 https://doi.org/10.3311/PPee.21969

Periodica Polytechnica Electrical Engineering and Computer Science, 67(4), pp. 384–393, 2023

Exploring the Use of Particle and Kalman Filters for Obstacle Detection in Mobile Robots

Zoltán Gyenes[1,2*], Ladislau Bölöni[2], Emese Gincsainé Szádeczky-Kardoss[1]

[1] Department of Control Engineering and Information Technology, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Műegyetem rkp. 3., H-1111 Budapest, Hungary
[2] Department of Computer Science, University of Central Florida, 4328 Scorpius St, 32816-2362 Orlando, FL, USA
[*] Corresponding author, e-mail: zgyenes@iit.bme.hu

Received: 31 January 2023, Accepted: 04 April 2023, Published online: 22 May 2023

Abstract

The present study aims to explore the adaptation of estimation methodologies, specifically Particle filters and Kalman filters, for the purpose of determining the position and velocity vector of obstacles within the operational workspace of mobile robots. These algorithms are commonly employed in the motion planning tasks of mobile robots for the estimation of their own position. The proposed methodology utilizes LiDAR sensor data to estimate the position vectors and calculate the velocity vectors of obstacles. Additionally, an uncertainty parameter can be determined using the introduced perception method. The performance of the newly adapted algorithms is evaluated through comparison of the absolute error in position and velocity vector estimations.

Keywords

robotics, state perception, LiDAR sensor

1 Introduction

The motion planning of mobile robots in dynamic environments poses significant challenges. To generate effective evasive maneuvers, mobile robots may utilize internal or external sensors to gather data.

By applying state estimation methodologies to this sensor data, the state of the mobile agent can be estimated at discrete time intervals. There exists a variety of estimation methods that can be employed for this purpose.

The Kalman filter, a classic algorithm in the field of state estimation, was initially developed to tackle the problem of estimating the state of linear systems [1, 2]. Later, the Particle filter, based on Bayesian filtering, was proposed as a solution for nonlinear systems with non-Gaussian distributions [3]. However, despite the effectiveness of these algorithms and their extensions [4, 5] in determining the location of the robot, they were not capable, or they were not focused on providing information about obstacles in the workspace.

The primary objective of this study was to investigate the adaptation of a perception method for the estimation of obstacle position vectors using LiDAR sensor data. To achieve this goal, both Kalman filter and Particle filter methods were employed for state perception. The performance of the proposed methodologies was evaluated through comparison of the results obtained.

The key findings of this paper can be summarized as:

- We propose an adaptation of a state perception method for the estimation of obstacle position and velocity vectors in the workspace of a mobile agent.
- The proposed method leverages both Kalman filter and Particle filter algorithms for state perception of the obstacles using LiDAR simulation data.
- An uncertainty parameter was also introduced, which can be used in the mobile robot's motion planning task.
- A set of experimental evaluations were conducted to compare the performance of the Kalman filter-based perception algorithm with that of the Particle filter-based solution.

Section 2 presents a review of relevant prior research on estimation methods in the field of robotics. Section 3 provides an overview of the Kalman filter and Particle filter algorithms. The measurement method with the introduced LiDAR simulation is presented in Section 4.1 and the proposed perception methods are detailed in Section 4.2,

Cite this article as: Gyenes, Z., Bölöni, L., Gincsainé Szádeczky-Kardoss, E. "Exploring the Use of Particle and Kalman Filters for Obstacle Detection in Mobile Robots", Periodica Polytechnica Electrical Engineering and Computer Science, 67(4), pp. 384–393, 2023. https://doi.org/10.3311/PPee.21969

where the extended Kalman filter-based approach is presented, and Section 4.3, where the Particle filter-based method is described. The performance of these methods is evaluated through a set of experiments and the results are presented in Section 5. Finally, in Section 6, the main contributions and implications of the proposed perception methods are summarized.

2 Previous work

In the field of mobile robotics, the Kalman filter algorithm has been widely utilized for the task of estimating the position and velocity of the mobile agent, also known as self-localization problem [6–8].

In recent years, Bayesian filtering methods have become increasingly prevalent in the field of state estimation for mobile robots, playing a significant role in advancing the capabilities of these systems [9–11].

Using the Extended Kalman Filter (EKF), the state estimation problem can be solved for not only linear but also for nonlinear systems [12]. The EKF algorithm linearizes the nonlinear model at each time step and then applies the Kalman filter to the linearized model, thus enabling the estimation of the state of the system. The EKF has been successfully applied in various domains, particularly in the field of mobile robotics for the localization problem of the agent [7, 13, 14]. It is a widely accepted method in the field of robotics and has been used in various applications and research studies in the last decade [15, 16].

The Unscented Kalman Filter (UKF) can also be used for the localization of the mobile agent considering a nonlinear system. Comparing the EKF and UKF, the UKF can generate a more appropriate solution for the localization problem, by avoiding the linearization step [17–19].

The Particle filter algorithm was first introduced in 1955 [20]. This approach simulated a large number of particles, or "molecules", to estimate the state of a system. In 1993, the Particle filter was re-named as the Bootstrap filter [3], as it was implemented as a recursive Bayesian filter. The core concept of this algorithm is to define a posterior distribution using a set of samples, each with different weights, that are calculated and updated at every sampling time using measurement data. This type of Bayesian filter has been found to achieve better results for nonlinear systems compared to the unscented and extended Kalman filters (EKF and UKF), even when using limited particles [21].

The Multi-robot collision avoidance with the localization uncertainty (CALU) algorithm utilizes the Particle filter for addressing the localization problem of mobile robots. The proposed method incorporates the Optimal Reciprocal Collision Avoidance (ORCA) method [22] for generating collision-free motion plans. The primary objective of this approach is to bound the error present in the localization process. The implementation of the algorithm is carried out using the Robot Operating System (ROS) framework.

The Simultaneous Localization and Mapping (SLAM) algorithm consists of two steps: continuously constructing and updating a map of the environment while simultaneously keeping track of an agent or robot's location within it [23–25]. The SLAM algorithm was also used with the combination of the Kalman filter [26] and the Particle filter [26, 27]. At the initial SLAM algorithm, the segmentation of the obstacles from the environment cannot be generated and only the position information of the map can be determined. In our approach, we can calculate also the velocity vectors of the different obstacles that occur in the workspace of the agent.

3 Background

3.1 Kalman filter algorithm

In Section 3.1, the Kalman filter algorithm is presented in a structured manner. The methodology of the algorithm is divided into two key components: the time update and measurement update processes. These two components are crucial in achieving the desired state estimation for the system under consideration. In this paper, we use linear system models in discrete time.

The time update can be defined in Eq. (1):

$$\bar{\boldsymbol{x}}_k = \boldsymbol{A}\hat{\boldsymbol{x}}_{k-1} + \boldsymbol{B}\boldsymbol{u}_{k-1}, \tag{1}$$

where k is the actual discrete time step, $\boldsymbol{A}$ is a square matrix, $\boldsymbol{B}$ is a column vector in the case, when the system has one input, $\bar{\boldsymbol{x}}_k$ is the a priori state estimation, $\hat{\boldsymbol{x}}_{k-1}$ is the previous state prediction using the Kalman filter and $\boldsymbol{u}_{k-1}$ is the control input.

Additionally, at the system equations, the output can be calculated as:

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \left(\boldsymbol{D}\boldsymbol{u}_k\right), \tag{2}$$

where $\boldsymbol{C}$ is a vector or matrix depending on the size of the output, $\boldsymbol{D}$ is a scalar or a vector depending on the number of the outputs.

After that, the error covariance $\boldsymbol{M}_k$ can be calculated:

$$\boldsymbol{M}_k = \boldsymbol{A}\boldsymbol{\Sigma}_{k-1}\boldsymbol{A}^T + \boldsymbol{R}_v, \tag{3}$$

where $\boldsymbol{\Sigma}_{k-1}$ is the variance of the prior estimate and $\boldsymbol{R}_v$ is the covariance matrix of the process noise.

The Kalman gain $\boldsymbol{K}_k$ can be calculated, which minimizes the a posteriori error covariance $\boldsymbol{\Sigma}_k$:

$$\boldsymbol{K}_k = \boldsymbol{M}_k \boldsymbol{C}^T \left(\boldsymbol{C} \boldsymbol{M}_k \boldsymbol{C}^T + \boldsymbol{R}_z \right)^{-1}, \tag{4}$$

where $\boldsymbol{R}_z$ is the covariance matrix of the measurement noise.

The a posteriori error covariance is:

$$\boldsymbol{\Sigma}_k = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{C} \right) \boldsymbol{M}_k. \tag{5}$$

Finally, receiving a measurement from the sensor ($\boldsymbol{y}_k$), the measurement update can be calculated for the estimated state:

$$\boldsymbol{x}_k = \bar{\boldsymbol{x}}_k + \boldsymbol{K}_k \left(\boldsymbol{y}_k - \boldsymbol{C} \bar{\boldsymbol{x}}_k \right). \tag{6}$$

3.2 Particle filter algorithm

In Section 3.2, we provide a detailed description of the steps involved in implementing the Particle filter algorithm. The Particle filter is a method for approximating the posterior distribution of a system using a discrete density. One of the key advantages of the Particle filter over other recursive Bayesian filtering methods is its ability to handle nonlinear dynamic models, in addition to linear ones. Additionally, the Particle filter can be applied to systems with non-Gaussian noise.

In the Particle filter algorithm, a weighted set of points ($\boldsymbol{S}_k$) is utilized to approximate the posterior distribution. The set comprises of pairs of $\left\langle \boldsymbol{x}_k^{(i)}, \; w_k^{(i)} \right\rangle$ where $\boldsymbol{x}_k^{(i)}$ represents a possible state of the i^{th} particle at time k, and $w_k^{(i)}$ denotes the likelihood (weight) of that state. The number of particles, represented by N, is not fixed and may change during the iteration. The vector of weights, denoted as $\boldsymbol{w}_k$, is constrained by the requirement that the sum of its elements is equal to 1 $\left(\sum_{i=1}^{N} w_k^{(i)} = 1 \right)$.

Prior to implementing the Particle filter algorithm, it is necessary to initialize the filter. This includes defining the state transition function, the resampling strategy, and the number of particles, denoted as N. The state transition function is mathematically represented in Eq. (7):

$$\boldsymbol{x}_{k+1} = f\left(\boldsymbol{x}_k, \boldsymbol{u}_k \right) + \boldsymbol{\xi}_k, \tag{7}$$

where f is a linear or nonlinear function, $\boldsymbol{u}_k$ means the actual control input and $\boldsymbol{\xi}_k$ is the system noise.

In the initialization step of the Particle filter algorithm, a random set of points, denoted as $\boldsymbol{x}_1^{(i)}$, is generated via sampling from the a priori distribution P_{x0}. The initial weight assigned to each particle is $w_{1|0}^{(i)} = 1/N$, where 1|0 indicates that the current weights are calculated at time step $k = 1$ using information from the prior iteration. In the absence of available measurement data at the beginning of the algorithm, the volume of the weights is set to $1/N$. The steps of the Particle filter algorithm, as outlined in [21], involve the following:

1. Step of measurement update:
   The process of updating the weights of the particles in the Particle filter algorithm is critical in ensuring accurate state estimation. This is accomplished by incorporating sensor measurements (z_k) into the weight update equation for each particle:
   $$w_k^{(i)} = \frac{w_{k-1}^{(i)} p\left(\boldsymbol{z}_k \mid \boldsymbol{x}_k^{(i)} \right)}{\sum_{j=1}^{N} w_{k-1}^{(j)} p\left(\boldsymbol{z}_k \mid \boldsymbol{x}_k^{(j)} \right)}. \tag{8}$$
2. Estimation:
   The approximated state can be calculated:
   $$\hat{\boldsymbol{x}}_k = \sum_{i=1}^{N} w_k^{(i)} \boldsymbol{x}_k^{(i)}. \tag{9}$$
3. Resampling:
   Usually, N samples will be selected from the set of the particles with replacement, considering the weights of the particles. Several resampling methods can be used that were investigated in previous research [28].
4. Time update:
   In this step, prediction can be defined:
   $$\boldsymbol{x}_{k+1}^{(i)} = \boldsymbol{A} \boldsymbol{x}_k^{(i)} + \boldsymbol{B} \boldsymbol{u}_k, \tag{10}$$
   when the system is a linear model. If it is not, then Eq. (7) can be used in the time update step. $w_{k+1|k}^{(i)}$ can be calculated using Eq. (8) with the new sample of particles after the resampling step.

4 Perception method of an obstacle with the Kalman filter and particle filter

4.1 Particle filter algorithm

This study introduces the utilization of a 2D LiDAR sensor in a simulated environment. The sensor boasts a maximum range of 12 meters and a resolution range of [0°, 1°], with a resolution of 0.5°, chosen for the scenario at hand. Fig. 1 depicts the LiDAR measurement data obtained. The position of obstacles in the workspace can be determined through segmentation of the measurement data and identification of the angles associated with said obstacles. The measurement noise is also simulated in the introduced environment. It is assumed that all obstacles present in the workspace possess a disk-like shape. These calculations are performed in the global coordinate system. The position of center point of the obstacle, represented by (p_x, p_y), can

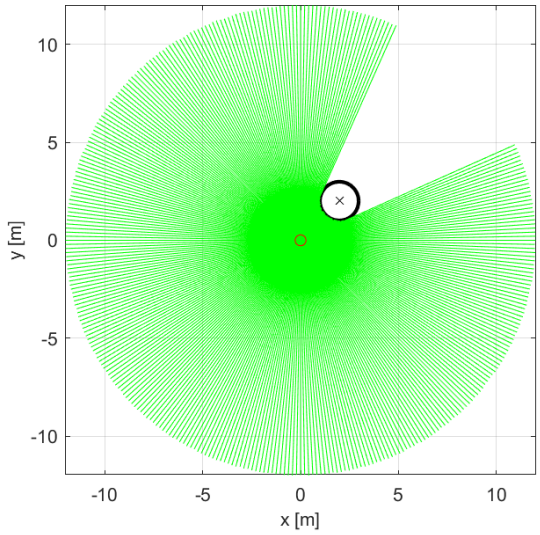

Fig. 1 Measurement data using the LiDAR sensor. The agent is in the origin position (red circle). The onboard sensor distance data can be seen with green lines. There is one obstacle in the workspace, represented by a black circle

then be calculated using the Least Square (LS) method. The methodology of obstacle measurement employed in this study is previously detailed in our prior work [29, 30].

4.2 Perception method with the Kalman filter

Two different methods can be introduced in the Kalman filter solution:

- The perceived state is the position of the obstacle (the velocity vector is calculated from the position): $\hat{\boldsymbol{x}}_{k_2} = \left[\hat{p}_x, \hat{p}_y\right]^T$.
- The perceived state includes the position and the velocity vector of the obstacle: $\hat{\boldsymbol{x}}_{k_4} = \left[\hat{p}_x, \hat{p}_y, \hat{v}_x, \hat{v}_y\right]^T$.

In the first aspect, the time update can be formulated with Eq. (11):

$$\bar{\boldsymbol{x}}_{k_2} = \hat{\boldsymbol{x}}_{k_2-1} + \left(\hat{\boldsymbol{x}}_{k_2-1} - \hat{\boldsymbol{x}}_{k_2-2}\right), \tag{11}$$

where it is assumed that the velocity vector is constant between the time steps. It is important to note that in the aforementioned equation, the control input is not present. This is due to the fact that during the estimation of the obstacle, it is not possible to control its motion, and thus, the position of the obstacle can only be estimated using sensor data acquired from a LiDAR sensor. The state vector in this scenario comprises the p_x and p_y coordinates of the obstacle.

In addition to the position of the obstacle, the time update step of the state vector can also include the velocity vector. This allows for the definition of the time update as:

$$\bar{\boldsymbol{x}}_{k_4} = \boldsymbol{A}\hat{\boldsymbol{x}}_{k_4-1}, \tag{12}$$

where the state vector consists p_x, p_y, v_x, v_y of the obstacle.

Equations (3) to (6) can be used in the same format as they were described in Section 3.1. The size of the covariance matrices has a size of [2 × 2] in the first case and [4 × 4] in the second case. Using the LiDAR sensor, the center point of the obstacle can be calculated as it was introduced in Section 4.1.

4.3 Perception method with the Particle filter

The Particle filter methodology is described here, which utilizes a weighted set of particles to represent the possible states of the system at a given sampling time. These states can be represented by the ***particles*** matrix, with dimensions of 2 rows and $\boldsymbol{N}$ columns, and the vector of weights, denoted by $\boldsymbol{w}$. The prediction of the states of the obstacles is accomplished utilizing measurement data obtained from the LiDAR sensor. The algorithm also accounts for the system and measurement noise. The methodology for executing the algorithm is outlined in Section 3.2, with the main steps being iteratively implemented.

4.3.1 Weight calculation method

Assuming that there are only disk-shaped obstacles in the workspace, the center point can be calculated using the measurement data of the LiDAR sensor. In this case, a measured x center (z_{px}) and measured y center (z_{py}) can be calculated using a *Least square error* estimation considering every point that is measured by the sensor as it was presented in Section 4.1.

Every particle denotes the center point of the obstacle (p_x and p_y). The measurement noise of the distance is denoted by (MN_d). Each particle has the x position ($\hat{z}_{px}(i)$) and the y position ($\hat{z}_{py}(i)$) data.

Firstly, the x position can be considered for the calculation of the weights:

$$w_{p_x}^{(i)} = \frac{1}{\sqrt{2\pi MN_d}} e^{-\frac{\left(z_{px} - \hat{z}_{px}(i)\right)^2}{2MN_d}}. \tag{13}$$

The weights of the x position must be normalized:

$$w_{px}^{(i)} = \frac{w_{px}^{(i)}}{\sum_{j=1}^{N} w_{px}^{(j)}}. \tag{14}$$

In this case, the sum of the weights of particles is equal to 1.

After that, the weights can be calculated considering the y position:

$$w_{py}^{(i)} = \frac{1}{\sqrt{2\pi MN_d}} e^{-\frac{\left(z_{py} - \hat{z}_{py}(i)\right)^2}{2MN_d}}. \tag{15}$$

The weights of y position must be normalized too:

$$w_{py}^{(i)} = \frac{w_{py}^{(i)}}{\sum_{j=1}^{N} w_{py}^{(j)}}. \tag{16}$$

The final weights of the particles can be calculated using the weights of the x and y positions:

$$w^{(i)} = w_{px}^{(i)} \times w_{py}^{(i)}. \tag{17}$$

The final weights must be also normalized:

$$w^{(i)} = \frac{w^{(i)}}{\sum_{j=1}^{N} w^{(j)}}. \tag{18}$$

4.3.2 State estimation method

After determining the weights of each particle, the estimated state of the obstacle, specifically its position, must be calculated. Various methods exist for determining this perceived state, one of which is through the computation of the mean of the particles.

$$\hat{\boldsymbol{S}} = \frac{\sum_{i=1}^{N} \boldsymbol{particles}_k^{(i)}}{N}, \tag{19}$$

where $\hat{\boldsymbol{S}}$ means the perceived state.

An alternative method for determining the perceived state is to utilize the weights of the particles as well. At this case Eq. (9) can be used.

In our perception method, we used the Eq. (9) to estimate the state of the obstacle.

4.3.3 Resampling algorithm and state transition method

We used the systematic resampling algorithm to select the particles at this step.

The state of the dynamic obstacle model consists of two coordinates (p_x, p_y) according to the position. The velocity (v_x, v_y) can be calculated after the position perception. At the state transition model, both the perceived position and the velocity are used. The state transition model can be described as it was introduced in Eq. (10).

At every time step, an uncertainty parameter (α) can be also calculated which can be useful in the motion planning algorithm for the mobile agent:

$$\alpha = \min\left(1, \max\left(std\left(\boldsymbol{particles}_k\right)\right)\right), \tag{20}$$

where *std* means the standard deviation which is calculated for both the x positions and y positions of the obstacles and the highest value will be selected for the uncertainty parameter. The maximum value is saturated to 1. The bigger the value of the uncertainty parameter is, the higher the uncertainty of the accuracy of the actual perception is.

5 Results

In Section 5, a comparison was made between the proposed Kalman filter-based perception methods and the previously introduced Particle filter-based algorithm.

The simulations were done using the following environment:

- Processor: Intel(R) Core(TM) i5-3320M CPU @ 2.60 GHz;
- Operation system: Win10, 64 bites;
- Memory (RAM): 8.00 GB;
- MATLAB 2021a.

5.1 One moving obstacle with constant velocity vector

In Section 5.1, the investigation of a moving obstacle is presented. The initial measurement data with the LiDAR sensor can be seen in Fig. 2, the value of measurement noise (MN_d) was set in this example to 0.05 m.

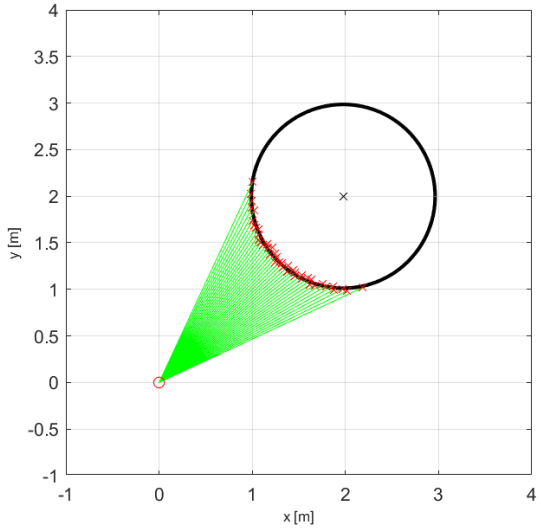

Fig. 2 Measurement with LiDAR sensor at the initial time step. The robot is in the origin, the centre point of the obstacle is in [2;2]. The red x-s are the measured points on the disk-shaped obstacle

In this example, a single moving obstacle was present within the operational space of the agent, possessing a constant linear velocity ([v_x = 0.25 m/s, v_y = 0.1 m/s]). At the Particle filter algorithm, N = 50000 particles were used. The absolute error in the perception of the p_x and p_y positions, as represented in Figs. 3 and 4. The red line illustrates the results obtained from the Particle filter-based solution, the blue line (Kalman filter 1) illustrates the absolute error of the Kalman filter-based method when only the position is included in the state vector, and the black line (Kalman filter 2) represents the results obtained from the Kalman filter-based method when the velocity vector is also included in the state vector. The simulation was conducted for a total of 100 iterations, with a sampling time of 0.1 seconds.

The results obtained from the Kalman filter-based solutions, and the Particle filter-based method were found to be similar. At the beginning of the simulation, the absolute error in position perception of the Particle filter-based solution was observed to be higher, however, after a few time steps, it was able to attain a more accurate solution in the perception. The average error of the p_x position perception was found to be 0.0228 m, for the Particle filter, 0.0245 m for the Kalman filter 1, and 0.0218 m for the Kalman filter 2 solutions. The average absolute errors in the y position were found to be 0.0144 m and 0.0195 meters and 0.0176 m, respectively.

The results of the v_x and v_y velocity perception using the introduced methods are presented in Figs. 5 and 6 respectively. In contrast to the results of the perception of the position vector, a notable difference can be observed between the Particle filter-based solution and the Kalman filter-based solutions. Specifically, the Particle filter-based solution demonstrates superior performance in terms of velocity

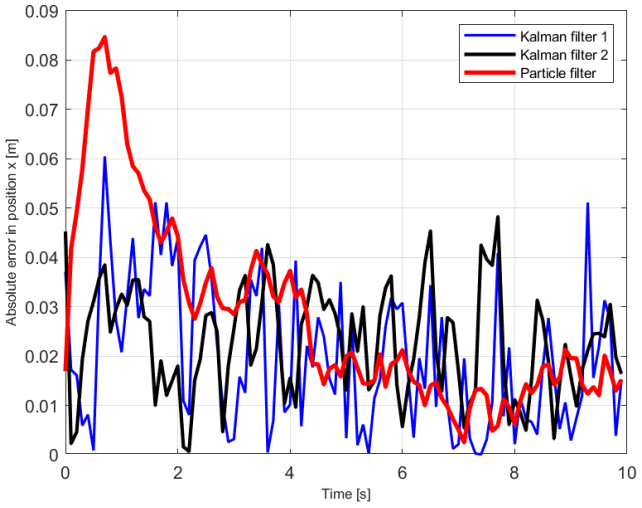

Fig. 3 Absolute error in the perception of the x position of the obstacle

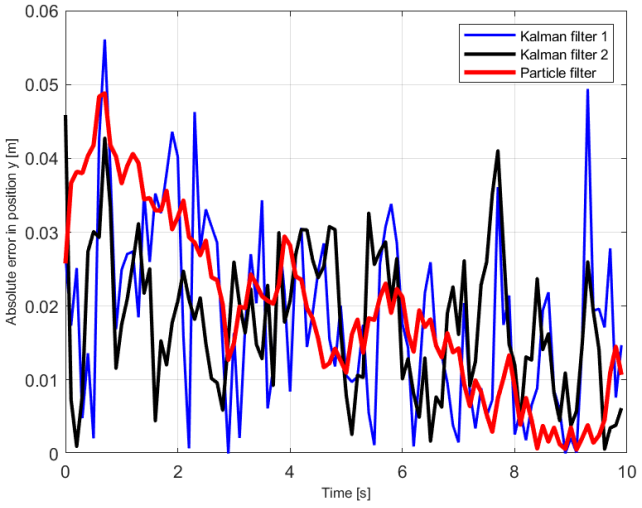

Fig. 4 Absolute error in the perception of the y position of the obstacle

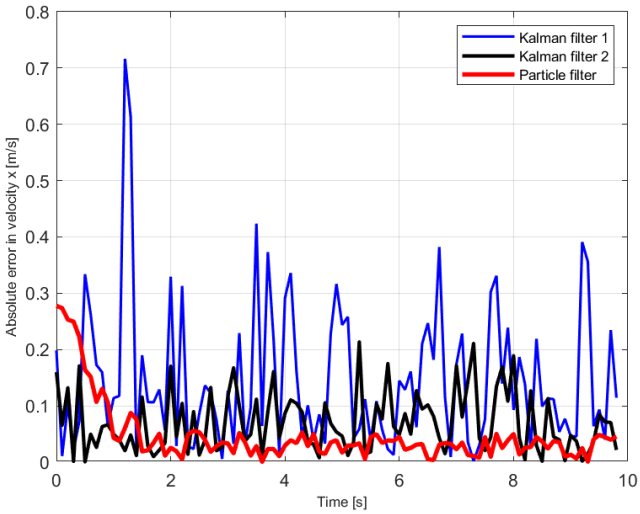

Fig. 5 Absolute error in perception of the x velocity (v_x) of the obstacle

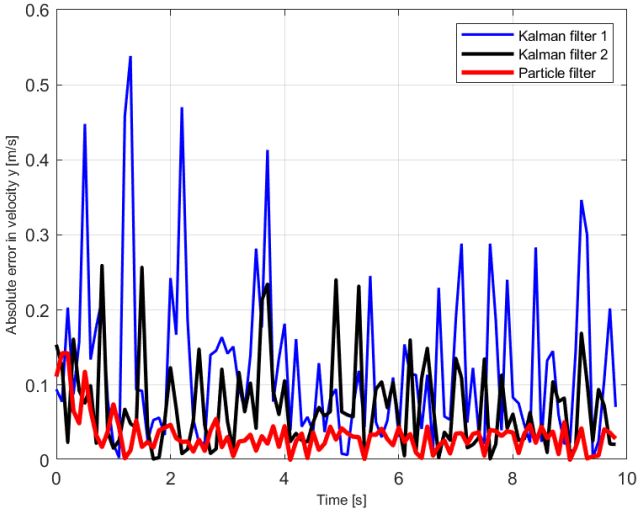

Fig. 6 Absolute error in perception of the y velocity (v_y) of the obstacle

perception, with an average absolute error of 0.0435 m/s for v_x and 0.0292 m/s for v_y. In comparison, the Kalman filter-based solutions yield significantly higher absolute errors of 0.1914 m/s and 0.1523 m/s at Kalman filter 1 method for v_x and v_y respectively. The Kalman filter 2 has an absolute error of 0.0818 m/s and 0.0905 m/s respectively. This suggests that the Particle filter-based method may be a more effective solution for velocity perception in this context.

The computational efficiency of the proposed Kalman filter-based perception methods were compared with that of the Particle filter-based algorithm using varying numbers of particles. The results, as illustrated in Table 1, demonstrate that as the number of particles increases, the running time also increases. When the number of particles is 100000, the running time exceeds 0.1 seconds, making it unable to provide real-time solutions. However, when the number of particles is less than 100000, the algorithm can generate online solutions. Additionally, it was observed that at smaller numbers of particles, the running time does not scale linearly. In this scenario, the results were tested with a particle number of 50000, striking a balance between computational efficiency and perception accuracy.

5.2 One moving obstacle with changing velocity vector

In this scenario, a moving obstacle was present in the workspace of the agent, with a variable linear velocity vector. The results of the evaluation of the absolute error in the p_x position perception can be observed in Fig. 7. As previously observed, it can be inferred that the Particle filter-based solution initially generates a higher average error. However, it is worth noting that when the obstacle changes its velocity vector (at time 10 s), the Particle filter-based perception solution generates a higher average error for a brief period, subsequently reaching a better perception of the position than the Kalman filter-based

Table 1 Running times considering the number of the particles

Number of particles (N)	Running time [s]
100	0.0039
500	0.0043
1000	0.0030
Kalman 1	0.0080
Kalman 2	0.0080
5000	0.0082
10000	0.0127
50000	0.0512
100000	0.1009
500000	0.5294

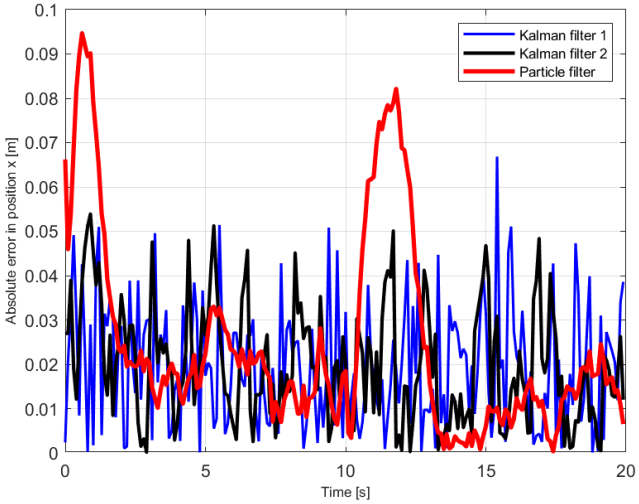

Fig. 7 Absolute error in perception of the x position of the obstacle in the second example

solutions. The Kalman filter-based solutions produce similar average errors throughout the motion of the obstacle.

Fig. 8 illustrates the results of the average absolute error in the v_x velocity perception obtained using the different methods introduced. It is evident that the Particle filter-based solution demonstrates the best performance, with an average error of 0.069 m/s. The Kalman filter 2 solution follows closely, with an average error of 0.0819 m/s. Conversely, the worst performance is observed in the Kalman filter 1 solution, with an average error of 0.2023 m/s.

5.3 One static obstacle and one moving obstacle

In this example, there are two obstacles present in the workspace of the agent, one static and one moving obstacle. The perception of multiple obstacles can be approached in

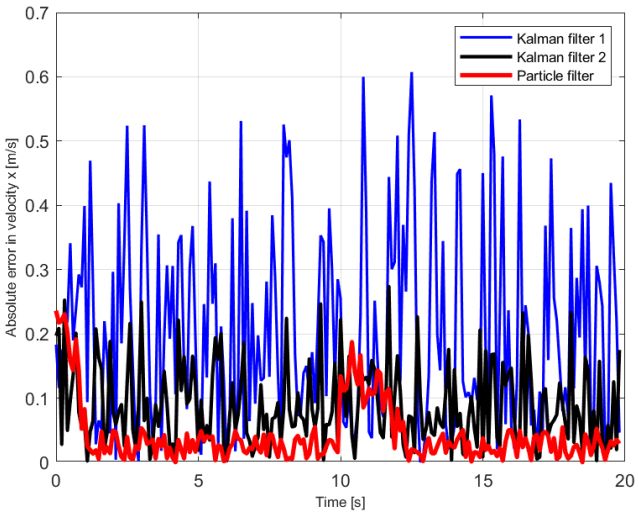

Fig. 8 Absolute error in perception of the x velocity of the obstacle in the second example

a variety of ways. One option is to employ different Particle or Kalman filters for each obstacle present in the agent's workspace. Another solution is to utilize a single Particle filter for the perception of the state of each obstacle. Given the previous results, this example presents only the Particle filter-based solution as the perception of the velocity vector is also deemed important.

As depicted in Fig. 9, the path of the two obstacles in the workspace is represented. The moving obstacle is located behind the static obstacle for a period of time (while it is moving from up to down), during which sensor information from the moving obstacle is not available to the agent. In this scenario, the agent relies on previous estimations to predict the path of the moving obstacle. Once the moving obstacle emerges from the coverage of the static obstacle, the perception can be updated. The uncertainty parameter, as introduced in Eq. (20), can be calculated at each time step during the motion of the obstacle.

In Fig. 10, a comparison is presented between the actual, measured, and estimated positions of the obstacles. It is evident that the position of the static obstacle can be accurately measured and estimated. However, when the moving obstacle is obscured by the static obstacle, the measurement will be significantly incorrect. The Particle filter-based estimation method is capable of utilizing previous estimation results in the absence of new measurement data. As can be seen, the estimation error increases with the duration of the absence of measurement information. Upon receipt of information from the moving obstacle again, the estimation method can promptly predict its state with minimal absolute errors.

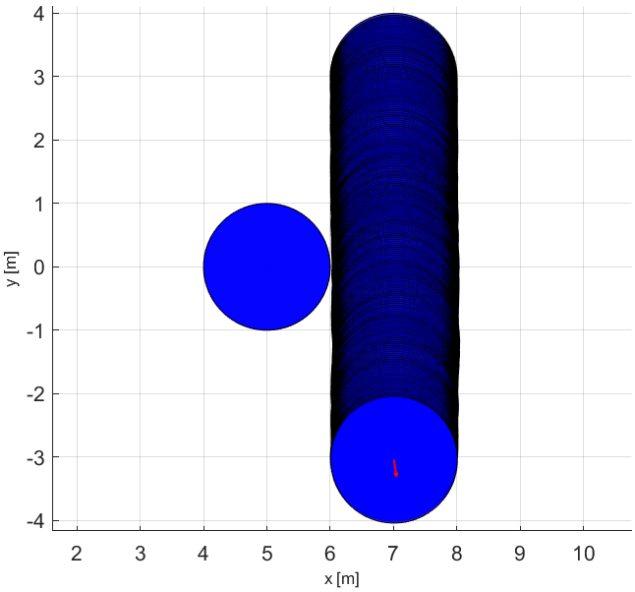

Fig. 9 Two obstacles are in the workspace of the agent. The moving obstacle executes its motion behind the static obstacle

Fig. 11 depicts the temporal evolution of the uncertainty parameter throughout the motion of the static and moving obstacle. The graph illustrates that the uncertainty parameter initially decreases at both obstacles, however, when the moving obstacle is obscured by the static obstacle, there is a significant increase in the uncertainty parameter, reaching a saturation value of 1. At this point, the absence of sensor information from the moving obstacle results in the dispersion of particles. In such instances, it is imperative for the robot to execute safe evasive maneuvers to avoid collisions. Considering the uncertainty parameter of the static obstacle, it is constant after a while because there is always available sensor information in every time step that can be received from the LiDAR sensor. In that

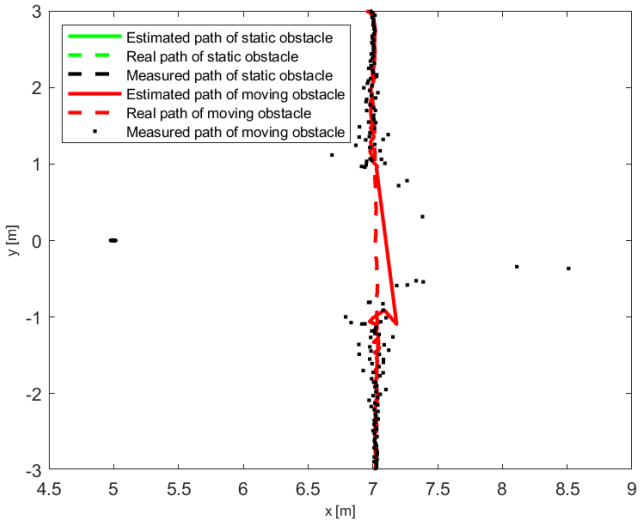

Fig. 10 Comparing the real, measured and estimated path of the obstacles

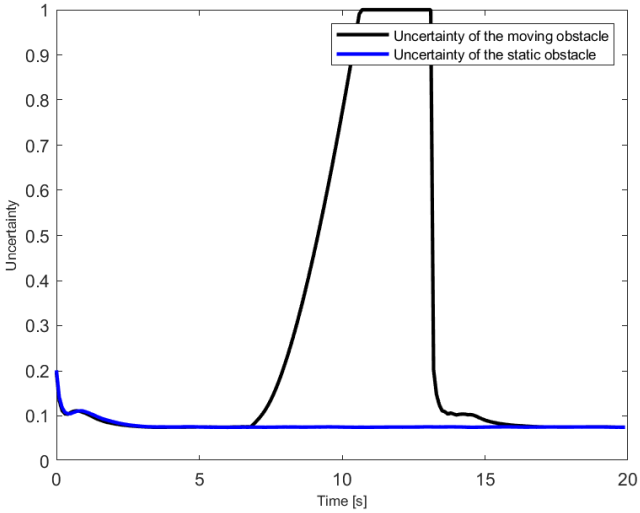

Fig. 11 Uncertainty parameter in every time step for the moving obstacle

case, the uncertainty parameter of the static obstacle is not 0 because of the measurement noise of the LiDAR sensor. The uncertainty parameters can also be utilized in conjunction with cost function-based motion planning algorithms, as described in [31].

6 Conclusion

In this paper, we presented and compared various perception methods that leverage LiDAR measurement data to estimate the position and velocity vectors of obstacles in the workspace of the agent. The results in terms of position perception were found to be comparable across the different methods, however, the Particle filter-based method demonstrated superior performance in velocity perception in comparison to the Kalman filter-based method. Additionally, we proposed a method for calculating the uncertainty parameter based on the current sensor information. As potential avenues for future research, these perception methods could be integrated into motion planning tasks for mobile robots. Furthermore, the introduced uncertainty parameter could be utilized in a cost-function-based velocity selection method for the mobile agent. Additionally, the uncertainty parameter could be also calculated using other techniques like considering the weights of the particles at every sampling time.

Acknowledgement

Supported by the ÚNKP-22-3-II new National Excellence Program of the Ministry for Innovation and Technology from the source of the National R&D and Innovation fund.

References

[1] Kalman, R. E. "A new approach to linear filtering and prediction problems", Journal of Fluids Engineering, 82(1), pp. 35–45, 1960. https://doi.org/10.1115/1.3662552

[2] Kalman, R. E., Bucy, R. S. "New results in linear filtering and prediction theory", Journal of Fluids Engineering, 83(1), pp. 95–108, 1961. https://doi.org/10.1115/1.3658902

[3] Gordon, N. J., Salmond, D. J., Smith, A. F. M. "Novel approach to nonlinear/non-Gaussian Bayesian state estimation", IEEE Proceedings F (Radar and Signal Processing), 140(2), pp. 107–113, 1993. https://doi.org/10.1049/ip-f-2.1993.0015

[4] Claes, D., Hennes, D., Tuyls, K., Meeussen, W. "Collision avoidance under bounded localization uncertainty", In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 2012, pp. 1192–1198. ISBN 978-1-4673-1737-5 https://doi.org/10.1109/IROS.2012.6386125

[5] Blok, P. M., van Boheemen, K., van Evert, F. K., IJsselmuiden, J., Kim, G.-H. "Robot navigation in orchards with localization based on Particle filter and Kalman filter", Computers and Electronics in Agriculture, 157, pp. 261–269, 2019. https://doi.org/10.1016/j.compag.2018.12.046

[6] Roumeliotis, S. I., Bekey, G. A. "Bayesian estimation and Kalman filtering: A unified framework for mobile robot localization", In: Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), San Francisco, CA, USA, 2000, pp. 2985–2992. ISBN 0-7803-5886-4 https://doi.org/10.1109/ROBOT.2000.846481

[7] Ahmad, H., Namerikawa, T. "Extended Kalman filter-based mobile robot localization with intermittent measurements", Systems Science & Control Engineering: An Open Access Journal, 1(1), pp. 113–126, 2013. https://doi.org/10.1080/21642583.2013.864249

[8] Chen, S. Y. "Kalman filter for robot vision: a survey", IEEE Transactions on Industrial Electronics, 59(11), pp. 4409–4420, 2012. https://doi.org/10.1109/TIE.2011.2162714

[9] Doucet, A., Godsill, S., Andrieu, C. "On sequential Monte Carlo sampling methods for Bayesian filtering", Statistics and Computing, 10(3), pp. 197–208, 2000. https://doi.org/10.1023/A:1008935410038

[10] Zhong, Z., Meng, H., Wang, X. "Extended target tracking using an IMM based Rao-Blackwellised unscented Kalman filter", In: 2008 9th International Conference on Signal Processing, Beijing, China, 2008, pp. 2409–2412. ISBN 978-1-4244-2178-7 https://doi.org/10.1109/ICOSP.2008.4697635

[11] Arasaratnam, I., Haykin, S., Elliott, R. J. "Discrete-time nonlinear filtering algorithms using Gauss–Hermite quadrature", Proceedings of the IEEE, 95(5), pp. 953–977, 2007. https://doi.org/10.1109/JPROC.2007.894705

[12] Jetto, L., Longhi, S., Venturini, G. "Development and experimental validation of an adaptive extended Kalman filter for the localization of mobile robots", IEEE Transactions on Robotics and Automation, 15(2), pp. 219–229, 1999. https://doi.org/10.1109/70.760343

[13] Chen, L., Hu, H., McDonald-Maier, K. "EKF based mobile robot localization", In: 2012 Third International Conference on Emerging Security Technologies, Lisbon, Portugal, 2012, pp. 149–154. ISBN 978-1-4673-2448-9 https://doi.org/10.1109/EST.2012.19

[14] Simanek, J., Reinstein, M., Kubelka, V. "Evaluation of the EKF-based estimation architectures for data fusion in mobile robots", IEEE/ASME Transactions on Mechatronics, 20(2), pp. 985–990, 2015. https://doi.org/10.1109/TMECH.2014.2311416

[15] Teslić, L., Škrjanc, I., Klančar, G. "EKF-based localization of a wheeled mobile robot in structured environments", Journal of Intelligent & Robotic Systems, 62(2), pp. 187–203, 2011. https://doi.org/10.1007/s10846-010-9441-8

[16] D'Alfonso, L., Lucia, W., Muraca, P., Pugliese, P. "Mobile robot localization via EKF and UKF: A comparison based on real data", Robotics and Autonomous Systems, 74, pp. 122–127, 2015. https://doi.org/10.1016/j.robot.2015.07.007

[17] Martinelli, F. "Robot localization: comparable performance of EKF and UKF in some interesting indoor settings", In: 2008 16th Mediterranean Conference on Control and Automation, Ajaccio, France, 2008, pp. 499–504. ISBN 978-1-4244-2504-4 https://doi.org/10.1109/MED.2008.4602030

[18] Medewar, P. G., Yadav, M., Patel, H. G. "A comparison between nonlinear estimation based algorithms for mobile robot localizations", In: 2019 IEEE 1st International Conference on Energy, Systems and Information Processing (ICESIP), Chennai, India, 2019, pp. 1–6. ISBN 978-1-7281-0420-1 https://doi.org/10.1109/ICESIP46348.2019.8938237

[19] Ko, N. Y., Kuc, T.-Y. "Fusing range measurements from ultrasonic beacons and a laser range finder for localization of a mobile robot", Sensors, 15(5), pp. 11050–11075, 2015. https://doi.org/10.3390/s150511050

[20] Rosenbluth, M. N., Rosenbluth, A. W. "Monte Carlo calculation of the average extension of molecular chains", The Journal of Chemical Physics, 23(2), pp. 356–359, 1955. https://doi.org/10.1063/1.1741967

[21] Gustafsson, F. "Particle filter theory and practice with positioning applications", IEEE Aerospace and Electronic Systems Magazine, 25(7), pp. 53–82, 2010. https://doi.org/10.1109/MAES.2010.5546308

[22] van den Berg, J., Guy, S. J., Lin, M., Manocha, D. "Reciprocal n-Body Collision Avoidance", In: Pradalier, C., Siegwart, R., Hirzinger, G. (eds.) Robotics Research, Springer, 2011, pp. 3–19. ISBN 978-3-642-19456-6 https://doi.org/10.1007/978-3-642-19457-3_1

[23] Chen, Y., Huang, S., Fitch, R. "Active SLAM for mobile robots with area coverage and obstacle avoidance", IEEE/ASME Transactions on Mechatronics, 25(3), pp. 1182–1192, 2020. https://doi.org/10.1109/TMECH.2019.2963439

[24] Kamburugamuve, S., He, H., Fox, G., Crandall, D. "Cloud-based parallel implementation of SLAM for mobile robots", In: ICC'16: Proceedings of the International Conference on Internet of things and Cloud Computing, Cambridge, UK, 2016, 48. ISBN 978-1-4503-4063-2 https://doi.org/10.1145/2896387.2896433

[25] Tee, Y. K., Han, Y. C. "LiDAR-based 2D SLAM for mobile robot in an indoor environment: A review", In: 2021 International Conference on Green Energy, Computing and Sustainable Technology (GECOST), Miri, Malaysia, 2021, pp. 1–7. ISBN 978-1-6654-1151-6 https://doi.org/10.1109/GECOST52368.2021.9538731

[26] Zhang, F., Li, S., Yuan, S., Sun, E., Zhao, L. "Algorithms analysis of mobile robot SLAM based on Kalman and particle filter", In: 2017 9th International Conference on Modelling, Identification and Control (ICMIC), Kunming, China, 2017, pp. 1050–1055. ISBN 978-1-5090-6576-9 https://doi.org/10.1109/ICMIC.2017.8321612

[27] Törnqvist, D., Schön, T. B., Karlsson, R., Gustafsson, F. "Particle filter SLAM with high dimensional vehicle model", Journal of Intelligent and Robotic Systems, 55(4), pp. 249–266, 2009. https://doi.org/10.1007/s10846-008-9301-y

[28] Gyenes, Z., Szádeczky-Kardoss, E. G. "Particle filter based obstacle's position estimation using lidar measurement data", In: 2021 Workshop on the Advances of Information Technology (WAIT), Budapest, Hungary, 2021, pp. 97–102. ISBN 9789634218449

[29] Gyenes, Z., Szádeczky-Kardoss, E. G. "Particle filter-based perception method for obstacles in dynamic environment of a mobile robot", In: 2021 25th International Conference on Methods and Models in Automation and Robotics (MMAR), Międzyzdroje, Poland, 2021, pp. 97–102. ISBN 978-1-7281-7381-8 https://doi.org/10.1109/MMAR49549.2021.9528442

[30] Gyenes, Z., Bölöni, L., Szádeczky-Kardoss, E. G. "Can Genetic Algorithms Be Used for Real-Time Obstacle Avoidance for LiDAR-Equipped Mobile Robots?", Sensors, 23(6), 3039, 2023. https://doi.org/10.3390/s23063039

[31] Gyenes, Z., Szadeczky-Kardoss, E. G. "Motion planning for mobile robots using the safety velocity obstacles method", In: 2018 19th International Carpathian Control Conference (ICCC), Szilvasvarad, Hungary, 2018, pp. 389–394. ISBN 978-1-5386-4763-9 https://doi.org/10.1109/CarpathianCC.2018.8473397