Probabilistic Mobile Manipulation in Dynamic Environments, with Application to Opening Doors

Anna Petrovskaya and Andrew Y. Ng
Stanford University
Computer Science Department
{ anya, ang }@cs.stanford.edu

Abstract

In recent years, probabilistic approaches have found many successful applications to mobile robot localization, and to object state estimation for manipulation. In this paper, we propose a unified approach to these two problems that dynamically models the objects to be manipulated and localizes the robot at the same time. Our approach applies in the common setting where only a low-resolution (10cm) grid-map of a building is available, but we also have a high-resolution (0.1cm) model of the object to be manipulated. Our method is based on defining a unifying probabilistic model over these two representations. The resulting algorithm works in real-time, and estimates the position of objects with sufficient precision for manipulation tasks. We apply our approach to the task of navigating from one office to another (including manipulating doors). Our approach, successfully tested on multiple doors, allows the robot to navigate through a hallway to an office door, grasp and turn the door handle, and continuously manipulate the door as it moves into the office.

1 Introduction

Many believe that general-purpose robots will soon inhabit home/office environments and carry out a large variety of tasks, for example fetching items, delivering messages, or cleaning up a room. At a bare minimum, such robots must navigate in these environments as well as interact with them. In this paper, we present a unified probabilistic approach to state estimation for simultaneous manipulation (of objects in the environment) as well as global navigation. Using this approach, we successfully enable a mobile manipulation platform to navigate from far away up to a door, manipulate the door handle so as to open the door, and enter an office while simultaneously continuing to manipulate the door. This work was done as part of the STAIR (STanford Artificial Intelligence Robot) project, which has the long-term goal of building a useful robotic assistant that can carry out home/office tasks such as those described above.

Over the last decade, probabilistic techniques have found great success in mobile robot navigation (e.g., [Fox *et al.*, 1999a]). In much of this literature, the environment is modeled as static (unchanging), and there is no interaction between the robot and the environment. More recently, a number of authors have developed models for non-static environments. For example, [Biswas *et al.*, 2002; Anguelov *et al.*, 2004; 2002; Hähnel *et al.*, 2003] use an off-line EM algorithm to differentiate between static and non-static parts of an environment. A few algorithms also perform on-line mapping while taking into account non-static information. [Wolf and Sukhatme, 2004] use separate occupancy grids for dynamic obstacles (e.g., moving people) and static obstacles; [Stachniss and Burgard, 2005] maintain clusters of local grid maps corresponding to different observed configurations of the environment; and [Biber and Duckett, 2005] model temporal changes of local maps.

Figure 1: STAIR robot platform manipulating a door during one of our experiments.

Robots typically interact with the environment using manipulators. Most work on manipulation focuses on properties of specific objects to be manipulated, rather than on moving in or understanding the global environment (e.g. [Shekhar, 1986; Moll and Erdmann, 2003]). With a few exceptions (e.g., [Petrovskaya *et al.*, 2006; Slaets *et al.*, 2004]), most of this literature also does not have probabilistic basis, and thus at first blush it appears difficult to derive a single unifying model that seamlessly integrates navigation and manipulation.

The task of mobile manipulation combines both navigation and manipulation. Most current work in mobile manipulation treats these as two tasks to be solved separately: First, mobile robotics techniques are used to navigate to a specific point; then, a separate set of techniques is used to localize objects to be manipulated. For example, in the context of door opening, navigation and manipulation of the door handle were considered in [Rhee *et al.*, 2004] and [Petersson *et al.*, 2000]. In both approaches, navigation to the door was performed as a separate task. The door handle is then localized only after the robot is already in front of the door, and a combination of visual servoing and tactile feedback is then used to grasp the door handle, and finally the door is opened by having the robot follow (with some compliance) a pre-scripted set of motions. Localization during door opening or while entering the doorway was not considered.

In this paper, we present a unified, real-time, algorithm that simultaneously models the position of the robot within the environment, as well as the objects to be manipulated. It allows us to consider manipulation of large objects, such as doors, filing cabinets and chairs. When the state of these objects changes, it significantly impacts navigation tasks. Thus our goal is to simultaneously model a dynamic environment as well as localize ourselves within it. Because this objective is reminiscent to that of simultaneous localization and mapping (SLAM), we will find that we can borrow many ideas from SLAM ([Thrun *et al.*, 2005]). However, our objective is also different in two significant ways: First, the environment changes very significantly as we interact with (manipulate) it, and second, the precision required (1-5mm) for manipulation is 1-2 orders of magnitude higher than is typical for most SLAM applications.

Tested successfully on multiple doors, our approach enables our robot to navigate towards, manipulate, and move through a door. (Figure 1.) In contrast to prior art on door opening, we are able to estimate parameters with high precision even during motion of the robot. Thus no additional delay is required to locate the door handle once the robot reaches the door. Further, using the same, seamlessly integrated probabilistic model, the robot is able to precisely estimate the position of the door even while the robot and/or door are in motion, so that the robot can continuously manipulate the door even while it is passing through it.

2 Representation

2.1 Probabilistic model and notation

One of our tasks is to determine the robot's position and orientation within an environment. We denote the robot's pose by $X = (x, y, \theta)$. In this paper we will restrict our attention to manipulating a single dynamic object placed in the environment. Concretely, consider an example where the position of the object is known (a reasonable assumption for doors, filing cabinets, elevators, etc.), but whose shape is governed by an object state parameter α. For example α could be the angle at which a door is open, or the extent to which a drawer is pulled out of a filing cabinet.[1]

[1] In our door-opening application, $\alpha \in \Re$ is a real number denoting the angle of the door. The generalization to vector-valued $\alpha \in \Re^n$, including settings where α also captures the position/orientation of the object being manipulated, offers no special difficulties.

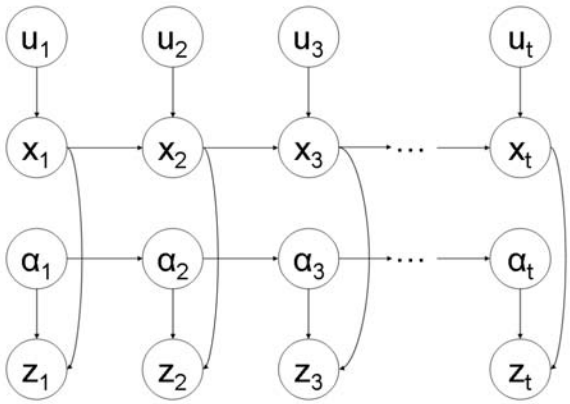


Figure 2: Dynamic Bayesian network model of the robot pose X_t, object state α_t, measurements z_t, and controls u_t.

At each time step t, we give the robot a new motion command, u_t, and obtain a new measurement z_t from its sensors. The robot pose and the object state evolve over time, and at time t are denoted by X_t and α_t respectively. We model X_t, α_t, z_t, and u_t jointly using the dynamic Bayesian network shown in Figure 2. In detail, given the robot pose and a new control, the pose evolves according to a probabilistic motion model derived from robot kinematics:

$$p(X_t|X_{t-1}, u_t).$$

The object state evolves according to:

$$p(\alpha_t|\alpha_{t-1}).$$

Similarly, sensor measurements are governed by a measurement model (discussed in section 2.3 in more detail):

$$p(z_t|\alpha_t, X_t).$$

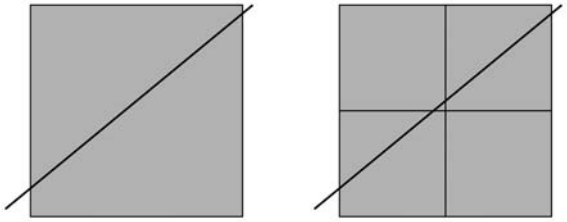
We define the robot trajectory to be a vector of all robot poses up to the current time step, written $X^t = (X_1, X_2, ..., X_t)$. Similarly, we write z^t and u^t to denote all measurements and controls up to time t.

2.2 Representation of environment

Following standard practice in mobile robot navigation, we represent the environment using an occupancy grid map (Figure 3a) of the form typically constructed by mobile robots using SLAM. These coarse maps typically use grid cells that are 10cm x 10cm, and are thus well suited to navigation where 10cm resolution is acceptable. However, manipulation tasks require 1-5mm precision, and constructing a 2mm grid map of an entire building is clearly impractical—both from a memory storage point of view, and because these maps are typically built using noisy sensors. Consequently, we choose to use a combination of high and low resolution maps. We use a high resolution map only for the parts of environment (i.e., the objects) we are interested in manipulating. In the present paper, we use models comprising polygonal objects ("polygon models") to represent these objects at high-resolution;

(a) (b) (c)

Figure 3: (a) Occupancy grid map. (b) Actual environment. (c) Our representation: polygon model is "pasted" onto a grid map. The green box shows the bounding box of the polygon model; and the black lines show the polygon model. (Best viewed in color.)



(a) (b)

Figure 4: Example of laser ray traveling through grid cells.

this representation is well-suited to modeling doors, filing cabinets, straight walls, etc. Figure 3c shows an example polygon model of a door together with the surrounding grid map. Our polygon models also allow us to model the changes in the shape of articulated objects (e.g., opening doors or filing cabinets) in a very natural and efficient manner, simply by letting the position or orientation of some of the polygons be parameters.

Thus, a complete representation of the environment consists of a combination of an occupancy grid map and a polygon model. Further, the polygon model's shape is governed by object's state parameter α.

Our choice of this combination of models is motivated by our goal of having the robot be able to open any door (and enter any office) in our office building. Since all offices in our building are built on a common theme, all doors are essentially identical, and thus it suffices to build only a single polygon model of a door (via careful measurement). Wherever a door is present in the building, this same polygon model can then be rapidly "pasted" onto a 10cm-resolution grid map that has been built via standard SLAM techniques. This allows us to very rapidly acquire a map of the entire building, including 0.1cm-resolution models of all the doors in it.

2.3 Measurement model

This section describes the measurement model $P(z_t|X_t, \alpha_t)$ used in our approach. For this work, we focused on using a single sensor: a SICK laser scanner. Because our map comprises both low-resolution (10cm-grid cells) and high-resolution (1-mm, polygon model) components, we desire a measurement model that has a consistent interpretation regardless of the resolution at which the map is represented. Specifically, we know that the 10cm grid cells are inaccurate—the building walls, chairs, etc., are unlikely to be aligned with the global 10cm grid—and thus we wish to model the 10cm grid map as probabilistically impeding the laser ray in a way that is noisier than the more precise polygon model.

In detail, the 10cm grids are usually only partially occupied, and thus there is a chance of the laser passing through it entirely. Thus, rather than simply modeling each grid cell as occupied or unoccupied, we will instead associate with each grid cell a probability that a laser ray terminates within that grid cell. Because our map has multiple resolutions, it is insufficient to associate with each grid cell (depending on the type of material, say) a probability of that grid cell impeding the laser. To understand this, consider the toy map shown in Figure 4, which we can choose to represent via either a low-resolution (10cm x 10cm) grid, or a higher-resolution (5cm x 5cm) grid. If we model a grid-cell as having a probability p of impeding a laser, then the chance of the laser being impeded by the 10cm x 10cm region in Figure 4a is p, whereas the chance for the map on the right is p^3 (since it passes through three grid-cells). Clearly, it is undesirable that the measurement model change just because we chose to represent an object at a different resolution.

There are a variety of solutions to this, but we consider the most natural one to be the probabilistic interpretation of occupancy grid maps proposed by [Eliazar and Parr, 2004]. Their interpretation was motivated by the observation that the more naive model (using a fixed probability p for each grid cell) shows anomalous effects depending on the angle at which a laser travels relative to the grid lines, even if all the grid cells are the same resolution. However, the same interpretation turns out to also elegantly address our problem of using multi-resolution maps.

In this model, each cell of a grid map is associated with an **opacity** ρ.[2] The probability of a laser ray being interrupted by this cell depends both on the opacity and on the distance the ray travels within the cell. In detail, the probability of a ray terminating (i.e., being interrupted) while traveling from

[2]The chance of a laser terminating/being interrupted is a decreasing function of ρ, so this parameter is perhaps better thought of as "transparency" rather than "opacity." However, for consistency we will use [Eliazar and Parr, 2004]'s terminology.

point r_1 to point r_2 in a medium of opacity ρ is defined as:

$$P(terminate(\rho, r_1, r_2)) = 1 - e^{-\frac{||r_1 - r_2||}{\rho}}$$

Immediately, we see that the probabilities of the ray terminating under the maps in Figure 4a or 4b are the same, since the total distance is the same in either case (assuming that all the grid-cells have the same opacity ρ). Thus, this model allows us to give a consistent probabilistic interpretation to multi-resolution maps.

More generally, suppose a laser travels in a direction that (if it were unimpeded) would take it through N different regions in the map. Here, a "region" can be a grid cell (from the low-resolution map) or a polygonal region (from the polygon map), such as a polygon that represents the shape of a door, or one that represents part of the door-frame. We let $r_0, r_1, \ldots, r_N$ denote the points at which the laser ray would transition from one region to another (if it were to pass through all regions unimpeded), with r_0 denoting the origin of the laser ray. We also let $\rho_1, \ldots, \rho_N$ denote the opacities of these regions. The probability of the ray terminating within the i-th region is then:

$$P(terminate(\rho_i, r_{i-1}, r_i)) \prod_{k=1}^{i-1} (1 - P(terminate(\rho_k, r_{k-1}, r_k)))$$
$$= (1 - e^{-||r_{i-1} - r_i||/\rho_i}) \prod_{k=1}^{i-1} e^{-||r_{k-1} - r_k||/\rho_k}.$$

This allows us to define the probability that the laser terminates at any specific range r. Finally, if the laser terminates at a certain range r, we model the actual observed laser measurement z to be the range corrupted by Gaussian noise:

$$z = r + \mathcal{N}(0, \sigma^2_{rng}) \quad (1)$$

3 Inference

3.1 Rao-Blackwellization

We now describe an inference algorithm that reasons about the robot trajectory and the object state based on a set of measurements and controls. Specifically, we compute the following belief:

$$Bel_t = p(\alpha_t, X^t | z^t, u^t).$$

Note that the belief includes the entire robot trajectory up to time t, but only the latest object state. This choice turns out to be important for deriving an efficient Rao-Blackwellized filter. (This is a consequence of the fact that the current belief about the state of the door depends on the entire past trajectory—which, e.g., indicates which of the past sensor measurements were directed at the door. A similar derivation is also used in [Montemerlo, 2003; Murphy and Russell, 2001].)

In detail, we apply a Rao-Blackwellized particle filter (RBPF), where in each particle we encode the entire robot trajectory and a posterior distribution of the object state. Thus, we split up the belief into two conditional factors:

$$Bel_t = p(X^t | z^t, u^t) p(\alpha_t | X^t, z^t, u^t).$$

The first factor encodes the robot trajectory posterior:

$$R_t = p(X^t | z^t, u^t).$$

The second factor encodes object state posterior, conditioned on the robot trajectory:

$$S_t = p(\alpha_t | X^t, z^t, u^t).$$

The factor R_t will be approximated using a set of particles; the factor S_t, which estimates the angle of the door, will be approximated using a Gaussian distribution (one Gaussian per particle).

3.2 Robot trajectory estimation

Within each particle we record a guess of the robot trajectory X^t, and a Gaussian approximation S_t to the object state. We will denote a particle by $q^t_m = (X^{t,[m]}, S^{[m]}_t)$ and a collection of particles at time t by $Q_t = \{q^t_m\}_m$. We compute Q_t recursively from Q_{t-1}. Suppose that at time step t, particles in Q_{t-1} are distributed according to R_{t-1}. We compute an intermediate set of particles $\bar{Q}_t$ by sampling a guess of robot pose at time t from the motion model. Thus, particles in $\bar{Q}_t$ are distributed according to the robot trajectory prediction distribution:

$$\bar{R}_t = p(X^t | z^{t-1}, u^t).$$

To ensure that particles in Q_t are distributed according to R_t (asymptotically), we generate Q_t by sampling from $\bar{Q}_t$ with replacement in proportion to importance weights given by $w_t = R_t / \bar{R}_t$. Section 3.4 explains how these weights are computed. For now, we note that since only the latest robot pose is used in the update equations, we do not need to actually store entire trajectories in each particle. Thus the memory storage requirements per particle do not grow with t.

3.3 Object state estimation

We use a Gaussian/extended Kalman filter (EKF) approximation to estimate the object state posterior, S_t. Thus we keep track of the mean μ_t and variance σ_t of the approximating Gaussian in each particle: $q^t_m = (X^{t,[m]}, \mu^{[m]}_t, \sigma^{[m]}_t)$.

Since S_t involves only the latest object state α_t (and not the entire object state history α^t), storage and computation requirements here also do not grow with t. We have:

$$\begin{aligned} S_t &= p(\alpha_t | X^t, z^t, u^t) \\ &\propto p(z_t | \alpha_t, X^t, z^{t-1}, u^t)\ p(\alpha_t | X^t, z^{t-1}, u^t) \\ &= p(z_t | \alpha_t, X_t)\ p(\alpha_t | X^t, z^{t-1}, u^t). \end{aligned} \quad (2)$$

The first step above follows from Bayes' rule; the second step follows from the conditional independence assumptions of our model (Figure 2). The expression (2) is a product of a measurement likelihood term and an object state (dynamical model) prediction term, which is defined (similarly to $\bar{R}_t$) as:

$$\begin{aligned} \bar{S}_t &= p(\alpha_t | X^t, z^{t-1}, u^t) \\ &= \int_{\alpha_{t-1}} p(\alpha_t | \alpha_{t-1}) p(\alpha_{t-1} | X^{t-1}, z^{t-1}, u^t) d\alpha_{t-1} \end{aligned}$$

Because $p(\alpha_{t-1} | X^{t-1}, z^{t-1}, u^t)$ is already approximated as a Gaussian (represented by a Rao-Blackwellized particle from the previous timestep) and we use a linear-Gaussian model for $p(\alpha_t | \alpha_{t-1})$, we can easily compute the mean and

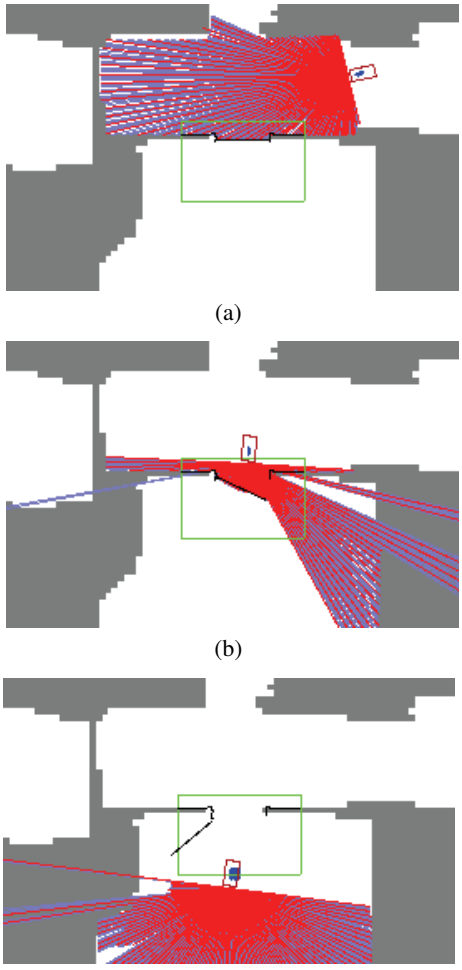


Figure 5: Robot localization with estimation of door state. The robot is denoted via the small rectangle; the rays emanating from the robot show the laser range scans; and the estimated door angle is also shown in the figures. The sequence of three images show the robot approaching, opening, and having passed through the door.

variance of $\bar{S}_t$ above in closed form. Similarly, by using a Laplace approximation to obtain a Gaussian approximation to the measurement model $p(z_t|\alpha_t, X_t)$, using Equation (2) we can also compute the mean and variance of S_t in closed form.

3.4 Computing importance weights

Briefly, following the derivation in [Montemerlo, 2003], it is straightforward to show that the importance weights w_t should be:

$$w_t \quad = \quad R_t/\bar{R}_t = \frac{p(X^t|z^t, u^t)}{p(X^t|z^{t-1}, u^t)} = \mathbb{E}_{\bar{S}_t}[\, p(z_t|\alpha_t, X_t)\,]$$

In words, the importance weights are the expected value (over the object state prediction distribution) of the measurement likelihood. Since the two terms $p(z_t|\alpha_t, X_t)$ and $\bar{S}_t$ are already approximated as Gaussians (as described in Section 3.3), this expectation can be expressed as an integral over a product of two Gaussians, and can thus be carried out in closed form.

3.5 Using Scaling Series to increase precision

For mobile robot navigation, 10cm precision is usually acceptable and gives reasonable performance. Thus in deployed implementations it is fairly common practice to assume a large laser noise variance and use only a few laser rays per measurement update, which results in an approximation to the next-state R_t that has fairly large variance. However, to perform manipulation tasks we require sub-centimeter precision in localization. Achieving this requires that we use most of the laser measurements in every update, and assume a realistic (small) variance in the laser readings. This results in a very peaked measurement model, in which most of the probability mass of our robot's position estimate is supported by a very small region (of perhaps 1-5mm diameter). A consequence of this is that it becomes difficult during the usual importance sampling step to draw a sufficient number of particles from this region to represent it well.

In [Petrovskaya *et al.*, 2006], a tactile localization application was considered that also had the similar problem of a very sharply peaked measurement model. They proposed a "Scaling series algorithm" to efficiently produce a much more informed proposal distribution, one that is concentrated around the areas of high probability mass. We refer the reader to [Petrovskaya *et al.*, 2006] for details on the scaling series, but briefly, the algorithm works by performing a series of successive refinements, generating an increasingly informative proposal distribution at each step of the series. The successive refinements are performed by gradually annealing the noise variance parameter within the measurement model from an artificially high value down to a realistic variance setting.

In our setting, we applied the scaling series algorithm to choose the proposal distribution on each step of importance sampling in our particle filter. To do this, we annealed the measurement noise variance parameter σ_{rng} in Equation 1 and performed a series of successive refinements. This resulted in a much more informed proposal distribution, which allowed us to perform localization using only about 100 particles per step.

4 Experimental Results

We apply our algorithm to the task of manipulating door handles and doors. The STAIR (STanford Artificial Intelligence Robot) project is an ambitious, long-term (10-15 year) project that seeks to build a useful home/office robotic assistant. Thus the ability to use doors and enter offices and conference rooms is of great practical importance to the robot.

We obtained 10cm resolution occupancy grid map by using standard SLAM algorithms with a mobile robot equipped with a laser. (See Figure 3a.) Further, as discussed in Section 2.2, because all doors in our building are essentially identical, it is possible to build a single precise polygon model of a door, and then rapidly "paste" the same model into the grid map at all places where a door exists. Our polygon model includes the door itself, the door frame, and a small surrounding region, and also encodes the position of the door hinge.

Datasets	Algorithms AMCL	RBPF	RBPF-SS
1	20.158cm	1.639cm	0.239cm
2	29.557cm	0.599cm	0.392cm
3	52.167cm	1.335cm	0.272cm
4	04.582cm	1.263cm	0.115cm
5	14.956cm	1.668cm	0.446cm
6	88.873cm	2.597cm	0.550cm
7	04.653cm	1.481cm	0.253cm
8	08.925cm	1.329cm	0.102cm
9	08.993cm	1.610cm	0.172cm
10	13.589cm	3.052cm	0.499cm
11	05.921cm	1.524cm	0.273cm
12	98.063cm	1.962cm	0.560cm
Overall RMS	42.975cm	1.779cm	0.358cm

Table 1: Positioning RMS error comparison of three localization algorithms: Adaptive MCL (AMCL), Rao-Blackwellized Particle Filter (RBPF), and full proposed algorithm (RBPF-SS). Each of the 12 experiments shown was an average over 10 runs of each algorithm; final row shows the overall RMS error.

(Figures 3b and 3c show a door and its polygon model representation.) Although not part of the polygon model, we note that the door handle is also at a fixed (known) position relative to the surface of the door, and can thus be straightforwardly computed if the position of the door (including the opening angle α of the door) is known.

The STAIR mobile manipulation platform comprises a Segway mobile platform, a Harmonic Arm manipulator, and a SICK laser. The arm has a fairly limited operational range (workspace), and has barely enough power to turn the door handles—it is able to do so only from certain configurations, where the load is spread more evenly among its motors—and thus there is only a very small 3cm x 3cm region from where the robot is physically capable of opening the door. Even within this region, localization accuracy of about 1-5mm is necessary to correctly grasp, turn, and manipulate the handle.

Several videos showing results of the robot opening a door are available at the website

http://cs.stanford.edu/∼anya/stair/

The robot navigates to the door, turns the handle and opens the door slightly; then as it is moving through the door, the arm continues to manipulate the door by continuously pushing it open in front of the robot. Our state estimation algorithm is used in real-time to continuously estimate the position of the robot and the opening angle of the door, and thereby control the arm to continuously push the middle of the door. Even though the map changes drastically each time the robot opens the door and moves through it (see Figure 5), in our experiments the proposed approach invariably gives precise state estimates and results in successfully manipulating and navigating through the door. Tested 12 times (3 times on each of 4 doors), the algorithm succeeded each time in giving sufficiently accurate state estimates to open, continuously manipulating, and move through the door.

Although our algorithm allowed us to solve the practical problem of going through doors in our building, we now also present a more formal evaluation of its performance.

For the experiments presented below, we collected several minutes of laser and odometry data of the robot's approach towards a door in twelve distinct test situations. We considered 4 different doors (in different parts of the building); we considered each door in 3 different positions (closed, open, half-open). The purpose of this set of experiments was to test our approach against others in identical conditions. To ensure fairness of comparison, the same real-time computation requirements were imposed on all three algorithms. Since the algorithms considered are non-deterministic, we ran each algorithm 10 times for each dataset. To give a quantitative evaluation we computed (for each algorithm) the root-mean-square (RMS) error with respect to ground truth among 10 runs for each test situation, and then averaged over all twelve test situations (summarized in Table 1). For the ground truth, we used the maximum a posteriori estimate of the door and robot position using a fine (2mm) grid within a 10cm area around the robot's final position in each test case.

To provide a baseline comparison, we used the Adaptive MCL localization algorithm implemented in Player (see [Gerkey *et al.*, 2003]). This implements the KLD MCL method proposed in [Fox, 2001], and uses a 10cm occupancy grid map. In agreement with results reported by [Stachniss and Burgard, 2005], we noticed that if the actual door state does not correspond to the mapped door state, the robot gets "lost," which manifests itself as increased positioning error. If the door state does correspond to the map, the robot is able to localize with a RMS error of 12.6cm.

We also tested an "intermediate" algorithm that uses the polygon map and grid map combination, also using a Rao-Blackwellized particle filter, but without the scaling series algorithm to choose its proposal distributions. Empirically, this algorithm is able to localize and estimate the door state fairly accurately. The RMS error of robot pose in this scenario was 1.78cm, which is insufficient accuracy for manipulating the door handle.

Our full algorithm, using the scaling series proposal distributions (and the same update rate as the intermediate algorithm) is able to estimate robot pose with an RMS error of about 3mm. Using this algorithm, we were able to reliably open multiple doors.

5 Discussion and Conclusions

One frequently discussed difficultly of Rao-Blackwellized algorithms, specifically of FastSLAM, is that of extinction of particles. In FastSLAM, if a robot does not visit part of a map for a long time, then because the map is *static*, through the normal death of particles there will be very little diversity in its posterior representation of that part of the map. Less formally, the algorithm becomes overly confident in its estimate of the map, which makes it difficult for the robot to accurately estimate that part of the map if it later returns to it. (See discussion in [Montemerlo, 2003].) In contrast, because we are estimating a *dynamic* parameter—namely the opening angle of the door, which is modeled as a dynamic, changing, variable—this is not a problem for our algorithm. Specifi-

cally, if the robot wanders away from the door for a long time, then its posterior estimate of the door's distribution will converge to its stationary (uncertain) distribution, and thus the particle filter will correctly capture the fact that we should be very uncertain about the state of a door that we have not seen for some time.

One possible direction for future work is consideration of highly crowded environments. While we did have occasional passersby during our experiments (and our algorithm was resilient to these effects), overall the amount of unmodeled effects was low and the map provided a good representation of the environment. Unmodeled effects can be considerably more frequent in highly crowded or cluttered environments, e.g. high traffic public areas such as a museum or cinema theater. These environments have been considered (from a mobile robot localization perspective) by [Fox *et al.*, 1999b], who proposed a range data filtering technique to improve robustness of localization. Similar techniques can be added to our algorithm to increase robustness for mobile manipulation in these environments.

While designing the algorithm, we also had in mind the applications of manipulating (opening/closing) a filing cabinet sliding drawer, and moving a piece of furniture (e.g., a chair). Either of these fit into our framework very naturally (α = drawer position; or α = chair position), and we believe our approach will extend straightforwardly to such applications as well.

In summary, we have presented a single, unified, probabilistic model for simultaneously localizing a mobile manipulator robot and estimating the state of an object being manipulated. Our algorithm uses a combination of a high- and a low-resolution map, and was successfully applied to door manipulation, a task which requires very precise state estimation.

Acknowledgments

We give warm thanks to Morgan Quigley, Jamie Schulte, Jimmy Zhang, David Lee and Francois Conti for their help with the STAIR robot, and to Pieter Abbeel, Sebastian Thrun, Mike Montemerlo, and the anonymous reviewers for their insightful comments and suggestions. This work was supported by DARPA under contract number FA8750-05-2-0249. Support from the Honda Research Institute is also gratefully acknowledged.

References

[Anguelov *et al.*, 2002] D. Anguelov, R. Biswas, D. Koller, B. Limketkai, S. Sanner, and S. Thrun. Learning hierarchical object maps of non-stationary environments with mobile robots. In *Proc. of UAI*, 2002.

[Anguelov *et al.*, 2004] D. Anguelov, D. Koller, E. Parker, and S. Thrun. Detecting and modeling doors with mobile robots. In *Proc. of ICRA*, 2004.

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

[Biswas *et al.*, 2002] R. Biswas, B. Limketkai, S. Sanner, and S. Thrun. Towards object mapping in dynamic environments with mobile robots. In *Proc. of IROS*, 2002.

[Eliazar and Parr, 2004] A. Eliazar and R. Parr. DP-SLAM 2.0. In *Proc. of ICRA*, 2004.

[Fox *et al.*, 1999a] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proc. of the National Conference on Artificial Intelligence*, 1999.

[Fox *et al.*, 1999b] D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 1999.

[Fox, 2001] D. Fox. KLD-sampling: Adaptive particle filters. In *Proc. of NIPS*, 2001.

[Gerkey *et al.*, 2003] B. Gerkey, Vaughan R. T., and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proc. of ICAR*, 2003.

[Hähnel *et al.*, 2003] D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *Proc. of ICRA*, 2003.

[Moll and Erdmann, 2003] M. Moll and M. A. Erdmann. *Reconstructing the Shape and Motion of Unknown Objects with Active Tactile Sensors*, pages 293–310. Springer Verlag, 2003.

[Montemerlo, 2003] Michael Montemerlo. *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association*. PhD thesis, Robotics Institute, Carnegie Mellon University, 2003.

[Murphy and Russell, 2001] K. Murphy and S. Russell. *Rao-blackwellized particle filtering for dynamic bayesian networks*. Springer, 2001.

[Petersson *et al.*, 2000] L. Petersson, D. Austine, and D. Kragic. High-level control of a mobile manipulator for door opening. In *Proc. of IROS*, 2000.

[Petrovskaya *et al.*, 2006] A. Petrovskaya, O. Khatib, S. Thrun, and A. Y. Ng. Bayesian estimation for autonomous object manipulation based on tactile sensors. In *Proc. of ICRA*, 2006.

[Rhee *et al.*, 2004] C. Rhee, W. Chung, M. Kim, Y. Shim, and H. Lee. Door opening control using the multi-fingered robotic hand for the indoor service robot. In *Proc. of ICRA*, 2004.

[Shekhar, 1986] Khatib O. Shimojo M. Shekhar, S. Sensor fusion and object localization. In *Proc. of ICRA*, 1986.

[Slaets *et al.*, 2004] P. Slaets, J. Rutgeerts, K. Gadeyne, T. Lefebvre, H. Bruyninckx, and J. De Schutter. Construction of a geometric 3-D model from sensor measurements collected during compliant motion. In *Proc. of ISER*, 2004.

[Stachniss and Burgard, 2005] C. Stachniss and W. Burgard. Mobile robot mapping and localization in non-static environments. In *Proc. of AAAI*, 2005.

[Thrun *et al.*, 2005] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[Wolf and Sukhatme, 2004] D. Wolf and G. S. Sukhatme. Online simultaneous localization and mapping in dynamic environments. In *Proc. of ICRA*, 2004.