Mobile Robot Localization with Reinforcement Learning Map Update Decision aided by an Absolute Indoor Positioning System

Luís Garrote[1], Miguel Torres[1], Tiago Barros[1], João Perdiz[1], Cristiano Premebida[2] and Urbano J. Nunes[1]

Abstract— **This paper introduces a new mobile robot localization solution consisting of two main modules: a Particle-Filter based Localization (PFL) and a Reinforcement-Learning based map updating, integrating relative measurements and absolute indoor positioning sensor (A-IPS) data. Concerning localization using 2D-LiDARs, featureless areas are known to be problematic. To solve this problem a classic PFL approach was modified to incorporate A-IPS position measurements in the prediction and update stages. The localization approach has the particularity of including the possibility of updating the map whenever major modifications are detected in the environment in relation to the current localization map. Due to the random sampling-based nature of the PFL, an update solution is not trivial as in classic mapping approaches since small inconsistencies in the estimated pose can lead to erroneous map associations. The proposed method learns to decide by assigning higher rewards the greater is the overlap between the map and the 2D-LIDAR scans, via RL, and then a proper update of the map is achieved. Validation of the proposed pipeline was carried out in a differential drive platform with algorithms developed in ROS. Tests were performed in two scenarios in order to assess the performance of both the localization module and the map update stage. The results show that the proposed method offers localization improvements in relation to known approaches, and consequently the concept validity and promising perspectives for the proposed map update decision framework.**

I. Introduction

Localization in robotics has been the focus of extensive research, with wide applications ranging from Automated Guided Vehicles (AGVs) to service and social robots. There have been many proposed solutions for the localization of these robotic systems and, within those solutions, laser-based localization techniques are some of the most popular. Although laser-based techniques mainly use laser scanners (or 2D LiDARs), other sensor modalities are also used to complement some of the technology limitations. Most of the present localization methods rely on multimodal sensor data, because laser-based systems alone struggle in accurately localizing robots in featureless areas (*e.g.*, corridors) [1], [2], or in areas with multiple dynamic obstacles. Besides LiDARs, some of the most popular sensors used in indoor localization are cameras, wheel encoders (odometry), and Inertial Measurement Units (IMUs). The use of absolute indoor positioning systems tends to be indispensable in order to have a more robust positioning system when in the presence of featureless areas where other 'local' approaches such as laser-based are prone to fail. Within laser-based approaches, Particle Filter (PF) localization approaches have been widely adopted due to a low software implementation overhead [3] but, due to lack of sensor information in real-world circumstances, the hypothetical poses (particles) can diverge from the real pose [4]. One of the most popular PF methods is the Adaptive Monte Carlo Localization (AMCL), which uses pose data (*e.g.*, odometry), a prior occupancy-grid map and laser scanner data to estimate a mobile robot's pose. An open version of the AMCL algorithm is available on ROS [5]. However, the available version was only designed to deal with two data inputs (*pose* and laser scanner data sources). When deploying a robot in a new environment it is common to provide an *a priori* representation. Some methods are robust to small changes in the environment but for multiple changes the pose estimates tend to diverge as the overlaps from the laser scan and the representation are reduced. Although occlusions caused by dynamic elements in a scene are a complex problem, static elements that were recently added to the scene can be updated and aid the localization process (*e.g.*, trash cans, cabinets or tables). Using learning techniques, such as RL, some patterns can be learned in order to assess if a laser scan can be updated in the environment representation.

In this work a new localization approach is proposed, developed in ROS environment, which supports pose data, data from an absolute indoor positioning system (A-IPS), and laser scanner's measurements. The localization approach, henceforth named **RLmPFL** (Reinforcement-learning based mapping in a Particle-filter based localization) approach, uses an *a priori* occupancy-grid map that is updated by a RL inspired map update decision approach. It is important to note that this work does not intend to be a replacement for a Simultaneous Localization and Mapping (SLAM) approach, but to be a complementary module that can be deployed in structured or semi-structured scenarios in order to reliably update a localization's map or used together with a SLAM approach to attain a more reliable update.

II. Related Work

Within localization methods for mobile robots some of the most well-known methods are based on the following stochastic filters frameworks [6]: the Extended Kalman Filter (EKF), the Unscented Kalman Filter (UKF), Monte Carlo Localization (MCL) or Particle Filter (PF) and AMCL. In the MCL/PF method, which is of particular importance in this work, the representation of the probability density function

[1] Authors are with the Institute of Systems and Robotics, Department of Electrical and Computer Engineering, University of Coimbra, Portugal. Emails: {garrote, tiagobarros, joao.perdiz, urbano}@isr.uc.pt.

[2] C. Premebida is with the Dept. of Aeronautical and Automotive Engineering, Loughborough University, UK. Email: c.premebida@lboro.ac.uk

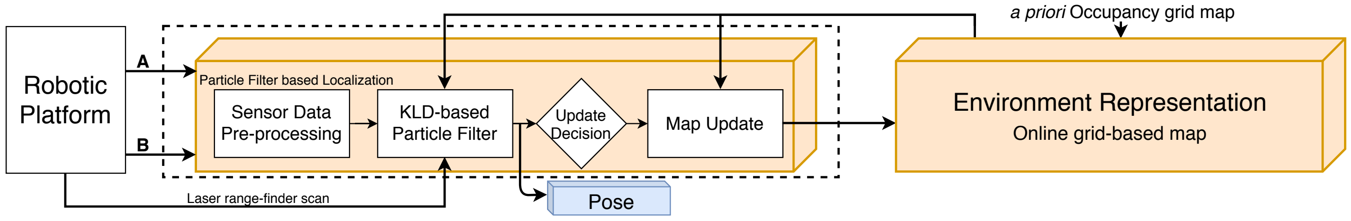

a priori Occupancy grid map
Robotic Platform
A
B
Particle Filter based Localization
Sensor Data Pre-processing
KLD-based Particle Filter
Update Decision
Map Update
Environment Representation
Online grid-based map
Laser range-finder scan
Pose

Fig. 1. Block diagram of the proposed RLmPFL approach. **A** and **B** represent respectively relative measurements and absolute measurements.

(*PDF*) is made by maintaining a set of samples (particles) that are randomly drawn from the *PDF* [7]. This method is able to represent multi-modal distributions and can be applied to both local and global localization problems [6]. PFs are easily implemented, which makes them an attractive solution for localization methods in mobile robotics. The benefits of a multi-sensor fusion localization approach are demonstrated in [8]. Information is processed from a laser range finder, a WiFi card, a compass, and a group of external cameras. Four metrics are used to evaluate multiple combinations of these four sensors, with the full set of sensors being the best globally. In order to decrease accumulation of error during vehicle localization, [9] introduces a multi-point joint particle filter (MPJPF) method acting upon information provided by visual odometry (VO) data. The method swaps the sequence of points used in VO for a simpler, faster anchor point (AP) representation. A Map-Aware Particle Filter (MAPF) was presented in [10] to optimize the weight of particles in a localization method based on map information, which is used to construct a proximity map of grid points. Particles are optimized by selecting those where prior odometry data does not create a conflict with the proximity map. Reinforcement Learning has also been combined with PFs. In [11], a learning framework is proposed that employs a PF inside a Q-learning RL paradigm. By observing the expected values of generated particles in the state space it was possible to isolate those with positive values so the system could more quickly learn which states produce the most favorable outcomes. Map update techniques have been incorporated in SLAM, such as in [12], to improve mapping accuracy in a dynamic environment by using long-term memory to distinguish between static and dynamic points in a 3D map. Alternatively, in [13] a Rao-Blackwellized PF SLAM approach was proposed where each particle in the PF keeps a map proposal and is independently updated.

RL-based techniques have been employed in map update approaches as well. In [14], SARSA (a RL method) is used to improve the performance of map-merging in a Multi-Robot SLAM problem that is applicable to the navigation of a large environment. In [15] mapping errors caused by robot motion were addressed. A policy search using a dynamic programming method is developed to create a RL model of trajectory control on a one-step policy basis. Reward is attributed based on the variation, between trials, of the uncertainty of robot pose estimation at each way-point in the trajectory. In [16], RL is integrated in a SLAM approach where RL is used in a global training prior the navigation takes place to create a global map of valid states and actions. These are loaded on request whenever the robot is in a specific location, creating a local RL model that can be inexpensively used for local motion planning. Concerning the use of indoor positioning systems, in [17] a PF approach fusing IMU/inertial and ultra wide band ranging information was proposed to track the relative positioning of multiple users.

III. Methodology and Proposed Method

In this section the proposed RLmPFL approach is presented in detail (see Fig. 1).

A. Sensor Data Processing

The design of a multimodal sensor fusion approach requires the correct pre-processing of the incoming sensor measurements. Since not all sensors share a common representation and a common frame, absolute positioning and relative pose measurements were considered. Also, to avoid wasting resources in redundant localization estimates, taking into account the different rates and the sampling-based nature of the PF, a threshold had been defined for the relative sensor data, which only activates the localization method when a defined minimal translation and/or rotation is accumulated by this module. The threshold values Δd and $\Delta\theta$ serve to preclude the sensor data from being used unless one of those values of displacement has been reached since the last estimated pose. An absolute positioning measurement always activates the particle filter loop. The motion model presented in [6], where the robot's motion is defined by a translation and two rotations (δ_{trans}, δ_{rot1} and δ_{rot2}), is used to compute the displacement between consecutive relative measurements. A local pose is then updated based on that model. If one of the thresholds for angular or linear displacement is met, the local translation and rotations are used to update the PF loop. The relative sensor accumulation is reset after each loop.

B. KLD-Based Particle Filter Localization

Let the set of particles $\mathscr{X}$ at a given step be defined as $\mathscr{X} = \{\mathbf{x}_k, w_k\}, k = 1, \cdots, n_p$ where $\mathbf{x}_k = (x_k, y_k, \theta_k)$ is the particle's pose/state, w_k a particle's weight and n_p the number of particles. Each particle corresponds to a pose hypothesis of the robotic platform's location. As presented in [18] and shown in Fig. 3, a particle filter can be represented by three main procedures; 1) **Prediction** of the next particle's

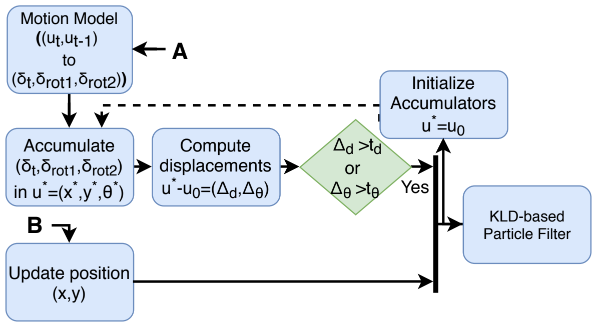

Motion Model
$((u_t,u_{t-1})$ to $(\delta_t,\delta_{rot1},\delta_{rot2}))$
A
Initialize Accumulators
$u^*=u_0$
Accumulate
$(\delta_t,\delta_{rot1},\delta_{rot2})$ in $u^*=(x^*,y^*,\theta^*)$
Compute displacements
$u^*-u_0=(\Delta_d,\Delta_\theta)$
$\Delta_d > t_d$ or $\Delta_\theta > t_\theta$
Yes
KLD-based Particle Filter
B
Update position (x,y)

Fig. 2. Sensor data pre-processing pipeline (**A**: relative measurements; **B**: A-IPS position measurements).

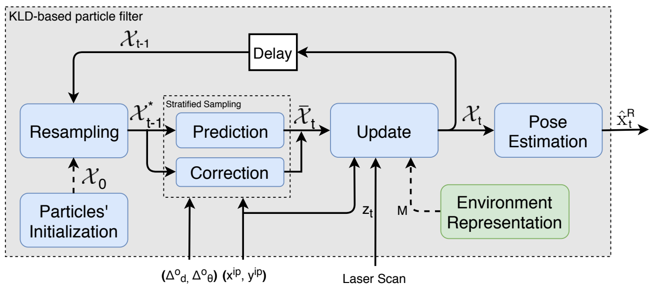


Fig. 3. KLD-based particle filter block scheme with: $\mathscr{X}_t$ and $\mathscr{X}_{t-1}$ being the actual and the previous particle sets; $\mathscr{X}^*_{t-1}$ the rearranged previous particle set; $\bar{\mathscr{X}_t}$ the actual predicted set; $\mathscr{X}_0$ the initial set; $\hat{\mathbf{x}}^{\mathbf{R}}_{\mathbf{t}}$ the actual estimated pose; M the prior map; z_t the laser scan observation; $(\Delta^o_d, \Delta^o_\theta)$ the linear and angular displacements from the relative sensor and (x^{ip},y^{ip}) the position from the A-IPS.

state given the sensor measurements; 2) **Update** of each particle's weights considering an observation of the environment; 3) **Resampling** of the particles taking into account the particle's weight distribution. Other important procedures are the initialization of resources (**Initialization**) and the pose estimation procedure (**Pose Estimation**). The proposed PF builds on the Kullback-Leibler distance (KLD) method [18]. The main modifications presented by this PF-based localization approach are in the prediction step, where it is possible to receive relative measurement data (Odometry) and absolute position measurements (A-IPS), and then fuse the data to predict the next state of the particles. Also, in the update stage, the absolute sensor is used to compute each particle's weight. The structure of the filter is shown in Fig. 3.

1) Prediction: The prediction stage is responsible for the computation of each particle's next state given the sensor measurements. Considering the aforementioned sensor data processing pipeline, multiple types of data might be available and depending on those types, different fusion methods can be applied. Linear and angular displacements for the relative "sensor" from the pre-processing stage are converted into δ_{trans}, δ_{rot1} and δ_{rot2}. A sampling-based stratified approach ($\sigma_{A-IPS} + \sigma_O = 1$) inspired by [17] is taken to handle multiple sensor data where the sensor is randomly picked considering the sensor's stratified representation (e.g., considering $\sigma_{A-IPS} = 0.1$ and $\sigma_O = 0.9$, this means that 10% of the particles are updated with A-IPS data while 90% of the particles are updated with relative data). For relative measurements we assume the motion model with noise described in [6]:

$$\begin{cases} \hat{\delta}_{trans} = \delta_{trans} - \sim \mathcal{N}(0, \alpha_1^2\delta_{rot1} + \alpha_2^2\delta_{trans}) \\ \hat{\delta}_{rot1} = \delta_{rot1} - \sim \mathcal{N}(0, \alpha_3^2\delta_{trans} + \alpha_4^2\delta_{rot2}) \\ \hat{\delta}_{rot2} = \delta_{rot2} - \sim \mathcal{N}(0, \alpha_1^2\delta_{rot1} + \alpha_2^2\delta_{trans}) \end{cases} \tag{1}$$

$$\mathbf{x_t^{[n]}} = \begin{cases} x_t^{[n]} = x_{t-1}^{[n]} + \hat{\delta}_{trans}\cos(\hat{\theta}_{t-1} + \hat{\delta}_{rot1}) \\ y_t^{[n]} = y_{t-1}^{[n]} + \hat{\delta}_{trans}\sin(\hat{\theta}_{t-1} + \hat{\delta}_{rot1}) \\ \theta_t^{[n]} = \hat{\theta}_{t-1} + \hat{\delta}_{rot1} + \hat{\delta}_{rot2} \end{cases} \tag{2}$$

The state of a particle n at iteration t, selected for A-IPS update using the stratified approach (see **Correction** in Fig. 3), is given by

$$\mathbf{x_t^{[n]}} = \begin{cases} x_t^{[n]} = x_{ip} - \sim \mathcal{N}(0, \sigma_x^2) \\ y_t^{[n]} = y_{ip} - \sim \mathcal{N}(0, \sigma_y^2) \\ \theta_t^{[n]} = \hat{\theta}_{t-1}^R - \sim \mathcal{N}(0, \sigma_\theta^2) \end{cases} \tag{3}$$

where (x_{ip}, y_{ip}) is the A-IPS position measurement and $(\sigma_x, \sigma_y, \sigma_\theta)$ noise-dependent parameters.

2) Update: The update stage computes the weight ($w_t = p(z_t|x_t)$) of a particle by using an adaptation of the "beam range finder model" algorithm described in [6]. A standard approach, used to avoid redundancy and not waste processing resources, would be to process only a subset of the laser scan points to compute the new weight of a particle. The modified "beam range finder model" [6] is mathematically expressed as

$$w_t = e^{-\left(\frac{\left|(\mathbf{x_t^{[n]}} - p_{ip}\right|}{\sigma_{ip}}\right)^2} + \sum_{i=1}^{n_l} e^{-\left(\frac{\left|(p_s^o \underset{M}{\rightarrow} s_i) - s_i\right|}{\sigma_r}\right)^2} \tag{4}$$

where σ_r and σ_{ip} act as shape parameters, n_l is the number of processed scan points, p_{ip} an A-IPS position measurement and $\underset{M}{\rightarrow}$ represents a raytracing operator (in the World frame) that performs a raytracing-like operation between the sensor frame origin (p_s^o) and a laser scanner point s_i in the environment representation M, and provides a collision point. The first occupied cell in the raytracing operation represents the expected position of an obstacle; if no occupied cell is found then p_s^o is returned.

3) Resampling: Contrarily to the classical particle filter, the KLD algorithm forces each particle to be iterated and the KLD condition is evaluated until the number of particles is enough to represent the expected distribution or a maximum number of particles (n_{max}) was reached. To this end, the first step is to sort the particles from the particle's set with the lowest to the one with the highest normalized weight. After this, for each iteration of the KLD loop a particle is sampled, and goes through the prediction and the update stages, until the number of processed particles is equal to n_{max} or the KLD condition is reached. The particles in this work are sampled according to the Multinomial Resampling method [19]. In order to compute the KLD condition, a

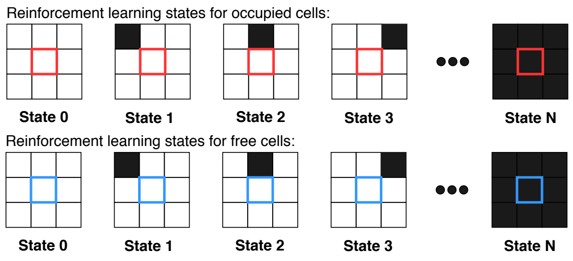

Reinforcement learning states for occupied cells:
State 0
State 1
State 2
State 3
State N
Reinforcement learning states for free cells:
State 0
State 1
State 2
State 3
State N

Fig. 4. Representation of the possible states for both RL models. For a 3×3 grid the number of possible combinations (N) is 512.

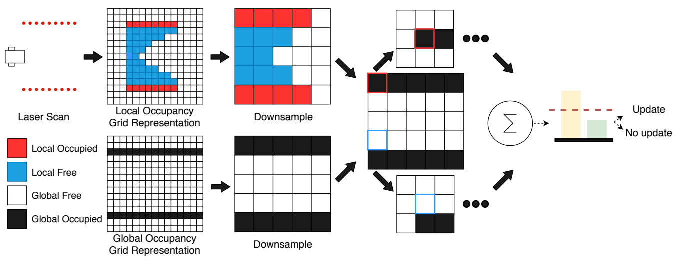


Fig. 5. Representation of the decision stage of the proposed RL pipeline.

2D grid representation was employed where each cell was represented by an angular histogram.

4) Pose Estimation: After performing a complete KLD iteration with the new particle states and their normalized weights, the robot's pose is computed according to the following expressions [20]:

$$\hat{\mathbf{x}}_{\mathbf{t}}^{\mathbf{R}} = \begin{cases} \hat{x}_t^R = \sum_{i=1}^{n_p} x_t^{[i]} w_t^{[i]} \\ \hat{y}_t^R = \sum_{i=1}^{n_p} y_t^{[i]} w_t^{[i]} \\ \hat{\theta}_t^R = \arctan 2(\sum_{i=1}^{n_p} \sin(\theta_t^{[i]}) w_t^{[i]}, \sum_{i=1}^{n_p} \cos(\theta_t^{[i]}) w_t^{[i]}) \end{cases} \tag{5}$$

where $\hat{\mathbf{x}}_{\mathbf{t}}^{\mathbf{R}}$ is the estimated robot's pose.

C. Update Decision

Due to the sampling nature of the PFL, the update of the map is not trivial. If the particles are scattered all over the map, the map may be incorrectly updated. The same is valid for featureless areas such as corridors where the update metric in the PFL may increase the weight of unwanted particles. The proposed approach, using RL, aims to learn patterns (represented by states) between the last laser scan (L) reading and the actual environment representation (M). The proposed RL approach consists of two stages: an offline learning stage and an online decision stage as shown in **Algorithm** 1.

For the learning stage, given the pose from the aforementioned PFL method ($\hat{\mathbf{x}}_{\mathbf{t}}^{\mathbf{R}}$), the environment model and the last laser scan (L), the first step consists in computing the RL states (**GetStates**). To compute the states, the first step consists in generating a ternary (occupied, unknown, free state) local grid-map for each range in the laser scan. In this context, a local grid-map is a representation centered in the laser scan sensor's frame while a global grid-map is the environment representation. The local grid-map is all initialized with unknown state. If a range is higher than a maximum distance, a ray-tracing approach (based on the Bresenham's algorithm) is employed to compute all the free cells between the sensor's origin frame and the measured point (i.e., the range is converted from polar to Euclidean coordinates). Otherwise, if the range is smaller than a maximum distance and larger than a minimum safety range, the point is marked as occupied and the ray-traced cells between the sensor's origin and the point are marked as free. After the generation of the local grid-map, two sets of free cells and occupied cells are generated, a translation is applied to each point in the sets using the pose ($\hat{\mathbf{x}}_{\mathbf{t}}^{\mathbf{R}}$) - a downsampling is applied at this stage to reduce the number of points. For each new point in each set, a hash value is computed considering the surrounding obstacles (in a 3×3 window). Each hash value is unique and represents a state to be modeled in one of two Q-Matrices, one that learns from the environment configuration centered in free points and another that learns from the environment configuration centered in occupied cells. The proposed RL considers a 3×3 mask centered in each of the aforementioned points (see Fig. 4). This gives a total of 512 possible combinations of occupied and free cells. Given the two sets of identifiers (S_{free} and S_{occ}), the following step consists in computing the weight, or overlapping score, between the environment representation and the laser scan converted to Euclidean points and projected in the World frame (P_W). The score (**GetScore**) is calculated as

$$w = \frac{\sum_{j=1}^{|P_W|}(|x_{near}^i - P_W^i| \wedge 0)}{|L|} \tag{6}$$

where x_{near}^i is the nearest occupied position to P_W^i in the environment representation. The reward (**GetReward**) is computed from the score and is expressed by

$$R(w) = K_1 + \frac{K_2}{1+e^{K_3(w-K_4)}} \tag{7}$$

where K_1, K_2, K_3 and K_4 are behaviour-dependent defined values. The action that originated the reward can systematically be computed from the overlapping score based on a threshold (**GetAction**).

For each set of free and occupied identifiers, the corresponding Q-Values are updated. If an identifier is not present in a Q-Matrix, its Q-Value is initialized with zero. The update follows the classic Q-Learning update equation. The Q_{free}^+ and Q_{occ}^+ are the previous iteration maximum Q-Value (for the free and occupied RL models respectively) for any action of the processed states and are updated at the end of the learning stage.

The decision stage (see Fig. 5) uses the two models trained in the previous stage to make a decision on whether a given laser scan should be used,or not used, to update the environment model. The first step consists of computing the states for each occupied and free cells in the local representation (**GetStates**); in this step the corresponding points in the

Algorithm 1: MapUpdateRL method.

```
Input: Environment Model ($M$), Pose ($\hat{\mathbf{x}}_t^{\mathbf{R}}$), Laser scan ($L$), Decision threshold ($t_d$);
1  Initialization:
2  $Q_{free} \leftarrow \emptyset$, $Q_{occ} \leftarrow \emptyset$, $Q^+_{free} \leftarrow 0$, $Q^+_{occ} \leftarrow 0$;
3  Learning Loop:
4  $S_{free}, S_{occ} \leftarrow$ GetStates($M$,$L$,$\hat{\mathbf{x}}_t^{\mathbf{R}}$);
5  $w \leftarrow$ GetScore($M$,$L$,$\hat{\mathbf{x}}_t^{\mathbf{R}}$); $R \leftarrow$ GetReward(w); $a \leftarrow$ GetAction(w);
6  $q^+_{free} \leftarrow 0$, $q^+_{occ} \leftarrow 0$;
7  foreach $s \in S_{free}$ do
8      if $s \notin Q_{free}$ then
9          $Q_{free}(s) \leftarrow 0_{(|A|\times|1|)}$;
10     $Q_{free}(s,a) \leftarrow Q_{free}(s,a) + \alpha(R + Q^+_{free} - Q_{free}(s,a))$;
11     $q^+_{free} \leftarrow \max(q^+_{free}, Q_{free}(s,:))$;
12 foreach $s \in S_{occ}$ do
13     if $s \notin Q_{occ}$ then
14         $Q_{occ}(s) \leftarrow 0_{(|A|\times|1|)}$;
15     $Q_{occ}(s,a) \leftarrow Q_{occ}(s,a) + \alpha(R + Q^+_{occ} - Q_{occ}(s,a))$;
16     $q^+_{occ} \leftarrow \max(q^+_{occ}, Q_{occ}(s,:))$;
17 $Q^+_{free} \leftarrow q^+_{free}$, $Q^+_{occ} \leftarrow q^+_{occ}$;
18 Decision:
19 $D \leftarrow 0_{(|A|\times|1|)}$;
20 $S_{free}, S_{occ}, P_{free}, P_{occ} \leftarrow$ GetStates($M$,$L$,$\hat{\mathbf{x}}_t^{\mathbf{R}}$);
21 foreach $(s,p) \in (S_{free}, P_{free})$ do
22     $a \leftarrow \underset{a_i \in A}{\text{argmax}}\ Q_{free}(s,a_i)$;
23     $D(a) \leftarrow D(a)$ + DecisionWeight($||p||$);
24 foreach $(s,p) \in (S_{occ}, P_{occ})$ do
25     $a \leftarrow \underset{a_i \in A}{\text{argmax}}\ Q_{occ}(s,a_i)$;
26     $D(a) \leftarrow D(a)$ + DecisionWeight($||p||$);
27 D $\leftarrow \frac{D}{\sum_{i=1}^{|A|}(D(i))}$;
28 if $\exists\ D(a_i) \geq t_{RL}$ ,$i = 1 \cdots |A|$ then
29     a $\leftarrow \underset{a_i \in A}{\text{argmax}}\ D(a_i)$;
30 else
31     a $\leftarrow$ -1;
Output: $a$
```

Euclidean space for each state are also computed. For each state, the decision is obtained based on the computation of the argument of the maximum Q-Value for that state. A decision set (D) with the same size as the number of actions is initialized with zero and is updated by each state's computed decision. A penalty factor w_d is also computed in order to give more importance to decisions closer to the sensor frame origin, $w_d = K_5 + \frac{K_6}{1+e^{-K_7(||d||-K_8|)}}$, where K_5, K_6, K_7 and K_8 are behaviour-dependent weights and d is the distance between the sensor frame origin and the correspondent state's position. After processing all the states, the decision set is normalized and if one of the decisions is greater than a given threshold (t_{RL}), the corresponding decision is used, otherwise a non-update decision is issued.

D. Map Update

The environment representation uses a 2D occupancy grid with log odds update [21], [6]. The probability that the cell c is occupied given the observations $z_{1:t}$ is given in log odds

$$l(c|z_{1:t}) = \log\frac{p(c|z_t)}{1-p(c|z_t)} - \log\frac{p(c)}{1-p(c)} + \log\frac{p(c|z_{1:t-1})}{1-p(c|z_{1:t-1})} \quad (8)$$

with $p(c)$ the prior probability ($p(c) = 0.5$), $p(c|z_{1:t-1})$ the previous estimate and $p(c|z_t)$ denotes the probability that the cell c is occupied given the measurement z.

IV. Experimental Validation

The experimental platform used in the validation of the proposed approach is a differential drive AGV prototype based on the InterBot architecture [22]. The platform is equipped with an Hokuyo UTM-30LX/LN laser scanner, a Xsens Mti-G IMU, and two encoders with 980 pulses per revolution. The drive of the two DC motors is performed by a RoboteQ SDC2130 power driver. The platform is also equipped with a laptop running ROS [5] responsible for data processing and communicating with the sensors and actuators of the platform. The parameters used for the validation of the proposed RLmPFC localization approach are presented in Table I.

A. Localization

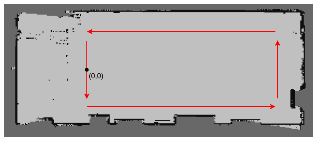
The A-IPS system from Marvelmind Robotics was deployed in both scenarios shown in Fig. 6. A mobile beacon had been mounted on the robotic platform and static beacons were placed on the walls with at least 3 beacons always in line-of-sight. Initial tests with the technology reported a precision of $10-15$ cm in line-of-sight. With occlusions, the error in the positioning increases as shown in Fig. 7. In the scenario shown in Fig. 6a, 6 static beacons were deployed. In the scenario shown in Fig. 6b, 11 static beacons were deployed with the leftmost area of the map without beacons. To compare the localization methods, the laser scan is used to estimate an overlapping score ($M1$), by computing the distance of the scan point to the nearest occupied cell in the *a priori* map (the score decreases with the increase of the distance) using a Gaussian function ($\mathcal{N}_M$) with zero mean

TABLE I
VALIDATION PARAMETERS.

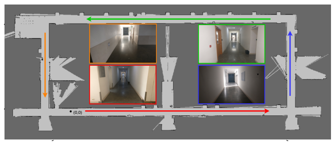
Parameter	Value	Parameter	Value	Parameter	Value
K_1	-0.5	t_{RL}	0.99	α_1	0.0208
K_2	1.0	t_{xy}	0.01	α_2	0.001
K_3	3	$t_{c\theta}$	0.12	α_3	0.0208
K_4	0.6	t_d	0.1	α_4	0.0802
K_5	0.1	t_θ	0.01	σ_{ip}	0.05
K_6	0.9	σ_x	0.5	σ_r	0.1
K_7	-3	σ_y	0.05	n_{max}	1740
K_8	5	σ_θ	0.2	σ_M	0.1
α	0.5	α_{A-IPS}	0.1	-	-

TABLE II
$M1$ AVERAGE AND THE RMSE, IN (%), EVALUATION FOR FIG. 6A.

Method	O	EKF(O)	EKF(O,IMU)	AMCL(O)	RLmPFL(O,A-IPS)
Mean$_{M1}$	15.71	25.48	28.29	68.49	71.61
RMSE$_{M1}$	84.29	74.51	71.71	31.51	31.03



(a) Occupancy grid map for the ISR-UC experimental room (5.2×13.1 m).



(b) Occupancy grid map for the Floor 0 of the ISR-UC (20×54 m).

Fig. 6. Occupancy grid map representation of the scenarios generated by a modified Hector_SLAM approach. The robotic platform was placed near the origin point (0,0) in each test. Arrows represent the direction of the performed tests.

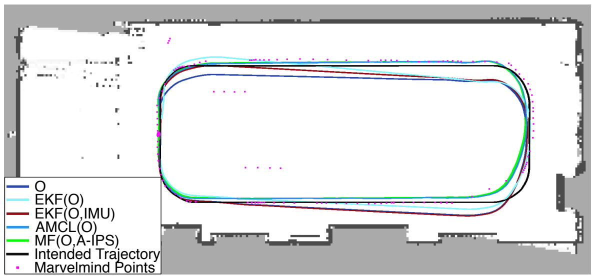


Fig. 7. Obtained trajectories for the methods evaluated in the scenario presented in Fig. 6a.

and σ_M standard deviation

$$M1 = \frac{\sum_{i=1}^{|S|}\sum_{j=1}^{|s_i|} \mathcal{N}_M(0, |x^j_{near} - s_{ij}|)}{|S|}. \quad (9)$$

To assess the performance of the proposed pipeline, the scenario in Fig. 6a was used. The following methods where evaluated: Odometry (**O**); EKF with Odometry (**EKF(O)**); EKF with Odometry and IMU (**EKF(O,IMU)**); AMCL ROS package using Odometry (**AMCL(O)**); and the proposed approach (**RLmPFL(O,A-IPS)**). The results for the scenario are shown in Fig. 7 and Table II.

The values in Table II are normalized by the number of laser scan readings (1080 readings). Using the same parameters in both the proposed approach and the AMCL package, the presented results show that the proposed localization approach provides better results, in particular, due to the use of A-IPS the convergence of the particles was faster in the initial steps and when odometry was not enough. Localization results in a more challenging scenario (from Fig. 6b) are described in the next subsection.

B. Map Update

To validate the proposed RL map update decision approach, the results with three methods are compared: no decision module i.e., classic mapping approach expressed by (8), confidence interval, and the proposed RL approach. The confidence interval approach is inspired in [23]; the first step consists in computing the weighted mean and variance for (x, y, θ) and the current set of particles in the PF,

$$\begin{aligned} \bar{x} &= \sum_{n=1}^{n_p} x_n w_n & \bar{\theta}_1 &= \sum_{n=1}^{n_p} w_n \sin(\theta_n) \\ \bar{y} &= \sum_{n=1}^{n_p} y_n w_n & \bar{\theta}_2 &= \sum_{n=1}^{n_p} w_n \cos(\theta_n) \end{aligned} \quad (10)$$

$$\begin{aligned} \sigma_x^2 &= \sum_{n=1}^{n_p} w_n(x_n - \bar{x})^2 & \sigma_{\theta_1}^2 &= \sum_{n=1}^{n_p} w_n(\sin(\theta_n) - \bar{\theta}_1)^2 \\ \sigma_y^2 &= \sum_{n=1}^{n_p} w_n(y_n - \bar{y})^2 & \sigma_{\theta_2}^2 &= \sum_{n=1}^{n_p} w_n(\cos(\theta_n) - \bar{\theta}_2)^2 \end{aligned} \quad (11)$$

where the confidence intervals are computed as

$$C_x = \left[\bar{x} - 2\sqrt{\sigma_x^2},\ \bar{x} + 2\sqrt{\sigma_x^2}\right], C_y = \left[\bar{y} - 2\sqrt{\sigma_y^2},\ \bar{y} + 2\sqrt{\sigma_y^2}\right]$$

$$C_\theta = \begin{bmatrix} atan2(\bar{\theta}_1 - 2\sqrt{\sigma_{\theta_1}^2},\ \bar{\theta}_2 - 2\sqrt{\sigma_{\theta_2}^2}) \\ atan2(\bar{\theta}_1 + 2\sqrt{\sigma_{\theta_1}^2},\ \bar{\theta}_2 + 2\sqrt{\sigma_{\theta_2}^2}) \end{bmatrix} \quad (12)$$

while the update decision is

$$a = \begin{cases} 1\ ,\ |C_x + C_y| \leq t_{xy} \wedge |C_\theta| \leq t_{c\theta} \\ 0\ ,\ otherwise. \end{cases} \quad (13)$$

The proposed RL approach requires a training step; thus, the scenario from Fig. 6a was used to train the free and occupied RL models. A simulation-environment was created to generate random robot poses inside the scenario and compute a virtual laser scan. Samples were taken from optimal conditions (the scan completely overlapped the map) and in conditions with noise added to the laser scan, actions were computed accordingly. The results for the three methods are shown in Fig. 8. Both maps in Fig. 6 were generated with a modified Hector_SLAM [24] package and annotated by hand in order to fill small "gaps". In particular, for the map in Fig. 6b, artificial landmarks where added in some of the featureless areas (corridors) in order to aid the scan matching process of the Hector_SLAM package. These maps where generated with datasets from previous works [22], and parts of the scenario in Fig. 6b were changed over time, such as the position of trash cans. For each method, at each iteration an update of the map was only considered in a radius of 5 meters. The obtained results show that it is not suitable for a critical application to directly map using only the obtained pose as small deviations in the estimate of the robot's orientation can lead to the complete failure of the mapping approach. The use of a confidence interval can provide interesting results to some extent but this approach can also fail in featureless regions or areas where the A-IPS provides erroneous or absent measurements. With the proposed approach the map was updated without modifying the baseline structure of the map and with the position of some the elements in the scene correctly modified. In Fig. 9 some of the updated areas are highlighted. It is important to note that due to the small number of features in the bottom part of the map, the number of updates is almost non existent.

V. Conclusion

In this paper, a new approach is developed by leveraging a PF-based method in order to combine measurement from an absolute indoor positioning system in order to increase reliability on localization. Additionally, a map update framework based on Reinforcement Learning (RL) has been proposed as well. The framework presented here has the key aim of providing a robust solution for indoor long term applications such as service robots and/or AGVs - industrial environments. The proposed localization and map update approaches

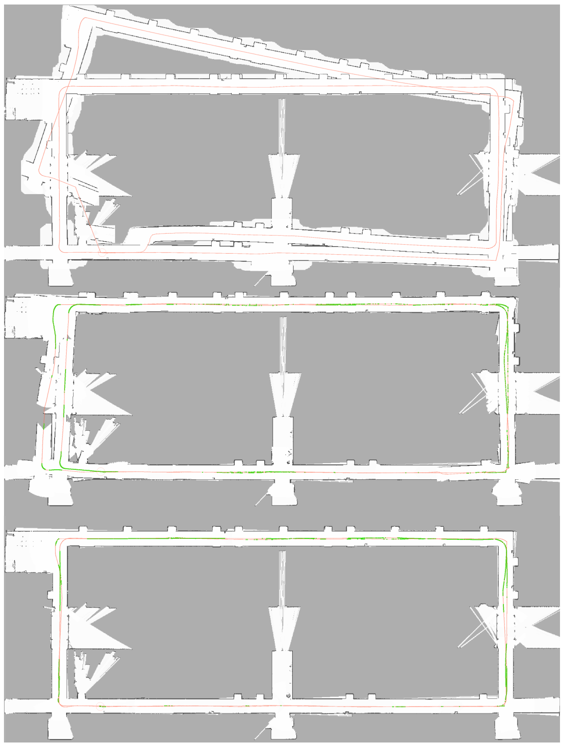


Fig. 8. Grid-map representations after map update for the evaluated methods. The localization of the robotic platform is marked in light red, and in green the map update points. From top to bottom: no decision module, confidence interval decision and the proposed RL approach.

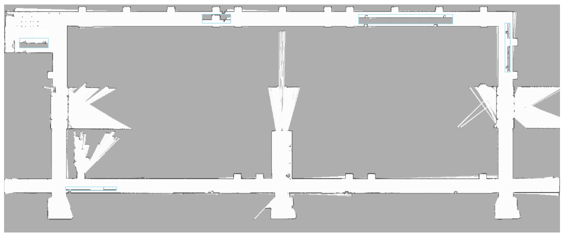


Fig. 9. Grid-map representation from Fig. 6b with the updated areas highlighted.

achieved promising results with the localization providing a stable and accurate pose, and the map update strategy obtained a very consistent update of the map.

Acknowledgments

This work was supported by the Portuguese Foundation for Science and Technology (FCT) under the PhD grant with reference SFRH/BD/88459/2012 with funds from QREN – POPH and the European Social Fund from the European Union. This work was also supported by the projects P2020-AgvPosys with reference CENTRO-01-0247-FEDER-003503 and MATIS - CENTRO-01-0145-FEDER-000014, Portugal. It was also partially supported by FCT through grant UID/EEA/00048/2013.

References

[1] D. Ronzoni, R. Olmi, C. Secchi, and C. Fantuzzi, "AGV global localization using indistinguishable artificial landmarks," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[2] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust Monte Carlo localization for mobile robots," *Artificial intelligence*, vol. 128, no. 1-2, pp. 99–141, 2001.

[3] D. Fox, J. Hightower, L. Liao, D. Schulz, and G. Borriello, "Bayesian filtering for location estimation," *IEEE pervasive computing*, no. 3, pp. 24–33, 2003.

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

[5] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA workshop on open source software*. Japan, 2009.

[6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[7] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Monte Carlo localization for mobile robots," in *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1999.

[8] A. Canedo-Rodríguez, V. Alvarez-Santos, C. V. Regueiro, R. Iglesias, S. Barro, and J. Presedo, "Particle filter robot localisation through robust fusion of laser, WiFi, compass, and a network of external cameras," *Information Fusion*, vol. 27, pp. 170–188, 2016.

[9] S. Gu, Z. Xiang, Y. Zhang, and Q. Qian, "A multi-position joint particle filtering method for vehicle localization in urban area," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[10] A. R. Rormero, P. V. K. Borges, A. Pfrunder, and A. Elfes, "Map-aware particle filter for localization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[11] A. Notsu, K. Honda, H. Ichihashi, Y. Komori, and Y. Iwamoto, "Improvement of particle filter for reinforcement learning," in *2011 10th International Conference on Machine Learning and Applications and Workshops*, 2011.

[12] F. Pomerleau, P. Krüsi, F. Colas, P. Furgale, and R. Siegwart, "Long-term 3D map maintenance in dynamic environments," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[13] 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.

[14] P. Dinnissen, S. N. Givigi, and H. M. Schwartz, "Map merging of multi-robot SLAM using reinforcement learning," in *2012 IEEE International Conference on Systems, Man, and Cybernetics*, 2012.

[15] T. Kollar and N. Roy, "Using reinforcement learning to improve exploration trajectories for error minimization," in *2006 IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

[16] N. Arana-Daniel, R. Rosales-Ochoa, and C. López-Franco, "Reinforced-SLAM for path planing and mapping in dynamic environments," in *2011 8th International Conference on Electrical Engineering, Computing Science and Automatic Control*, 2011.

[17] R. Liu, C. Yuen, T. Do, D. Jiao, X. Liu, and U. Tan, "Cooperative relative positioning of mobile users by fusing IMU inertial and UWB ranging information," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 5623–5629.

[18] D. Fox, "Adapting the sample size in particle filters through KLD-sampling," *The International Journal of Robotics Research*, vol. 22, no. 12, pp. 985–1003, 2003.

[19] R. Douc and O. Cappe, "Comparison of resampling schemes for particle filtering," in *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, Sept 2005.

[20] N. M. Nasrabadi, "Pattern recognition and machine learning," *Journal of Electronic Imaging*, vol. 16, 2007.

[21] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *IEEE ICRA*, 1985.

[22] R. Cruz, L. Garrote, A. Lopes, and U. J. Nunes, "Modular software architecture for human-robot interaction applied to the InterBot mobile robot," in *2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, April 2018.

[23] F. F. Campillo, G. G. Canales, and L. Mevel, "Particle based confidence intervals," in *2. International Operational Modal Analysis Conference*, Copenhague, Denmark, Apr. 2007.

[24] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable SLAM system with full 3D motion estimation," in *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, November 2011.