Map-Aware Particle Filter for Localization

Adrian Rechy Romero[1,3], Paulo V K Borges[1], Andreas Pfrunder[1,2], Alberto Elfes[1]

***Abstract*— This work presents a method to improve vehicle localization by using the information from a prior occupancy grid to bound the possible poses. The method, named Map-Aware Particle Filter, uses a nonlinear approach to map-matching that can be integrated into a particle filter framework for localization. Each particle is re-weighted based on the validity of its current position in the map. In addition, we buffer the trajectory followed by the vehicle and then append it to each particle's pose. We then quantify the overlap between the trajectory and the map's free space. This serves as a measure of each particle's validity given the trajectory and the shape of the map. We evaluated the method by performing experiments with different types of localization sensors: First, (i) we significantly reduced the drift inherent to dead reckoning. By only using wheel odometry and map information we achieved loop closure over a distance of approximately 3 km. We also (ii) increased the accuracy of GPS localization. Finally, (iii) we fused a fragile 2D LiDAR localization with the map information. The resulting system had a higher robustness and managed to close the loop in an outdated map where it had failed before.**

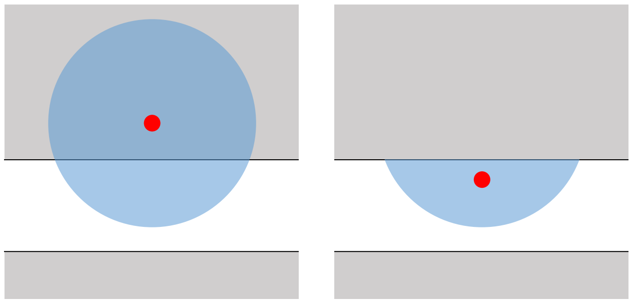
I. Introduction

Sensor-based localization is an essential feature of robot navigation systems. Yet, factors such as sensor uncertainty, sensor fusion errors, and inaccurate prior maps can lead to incorrect pose estimates.

In this paper we propose the enhancement of the vehicle's pose estimation by bounding it to the geometry of a prior map. The spatial map representation chosen is an occupancy grid. Occupancy grids [1] are discretized representations of an environment where each grid cell denotes the probability of an obstacle being present. They are a popular mapping strategy in robotics and can be used for navigation purposes such as path planning and localization.

The example shown in Fig. 1 shows a simple pose correction through the use of an occupancy grid map. Here we exemplify the measurement of an absolute localization sensor such as a GPS. In Fig. 1a the red dot represents the Maximum Likelihood Estimation (MLE) of the vehicle's position. The blue area represents an envelope of alternative possible positions. In this case the MLE corresponds to the mean position within the blue envelope. The white area in the occupancy grid corresponds to the road, the free space. The gray area is the space located behind the obstacles, which are shown in black. In this example the MLE and most of the possible positions are outside of the road area. Since the gray areas are behind the obstacles, it is evident that the vehicle cannot access them.

[1] Autonomous Systems Laboratory, CSIRO, Australia
[2] Autonomous Systems Lab, ETH Zurich
[3] adrian.rechy.romero@gmail.com


(a) Maximum Likelihood Estimation (MLE). The red dot corresponds to the mean and the blue area to the range of possible positions.

(b) Maximum A Posteriori Estimation (MAP). The range of possible positions and the mean now lie within the accessible areas.

Fig. 1: By combining a measurement with the occupancy grid it is possible to obtain a better position estimation that yields a valid mean position and reduced uncertainty. The white space represents the traversable areas. Gray corresponds to unknown space that lies behind the obstacles (black).

A correction can be made where only the accessible areas are taken into consideration. Thus, the MLE and the map information are fused to generate the Maximum A Posteriori Estimate (MAP) as shown by the red dot in Fig. 1b. The new estimate is still loyal to the measurement's distribution and lies within the physically plausible areas of the map.

Relative localization estimations can also benefit from the map information. To illustrate the concept consider the localization problem presented in Fig. 2. Let us assume that the vehicle is known to have followed the trajectory shown in Fig. 2b. By comparing the shape of the trajectory with that of the map (Fig. 2a) it is possible to infer the pose of the vehicle. This means that a relative position tracker (odometry) could be used to perform absolute localization. However, the cummulative drift of relative sensors render them inadequate for such purpose over long distances.

In our proposal we follow the assumption that drift is minimal over short distances. It is then possible to recursively compare the last section of the trajectory with the map. This allows us to use the information of the map's shape to compensate for drift over long distances. However, in order to cope with the nonlinearities of a 2D Occupancy Grid a similarly nonlinear approach is needed.

A. *Map-Aware Particle Filter (MAPF)*

In this paper we propose a new method, the Map-Aware Particle Filter (MAPF). This is an extension of the Particle Filter algorithm [2] that incorporates map information

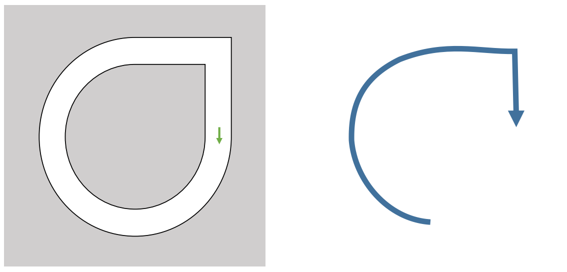

(a) The green arrow corresponds to the current pose. (b) Trajectory tracked by a relative localization sensor.

Fig. 2: By comparing the trajectory and the shape of the map it is possible to infer the pose of the vehicle.

into the pose estimation. Particle filters have an intrinsic capability to handle nonlinear systems. Thus, we can easily incorporate the information of an occupancy grid as an additional "sensor".

We present a method to efficiently obtain a particle's distance to the free space (Section III-A). By penalizing each particle by this distance we promote the likelihood of the estimation lying on the road (Section III-B). Furthermore, we also propose the use of a trajectory buffer that keeps track of the last odometric measurements. This trajectory is then appended to each particle's pose. By quantifying the trajectory's overlap with the free space in the map we can evaluate how well each particle's pose explains the shape of the map (Section III-C).

We evaluated the MAPF on an autonomous ground robot (Fig. 3). The test vehicle is equipped with wheel encoders for odometry, a GPS sensor and an array of 2D LiDARs. A 3D LiDAR is also available but not used for localization.

We evaluated the impact of the MAPF for each of the localization sensors. The results (Section V) show that, in each case, localization benefited from the inclusion of the map information. The pose estimation of the wheel odometry alone increased in accuracy. Through MAPF the vehicle was able to localize itself over a total trajectory of approximately 3 km. Similarly, the accuracy of GPS-based localization was also increased. Furthermore, we fused a fragile 2D LiDAR localization with the map information. The resulting system had a higher robustness and managed to close the loop in an outdated map where it had failed before.

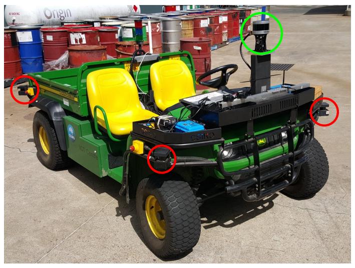


Fig. 3: The test vehicle is a John Deere Electric Gator. The 2D LiDARs are marked with red circles and the 3D Velodyne LiDAR with a green circle.

B. Related Work

Map-matching is the fusion of sensor measurements with map information in order to improve localization [3]. Common applications include improving the accuracy of GPS or dead reckoning position estimates by comparing them to a digital map. Most of the previous research in this area performs map-matching using geometric map representations, where roads are modeled as a collection of lines and curves. The sensor-based estimated pose is then corrected by matching it to the closest point on a line [4] or curve [4], [5], [6] in its vicinity. More advanced methods take into account the connections between road links [7], [8], [9], or the probabilistic representation of sensor measurements [9]. A detailed classification and analysis of different map-matching algorithms on simplified roadmaps can be found in [3]. The same solutions can be used on discretized grid maps by analyzing them a priori and extracting geometric and topological data to create a simplified roadmap [10].

The inclusion of road information into a particle filter implementation was presented in [11], where a function that penalizes a particle's distance to the road is mentioned. However, a method to obtain such distance is not provided, and we propose our own metric in Section III-A. The analysis of the validity of each particle's movement is also presented in [12], [13], [14], [15], where particles are considered invalid if they traverse through a wall or obstacle on each update step. An approach to backtracking particle trajectories has also been proposed [14], [15], where each particle's trajectory history is stored such that their pose estimate can be analyzed and corrected whenever they end on an invalid position. Also, particle filter implementations where each particle is weighted through an analysis of the vehicle's speed and kind of road have been presented [16].

The MAPF performs map-matching with a stochastic spatial representation of the environment. It works directly on the occupancy grid, rather than on a highly simplified geometric model of the environment. As so, it does not require any complex analysis to extract information of the shape of the road. It works in this nonlinear space without making any simplifications other than the ones intrinsic to a 2D occupancy grid (namely, cell decomposition and 2D representation). More so, our approach does not require to keep in memory the previous poses of each of the particles. Instead, a single trajectory, reconstructed from the odometry measurements, is appended to each particle for evaluation.

II. Particle Filter

Although particle filtering is a well-known technique, we include the following explanation for clarity and notation consistency in our contributions in Sec. III. For improved readability, the shorthand expression f_a is used to represent $f_a(a)$ unless specified otherwise.

The Particle Filter is a nonlinear approach to state estimation that aims to represent (potentially complex and nonlinear) noise models through sampling [2]. As an estimation of the Bayesian tracking algorithm it aims to approximate the probability density function (PDF) $f_{x_k|z_{1:k}}$, where x_k

is the state at discrete time k and $z_{1:k}$ corresponds to the collection of measurements of all the sensors up to time k. The standard formulation splits the state estimation in two steps, prediction update and measurement update.

For the prediction step, the state of each particle n is estimated based on a model of the system's state evolution:

$$\tilde{x}_k^n = q_{k-1}(x_{k-1}^n, u_{k-1}, v_{k-1}^n), \quad (1)$$

where u is the control input and v is noise with known PDF f_{v_k} and whose values are generated through Monte-Carlo sampling. Note that the initial state of each particle is initialized through a Monte-Carlo sampling of a given initial state PDF $x_0^n \sim f_{x_0}$.

The measurement update step aims to correct the state estimation through the inclusion of sensor measurements z_k. This is based on a measurement model $h_k(x_k, w_k)$, where w is the measurement noise with known PDF f_w, and $f_w(\cdot) \perp f_v(\cdot) \perp f_{x_0}(\cdot)$.

$$z_k = h_k(x_k, w_k) \quad (2)$$

In this step each particle is assigned a weight β^n based on the likelihood of the sensor measurement given the particle's state.

$$\beta^n = \alpha \cdot f_{z_k|x_k^n}, \quad (3)$$

where α is a normalization factor such that

$$\sum_{n=1}^{N} \beta^n = 1. \quad (4)$$

A resampling step is then performed where each particle has a probability β^n of being resampled. The distribution of the particles is then a representation of $f_{x_k|z_{1:k}}$ [2].

III. Map-aware Particle Filter

We propose enhancing the measurement update section of the Particle Filter algorithm with information from the occupancy grid. Rather than fully eliminating all of the particles that lie outside of the free area, we allow for small deviations. This ensures that the system is robust against small changes in the environment and inaccuracies in the map. Essentially, each particle's weight becomes a function of its proximity to the free space areas in the map. Since calculating this distance is a costly operation, we propose analyzing each cell in the map beforehand such that reweigthing becomes a look-up operation and real-time performance can be maintained.

A. Creation of the Proximity Map

The Proximity Map $\mathbf{P} \in \mathbb{R}^2$ is a grid map where each cell is given a numerical value that represents the amount of cells between itself and the free space in the map (i.e. the road). It is analogous to the "likelihood fields" used for beam endpoint models [17] but, rather than measuring the distance to the obstacle, it measures the distance to the free space. $\mathbf{P}$ can be computed through a dynamic programming approach, starting with the knowledge of the free space cells in the map and expanding it by sequentially exploring the neighboring cells, as detailed in Algorithm 1. In here, c represents an individual cell in the map, $\mathcal{C}$ corresponds to the set of all cells and $\mathcal{F}$ to the set of free-space cells from the occupancy grid. $\mathcal{B}$ is the set of cells that are to be explored. This set is updated on each iteration. The distance from each cell to the free space, measured in number of cells, is then obtained recursively. Say $neigh(c, \mathcal{B})$ is true whenever cell c is a neighbor of any of the cells in $\mathcal{B}$.

Algorithm 1 Algorithm to initialize the proximity map

```
1: $\mathbf{P}(c) = -1 \quad \forall c \in \mathcal{C}$
2: $\mathcal{B} = \mathcal{F}$
3: $\mathbf{P}(c) = 0 \quad \forall c \in \mathcal{B}$
4: $current\_distance = 1$
5: while $|\mathcal{B}| > 0$ do
6:     $current\_distace + +$
7:     $\mathcal{B} = \{c \,|\, c \in \mathcal{C}, neigh(c, \mathcal{B}), \mathbf{P}(c) == -1\}$
8:     $\mathbf{P}(c) = current\_distance \quad \forall c \in \mathcal{B}$
9: end while
```

B. Enhancement of the measurement update step

The distance to the road can be used as an additional sensor input by creating a measurement model that penalizes particles for being away from the road [11]. In order to achieve real-time performance, we propose the use of $\mathbf{P}(c)$ to obtain the number of cells between each particle and the road. The distance d_{road}^n can then be computed with the map resolution r_{pm}. Assuming that the particle n lies in cell c the calculation is as follows

$$d_{road}^n = \mathbf{P}(c) \cdot r_{pm}. \quad (5)$$

An inverse exponential function

$$f_{z_k^{pm}|x_k^n} = \exp\left(-\lambda_{pm} \cdot d_{road}^n\right), \quad (6)$$

is used as a sensor model. The decay rate λ_{pm} is a soft constraint that defines how harshly the particles will be penalized for being far from the road. Let z^s represent the sensor measurements and z^{pm} represent the measurement model of $\mathbf{P}$. Given the conditional independence of both measurements given x_k^n,

$$f_{z_k^s, z_k^{pm}|x_k^n} = f_{z_k^s|x_k^n} \cdot f_{z_k^{pm}|x_k^n}. \quad (7)$$

Therefore,

$$\beta^n = \alpha \cdot f_{z_k^s|x_k^n} \cdot f_{z_k^{pm}|x_k^n}. \quad (8)$$

C. Including trajectory information

Odometric position estimation generally drifts over distance traveled. However, within short distances it can be very accurate. Thus, accuracy can also be expected if we propagate the current pose backwards in time over a short distance. As shown in Fig. 2 this short trajectory could be compared to the shape of the map to improve localization.

For this purpose we propose the use of a relative position buffer that tracks the changes in odometry in order to recreate

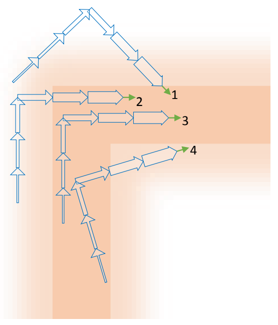


Fig. 4: The odometric trajectory (blue arrows) is appended to each of the particles' poses (green arrows). Each particle is then weighted based on the overlap between the appended trajectory and the Proximity Map $\mathbf{P}$ (orange gradient).

the last section of the trajectory. This trajectory can then be appended to each particle's pose and compared with the Proximity Map. This is illustrated in Fig. 4.

In the example the odometric trajectory consists of a right turn, shown by the blue arrows. This trajectory is then appended to each of the particles' poses (the green arrows) so that it can be compared with the Proximity Map $\mathbf{P}$. The Proximity Map is shown as an orange gradient, where the solid color represents the road and the reduced tonalities represent the lower weights as the distance to the road increases. By measuring the overlap between each particle's appended trajectory and the weight of $\mathbf{P}$ it is possible to evaluate how well each particle can explain the shape of the road via its current pose and estimated trajectory. It is then evident that particle "1" offers a conflicting pose hypothesis given that particular trajectory and map. Meanwhile, the pose of particle "3" offers a harmonious one.

The buffer is bounded by two parameters: the total length to be tracked d_{max} and the trajectory resolution r_{traj} that defines the distance between trajectory points as shown in Fig. 5. More so, as we know that the odometry estimate degrades with distance it is possible to give a higher weight to the most recent ones. This is represented in Fig. 4 by the varying width of the blue arrows. For instance, even though particle "4" has most of its trajectory points within the road, the first two, which have the highest weight, are outside of the road. Therefore, the particle would receive a low weight.

Given a distance $d^j_{vehicle}$ from a given trajectory point j to the vehicle and a function $g(d^j_{vehicle})$ that weights each trajectory point,

$$f_{z^{pm}_k|x^n_k} \propto \sum_{j=0}^{m} g(d^j_{vehicle}) \cdot f_{z^{pm}_k|x^{n,j}_k} \tag{9}$$

A weighted sum was chosen instead of a product to avoid a single invalid trajectory position to cancel the combined weight.[1]

[1]Note that equations (6) and (9) are not formal probability definitions as $\sum f_{z_k|x^n_k} \neq 1$. However, the normalization step in (4) enforces the "sum to 1".

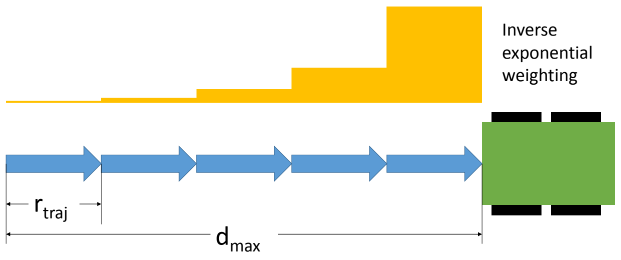


Fig. 5: Each trajectory point is given a weight based on its distance from the vehicle.

Based on experimental evaluation, we found $g(d^j_{vehicle})$ to be adequately modeled through an exponentially decaying function. Given that smaller values of j are closer in time to the current pose,

$$g(d^j_{vehicle}) = \exp\left(-\lambda_{traj} \cdot d^j_{vehicle}\right), \tag{10}$$

where λ_{traj} is a decaying factor that determines how much the buffer positions that lie farther away from the vehicle will be penalized. The weighting of the trajectory points can be visualized on Fig. 5.

Note that if the same relative localization sensor is used for the state evolution (1) and the trajectory evaluation then the odometric information is accounted for twice. One way to avoid this would be to use two different sensors. Alternatively, a single odometric sensor could be used given some constraints. First, d_{max} should be small, following the assumption that the drift is minimal over a short distance. The acceptable values would depend on the characteristics of each sensor. Second, the road width should be large enough to allow for small drifts in the odometry. Given these constraints the map's shape would have a much larger effect on the measurement than the odometric drift.

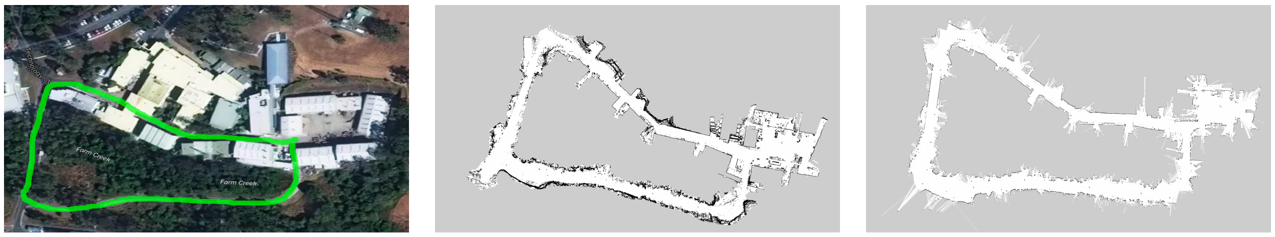
D. *Computational cost*

The time complexity of the measurement state of a traditional particle filter is **at least** linear with the number of particles $O(N)$ (i.e. evaluates every particle). The addition of trajectory information adds a complexity proportional to the number of particles N and the number of trajectory points to be used J. If the lookup operation for a particle's position in the proximity map is done in constant time, then the added complexity of the model is $O(N \cdot J)$. This leads to a total complexity of the measurement step of $O(N(1+J)) = O(N \cdot J)$.

Note that if only the proximity map is used without including trajectory information (i.e. $J = 1$ since we only check the current position) the total complexity of the measurement step remains $O(N)$.

IV. Experiments

Experiments were performed on an autonomous ground robot platform, a John Deere Gator TE electric, shown in Fig. 3. It is equipped with four 2D LiDARs, a GPS, two



(a) (b) (c)

Fig. 6: Satellite view of the test area (a) and the occupancy grids used for the experiments. The first map (b) was built with a 3D laser SLAM algorithm and is used to test accuracy. The second map (c) was built with a 2D laser SLAM algorithm and has loop-closure errors and outdated obstacles. It is used to test robustness.

wheel encoders for odometry and a nodding 3D LiDAR. Due to the different nature of each of these sensors, multiple experiments were designed to evaluate the effect of the Map-aware Particle Filter on each of them.

The software is written in *C++* and built upon the *Robot Operating System (ROS)* framework. In addition, the 2D laser localization method used in this work is an implementation of the *Adaptive Monte-Carlo Localization* [17] which is available through the *amcl* package [18] from *ROS*.

Note that the performance of the method proposed in this paper is highly dependent on the shape of the occupancy grid. For our experiments we used the maps shown in Fig. 6. The first one corresponds to a map built with the nodding 3D laser, a Velodyne PUCK VLP-16 [19]. Mapping was performed by using the 3D SLAM pipeline presented in [20]. In order to get a 2D occupancy grid, an offline ray tracing algorithm was applied while ignoring the obstacles that lied above and below user defined heights. A detailed description of the mapping algorithm is not in the scope of this paper and will be a topic of future publications.

The sensors used for the second map were four 2D Hokuyo UTM-30LX LIDARs, placed on the corners of the vehicle at a height of $0.79\,\mathrm{m}$. These lasers have an angular resolution of $0.25°$, a scan angle of $270°$, run at a frequency of $30\,\mathrm{Hz}$ and have a maximum detection distance of $30\,\mathrm{m}$. The map was built with the *gmapping* algorithm [21], [22] using the *slam_gmapping* wrapper [23].

The map built with the 3D lasers is newer and more accurate than the one built with the 2D laser SLAM. Therefore, the former is used to test accuracy while the latter is used to test robustness.

A. Model selection

The models used for $f_{z_k^{pm}|x_k^n}$ correspond to (6) and (10). Also, the following parameters were chosen empirically based on experimental results: $\lambda_{pm} = 1$, $\lambda_{traj} = 0.1$, $r_{traj} = 5$, $\alpha_{slow} = 0.001$, $\alpha_{fast} = 0.1$. Note that the *amcl* package [18] from *ROS* also has two parameters α_{slow} and α_{fast} which are not the same as the ones used for the MAPF . They are meant to determine when to generate random poses as a recovery measure. In our experiments they were both set to zero (deactivated).

We evaluated the MAPF with and without the incorporation of trajectory information. First we evaluate the performance with the Proximity Map alone. A second test uses a trajectory of length $d_{max} = 30$.

The particle filter implementation was partially based on the *amcl* package and had a minimum and maximum number of particles of 500 and 3000, respectively. Furthermore, for laser localization we used the likelihood field model [17]. The motion model (1) for differential robots present in the *amcl* package was also used. Note that the motion model was solely used for the state evolution of the particle filter and is independent of the trajectory buffer used by the MAPF.

B. Pure Odometric localization

A comparison was made between the pose estimation of the wheel odometry with and without the map correction. The purpose of this experiment was to evaluate the effect of the MAPF for relative pose estimators. The pose estimation from the 2D lasers was used as ground truth as it was the most accurate sensor available.

For this experiment the map from Fig. 6b was used and a trajectory of five consecutive loops was performed around the circuit. This trajectory was meant to prove that loop closure can be achieved through odometry sensing alone with the help of the MAPF. Results for this experiment can be found in Section V-A.

C. GPS bounding

The effect of using the MAPF for the GPS pose estimation was also evaluated. In this case, the objective was to quantify the improvement of an absolute localization method through the inclusion of the map information. Once again, the trajectory built by the 2D lasers was used as ground truth.

The map from Fig. 6b was used as it is more accurate. A total of 10 tests were performed over the same trajectory, which is a single loop around the map. The results for this experiment can be found in Section V-B.

D. Robustness

We also made a comparison between the robustness of our pure odometric approach and that of a 2D laser localization method that used a popular Particle Filter algorithm (*Adaptive Monte-Carlo Localization* [17]). For this, the outdated and less accurate map from Fig. 6c is used to localize the

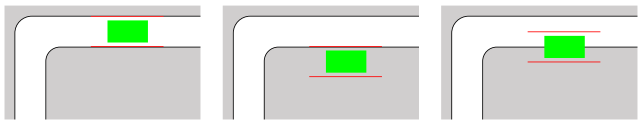

(a) Correct position estimation. (b) Wrong position estimation with high weight. (c) Better position estimation with lower weight.

Fig. 7: The laser localization might give higher weight to an incorrect solution over one that is closer to the correct one. The laser scan is represented by the red lines.

vehicle as it performs several loops around the industrial complex. The pure odometric localization is proven to be more robust, although not as precise as the laser localization. Both sensors are then fused in order to obtain a state estimation that is both robust and precise. The results for this experiment are presented in Section V-C.

1) Fusion challenge: The performance of the 2D laser localization method is highly dependent on an accurate initialization. It localizes itself by matching laser scans to the obstacles in the occupancy grid and thus, once it is lost, it might match the laser scans with incorrect obstacles. If this happens, the laser localization might not have a tendency towards finding its way back into the correct pose. This is illustrated in Fig. 7. The weight given to a wrong estimation such as the one in Fig. 7b is higher than the one given to a better estimation such as the one in Fig. 7c. Thus, the filter does not tend to converge into the right solution.

In such scenario the laser localization's incorrect certainty about a wrong hypothesis could impair the sensor fusion. When trying to use the fusion method described in Section III-B the laser hypothesis would have such a high certainty that it would overpower the map-related weight and avoid its correction.

2) Fusion solution: A different fusion method is proposed where the lasers' contribution to the particles' weight β^n is ignored when the performance of the laser localization decreases.

In order to track the performance of the laser localization an adaptation of a method described in [17] is used. In the original version, the average particle weight is tracked through ω_{slow} and ω_{fast}, which act as exponential filters over long and short time respectively. These parameters would then be used to determine when to generate random particles on the map so that the vehicle could localize itself again (solve the kidnapped robot problem). In our case, we do not use this method to generate random particles but, instead, we use it to evaluate the current laser performance. For this, instead of tracking the average particle weight, we track the highest one ω_{max}. The following equations describe how the variables are updated after each measurement update.

$$\omega_{max} = \max_n f_{z_k^{lasers}|x_k^n} \tag{11}$$

$$\omega_{slow} = \omega_{slow} + \alpha_{slow} \cdot (\omega_{max} - \omega_{slow}) \tag{12}$$

$$\omega_{fast} = \omega_{fast} + \alpha_{fast} \cdot (\omega_{max} - \omega_{fast}) \tag{13}$$

Where α_{slow} and α_{fast} describe the decaying factor of the exponential filters and $0 \leq \alpha_{slow} \ll \alpha_{fast}$. The switching then works as shown in Algorithm 2.

Algorithm 2 Laser fusion method

```
if $(1 - \frac{\omega_{fast}}{\omega_{slow}}) \geq \tau$ then
    Use (8)
else
    Use $\beta^n = \alpha \cdot f_{z_k^{pm}|x_k^n}$
end if
```

Here, $0 \leq \tau \leq 1$ is a user-defined threshold. In our case, it was empirically set to 0.5.

Note that both the beam model for laser localization and the analysis of the appended trajectories of each particle are computationally expensive tasks. When trying to use both simultaneously the laser localization turned even more fragile as the update frequency of the filter decreased. Furthermore, the accuracy of the LiDAR is already high and would not benefit much from the low accuracy of the corrected odometry. Therefore, this experiment was only done with the MAPF implementation that did not include the trajectory information.

V. Results

A. Pure Odometric Localization

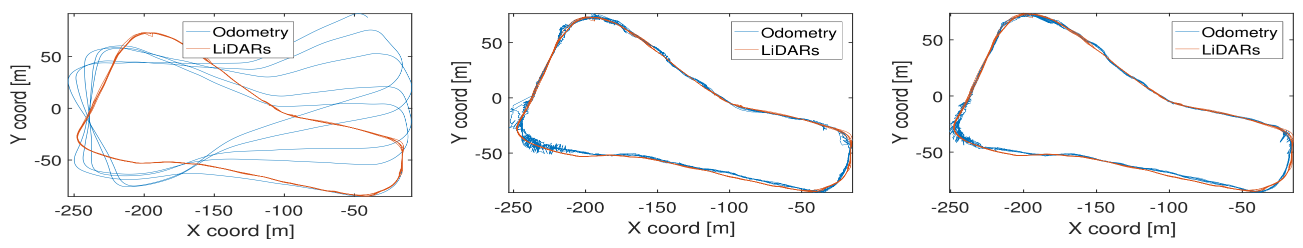
The results for the experiment described in Section IV-B, where several loops were ran while localizing only with the wheel odometry, are shown in Fig. 8. Similarly, Table I shows the mean squared error (MSE) of each trajectory when compared to the 2D LiDAR Localization. The drift is evident when no map correction is being made. Meanwhile, a map correction without trajectory information reduces the drift but is still not as precise as when the trajectory is being tracked. The graphs from Fig. 9 show the evolution of the squared error as the vehicle navigates. In this graph $d_{max} = 0$ corresponds to the implementation of the MAPF without using the trajectory information. It can be seen that the drift is largely reduced for both map-corrected trajectories. However, when no trajectory information is used the system does appear to start drifting by the end of the graph. This is expected as in this case the MAPF would only manage to keep the estimate within the road boundaries but would not correct for its position within the length of the road.

TABLE I: Mean Squared Error (MSE) for the distance and orientation between odometric trajectories and the 2D laser's.

	MSE dist. [m^2]	**MSE orient. [rad^2]**
No map correction	2504.5	0.2925
MAPF no trajectory info.	170.23	0.0634
MAPF $d_{max} = 30$m	61.7806	0.0231

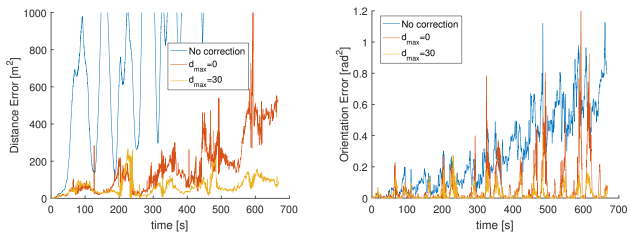
B. GPS Bounding

The use of the proximity map $\mathbf{P}$ to bound the GPS' pose estimation, as detailed in Section IV-C, yielded a higher



(a) No map correction. (b) MAPF without trajectory info. (c) MAPF with $d_{max} = 30$m.

Fig. 8: Odometry trajectory with different levels of map correction. The use of the map information reduced the drift. Furthermore, the use of trajectory information yielded a more precise estimation.



(a) Position error. (b) Orientation error.

Fig. 9: Error evolution for the odometric trajectories.

position accuracy. This can be observed in Table II. The distance error reduced by 17%, although not much improvement was obtained by including the trajectory information for the map analysis. The trajectories generated are also plotted in Fig. 10.

TABLE II: Mean Squared Error (MSE) for the distance between GPS estimates and the 2D laser's.

	MSE distance $[\mathbf{m^2}]$
No map correction	91.1585
MAPF without trajectory information	77.4681
MAPF with trajectory $d_{max} = 30$	75.4853

Note that the trajectory from Fig. 10b appears to be closer to the ground truth than the one from Fig. 10c despite yielding a similar error measure. We attribute this to the fact that the use of a longer odometric trajectory buffer allowed the vehicle to deviate farther from the free-space areas in some sections of the map. This is particularly evident in the right-most section where the GPS and LiDAR measurements showed the largest difference. However, the increase of accuracy in the remaining portions of the map compensated for this deviation.

C. *Robustness analysis*

This section presents the results for the experiment defined in Section IV-D.

The trajectories tracked by the 2D LiDAR and the corrected odometric trajectory are compared in Fig. 11. As can be seen, the laser-based localization (which used a popular Particle Filter algorithm) got lost after the first loop closure. Meanwhile, our enhanced odometric estimation, although imprecise, was robust enough to complete the loop several times.

The result of the fusion of both can be visualized in Fig. 12. The xy plane corresponds to the map coordinates while the color represents ω_{max}, the maximum weight among the particles as assigned by the laser measurements. In this case we use ω_{max} as a measure of the performance of the laser localization. The vehicle had a tendency to switch modes to purely odometric localization after the curve in the top-right corner of the map. This is evident by the sudden drop in ω_{max} and the decrease in precision. As can be seen in Fig. 6, the lower part of the map is not as structured as the upper half. This corresponds to a section of the operational space that is surrounded by vegetation.

After getting lost in the curve, the vehicle could not localize itself with the LiDAR until it returned to a more structured environment. The vehicle then changed modes, which is evident by observing the sudden increase in both, ω_{max}, and the precision of the estimations in the middle-left section of the map. It is evident that the system sacrificed accuracy in the lower half of the map in order to increase its robustness.

VI. Conclusions

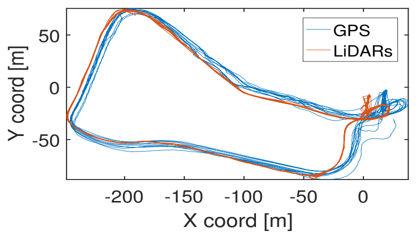
This work presents a method to incorporate the information from the occupancy grid to improve the accuracy of a relative localization sensor (wheel odometry) into being able to perform multiple loop closures. More over, it has also been shown to increase the accuracy of an absolute localization sensor (GPS) and to increase the robustness of an accurate, but fragile, Markovian localization method (2D LiDAR localization). This improvement in performance was obtained through a simple implementation that utilizes the framework of a Particle Filter localization algorithm.

The benefit from using the method presented is highly dependent on the shape of the occupancy grid. Therefore, future work could present a methodology to perform an a priori analysis of the map in order to determine the expected performance of the algorithm in different sections of the operational space.

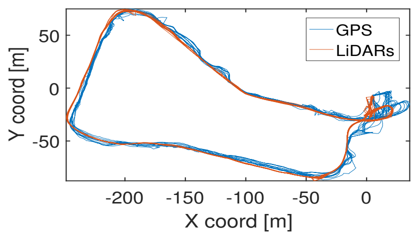
References

[1] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

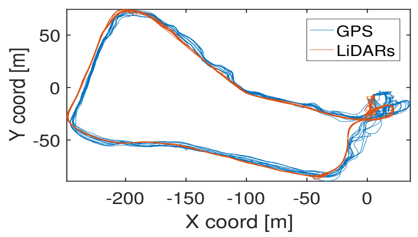
[2] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 2. IEEE, 1999, pp. 1322–1328.



(a) No map correction.



(b) MAPF without trajectory info.



(c) MAPF with $d_{max} = 30$m.

Fig. 10: GPS trajectories with different levels of map correction. When using MAPF the trajectories better resembled the ones from the lasers. Fig. 10c appears to deviate more from the ground truth in the rightmost section of the map. However, the increased accuracy in the remaining portions of the map compensated for this deviation as seen in Table II.

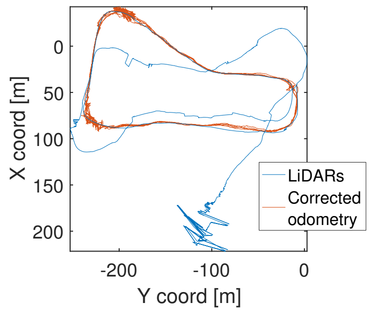


Fig. 11: When operating on an outdated map, the lasers got lost after just one loop. In contrast, the corrected odometry managed to complete the five loops without getting lost.

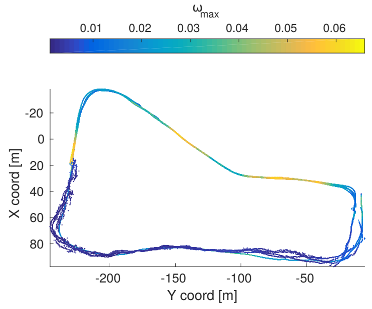


Fig. 12: Results of the fusion of 2D LiDAR localization and odometry through the MAPF. ω_{max} indicates the maximum weight that the LiDAR gave to the particles, a measure of its performance. Whenever the Lasers performed poorly (lower half of the map) the system ignored their measurements and relied solely on odometry and map information. This behavior allowed the system to perform several loops without getting lost.

[3] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, "Current map-matching algorithms for transport applications: State-of-the art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 15, no. 5, pp. 312 – 328, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X07000265

[4] D. Bernstein and A. Kornhauser, "An introduction to map matching for personal navigation assistants," 1998.

[5] C. White, D. Bernstein, and A. Kornhauser, "Some map matching algorithms for personal navigation assistants," *Transportation Research Part C: Emerging Technologies*, vol. 8, no. 1-6, pp. 91–108, 2000.

[6] G. Taylor, G. Blewitt, D. Steup, S. Corbett, and A. Car, "Road reduction filtering for gps-gis navigation," *Transactions in GIS*, vol. 5, no. 3, pp. 193–207, 2001. [Online]. Available: http://dx.doi.org/10.1111/1467-9671.00077

[7] M. A. Quddus, W. Y. Ochieng, L. Zhao, and R. B. Noland, "A general map matching algorithm for transport telematics applications," *GPS Solutions*, vol. 7, no. 3, pp. 157–167, 2003. [Online]. Available: http://dx.doi.org/10.1007/s10291-003-0069-z

[8] H. Yin and O. Wolfson, "A weight-based map matching method in moving objects databases," in *Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on*, June 2004, pp. 437–438.

[9] W. Y. Ochieng, M. A. Quddus, and R. B. Noland, "Map-matching in complex urban road networks," 2003.

[10] S. Taneja, B. Akinci, J. H. G. Jr., and L. Soibelman, "Algorithms for automated generation of navigation models from building information models to support indoor map-matching," *Automation in Construction*, vol. 61, pp. 24 – 41, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0926580515002034

[11] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, Feb 2002.

[12] Widyawan, M. Klepal, and D. Pesch, "A bayesian approach for rf-based indoor localisation," in *2007 4th International Symposium on Wireless Communication Systems*, Oct 2007, pp. 133–137.

[13] P. Davidson, J. Collin, and J. Takala, "Application of particle filters for indoor positioning using floor plans," in *Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS), 2010*, Oct 2010, pp. 1–4.

[14] Widyawan, M. Klepal, and S. Beauregard, "A backtracking particle filter for fusing building plans with PDR displacement estimates," in *Positioning, Navigation and Communication, 2008. WPNC 2008. 5th Workshop on*, March 2008, pp. 207–212.

[15] S. Beauregard, Widyawan, and M. Klepal, "Indoor PDR performance enhancement using minimal map information and particle filters," in *2008 IEEE/ION Position, Location and Navigation Symposium*, May 2008, pp. 141–147.

[16] A. U. Peker, O. Tosun, and T. Acarman, "Particle filter vehicle localization and map-matching using map topology," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, June 2011, pp. 248–253.

[17] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[18] "amcl," http://wiki.ros.org/amcl, accessed: 2016-05-24.

[19] Velodyne Acoustics, Inc, "Velodyne PUCK (VLP-16)," accessed: 2016-05-23. [Online]. Available: http://velodynelidar.com/vlp-16.html

[20] M. Bosse and R. Zlot, "Continuous 3d scan-matching with a spinning 2d laser," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, May 2009, pp. 4312–4319.

[21] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, Feb 2007.

[22] ——, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, April 2005, pp. 2432–2437.

[23] "slam_gmapping," http://wiki.ros.org/slam_gmapping, accessed: 2016-05-23.