposed method can better preserve particle diversity under non-Gaussian disturbances. These results demonstrate that MC-PF provides superior robustness and localization accuracy under non-Gaussian measurement noise, as shown in Figures 9 and 10.

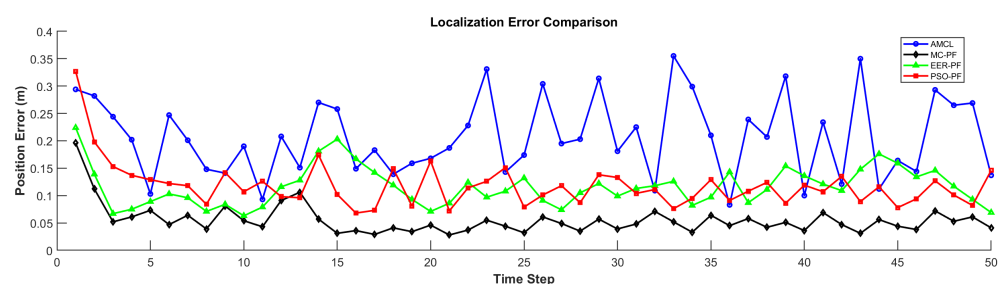


Figure 9. Comparison of localization errors among the four algorithms.

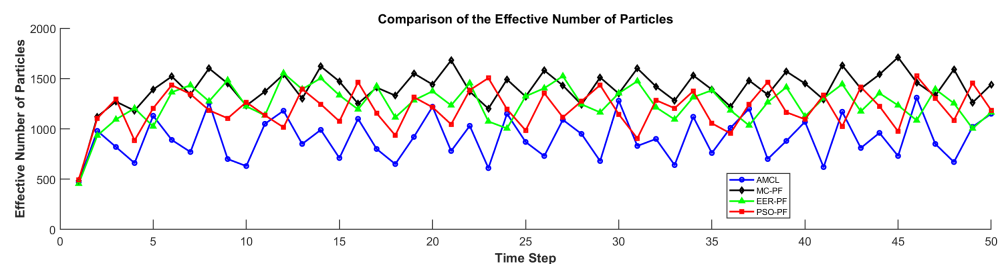


Figure 10. Comparison of ESS values among the four algorithms.

In addition, we construct an additional indoor scene with multiple turns, narrow passages, and symmetric corridors to further evaluate the performance of the proposed method under a complex map topology, as shown in Figure 11a. The robot trajectory, presented in Figure 11b, passes through several similar regions that may produce ambiguous LiDAR observations and multiple particle clusters. Therefore, this scenario is more challenging than the relatively open scenario and can be used to evaluate the localization performance of different algorithms.

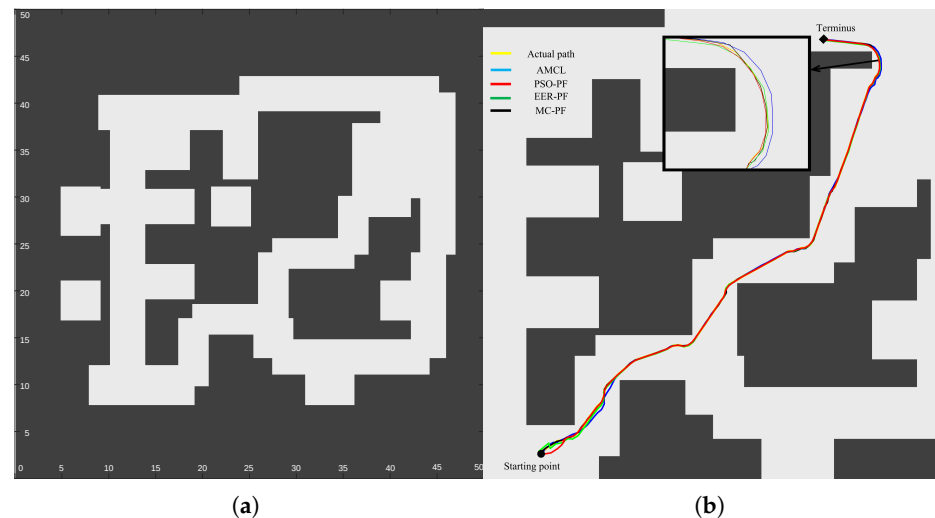


Figure 11. (a) Simulated scene structure 2. (b) Robot trajectory.

The localization errors and ESS of AMCL, PSO-PF, EER-PF and MC-PF are compared in Figures 12 and 13. Due to the presence of multiple turns and symmetric corridors, the localization performance of all four algorithms decreases to varying degrees. AMCL achieves an average localization error of 0.1227 m and an average ESS of 956. The average localization errors of PSO-PF and EER-PF are 0.0734 m and 0.068 m, while their average ESS values are 1053 and 1104, respectively. In comparison, MC-PF achieves the lowest average localization error of 0.0352 m and maintains the highest average ESS of 1439. These results indicate that the proposed method can retain more effective particles and reduce localization errors in environments containing repeated geometric structures.

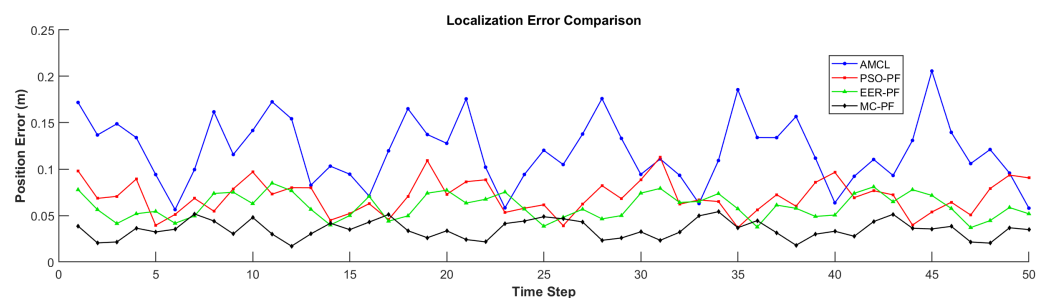


Figure 12. Comparison of localization errors among the four algorithms in Scenario 2.

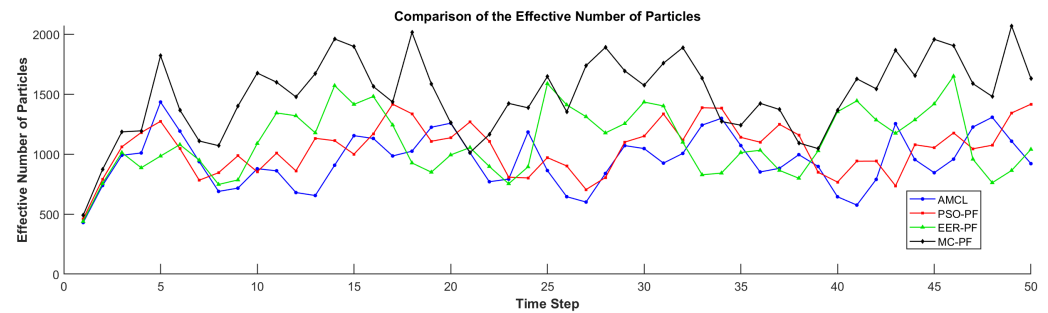


Figure 13. Comparison of ESS values among the four algorithms in Scenario 2.

5.3. Simulation Based on a Realistic Indoor Map

Although the preceding experiments verify the basic effectiveness of the proposed algorithm in relatively simple scenarios, actual indoor environments usually have more complex structural features, which bring greater challenges to localization. To further evaluate the applicability and stability of the algorithm in complex environments, a simulated indoor scenario with more complex structures is constructed based on the Intel Laboratory map, as shown in Figure 14a. The robot motion trajectory is shown in Figure 14b. By conducting simulation experiments in this scenario, the effectiveness of the proposed algorithm can be evaluated more comprehensively.

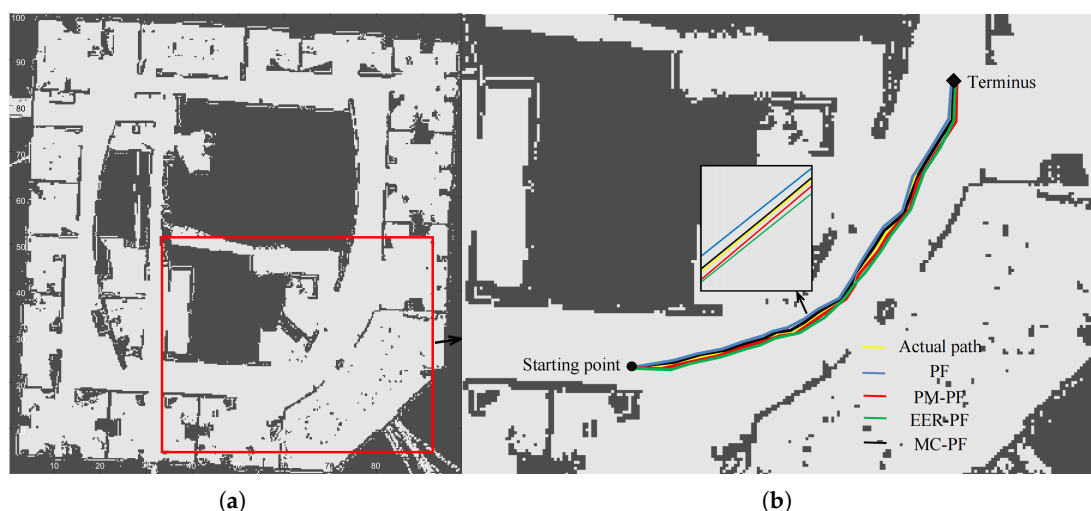


Figure 14. (a) Simulated scene based on the Intel Laboratory map. (b) Robot trajectory.

In complex indoor scenarios, the localization errors and the ESS of the four algorithms are compared and analyzed, as shown in Figures 15 and 16. The results show that under the interference of multiple obstacles and symmetrical narrow channels, the localization performance of each algorithm decreases. The average localization error of AMCL is 0.1263 m, and the ESS is 971. The localization errors of EER-PF and PSO-PF are 0.0851 m and 0.0828 m, respectively, and the corresponding ESS values are 1157 and 1183, respectively. The MC-PF algorithm (Algorithm 2) maintains the lowest average localization error of 0.0398 m throughout the experiment and a relatively high average ESS of 1386. The results show that the proposed method remains stable in complex environments, and it can effectively suppress particle degeneracy and improve state estimation accuracy.

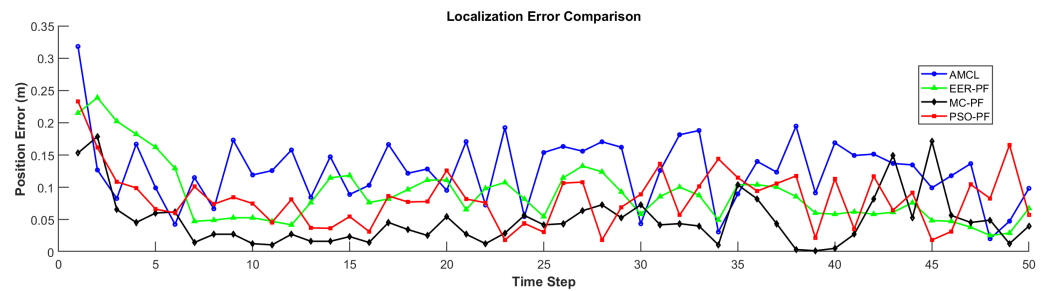


Figure 15. Comparison of localization errors among the four algorithms in Scenario 3.

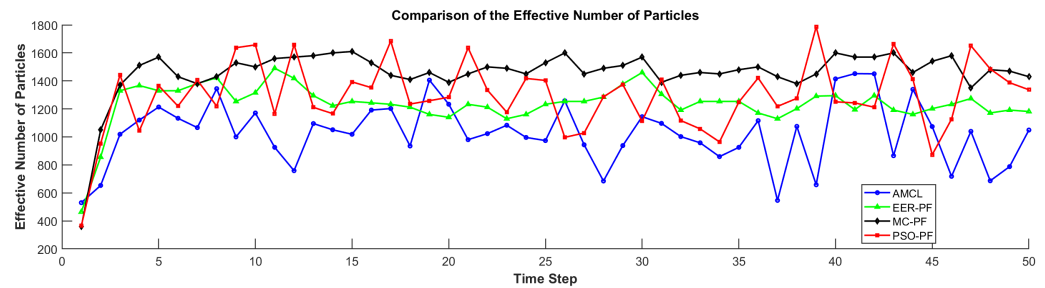


Figure 16. Comparison of ESS values among the four algorithms in Scenario 3.

Figure 17 shows the average localization error of the robot using different numbers of particles. In the experiment, the number of particles is gradually increased from 500 to 6000, and the performance of the algorithms is compared under identical experimental settings. As can be seen from the figure, when the confidence interval variables are the same, more particles will reduce the localization error of the algorithm. When the number of particles reaches 3000, the localization error tends to stabilize, and further increasing the particle number provides only limited accuracy improvement while increasing the computational cost. Therefore, 3000 particles are selected as a compromise between localization accuracy and computational efficiency under the current simulation settings.

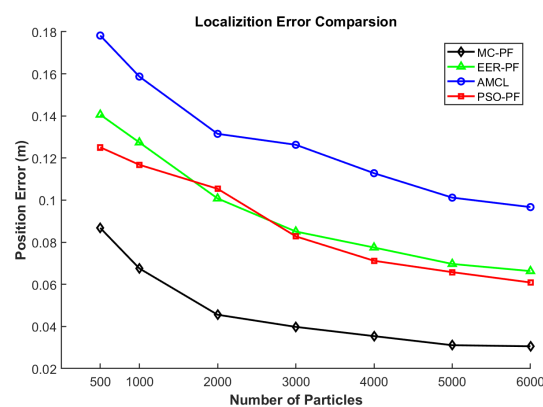


Figure 17. Localization error results of four algorithms with different particle numbers.

Based on the particle number parameters determined in the above experiments, the influence of the effective measurement distance on the localization accuracy is further analyzed. During the experiment, the number of particles and all other experimental parameters remain unchanged, and only the effective measurement distance is changed. Table 3 gives the comparison results of localization errors under different effective measurement distances.

Table 3. Mean localization error for different effective measurement distances.

1	Effective Measurement Distance (m)	Average Localization Error (m)	Average Computation Time per Step (s)
AMCL	1	0.2387	0.186
	3	0.1962	0.224
	5	0.1685	0.231
	8	0.1391	0.318
	10	0.1263	0.384
	12	0.1187	0.397
	15	0.1094	0.472
MC-PF	1	0.09794	0.266
	3	0.06539	0.308
	5	0.05386	0.350
	8	0.04665	0.467
	10	0.03984	0.585
	12	0.03477	0.610
	15	0.03226	0.642

As shown in Table 3, the effective measurement distance has a significant influence on both localization accuracy and computational efficiency. When the effective measurement distance is small, the available environmental information is limited, which reduces the discriminative ability of the particle-weight update and leads to relatively large localization errors. As the effective measurement distance increases, more LiDAR and map information is used for matching, and the localization accuracy gradually improves. However, the computational cost also increases because more map-matching calculations are required. When the effective measurement distance reaches 15 m, the average computation time is 0.642 s per step, which may not satisfy the requirements of high-frequency real-time localization tasks. Therefore, a larger measurement distance is not always preferable in practical applications. Considering the trade-off between localization accuracy and computational efficiency, an effective measurement distance in the range of 8–12 m is recommended in this study.

To further evaluate the long-term localization stability of different methods, we compare the cumulative drift over 50 motion steps in three representative scenarios. The cumulative drift reflects the accumulated deviation tendency of each localization method during continuous robot motion. As shown in Figure 18a–c, the proposed method consistently produces a smaller cumulative drift than the other comparison methods in all three scenarios. Across the three scenarios, the proposed method consistently achieves lower cumulative drift, demonstrating superior localization stability and robustness against error accumulation in different indoor environments. Overall, the results show that the proposed method can better limit localization error accumulation during continuous movement.

To reduce the influence of random noise and further evaluate the repeatability of the different localization algorithms, 50 independent Monte Carlo experiments are conducted in each of the three representative scenarios under the same parameter settings. For each experiment, the mean localization error over the entire robot trajectory is calculated. Figure 19a–c presents the mean localization errors obtained by AMCL, EER-PF, PSO-PF, and MC-PF in the 50 experiments, while Table 4 summarizes the corresponding mean errors and standard deviations. Compared with the other methods, MC-PF consistently achieves lower mean localization errors with relatively small fluctuations in all three scenarios, demonstrating its localization accuracy and repeatability under different random noise realizations.

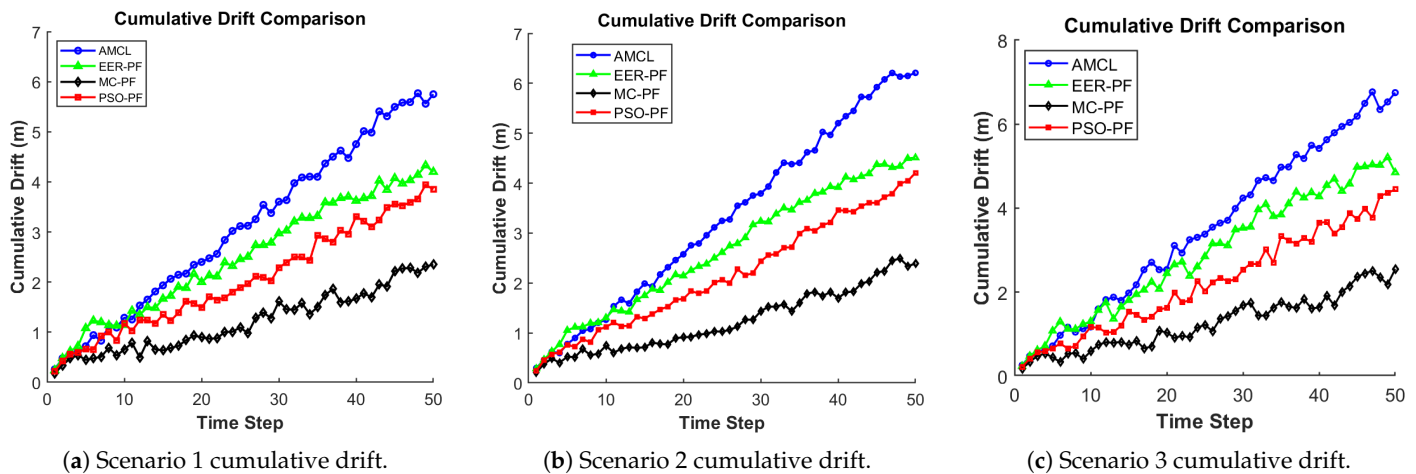


Figure 18. Cumulative drift comparison of the four algorithms in three scenarios.

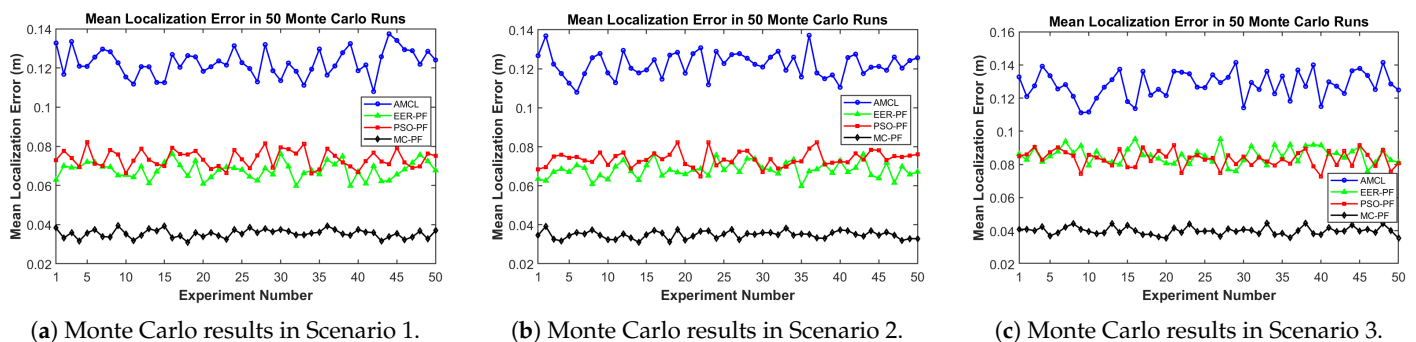


Figure 19. Comparison of mean localization errors among the four algorithms over 50 Monte Carlo experiments in three scenarios.

Table 4. Mean localization errors and standard deviations over 50 Monte Carlo experiments in three scenarios.

Scenario	AMCL	EER-PF	PSO-PF	MC-PF
Scenario 1	0.1184 ± 0.0065	0.0528 ± 0.0033	0.0530 ± 0.0031	0.0277 ± 0.0017
Scenario 2	0.1229 ± 0.0063	0.0683 ± 0.0049	0.0738 ± 0.0037	0.0346 ± 0.0020
Scenario 3	0.1281 ± 0.0082	0.0854 ± 0.0053	0.0835 ± 0.0048	0.0417 ± 0.0025

6. Conclusions

This paper proposes a map-constrained particle filtering method for indoor robot localization under accumulated motion errors. The proposed MC-PF includes two main improvements. First, a map-constrained weight update model jointly exploits obstacle-detection measurements and the free-space information implied by obstacle non-detection. Second, a limited iterative particle-movement strategy is designed to guide sampled particles toward high-likelihood regions while avoiding excessive concentration.

Simulation experiments under different map structures, non-Gaussian noise, and a realistic indoor map show that MC-PF achieves lower localization errors and maintains more effective particles than AMCL, EER-PF, and PSO-PF. In the realistic indoor map MC-PF reduces the mean localization error by approximately 68.5%, 53.2%, and 51.9%. Moreover, cumulative-drift comparisons under continuous motion demonstrate that the proposed method effectively suppresses error accumulation and improves long-term localization stability. The results confirm that incorporating both obstacle-detection and obstacle

non-detection information improves localization accuracy and robustness. For practical deployment, the noise covariance should be calibrated using data collected from the target robot, while the simulated noise levels can serve as initial settings. Since the computational load increases with the particle number, LiDAR beams, and iteration number, embedded implementations may use adaptive particle numbers, beam subsampling, and optimized parallel computation. The current MATLAB runtime should be regarded as a relative benchmark, as limited computational resources, sensor synchronization errors, dynamic obstacles, and map inaccuracies may introduce additional challenges on real robots. Future work will validate the proposed method on an embedded robotic platform.

Author Contributions: Conceptualization, C.H.; methodology, C.H.; software, Q.G.; validation, Q.G. and R.J.; formal analysis, Q.G. and R.J.; writing—original draft preparation, Q.G.; writing—review and editing, C.H. and Q.G.; visualization, R.J.; supervision, C.H. and R.J.; project administration, C.H. and R.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China (Grant No. 62403295), the Natural Science Basic Research Program of Shaanxi (2025JC-YBQN-903), and the Scientific Research Program of Education Department of Shaanxi Provincial Government (25JK0559).

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Acknowledgments: During the preparation of this article, the authors used ChatGPT-5.5 solely for language polishing. The authors reviewed and edited the output and take full responsibility for the content of this publication.

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

References

1. Zhu, Y.; Fang, X.; Li, C. Multi-Sensor Information Fusion for Mobile Robot Indoor-Outdoor Localization: A Zonotopic Set-Membership Estimation Approach. *Electronics* **2025**, *14*, 867.
2. Liu, X.; Gong, G.; Hu, X.; Shang, G.; Zhu, H. Cognitive Enhancement of Robot Path Planning and Environmental Perception Based on Gmapping Algorithm Optimization. *Electronics* **2024**, *13*, 818.
3. Alqobali, R.; Alnasser, R.; Rashidi, A.; Alshmrani, M.; Alhmiedat, T. A Real-Time Semantic Map Production System for Indoor Robot Navigation. *Sensors* **2024**, *24*, 6691.
4. Liu, J.; Luo, J. PLM-SLAM: Enhanced Visual SLAM for Mobile Robots in Indoor Dynamic Scenes Leveraging Point-Line Features and Manhattan World Model. *Electronics* **2024**, *13*, 4592.
5. Liu, T.; Li, B. Tightly coupled integration of GNSS/UWB/VIO for reliable and seamless positioning. *IEEE Trans. Intell. Transp. Syst.* **2023**, *25*, 2116–2128.
6. Stefanoni, M.; Kovács, I.; Sarcevic, P.; Odry, Á. A Survey on the Main Techniques Adopted in Indoor and Outdoor Localization. *Electronics* **2025**, *14*, 2069.
7. Shan, D.; Su, J.; Wang, X.; Liu, Y.; Zhou, T.; Wu, Z. VID-SLAM: Robust Pose Estimation with RGBD-Inertial Input for Indoor Robotic Localization. *Electronics* **2024**, *13*, 318.
8. Kang, S.; Lee, H. Robust robot localization with visually adaptive consensus filters in dynamic corridor environments. *Eng. Sci. Technol.* **2025**, *64*, 101998.
9. Yang, X.; Huang, X.; Zhang, Y. Multi-Sensor fusion and semantic map-based particle filtering for robust indoor localization. *Measurement* **2025**, *242*, 115874.
10. Ai, Mengchi.; Hokmabadi, Ilyar Asl Sabbaghian.; Zhao, Xuan. A Novel Smartphone PDR Framework Based on Map-Aided Adaptive Particle Filter with a Reduced State Space. *ISPRS Int. J.-Geo-Inf.* **2025**, *14*, 476.
11. Liu, A.; Wang, J.; Lin, S.; Kong, X. A Dynamic UKF-Based UWB/Wheel Odometry Tightly Coupled Approach for Indoor Positioning. *Electronics* **2024**, *13*, 1518.
12. Wang, M.; Lian, Z.; Núñez-Andrés, M.A.; Wang, P.; Tian, Y.; Yue, Z.; Gu, L. Robot Localization Method Based on Multi-Sensor Fusion in Low-Light Environment. *Electronics* **2024**, *13*, 4346.
13. Yin, Z.; Yang, J.; Ma, Y. A robust adaptive extended kalman filter based on an improved measurement noise covariance matrix for the monitoring and isolation of abnormal disturbances in GNSS/INS vehicle navigation. *Remote Sens.* **2023**, *15*, 4125.

14. Yuan, S.; Wu, J.; Luan, F. Improvement of strong tracking UKF-SLAM approach using three-position ultrasonic detection. *Robot. Auton. Syst.* **2023**, *159*, 104305.
15. Durodié, Y.; Decoster, T.; Van Herbruggen, B. A UWB-Ego-Motion Particle Filter for Indoor Pose Estimation of a Ground Robot Using a Moving Horizon Hypothesis. *Sensors* **2024**, *24*, 2164.
16. Xia, Y.; Huang, Y.; Qin, H.; Shi, Y. Monte Carlo Localization Based on Off-Line Feature Matching and Improved Particle Swarm Optimization for Mobile Robots. *Intell. Serv. Robot.* **2024**, *17*, 777–791.
17. Zhu, H.; Luo, Q. Indoor localization of mobile robots based on the fusion of an improved AMCL algorithm and a collision algorithm. *IEEE Access* **2024**, *159*, 67199–67208.
18. Harder, D.; Shoushtari, H.; Sternberg, H. Real-Time Map Matching with a Backtracking Particle Filter Using Geospatial Analysis. *Sensors* **2022**, *22*, 3289.
19. Li, G.; Meng, J.; Xie, Y.; Zhang, X.; Huang, Y.; Jiang, L.; Liu, C. Reliable and Fast Localization in Ambiguous Environments Using Ambiguity Grid Map. *Sensors* **2019**, *19*, 3331.
20. Adurthi, N. Scan Matching-Based Particle Filter for LiDAR-Only Localization. *Sensors* **2023**, *23*, 4010.
21. Sasaki, M.; Tsuda, Y.; Matsushita, K. Development of Autonomous Mobile Robot with 3DLidar Self-Localization Function Using Layout Map. *Electronics* **2024**, *13*, 1082.
22. Cheng, Y.; Manduchi, R. PALMS: Plane-Based Accessible Indoor Localization Using Mobile Smartphones. In *Proceedings of the 14th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*; IEEE: New York, NY, USA, 2024; pp. 1–7.
23. Elhousni, M.; Zhang, Z.; Huang, X. LiDAR-OSM-Based Vehicle Localization in GPS-Denied Environments by Using Constrained Particle Filter. *Sensors* **2022**, *22*, 5206.
24. Cai, Y.; Kong, F.; Ren, Y. Occupancy grid mapping without ray-casting for high-resolution LiDAR sensors. *IEEE Trans. Robot.* **2023**, *40*, 172–192.
25. Hu, C.; Liang, Y.; Xu, L. Hybrid sampling-based particle filtering with temporal constraints. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *53*, 1104–1115.
26. Xue, L.; Zhong, Y.; Han, Y.; de la Puente P. Constrained Cubature Particle Filter for Vehicle Navigation. *Sensors* **2024**, *24*, 1228.
27. Xu, C.; Wang, X.; Duan, S. Spatial-temporal constrained particle filter for cooperative target tracking. *J. Netw. Comput. Appl.* **2021**, *176*, 102913.
28. Qibin, Zhang.; Peng, Wang. An improved particle filter for mobile robot localization based on particle swarm optimization. *Expert Syst. Appl.* **2019**, *135*, 181–193.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.