Lifelong Localization and Dynamic Map Estimation in Changing Environments

Author Names Omitted for Anonymous Review. Paper-ID [130]

***Abstract*—Robot localization systems typically assume that the environment is static, ignoring the dynamics inherent in most real world scenarios like parking lots, warehouses and even offices and households. In such environments the configuration of certain objects such as cars, goods, or furniture can change with time leading to inconsistent observations with respect to previously learned maps and thus decreasing the localization accuracy. In this paper we present a novel probabilistic approach to lifelong localization in changing environments, where the robot pose and the environment state are jointly estimated using a Rao-Blackwellized particle filter. To describe the environment, we utilize a hidden Markov model formulation. Exploiting several characteristics of this model, we can considerably speed up the estimation procedure. This makes it feasible to run our algorithm online. Experimental results obtained with a real robot in a dynamically changing environment demonstrate that our approach can reliably adapt to changes in the environment and that it significantly outperforms standard localization techniques.**

I. Introduction

Long term operation of mobile robots in changing environments has become a major focus of interest in robotics research in recent years, as this ability is required for robots navigating in the real world. One of the most challenging tasks in this context is that of dealing with the dynamic aspects of the environment. Many existing approaches to robot navigation assume that the world is static and apply models which treat dynamic objects as outliers [1, 2, 3]. For highly dynamic objects like moving people or cars, these methods typically work quite well, but they are less effective for *semi-static* objects. By semi-static objects we mean objects which change their state slowly or seldom, like parked cars, doors, pallets in warehouses or furniture which can be moved. In many realistic scenarios (see Figure 1), in which robots operate for extended periods of time, semi-static objects are ubiquitous and ignoring them can substantially deteriorate the navigation performance.

In this paper, we present a novel approach to lifelong localization in changing environments, which explicitly takes into account the dynamics of the environment. The approach is able to distinguish among objects that presents fast dynamic behaviors, e.g., cars and people, objects that can be moved around and change configuration, e.g., boxes, shelfs, doors, and objects that are static and do not move around, e.g., walls. To represent the environment, we use a *dynamic occupancy grid* [4], which employs hidden Markov models on a two-dimensional grid to represent the occupancy and the corresponding transition probabilities for each cell of this grid. We first learn the parameters of this model using a variant of the expectation maximization (EM) algorithm and then use this information to jointly estimate the pose of the robot and the state of the environment during global localization. We employ a Rao-Blackwellized particle filter (RBPF), in which the robot pose is represented by the sampled part of the filter, and the occupancy probability of a cell is represented in the analytical part of the factorization. We further propose a map management method, which uses a local map representation that is able to forget changes in a sound probabilistic way, by considering the mixing times of the associated Markov chain, and to minimize memory requirements.

Fig. 1. A robot navigating in a parking lot at noon (top) and at 6 pm (bottom). Note that despite being at the same spot in both cases, the observations will be substantially different due to the changed number of parked cars. The pictures have been anonymized for double blind submission: the robot has been obfuscated and real buildings have been replaced by famous ones (Dancing House and Taj Mahal.

Compared to previous approaches, our algorithm has several desirable advantages. First, it improves the robustness and accuracy of the pose estimate of the robot. Second, our method is able to provide an up-to-date map of the environment. Finally, our map management method considerably reduces the runtime of the process whilst minimizing the memory requirements. As a result, our approach allows a robot to

simultaneously perform the estimation of its pose and the potentially changing state of the environment in an online fashion. To the best of our knowledge, this is the first approach to address this problem in a general and systematic way. Previous attempts either focused on how to filter spurious observations due to dynamic objects, focused only on limited areas or individual elements of the map or specifically addressed the problem of pose tracking.

This paper is organized as follows. After discussing related work in Section II, we briefly summarize the dynamic occupancy grid representation in Section III. In Section IV, we explain the algorithms for the joint estimation and the map management. Finally, in Section V, we present extensive experiments carried out with a real robot, showing that in changing environments, our approach significantly outperforms state-of-the-art localization methods and generates consistent maps of the environment.

II. Related Work

Most mobile robot navigation systems rely on a map of the environment built beforehand in an offline phase. A popular approach to deal with subsequent changes in the environment is to filter out sensor measurements caused by dynamic objects. Several approaches rely on probabilistic sensor models that identify the measurements that are inconsistent with the map. For example, Fox *et al.* [1] use an entropy gain filter and a distance filter based on the expected distance of a measurement, while Schultz *et al.* [2] uses minima of the laser scan together with an already available map. Orthogonal to the work on localization in dynamic environments, many authors have addressed the problem of modeling such environments.

Other approaches specifically focus on building two maps, one for the static part of the environment and one that accounts for dynamic objects. Wolf and Sukhatme [5] propose a model that maintains two separate occupancy grids, one for the static parts of the environment and the other for the dynamic parts. Wang *et al.* [6] formulate a general framework for mapping and dynamic object detection and developed a system to detect if a measurement is caused by a dynamic object. Montesano *et al.* [7] extends the approach by jointly considering the problem of dynamic objet detection with the one of mapping, by including the error estimation of the robot in the classifier.

Despite the success of these techniques, they discard potentially valuable information about the environment and considers dynamic objects as outliers to be discarded. Instead of filtering out inconsistent measurements, Montemerlo *et al.* [8] use them for people tracking while localizing the robot. They show that the state of dynamic objects in the environment can be used to increase the robustness of the pose estimation process. Motivated by this idea, we utilize all sensor measurements to keep the map of the environment up-to-date while simultaneously localizing the robot within the updated map.

Biber and Duckett [9] propose a model that represents the environment on multiple timescales simultaneously. For each timescale a separate sample-based representation is maintained and updated using the observations of the robot according to an associated timescale parameter. Our extends it by providing a sound model of the change and the possibility of inferring the optimal timescal parameter for each grid cell. Recently, Yang and Wang [10] proposed the *feasibility* grids to facilitate the representation of both the static scene and the moving objects, where the dual sensor model is used to discriminate between stationary and moving objects in mobile robot localization. Those approaches, however, assumes that the position of the robot is known with a certain accuracy to compute and update the maps and therefore are not suited to be used in a global localization problem.

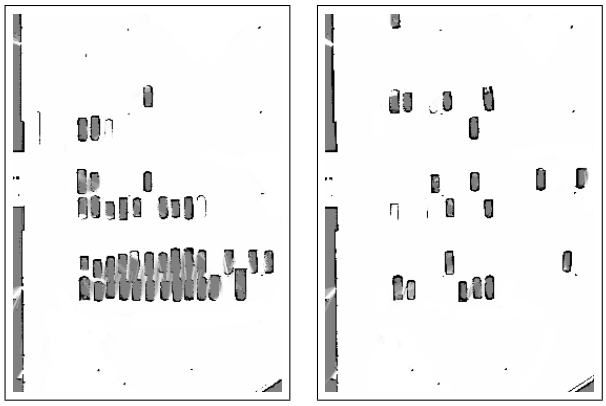

Fig. 2. Parked cars in a parking lot at 9am (left) and 6pm (right).

Whereas the majority of the work on mapping in dynamic environments assumes that a good estimate of the robot's pose is available, most of the work on mapping where the pose of the robot is not available (i.e., SLAM) assumes that the environment is static. Only few authors address the problem of jointly estimating the pose of the robot and the state of a dynamic environment. Murphy *et al.* [11] showed how a Rao-Blackwellized particle filter solution to the SLAM problem can deal with dynamic maps in a theoretical way. Their approach, however, is not able to handle real world scenarios and computes a map in the robot frame, thus limiting its applicability in global localization settings. Avots *et al.* [12], for example, use a Rao-Blackwellized particle filter to estimate the pose of the robot and the state of doors in the environment. They represent the environment using a reference occupancy grid where the location of the doors is known, but not their state (i.e., opened or closed). Petrovskaya and Ng [13] propose a similar approach where instead of a binary model, a parametrized model (i.e., opening angle) of the doors is used. Similar to these approaches, we also use a Rao-Blackwellized particle filter to estimate the pose of the robot and the state of the environment. Stachniss and Burgard [14] estimate typical configurations of dynamic areas in the environment of a mobile robot. Their approach clusters local grid maps to identify the possible configurations and uses these clusters to localize a mobile robot in a non-static environment. Similarly, Meyer-Delius *et al.* [15] keep track of the observations caused by unexpected objects in the environment using temporary local maps. The

robot pose is then estimated using a particle filter that relies both on these temporary local maps and on a reference map of the environment. Lately, an interesting approach to lifelong mapping in dynamic environments has been presented [16]. The approach focuses mainly on visual maps, and presents a framework where local maps (views) can be updated over time and new local maps are added/deleted when the configuration of the environment changed. In contrast to their methods, however, we estimate the state of the complete environment, and not only of a small, specific area or element. Additionally, we also learn the model parameters from data and we are able to generalize over unforseen environment configurations.

III. Dynamic Occupancy Grid

Occupancy grids [17] are one of the most popular representations of a mobile robot's environment. They partition the space into rectangular cells and store, for each cell, a probability value indicating whether the underlying area of the environment is occupied by an obstacle or not.

One of the main disadvantages of occupancy grids is that they assume the environment to be static. To be able to deal with changes in the environment we utilize a *dynamic occupancy grid* [4], a generalization of an occupancy grid that overcomes the static-world assumption by explicitly accounting for changes in the environment. A dynamic occupancy grid relies on an HMM (see Rabiner [18]) to explicitly represent both the belief about the occupancy state and state transition probabilities of each grid cell.

We assume that the map consists of a collection of individual cells, $m_t = \{c_t^{(i)}\}$, following the standard occupancy grid assumptions, and each cell is modeled using an Hidden Markov Model (HMM). The state transition probabilities $p(c_t \mid c_{t-1})$ of each HMM describe how the occupancy state of cell c changes between consecutive time steps. Since a cell c can be either free (*f*) or occupied (*o*), the state transition model is specified using only two transition probabilities, namely $p(c_t = f \mid c_{t-1} = f)$ and $p(c_t = o \mid c_{t-1} = o)$. Note that assuming a stationary process for the changes in the environment, these probabilities do not depend on the absolute value of t. Standard occupancy grids are a special case of dynamic occupancy grids where the transition probabilities $p(c_t = f \mid c_{t-1} = f)$ and $p(c_t = o \mid c_{t-1} = o)$ are 1 for all cells c.

The estimation of the occupancy state of a cell follows a Bayesian approach according to

$$p(c_t \mid z_{1:t}) = \eta\, p(z_t \mid c_t) \sum_{c_{t-1} \in \{f,o\}} p(c_t \mid c_{t-1})\, p(c_{t-1} \mid z_{1:t-1}), \quad (1)$$

where $p(z_t \mid c_t)$ and $p(c_t \mid c_{t-1})$ correspond respectively to the observation and transition models of the cell and η is a normalization constant. The observation model specifies the likelihood of measuring a cell as occupied or free and is assumed to depend only on the sensor used and not on the location, therefore it is specified beforehand and is the same for each HMM. As explained in [4], the transition model and the observation model for each cell is estimated from observed data using an instance of the expectation-maximization (EM) algorithm.

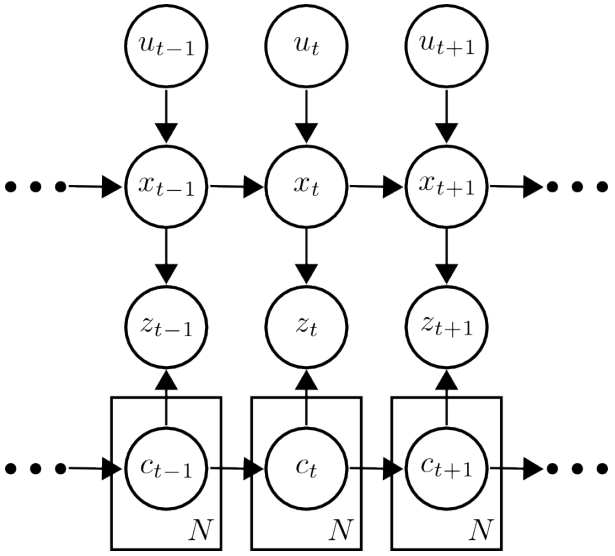

Fig. 3. Dynamic Bayesian network describing the localization problem addressed in the paper. Note the temporal connections among the individual cells of different steps.

IV. Simultaneous Localization and Dynamic State Estimation

In this section we describe our approach to simultaneously estimate the robot pose and the dynamic state of the environment. Although on first sight one can see the addressed problem as an instance of the better known simultaneous localization and mapping (SLAM), there are two main differences between them.

The first difference is the absence of a global reference frame in the SLAM problem. No global pose is required and the initial pose of the robot can typically be set freely. On the contrary, we explicitly address global localization as part of the estimation aspect. We have a global reference frame and the initial pose of the robot is unknown and assumed to be uniformly distributed over the whole environment. The second difference regards the dimensionality of the map. In the SLAM problem, the size of the map is not known in advance and can grow unbounded with time. In our problem the size of the map is known and we only focus on estimating the actual configuration of the dynamic objects present in the environments. A graphical model representation of the problem is illustrated in Figure 3. Despite the differences, the two problems do share the same state space, i.e., the robot pose and the state of the map, and one can exploit the same factorization that made Rao-Blackwellized particle filters a feasible solution to the SLAM problem [11, 19].

In the following we show how this factorization can be exploited and we derive the RBPF that will be used to estimate the posterior $p(x_t, m_t \mid z_{1:t}, u_{0:t-1}, m_{t-1})$ about the robot

pose $x_{1:t}$ and the configuration of the environment m_t, given the observations $z_{1:t}$, the odometry measurements $u_{0:t-1}$ and the prior over the map m_0. The key idea is to separate the estimation of the robot pose from the map estimation process,

$$p(x_t,m_t \mid z_{1:t}, u_{1:t-1}, m_{t-1}) = p(m_t \mid x_t, z_{1:t}, m_{t-1}) p(x_{1:t} \mid z_{1:t}, u_{1:t-1}, m_{t-1}). \quad (2)$$

This can be done efficiently, since the posterior over maps $p(m_t \mid x_{1:t}, z_{1:t}, m_{t-1})$ can be computed analytically given the knowledge of x_t, $z_{1:t}$ and m_{t-1} and using the forward algorithm for the HMM. The remaining posterior, $p(x_t \mid z_{1:t}, u_{1:t-1}, m_{t-1})$, is estimated using a particle filter which incrementally processes the observations and the odometry readings. Following Doucet *et al.* [20], we can use the motion model $x_t^{(i)} \sim p(x_t|x_{t-1}^{(i)}, u_{t-1})$ as proposal distribution $\pi(x_t)$ to obtain a sample of the robot pose. This recursive sampling schema allows us to recursively compute the importance weights using the following equation

$$\begin{aligned} w_t^{(i)} &= w_{t-1}^{(i)} \frac{p(z_t \mid x_t^{(i)}, z_{1:t-1}, m_{t-1}) p(x_t^{(i)} \mid x_{t-1}^{(i)}, u_{t-1})}{\pi(x_t^{(i)})} \\ &= w_{t-1}^{(i)} \frac{p(z_t \mid x_t^{(i)}, z_{1:t-1}, m_{t-1}) p(x_t^{(i)} \mid x_{t-1}^{(i)}, u_{t-1})}{p(x_t^{(i)} \mid x_{t-1}^{(i)}, u_{t-1})} \\ &= w_{t-1}^{(i)} p(z_t \mid x_t^{(i)}, z_{1:t-1}, m_{t-1}). \end{aligned} \quad (3)$$

The observation likelihood is then computed by marginalization over the predicted state of the map leading to

$$\begin{aligned} & p(z_t \mid x_t^{(i)}, z_{1:t-1}, m_{t-1}) \\ &= \int p(z_t \mid x_t^{(i)}, m_t) p(m_t \mid m_{t-1}^{(i)}) dm_t \\ &= \prod_j \mathcal{N}(z_t^j; \hat{z}_t^j, \sigma^2), \end{aligned} \quad (4)$$

were z^j is an individual range reading and $\hat{z}_t^j$ is the range to its closest cell in the map with an occupancy probability above a certain threshold. The map motion model $p(m_t \mid m_{t-1}^{(i)})$ is computed using the HMM and described in section III. Note that the *disappearance* of the integral is not an approximation but a direct consequence of using the likelihood field model [21].

The overall process for simultaneously estimating the robot's pose and the dynamic state of the environment proposed in this paper is summarized in Algorithm 1.

Algorithm 1: RBPF for Changing Environments

In: The previous sample set $\mathcal{S}_{t-1}$, the stable map m_0, the current observation z_t and odometry u_t
Out: The new sample set $\mathcal{S}_t$

$\mathcal{S}_t = \{\}$
foreach $s_{t-1}^{(i)}$ *in* $\mathcal{S}_{t-1}$ **do**
  $< x_{t-1}^{(i)}, m_{t-1}^{(i)}, w_{t-1}^{(i)} >= s_{t-1}^{(i)}$
  `// State prediction`
  $x_t^{(i)} \sim p(x_t|x_{t-1}^{(i)}, u_{t-1})$
  foreach c_{t-1} *in* $m_{t-1}^{(i)}$ **do**
    $p(c_t \mid z_{1:t-1}) = \sum_{c_{t-1} \in \{f,o\}} p(c_t \mid c_{t-1}) p(c_{t-1} \mid z_{1:t-1})$
  end
  `// Weight computation`
  $w_t^{(i)} = w_{t-1}^{(i)} \prod_j \mathcal{N}(z_t^j; \hat{z}_t^j, \sigma^2)$
  `// Map update`
  foreach c_t *in* $m_t^{(i)}$ **do**
    $p(c_t \mid z_{1:t}) = \eta p(z_t \mid c_t) p(c_t \mid z_{1:t-1})$
  end
  $\mathcal{S}_t = \mathcal{S}_t \cup \{< x_t^{(i)}, m_t^{(i)}, w_t^{(i)} >\}$
end
`// Normalize the weights`
normalize($\mathcal{S}_t$)
`// Resample`
$N_{eff} = \frac{1}{\sum_i (w_t^{(i)})^2}$
if $N_{eff} < T$ **then**
  $\mathcal{S}_t$ = resample($\mathcal{S}_t$)
end

A. Map Management

As we already mentioned above, the initial pose of the robot is unknown and assumed to be uniformly distributed over the environment. This forces us to use a high number of particles, generally above thousands, to accurately represent the initial distribution. Since every particle needs to have its own estimate of the map, memory management is a key aspect of the whole algorithm. In order to save memory, we want to only store the cells in the map that have been considerably changed from the a priori map m_0, which is shared among the diverse particles. This is done by exploiting two important aspects of the Markov chain associated to the HMM: the *stationary distribution* and the *mixing time*.

As the number of time steps for that no observation is available tends to infinity, the occupancy value of a cell converges to a unique stationary distribution π [22]. This stationary distribution represents the case where the environment has not been observed for a long time and is represented by our a priori map m_0. In the case of a binary HMM as the one used in this paper, this distribution is computed using the transition probabilities

$$\begin{bmatrix} \pi_f \\ \pi_o \end{bmatrix} = \frac{1}{p+q} \begin{bmatrix} q \\ p \end{bmatrix}, \quad (5)$$

where for notation simplicity we have

$$\begin{aligned} p &= p(c_t = o \mid c_{t-1} = f), \\ q &= p(c_t = f \mid c_{t-1} = o). \end{aligned}$$

Every time an individual particle observes the state of a cell for the first time, the state distribution of that particular

cell changes from the stationary one and the particle needs to store the new state of the cell. In order to reduce memory requirements, only a limited number of cells should be stored and a *forgetting* mechanism should be implemented. This can be done in a sound probabilistic way, by exploiting the mixing time of the associated Markov chain. The mixing time is defined as the time needed to converge from a particular state to the stationary distribution. The concrete definition depends on the measure used to compute the difference between distributions. In this paper we use the total variation distance as defined by Levin *et al.* [22]. Since our HMMs have only two states, the total variation distance Δ_t between the stationary distribution π and the occupancy distribution p_t at time t can be specified as

$$\Delta_t \quad = \quad |1-p-q|^t \Delta_0, \tag{6}$$

where $\Delta_0 = |p(c_t = f) - \pi_f| = |p(c_t = o) - \pi_o|$ is the difference between the current state $p(c_t)$ and stationary distribution π. Based on the total variation distance, we can define the mixing time t_m as the smallest t such that the distance Δ_{t_m} is less than a given confidence value ϵ. This leads to

$$t_m = \left\lceil \frac{ln(\epsilon/\Delta_0)}{ln(|1-p-q|)} \right\rceil. \tag{7}$$

In other words, the mixing time tells us how many steps are needed for a particular cell to return to its stationary distribution, given an approximation error of ϵ, i.e., how many steps a particle needs to store an unobserved cell before removing it from its local map and rely on the a priori map m_0.

It is worth to notice that the approach employed here is orthogonal to the work of Eliazar and Parr [23], where map memory consumption is reduced by exploiting parent-child relationship between resampled particles. Note, further, that when the global localization is initialized, the particle still need to store an individual map each, since no parent particle is present.

B. Time and Memory Complexity

In this section we describe the time and memory complexity of the presented algorithm in more details. First of all, the algorithm, as the original MCL one, depends on the number of particles used, which in turns depends on the size of the free space. Although, there exists techniques to estimate this number based on using statistical divergences with respected to a fixed target distribution, this goes beyond the purpose of this paper and we focus solely on the complexity of updating an individual particle.

The analysis of time and memory consuption strongly depends on how much we want to trade memory for time. To be more precise, in case we have infinite (or big enough) memory, we could simply use a full map for each particle, ending up with a costant time prediction and update for each particle, but at the cost of storing N copies of the map. Let N be the number of particles and M the size of the map, we have an asymptotic time complexity of $O(N)$ and an asymptotic memory complexity of $O(NM)$.

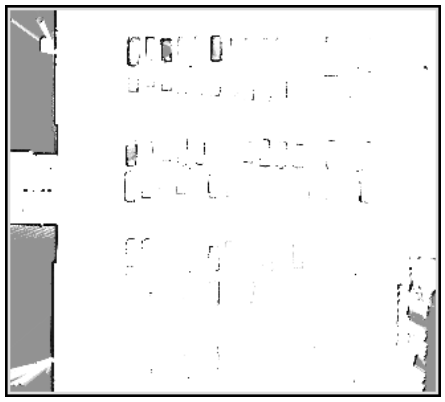

Fig. 4. Prior map used in the experiments.

On the other spectrum, if we fully exploit the mixing time and the stable distribution, each particle only need to store a small local map, consisting of the cells that have been observed lately. More formally, let consider k being the average mixing time and A the averae number of cells observed in each time step, we have that the average number of cells in the local map is given by kA. Since those values are costant both with respect to the map size and to respect with the particle numbers, we have the same asymptotic time complexity of the naive implementation, $O(N)$, but with a lower memory complexity of $O(M+N)$.

The cost that we pay is simply a multiplication factor for looking up the cells in the local map, which can be implemented efficiently using kd-trees or hash tables.

V. Experiments

We tested our proposed approach using a data set collected with a MobileRobots Powerbot equipped with a SICK LMS laser range finder. In the data set, the robot has been steered through a general parking lot, performing a run every full hour from 7am until 6pm during one day. The range data obtained from the twelve runs (data sets d_1 through d_{12}) corresponds to twelve different configurations of the parked cars, including an almost empty parking lot (data set d_1) and a relatively occupied one (data set d_8). A standard SLAM approach for static environments [19] has been used to correct the odometry of the robot in each data set and thereby obtain a good estimate of its pose to use as ground truth. The underlying assumption here is that the parking lot did not change considerably during a run. Furthermore, the trajectory and velocity of the robot during each run were approximately the same to avoid a bias in the complete data set.

In order to assess the performances of the localization approach, we compared it to standard Monte Carlo localization both in a global localization and pose tracking setting. Furthermore, we compared our approach with the one of Meyer-Delius *et al.* [15] but only in the pose tracking case, since it employs standard MCL before the filter reaches convergence. Comparison with other approaches is not possible since they

TABLE I
GLOBAL LOCALIZATION EXPERIMENT

Data set	MCL-GT			RBPF			MCL-S		
	Success	Error^2	σ^2	Success	Error^2	σ^2	Success	Error^2	σ^2
01	100%	0.21	0.36	50%	0.26	0.36	50%	0.26	0.18
02	100%	0.19	0.29	40%	0.10	0.08	33%	0.13	0.09
03	100%	0.13	0.19	80%	0.10	0.29	52%	0.19	0.17
04	100%	0.04	0.03	60%	0.08	0.14	53%	0.15	0.19
05	100%	0.07	0.18	54%	0.07	0.09	35%	0.15	0.18
06	100%	0.02	0.01	87%	0.02	0.02	45%	0.06	0.02
07	100%	0.06	0.08	59%	0.12	0.22	43%	0.14	0.20
08	100%	0.05	0.10	71%	0.03	0.02	28%	0.03	0.01
09	100%	0.02	0.01	53%	0.12	0.22	31%	0.06	0.02
10	100%	0.14	0.28	62%	0.13	0.31	34%	0.30	1.01
11	100%	0.11	0.11	38%	0.15	0.21	26%	0.24	0.29
12	100%	0.19	0.32	20%	0.16	0.14	22%	0.27	0.38
Total	100%	0.11	0.19	52%	0.11	0.18	36%	0.17	0.22

assume either a fixed number of configurations [14] or focused on a limited set of dynamic objects (e.g., doors) [12, 13]. For each data set, we compared our approach (RBPF), MCL using the standard occupancy grid (MCL-S), MCL using the ground-truth map for that specific data set (MCL-GT), and MCL using the temporary maps (MCL-TM). We performed 100 runs for each data set, where we randomly sampled the initial pose of the robot. In order to obtain a fair comparison, the same seed has been used to generate the initial pose, as well as to perform all the random sampling processes for each approach. All the approaches have been initialized with $10,000$ particles for global localization and 500 particles for pose tracking. They also used the same set of parameters, an occupancy threshold of 0.6 and a maximum distance of 1m for the likelihood field model.

The static maps used for MCL-GT have been computed using the corrected log files from the SLAM algorithm and rendering the points into an occupancy grid. The static map for MCL-S has been computed using all the logfiles as they would be a unique run of the SLAM algorithm over the whole day. The dynamic map has been estimated using n-fold cross validation over the corrected log files. The RBPF has been then initialized using the stable distribution of the HMM as prior map m_0. Figure 4 shows how the probability distribution of the prior map looks like.

The results of the global localization experiment are shown in Table I. The table shows the success rate of the global localization, as the percentage of time the filter converged to the true pose, and the residual squared error, with respective variance, after convergence. The success rate is reported relative to the result of MCL on the ground-truth map, in order to have a measure independent of the complexity of the environment. The results show that our approach outperforms the standard MCL on static maps both in terms of convergence rate and accuracy in localization.

Table II shows the results for the pose tracking experiment, where the filter is initialized around the true pose and keeps tracking the robot. The table shows the failure rate, i.e., the percentage of time the robot got lost during tracking, as well as the residual squared error in the case the tracking was succesful. The results of this experiment show that the performance of our approach in pose tracking is almost equivalent to MCL with the ground-truth maps, with a failure rate of only 2%. It is worth to notice the comparison with the temporary map approach [15] in this experiments. Looking at the tables, we get two important messages. The first message is that the proposed approach is always more precise in terms of residual error. This come with no surprise, since the estimation of the local surrounding is in some sense constraint with the map geometry and static objects can only appear in places where they have been seen during learning. On the contrary, the dynamic maps get initialized with the current pose estimate of the robot and introduce a bias in the estimation which is almost impossible to remove. The second message is that the proposed approach is more robust to the changes and initialization. This is evident from the failure rate, where the temporary maps approach is almost always on par with the RBPF but in three cases, and in one case is even worse than standard MCL. The problem is that if the temporary map is created from a wrong position, there is no possibility to recover, the worst case being when the observations matching the prior map are considered as outliers.

In terms of runtime and size of the local map, we experienced and average mixing time of $k = 10$ and an average size of the local map of about 250 cells. The original map size is 369x456 pixels with a resolution of 0.1 m, resulting in a memory saving of about three orders of magnitude with respect to the naive RBPF.

A frame to frame comparison between the proposed approach and standard MCL is shown in Figure 5. Both algorithms have the same parameters and the same seed for the random number generator. As it can be seen, MCL converges too fast to a wrong solution, believing that the measurements coming from parked cars (not present in the a priori map) have been generated by the wall on the bottom right (frame 8). The proposed approach, on the other hand, has a slower convergence rate due to the uncertainty in the map estimate. After a few frames (frame 12) it finally converges to the right

TABLE II
POSITION TRACKING EXPERIMENT

Data set	MCL-GT			RBPF			MCL-S			MCL-TM		
	Failure	Error^2	σ^2	Failure	Error^2	σ^2	Failure	Error^2	σ^2	Failure	Error^2	σ^2
01	0%	0.04	0.01	3%	0.09	0.03	5%	0.18	0.07	0%	0.25	0.16
02	0%	0.03	0.01	4%	0.08	0.05	24%	0.18	0.10	0%	0.16	0.04
03	0%	0.04	0.01	2%	0.05	0.04	10%	0.09	0.04	12%	0.63	0.45
04	0%	0.02	0.01	0%	0.04	0.01	10%	0.08	0.02	29%	0.63	0.51
05	0%	0.02	0.01	3%	0.03	0.04	13%	0.06	0.02	1%	0.51	0.31
06	0%	0.02	0.01	2%	0.02	0.01	26%	0.09	0.12	0%	0.21	0.05
07	0%	0.02	0.01	0%	0.03	0.01	34%	0.07	0.01	1%	0.44	0.24
08	0%	0.02	0.01	2%	0.02	0.01	35%	0.09	0.15	35%	0.59	0.56
09	0%	0.02	0.01	4%	0.03	0.01	37%	0.07	0.16	4%	0.49	0.31
10	0%	0.02	0.01	0%	0.03	0.01	36%	0.09	0.10	0%	0.32	0.12
11	0%	0.03	0.01	1%	0.05	0.02	42%	0.10	0.05	1%	0.47	0.28
12	0%	0.03	0.01	5%	0.06	0.01	44%	0.15	0.20	0%	0.23	0.04
Total	0%	0.03	0.01	2%	0.04	0.02	27%	0.10	0.08	7%	0.41	0.25

position. In the last frame one can see the updated map, which better reflects the current configuration of the environment.

Both experiments show two important aspects of the problem and of the solution adopted. The first aspect is that the problem is much more complex than global localization since the search space is bigger and deciding if a measurement is an outlier or is caused by a change of the configuration is not a trivial task. Furthermore, analyzing the performance results in pose tracking, we see that if the filter is initialized close to the correct solution, i.e., the search is reduced to the correct subspace, it is able to estimate the correct configuration. The second aspect is how the algorithm scales with different amounts of change in the environment configuration. In the first four data sets, the parking lot is almost empty and it becomes quite full in the last ones. This is evident, when analyzing the results of MCL on the static maps, since the performance gets worse with an increasing amount of change. On the other hand, the performance of our approach is less sensible to the amount of change in case of global localization and is even independent from that in case of position tracking, as can be seen from the two tables.

VI. CONCLUSIONS

In this paper, we presented a probabilistic localization framework which recursively estimates not only the pose of the robot, but also the state of the environment. Our algorithm uses a hidden Markov model to represent the dynamics of the environment and employs a Rao-Blackwellized particle filter to efficiently estimate the joint state. It explicitly exploits the properties of Markov chains to reduce the memory requirements so that it can be run online on a real robot. Our approach has the main advantages that it provides robust localization even in changing environments and that it additionally supplies up-to-date maps of the environment. We evaluated it using real-world data. The results demonstrate that our model significantly outperforms standard Monte-Carlo localization on static maps. This makes our method more suitable for long-term operation of mobile robots in changing environments.

In future, we would like to extend our model to reason about objects and not only about individual cells and experimenting with different models to encode the change (e.g. Dynamic Bayesian Networks and second order Hidden Markov Models). This will bring a novel perspective on how to reason about correlations in a grid map, as well as interestins issues such as moving object detection and motion segmentation. We also plan on

REFERENCES

[1] D. Fox, W. Burgard, and S. Thrun, "Markov localization for mobile robots in dynamic environments," *Journal of Artificial Intelligence Research*, vol. 11, 1999.

[2] D. Schulz, D. Fox, and J. Hightower, "People tracking with anonymous and id-sensors using rao-blackwellised particle filters," in *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, 2003. [Online]. Available: http://dl.acm.org/citation.cfm?id=1630659.1630792

[3] D. Hähnel, R. Triebel, W. Burgard, and S. Thrun, "Map building with mobile robots in dynamic environments," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2003.

[4] D. Meyer-Delius, M. Beinhofer, and W. Burgard, "Grid-based models for dynamic environments," Dept. of Computer Science, University of Freiburg, Tech. Rep., 2011.

[5] D. F. Wolf and G. S. Sukhatme, "Mobile robot simultaneous localization and mapping in dynamic environments," *Autonomous Robots*, vol. 19, no. 1, pp. 53–65, 2005.

[6] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *Int. J. Rob. Res.*, 2007. [Online]. Available: http://dl.acm.org/citation.cfm?id=1286136.1286141

[7] L. Montesano, J. Minguez, and L. Montano, "Modeling the static and the dynamic parts of the environment to improve sensor-based navigation," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2005. [Online]. Available: http://dx.doi.org/10.1109/ROBOT.2005.1570822

[8] M. Montemerlo, S. Thrun, and W. Whittaker, "Conditional particle filters for simultaneous mobile robot

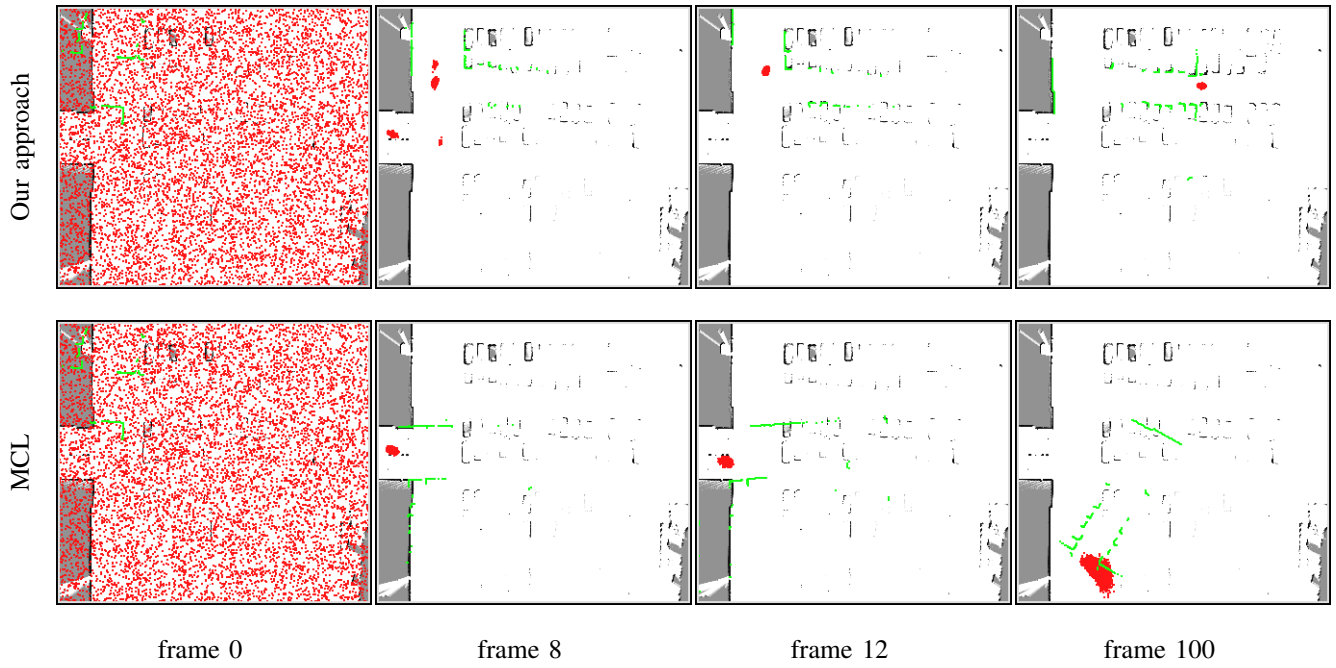

Fig. 5. Comparison between the proposed approach (top) and MCL (bottom) in a global localization setting. The MCL converges too fast and to a wrong position (frame 8), while the proposed approach needs more time to better estimate the current configuration (frame 12). The last frame shows the updated map with the current configuration.

localization and people-tracking," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2002.

[9] P. Biber and T. Duckett, "Dynamic maps for long-term operation of mobile service robots," in *Proc. of Robotics: Science and Systems (RSS)*, 2005.

[10] S.-W. Yang and C.-C. Wang, "Feasibility grids for localization and mapping in crowded urban scenes," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.

[11] K. Murphy, "Bayesian map learning in dynamic environments," in *Proc. of the Conf. on Neural Information Processing Systems (NIPS)*, Denver, CO, USA, 1999, pp. 1015–1021.

[12] D. Avots, E. Lim, R. Thibaux, and S. Thrun, "A probabilistic technique for simultaneous localization and door state estimation with mobile robots in dynamic environments," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2002.

[13] A. Petrovskaya and A. Y. Ng, "Probabilistic mobile manipulation in dynamic environments, with application to opening doors," in *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, 2007.

[14] C. Stachniss and W. Burgard, "Mobile robot mapping and localization in non-static environments," in *Proc. of the Nat. Conf. on Artificial Intelligence (AAAI)*, 2005.

[15] D. Meyer-Delius, J. Hess, G. Grisetti, and W. Burgard, "Temporary maps for robust localization in semi-static environments," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, 2010.

[16] K. Konolige and J. Bowman, "Towards lifelong visual maps," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1733343.1733559

[17] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 1985.

[18] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition." in *Proceedings of the IEEE*, vol. 77 (2), 1989, pp. 257–286.

[19] 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, 2007.

[20] A. Doucet, J. de Freitas, K. Murphy, and S. Russel, "Rao-Blackwellized partcile filtering for dynamic bayesian networks," in *Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI)*, Stanford, CA, USA, 2000, pp. 176–183.

[21] S. Thrun, "A probabilistic online mapping algorithm for teams of mobile robots," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 335–363, 2001.

[22] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times*. American Mathematical Society, 2008.

[23] A. I. Eliazar and R. Parr, "Dp-slam 2.0," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2004. [Online]. Available: http://dx.doi.org/10.1109/ROBOT.2004.1308006