Detecting and Modeling Doors with Mobile Robots

Dragomir Anguelov, Daphne Koller, Evan Parker, Sebastian Thrun
Computer Science Department
Stanford University, Stanford, CA 94305
Email:{drago, koller, nave, thrun} @cs.stanford.edu

Abstract— **We describe a probabilistic framework for detection and modeling of doors from sensor data acquired in corridor environments with mobile robots. The framework captures shape, color, and motion properties of door and wall objects. The probabilistic model is optimized with a version of the *expectation maximization* algorithm, which segments the environment into door and wall objects and learns their properties. The framework allows the robot to generalize the properties of detected object instances to new object instances. We demonstrate the algorithm on real-world data acquired by a Pioneer robot equipped with a laser range finder and an omni-directional camera. Our results show that our algorithm reliably segments the environment into walls and doors, finding both doors that move and doors that do not move. We show that our approach achieves better results than models that only capture behavior, or only capture appearance.**

I. Introduction

Consider a mobile robot, which is equipped with sensors and entrusted with the task of finding all doors in a hallway. The robot can try to find doors based on the fact that they move, distinguishing them from static walls. A differencing algorithm between sensor readings taken at different times will be rather successful in finding moving doors. What about the doors that never moved during the robot's data acquisition phase? If the robot knew the door width and color apriori, it would again be successful, but this information is usually not available. Instead, the robot can learn about width and color from doors that moved and apply the acquired knowledge to find the doors that did not move. This example suggests that a framework which jointly models geometric, color and motion attributes of objects can successfully generalize from object instances that a robot has detected to new object instances.

Surprisingly, object-level modeling and property generalization have remained largely unaddressed in the robotics literature. Environment models, or maps, play a pervasive role in mobile robotics, but most map representations have been chosen with the primary goals of navigation and localization in mind. Often, environment maps are represented as *occupancy grids* [20], [25], in which models are represented by grids of binary occupancy variables. An alternative representation is known as topological [7], [13], [18], [22]; here the environment is represented by a graph whose edges corresponds to places (e.g., intersections in a corridor environment), and whose arcs correspond to navigable connections between them. Other popular representations of maps are composed of point features [10], [14], raw sensor measurements [5], [11], [16], or line segments [6], [15]. The representations mentioned above (and most others that have been proposed) largely assume static environments, and do not model the world structure in an intuitive object-based framework. Recently, there has been some related work in the context of object-based modeling [2], [4] (see also [8]). The approach in [2] uses simple differencing to segment non-stationary objects, and learns their appearance models, represented as 2D occupancy grid maps. There are two restrictive assumptions in this model that we will not make. First, it avoids the segmentation problem entirely by assuming that objects are separable (not too close to each other) and that an entire scan of the circumference of an object can be obtained by the robot. Second, the use of object differencing implies that objects can be recognized only if they move at some point in time. In the case of door detection, doors can touch other objects, and can be partially occluded. We are also interested in finding doors that never moved.

Our approach demonstrates the advantages of object-based modeling by defining a probabilistic generative model of corridor environments containing door and wall objects. Each object is parameterized both by visual features — its shape and color — and by behavioral features — its motion model. Given a map containing a set of objects, our generative model defines the distribution over possible observations that the robot might make when scanning the environment. It expresses preferences for models, which match general knowledge about the objects (doors move and walls do not), and for models with coherent object attributes (doors usually have similar colors and widths). For a fixed set of objects, we can fit the model parameters using the *expectation maximization (EM)* algorithm [9]. Our algorithm also considers the introduction of new objects and evaluates their plausibility within the model. It can detect objects using both their behavioral features, i.e., their motion, and their appearance. Thus, using generalization, it enables a robot to recognize instances of door objects even if those instances never move. We present experimental evidence on real-world data acquired with a Pioneer robot, equipped with a SICK laser range finder and an omnidirectional camera. The raw data is pre-processed using a scan matching algorithm described in [12], which enables us to recover accurate estimates of the robot's pose. From this data we learn a model integrating object appearance and behavior properties and show it achieves better results than models which only capture behavior or models which only capture appearance.

II. Map Model

Object-based Modeling We begin by presenting a probabilistic framework for maps containing objects. Our results in this paper address a special case of this framework, where maps contain door and wall objects, modeled as line segments. While we believe that the framework and the ability to use EM for learning object-based maps are of independent interest, it is clear that our particular instantiation allows us to make certain strong assumptions which allow us to learn maps effectively.

A *mapping domain* is defined via a set $\mathcal{C}$ of *object classes*. In our example, we have two classes of objects, walls and doors, and any specific map will consist of some set of doors and some set of walls. Each class $C \in \mathcal{C}$ is associated with a set of parameters, divided into two types: *static parameters*, which represent attributes of the object whose value is fixed over time, and *dynamic parameters*, which can change between scans of the robot. For example, shape and color are typically static parameters, whereas position parameters can be static for stationary object classes such as walls, and dynamic for non-stationary object classes such as doors. Static parameters are further partitioned into two subsets: *global parameters* and *local parameters*. Global parameters are defined for an entire class of objects, whereas local parameters are specific to individual objects in the class. Global parameters are very common. For example: multiple walls (or doors) might share the same color, and multiple doors might have the same width.

A *world* is a set of J concrete objects, $\phi_1, \ldots, \phi_J$, such that each object ϕ_j is an instantiation of an object class in $\mathcal{C}$. In a dynamic environment, the world changes over time. We assume that time is represented as a set of discrete time points $t = 1, \ldots, T$. A *configuration* m^t specifies a value for each parameter of each object in the world at time t. Specifically, let ϕ be an object in the world from class C, and θ some parameter associated with C. In general, the map m^t contains a parameter θ^t_ϕ. However, if θ is a static parameter, then θ^t_ϕ is fixed over time, and we can therefore drop the dependence on t, using the notation θ_ϕ. If θ is also a global parameter, it is fixed for the entire class, and we can drop the dependence on ϕ, using the notation θ_C. The *map* is the set of all configurations: $\mathcal{M} = m^1, \ldots, m^T$, which includes all static and all dynamic parameters for every value of t.

Implementation with Walls and Doors In our application, a robot learns object maps of a typical corridor environment. In this case, we have two classes of objects: walls and doors. For each of these classes, we represent both their shape and their color features.

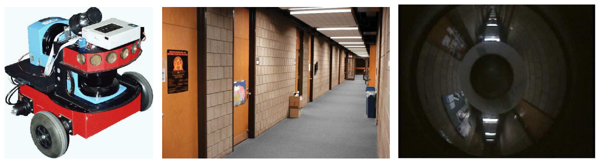
Fig. 1. a) Pioneer mobile robot, equipped with two 2D laser range finders and a panoramic camera; b) corridor environment with doors; c) panoramic camera image

Most simply, a wall can be represented as a single straight line segment, as proposed in [6], [17]. However, a single wall often contains openings, for hallways or doors. If we model such a wall as a collection of separate objects, we will fail to capture the fact that these segments are part of a single object and are therefore aligned. Without this assumption, the noise in the data will lead us to learn poorer maps, containing many segments whose orientation is slightly different.

Thus, a wall object W is defined via a set S_W of segments aligned along the same straight line. The line is described by the tuple: $\langle \alpha_W, \beta_W \rangle$, where α_W is a normal unit vector, and β_W is a scalar offset between the line and the origin of the coordinate system. Each segment along the line is represented by specifying its endpoints; given the line parameters and a reference point, each endpoint can be represented by a single parameter. Thus, we need a total of $2 \cdot |S_W| + 2$ independent parameters to represent a wall object W. In our framework, we take these parameters to be static and local.

A door object D is represented as a segment that can rotate around a hinge. The door class is associated with three parameters: h_D, a two-dimensional vector denoting the location of the door hinge; w, a scalar denoting the width of the door; and $\psi^t_D \in [-\pi/2, \pi/2]$, the angle of the door at time t. In our framework, h_D and w are static parameters and the angle ψ^t_D is a dynamic parameter (as indicated by time index t). Furthermore, we take the width w to be a global parameter (hence the missing subscript), whereas all other parameters are local. Thus, for a specific door object D we have the parameters: $\langle h_D, w, \{\psi^t_D\}^T_{t=1} \rangle$.

We now turn to the color model. In our setting, color is a global parameter, that is, we assume that objects in the same class have the same color. Of course, specific scans of an object typically vary in the actual color reading; this variability is modeled via a Gaussian distribution. The wall class is associated with two global parameters μ_W, σ^2_W that specify the mean and variance of this Gaussian density. Similarly, the door class is associated with two parameters μ_D, σ^2_D.

III. Data Model

The parameters in the object map are learned from data, acquired by a robot equipped with a forward-pointing laser range finder and an omni-directional camera (see Fig. 1). The raw data is pre-processed using a scan matching algorithm described in [12], which enables us to recover accurate estimates of the robot pose. The robot *pose* at time t will be denoted $s^t = (x^t, y^t, \gamma^t)$, where x^t and y^t represent the robot's location in Cartesian coordinates, and γ^t its heading direction. The robot's *trajectory* S is its simply the sequence of all its poses $s^1, s^2, \ldots, s^T$.

At each time t, the robot acquires a scan Σ^t of the environment. In each scan, the SICK range finder we use takes $K = 180$ range measurements, spanning a semicircle. A scan Σ^t is a set of K tuples, each consisting of a 1D range measurement d associated with an RGB color l:

$$\Sigma^t \quad = \quad \{(d_1^t, l_1^t), \ldots, (d_K^t, l_K^t)\}, \tag{1}$$

Each tuple (d_k^t, l_k^t) in the scan is associated with a direction ρ_k, which indicates the angle offset from γ^t, the heading direction of the robot. Knowledge of the robot pose enables us to project a measurement into the global coordinate frame. In particular, the mapping to world coordinates of the range measurement d_k^t is given by the geometric equation:

$$\pi_k^t \quad = \quad \begin{pmatrix} x^t + d_k^t \cos(\gamma^t + \rho_k) \\ y^t + d_k^t \sin(\gamma^t + \rho_k) \end{pmatrix} \tag{2}$$

Clearly, π_k^t is only meaningful if the sensor actually detects an object within its maximum sensor range. If a range sensor fails to detect an object, π_k^t is undefined; however, the range measurement may still carry information relevant to the object modeling problem.

Range Sensor Model An ideal, noise-free range sensor would always measure the distance to the nearest object in its sensor cone, up to the maximum sensor range. Of course, real measurements are subject to noise: the actual range is usually corrupted by noise, and sometimes sensors return values that are random relative to our object model (e.g., specular reflections or absorbing surface materials).

We model this phenomenon using the following sensor model. For each range measurement d_k^t, let ϕ_k^t denote the closest object in the map m^t along the measurement ray. (Note that ϕ_k^t is a function of the configuration m_k^t). Let δ_k^t denote the distance to this object. We also associate with each measurement a binary variable v_k^t, which denotes whether the range measurement is a valid one (derived from the object ϕ_k^t). If $v_k^t = 1$, the measurement is the correct distance to ϕ_k^t, corrupted by Gaussian noise, and cropped at the maximum sensor range $d_{\max}$:

$$p(d_k^t \mid s^t, m^t, v_k^t = 1) \quad = \quad \alpha \mathcal{N}\left(\min\{d_{\max}, \delta_k^t\}, \sigma^2\right)$$

Formally, we need the normalizing factor α (which depends on δ_k^t) to ensure that the conditional probability sums to 1 for the domain of d_k^t.

For the case when the measurement ray does not intersect any physical object (wall or door), we assume that $\delta_k^t = \infty$. The equation above is equivalent to explaining the reading with a virtual object located at the max-range mark.

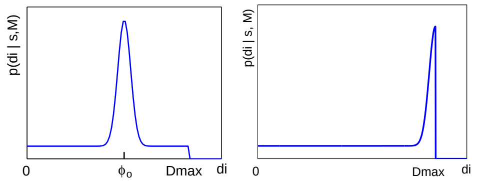

Fig. 2. Sensor model. a) The first physical object intersected is ϕ_o, b) The ray does not intersect any physical object.

If $v_k^t = 0$, the measurement is random within the valid measurement range $[0, d_{\max}]$, which is modeled by the following uniform density:

$$p(d_k^t \mid s^t, m^t, v_k^t = 0) \quad = \quad 1/d_{\max}.$$

The prior probability of a non-random measurement $p(v_k^t = 1)$ is denoted λ; it plays a role when 'recovering' expectations over the (unobservable) validity variables v_k^t. Graphical examples of the range sensor model are given in Fig. 2.

Color Model Technically, cameras possess different failure modes than range sensors. However, it will be mathematically convenient to stipulate the same validity variable v_k^t for both the range measurement d_k^t and the corresponding color value l_k^t. We make this assumption for simplicity both in the notation and the implementation, but our approach is easily generalized to accommodate separate validity variables. Thus, for $v_k^t = 1$, we have

$$p(l_k^t \mid s^t, m^t, v_k^t = 1) \ = \ \mathcal{N}(l_k^t; \mu_{\phi_k^t}, \sigma^2_{\phi_k^t}) \tag{3}$$

Otherwise, if $v_k^t = 0$, the color is drawn uniformly.

As we discussed, our implementation assumes that the color of each object is a global parameter. The color of an object ϕ in class C is defined via the parameter pair (μ_C, σ_C^2) defining a Gaussian distribution. Hence, in Eq. (3), we use the general class color parameters for the class of object ϕ_k^t rather than color parameters for the individual object.

IV. Map Likelihood, Prior, and Posterior

Data Likelihood In this section, we discuss the target function that is being optimized when learning the map. Assuming independence between the individual noise and validity variables for all measurements, the total likelihood of the data $\mathcal{D} = \{\Sigma_k^t\}$ and all validity variables $V = \{v_k^t\}$ is given by the following product:

$$p(\mathcal{D}, V \mid S, \mathcal{M}) \quad = \quad \prod_{t=1}^{T} p(\Sigma^t \mid s^t, m^t, v^t)\, p(v^t) \tag{4}$$

$$= \quad \prod_{t=1}^{T} \prod_{k=1}^{K} p(d_k^t \mid s^t, m^t, v_k^t)\, p(l_k^t \mid s^t, m^t, v_k^t)\, p(v_k^t)$$

Taking the negative logarithm of the resulting expression gives us the following target function, which we subsequently seek to minimize.

$$\log p(\mathcal{D}, V \mid S, \mathcal{M}) = \text{const.} + \sum_{t=1}^{T} \sum_{k=1}^{K} \{\log p(v_k^t) + + \log p(l_k^t \mid s^t, m^t, v_k^t) + \log p(d_k^t \mid s^t, m^t, v_k^t)\} \quad (5)$$

Map Prior Unfortunately, the number of objects in the map is unknown and has to be determined during mapping. One could easily optimize the data likelihood by introducing a unreasonably large numbers of objects in the map. To counteract this effect, we assume a prior over maps that penalizes maps in proportion to the number of parameters (this is a version of the well-known AIC criterion [1]). In our implementation, the number of parameters comprises all global parameters, whose number is denoted by M_G, all static local parameters, denoted M_S, and all dynamic parameters, denoted M_D. In addition, in order to prevent segments from being artificially lengthened without any supporting measurements, we introduce a prior to penalize long segments. In particular, as shape is a static parameter, we define $\mathcal{L}[\mathcal{M}]$ to be the total length of all segments in the map (in doors as well as walls). Overall, our prior then has the form:

$$p(\mathcal{M}) \quad \propto \quad \exp(-\rho_{\mathcal{G}}\, M_G - \rho_{\mathcal{S}}\, M_S - \;\rho_{\mathcal{D}}\, M_D - \rho_{\mathcal{L}}\mathcal{L}[\mathcal{M}])$$

The constants $\rho_{\mathcal{G}}, \rho_{\mathcal{S}}, \rho_{\mathcal{D}}, \rho_{\mathcal{L}}$ specify the strength of each component in the prior and are set by hand. Under this prior, the goal of learning is to identify a map $\mathcal{M}$ that maximizes the posterior. Note that this prior probability does not depend only on the total number of objects, but also on the number of segments in each wall object. This is because each segment accounts for two parameters in the wall object, as discussed above.

V. Learning Map Parameters With EM

In this section we assume a known set of object hypotheses, and show how to optimize the object parameters. The next section describes an outer loop to this procedure, which introduces new object hypotheses into the map.

In order to optimize the object parameters, we want to maximize the log-posterior probability of the map and the data:

$$p(\mathcal{M}, \mathcal{D} \mid S) \quad = \quad p(\mathcal{D} \mid S, \mathcal{M})\, p(\mathcal{M}) \qquad (6)$$

In general, this optimization problem is complex and multi-modal, so that a closed form solution is infeasible. Among many complicating factors is the fact that the validity variables $V = \{v_k^t\}$ are hidden. We therefore resort to finding a local maximum of Eq. (6) using the expectation maximization (EM) algorithm [9]. As usual, in EM we optimize the expected log-posterior $E_V[\log p(V, \mathcal{D}, \mathcal{M} \mid S) \mid \mathcal{D}, \mathcal{M}, S]$. The EM algorithm improves this quantity by iterating over two steps: the E-step and the M-step.

E-Step In the E-step, we are given a map $\mathcal{M}$, and we compute the expected values of the validity variables given the map and all available data. The individual validity variables v_k^t are independent of each other given the data and the map. Thus, we can easily compute

$$\begin{aligned} e_k^t \quad &:= \quad p(v_k^t \mid s^t, m^t, d_k^t, l_k^t) \qquad (7) \\ &\propto \quad p(l_k^t \mid s^t, m^t, v_k^t)\; p(d_k^t \mid s^t, m^t, v_k^t)\; p(v_k^t) \end{aligned}$$

Evaluating this probability involves querying the sensor model for a given map. Each correspondence v_k^t can only take on two values. This makes the calculation of v_k^t highly efficient. The time required to calculate these correspondence is linear in the number of measurements (and logarithmic in the size of the model, assuming efficient data structure for ray tracing).

M-Step Our task in the M-step is to find the model parameters that optimize the expected log-posterior, specified in Eq. (6). In our probabilistic model the range readings and colors are conditionally independent given the validity variables. Furthermore, they depend on disjoint sets of parameters. Thus, the expected log-probability decomposes into a sum of two unrelated terms — one involving only the shape parameters and the other only the color parameters — which can be optimized separately.

Unfortunately, the log-posterior for the shape is a discontinuous function, which involves discrete and continuous elements. We therefore decompose the maximization step into multiple sequential sub-steps, each of which optimizes a subset of all parameters in order to improve the log-posterior of the data. For walls, we have separate steps for optimizing the orientation parameters for a single line and the segment ends for a single line. For doors we have steps for optimizing the local parameters representing the position of the hinge and the angles, and steps for optimizing the global width parameter. As each of these optimization steps is executed separately, the resulting model is not a global optimum of the expected log-likelihood function (relative to the results of our E-step). However, this decomposition of the M-step enables us to utilize highly efficient solutions for the individual steps.

Wall Orientation This step determines the straight line parameters $\langle \alpha_W, \beta_W \rangle$ of a single wall in the model. It is easy to see that such parameters only depend on the range readings, and not the color measurements. Unfortunately, finding wall parameters that minimize the distance along all sensor rays (weighted by their expectation) defies a closed form solution. For efficiency purposes, we use an approximation motivated by the work of Liu *et al.* [15], and optimize a function where distance between the measurement and the object is evaluated using Euclidean distance:

$$(d_k^t - \delta_k^t)^2 \propto (\alpha_W^T \cdot \pi_k^t - \beta_W)^2 \qquad (8)$$

under the constraint that $\|\alpha_W\|_2 = 1$. Using this approximation, the optimization can be accomplished using standard weighted least-squares line fitting (see [15]).

Determining Wall Segments This optimization step determines the number and location of segments along a wall.

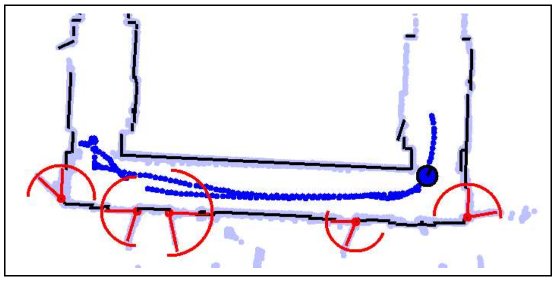

Fig. 3. (a) The Stanford hallway environment. (b) Stanford hallway map containing wall and door objects (dark lines: walls and door segments; semi-circles: dynamic range of the doors; light gray dots: laser data). Doors are allowed to have different widths in this experiment

At first glance, this may appear to be a difficult problem: for each point along the wall, we have to make a decision as to whether this point corresponds to a segment or not, that is, whether this point along the infinite line contains a physical wall in the world. However, this problem is simplified thanks to a number of critical observations. First, the problem is inherently discrete: Rather than considering arbitrary points along the line to be segment end points, our desire to minimize the sum of all segment length immediately implies that sensible endpoints must lie on points where sensor rays intersect the wall. Since there are only finitely many sensor measurements, we have to consider only finitely many candidates.

The number of segments and their end points can now be determined efficiently using dynamic programming. To understand the algorithm, let R_W be the set of all range measurements whose ray intersects with the line corresponding to the wall W. For each point $d_k^t \in R_W$, we have two choices: either this point is included in a segment, or it is not. This decision affects the object that the scan is expected to detect: Either it is the wall in question, or it is a possible wall behind it (assuming that such a wall exists in the mode; otherwise it will be max-range). If we denote the configuration that includes a wall at this point as m_k^{t+}, and the configuration that excludes it as m_k^{t-}, we effectively obtain two different expected log-posteriors derived from this local decision:

$$\begin{aligned} q_k^{t+} &= e_k^t \log p(d_k^t \mid s^t, m^+, v_k^t = 1) \\ q_k^{t-} &= e_k^t \log p(d_k^t \mid s^t, m^-, v_k^t = 1) \end{aligned} \qquad (9)$$

The difference of those values, $\Delta_k^t = q_k^{t+} - q_k^{t-}$, is the cost (in terms of log-likelihood) of including this point in a segment. Critically, the decisions for different points in R_W are independent, in that the overall difference in log-likelihood between two different segment configurations along the wall is simply the sum $\sum_{d_k^t \in R_W} \Delta_k^t$.

To obtain the log-posterior cost, we must factor in the effect of the prior, which adds to this term the length of the segment piece that results from including a point, and a possible parameter penalty for starting a new line segment.

The overall log-posterior cost can now be exactly optimized using a dynamic programming algorithm. The points are processed in linear order, starting from one end of the line. As the algorithm processes a point d_k^t, it keeps track of two costs: the optimal cost for the points processed so far if we include d_k^t in a segment, and the optimal cost for these points if we do not. Given both costs for d_k^t, we can compute the optimal pair of costs for the next point along the line. Once the processing of the line is done, we can (as usual) pass over the points in reverse order to select the overall configuration of segments that optimizes the cost.

Door Hinges and Angles Recall that a door D is a set of segments rotating around a hinge. It is defined by a static parameter h_D for the position of the hinge and a set of dynamic parameters ψ_D^t specifying its angle at different points in time. The problem of identifying the segment angles is similar to that of identifying wall orientations discussed above. We use the same approximation, taking into account the door hinge position, and thus optimize:

$$\sum_{t,k} e_k^t \{(\sin \psi_D^t \quad \cos \psi_D^t) \cdot (\pi_k^t - h_D)\}^2. \qquad (10)$$

Here "$\cdot$" is the vector dot product. Unfortunately, even this approximate problem does not have a closed form solution. However, the function is differentiable and we can optimize the hinge position and the angles by standard gradient descent.

Door Width Our model contains a global parameter w representing the width of all doors in the environment. We treat the scans of all doors we have found as scans of a single segment, whose length we want to optimize. This computation is straightforward, and uses essentially the same procedure as the one for optimizing the length of a wall segment.

Color Models As discussed, the color model for each object class C is a simple Gaussian distribution. Our approach estimates the mean color vector of each of the two classes using the standard ML estimator:

$$\mu_C = (\sum_{k,t} e_k^t l_k^t) / (\sum_{k,t} e_k^t)$$

The variance is computed analogously.

VI. Modifying the Model Structure

So far, we have assumed that the number of objects in the world, the walls and doors in our case, is fixed

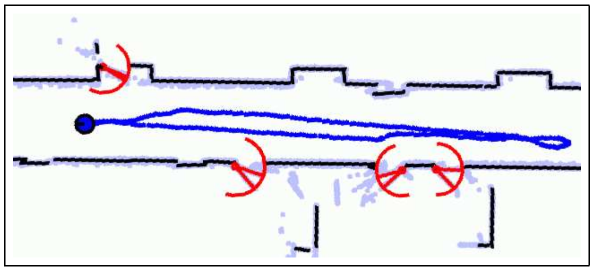
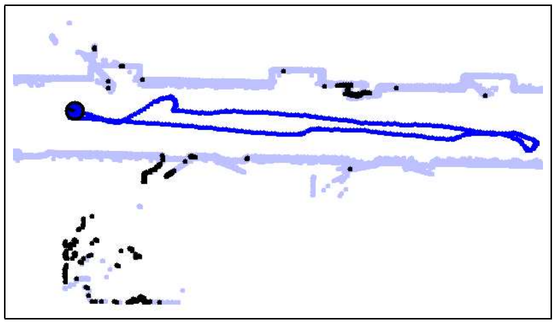

Fig. 4. (a) CMU hallway map learned from range data only. (b) Result from learning a mixture of two Gaussians on the color data only

and known. In this section, we address the question of selecting the right number of objects, or *map structure*. As is common in the literature on EM, our approach possesses an outer loop in which new objects are introduced, and others are terminated. The posterior log likelihood provides a clear measure with regards to the optimal number of objects in the map.

Unfortunately, adding objects with random parameters is highly inefficient, since the probability of such random objects to explain any previously unexplained measurement is usually extremely low. We therefore have developed a suite of initialization techniques that tend to insert objects at sensible locations and with sensible initial parameters.

Discovering Walls New walls are introduced by analyzing the set of all points π_k^t (not just the poorly explained ones) using a Hough transform [24]. This transform is analyzed for dominating line orientations in the environment, to recover an approximate prior over wall orientation (assuming that those are not uniformly distributed for most environments). In doing so, the initial orientation of walls is biased toward plausible orientations given the data.

Once a wall is suggested, we introduce it into the model, and then use the EM method described in the previous section to optimize the model parameters. Unfortunately, this approach is subject to local minima, which is particularly problematic when introducing a new wall into the map. The problem is due to the fact that before an object is introduced, the expectations step might have "explained away" an inconsistent sensor measurement by the hypothesis that it is noise. Put differently, the expectation e_k^t may be very close to zero *before* a new object is introduced.

Unfortunately, a single M-step is unable to take advantage of this reading: Even if the new object *perfectly* explains the measurement in question, the residual improvement in log-likelihood is at most e_k^t, the probability that the reading is viable. We address this problem by incorporating a "pseudo-E-Step" into the optimization associated with the introduction of a new object: When introducing a new object, we locally recalculate the expectations e_k^t for all measurements intersecting this object, temporarily allowing for the detection of objects that are occluded. This makes it possible to temporarily "discover" explanations of measurements that, after appropriate modification of the model in the M-step, will be plausible; in fact, the underlying model is equivalent to the one used in [15], which ignores occlusions when determining data association. While this temporary modification does not affect the overall optimization function (the log-posterior), which in fact models occlusions correctly, the ability to rapidly "re-acquire" points that were previously assumed to be noise improves the convergence of the algorithm significantly.

Discovering Doors In range data, doors can be recognized because of their motion. For example, a door may be open at one time and closed at another. Consider a segment that appears in some scan t, but moves to a different position in another scan t'. In most cases, the new position of the segment will cause some objects that are visible in scan t to be occluded in t' (or vice versa). Our algorithm searches for these occluding segments by building segment maps for individual scans. If straight line segments at time t are intersected by many scans at t', it is suggested as a potential door and the map is extended accordingly.

Additionally, doors can be suggested if we have prior or induced estimate of their color. Our algorithm searches for segments whose color is better explained by our model of door color than our model of wall color. An interesting capability of our framework is that we continue updating our color model as we find doors (based on motion, for example), which improves our color-based suggestions as we find an increasing number of doors that never moved.

Once a door is suggested, the algorithm estimates the (static and dynamic) parameters for the door by running the EM algorithm described above. It then computes the overall log-posterior score of the map containing the door. If this score is higher than that of the map without this door (i.e., when the segment is static), the door hypothesis is adopted. Otherwise, it is abandoned.

VII. Experimental Results

We conducted systematic experiments to determine the capability of our algorithm to detect door and wall objects, and to generalize from properties of detected object instances to new instances. We used real data collected by a physical robot in two different hallway settings at Stanford

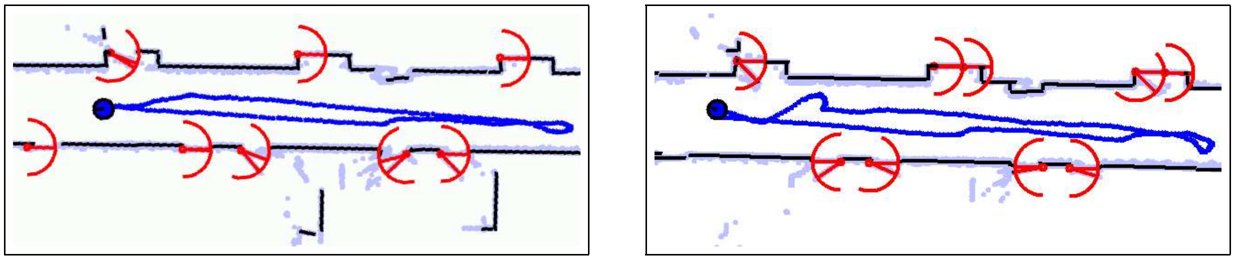

Fig. 5. (a) CMU map learned using both color and range data. (b) Generalizing the class parameters from one map to another. This is a map learned on a different robot pass using global class parameters learned on the previous pass

and CMU. The robot, shown in Fig. 1, is equipped with a SICK laser range finder and a panoramic camera. Stripes of color measurement were extracted from the camera image and associated with corresponding range measurements. As noted above, the pose information was recovered with sufficient accuracy using the algorithm described in [12]. The localization algorithm is run as a preprocessing step and is independent of the object-based mapping method proposed in this paper. All experimental data was collected in hallways (see Fig. 1), with doors changing orientation at random. None of the objects were labeled by hand; instead, the robot had to segment the data, identify objects, and learn models using EM.

All maps from both environments are learned using the same parameter setting: $\lambda = 0.7$, $\sigma = 5$cm, $\rho_{\mathcal{S}} = 35$, $\rho_{\mathcal{D}} = 1.3{*}\rho_{\mathcal{S}}$, $\rho_{\mathcal{L}} = 0.7$/cm. The results reported below are not particularly sensitive to this parameter setting. In our experiments, doors are stationary while the robot is passing through the hallway, but move between robot passes.

Our main experimental results pertain to the question of whether suitable object maps can be recovered using our approach. In particular, we sought to understand the extent to which the combination of multiple cues, motion and appearance, can aid our discovery of objects. Our main finding, in the corridor domain, shows that the combination of both features is indeed superior to the use of a single feature alone. In fact, our approach shows significant robustness in identifying walls and doors, and in learning accurate models.

In detail, our first set of experiments were targeted at the subproblem of finding doors based on dynamics, using range data alone. Fig. 3b) shows a typical result, obtained in one of our testing environments. As this image suggests, a number of doors was successfully identified, as were all major walls. However, two out of seven doors were misclassified as walls. One of them, shown on the upper left, never changed state, and hence is indistinguishable from a wall under our model. The other, on the bottom right, moved once, but it was invisible to the robot's scanner in its open position. These result suggest that our approach is quite robust with regards to finding doors that move, but is unable to discover doors that do not move.

Our second set of experiments was targeted at using appearance information (color, width) to further aid the recognition of doors. A typical example is shown in Fig. 5a). Here, the full model, as described in this paper, was trained using both range and color data. As this result suggests, all doors were found successfully; in fact, in all our runs involving this dataset the algorithm exhibited at least 84% reliability in identifying doors. The most remarkable finding, however, is that our approach succeeds in discovering four doors that never opened. These doors are the two rightmost ones on the top end and the two leftmost ones on the bottom end of the corridor. This example illustrates that our approach is indeed able to identify objects based on behavior features (motion), then learn appearance attributes (color, width) and use these learned attributes to identify other members of the same object class. To verify our hypothesis that both types of features indeed matter, we also trained a model based on color alone. The result in shown in Fig. 4b); learning mixture models with 2 or 3 Gaussians on the color data does not manage to separate door from non-door regions.

In a final series of experiments, we wanted to estimate the robustness of our approach to various modifications of the data set. We gradually added noise to the robot pose estimates of a no-color dataset, which significantly impacts the smoothness of the range data. At the same time, we kept the noise parameter σ of our sensor model unchanged. The results in Table 7 show a graceful decline in our capability to detect doors as the noise increases. In further experiments, we reduced the number of moving doors in the training set. We found empirically that the accuracy in identifying those doors that never opened does not depend on the number of doors that changed state in the training set, as long there was at least a single such door present.

Finally, Fig. 5b) shows the application of our learned model to a different data set collected in the same hallway, but without retraining the object model. In this run, one door is missed (bottom left), and different wings of double-winged doors are identified (but with incorrect hinge point, since those never moved). These results show that our approach is indeed capable of learning usable models of objects within our restrictive corridor setting.

Table 7
Performance of the door mapping algorithm on noisy localization data

Noise (cm/deg)	**0 / 0**	**2 / 0.3**	**4.0 / 0.6**	**6.0 / 0.9**
Doors Found	5	4	3	2
False Positives	0	0	0	0

VIII. Conclusions

This paper describes a probabilistic framework which models shape, color and motion properties of door and wall objects from robot sensor data. The model is learned with an instance of the EM algorithm, which detects doors and walls and learns their properties. The framework was validated experimentally on real-world datasets, demonstrating acquisition of accurate object maps and reliable detection of both moving and non-moving doors.

Our work can be extended in several ways. In this paper, we focused on object modeling, and rely on an existing software system to provide accurate pose estimates [12], [19]. Instead, the mapping problem can be addressed in the context of a robot which has to simultaneously localize itself in the environment (as in [21], [23]). Additionally, we have restricted ourselves to segment-based models of objects which largely remain at the same location. The general techniques we explored in this work can be extended to a much richer class of object models. A key decision in this case is to determine the parameterization of object shape, as most objects are not well-described using line segments. A volumetric representation as in [3] is one possibility. Defining an appropriate shape parameterization, especially when we are dealing with 3D geometry, is an area of open research.

Acknowledgment

This work has been sponsored by the Office of Naval Research, Young Investigator (PECASE) grant N00014-99-1-0464, the ONR Grant N00014-00-1-0637 under the DoD MURI program and DARPA's MARS Program (contracts N66001-01-C-6018 and NBCH1020014), which is gratefully acknowledged.

References

[1] H. Akaike. Information theory and an extension of the maximum likelihood principle. In *Proc. Second International Symposium on Information Theory*, 1973.
[2] 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. UAI*, 2002.
[3] D. Anguelov, R. Biswas, D Koller, B. Limketkai, and S. Thrun. Learning hierarchical object maps of non-stationary environments with mobile robots. In *Proc.UAI*, 2002.
[4] R. Biswas, B. Limketkai, S. Sanner, and S. Thrun. Towards object mapping in dynamic environments with mobile robots. In *Proc. IROS*, 2002.
[5] M. Bosse, P. Newman, M. Soika, W. Feiten, J. Leonard, and S. Teller. An atlas framework for scalable mapping. In *Proc. ICRA*, 2003.
[6] R. Chatila and J.-P. Laumond. Position referencing and consistent world modeling for mobile robots. In *Proc. ICRA*, 1985.
[7] H. Choset and J.W. Burdick. Sensor Based Planning: The Hierarchical Generalized Voronoi Graph. In *Proc. Workshop on Algorithmic Foundations of Robotics*, 1996.
[8] J. Coelho, J. Piater, and R. Grupen. Developing haptic and visual perceptual categories for reaching and grasping with a humanoid robot. *Robotics and Autonomous Systems*, 37(2-3):195–219, 2001.
[9] A. Dempster, N.Laird, and D.Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 1977.
[10] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. An experimental and theoretical investigation into simultaneous localisation and map building (SLAM). In P. Corke and J. Trevelyan, editors, *Lecture Notes in Control and Information Sciences: Experimental Robotics VI*, pages 265–274, London, 2000. Springer Verlag.
[11] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proc. CIRA*, 2000.
[12] D. Hähnel, D. Schulz, and W. Burgard. Map building with mobile robots in populated environments. In *Proc. IROS*, 2002.
[13] B. Kuipers and Y.-T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. Technical report, UT Austin, 1990.
[14] J.J. Leonard and H.J.S. Feder. A computationally efficient method for large-scale concurrent mapping and localization. In *Proceedings of the Ninth International Symposium on Robotics Research*, 1999.
[15] Y. Liu, R. Emery, D. Chakrabarti, W. Burgard, and S. Thrun. Using EM to learn 3D models of indoor environments with mobile robots. In *Proc. ICML*, 2001.
[16] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.
[17] C. Martin and S. Thrun. Real-time acquisition of compact volumetric maps with mobile robots. In *Proc. ICRA*, Washington, DC, 2002.
[18] M. J. Matarić. A distributed model for mobile robot environment-learning and navigation. Master's thesis, MIT, 1990.
[19] M. Montemerlo, N. Roy, and S. Thrun. Carnegie mellon robot navigation toolkit. Software package for download at www.cs.cmu.edu/∼carmen, 2002.
[20] H. P. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proc. ICRA*, 1985.
[21] Williams S., Dissanayake G., and Durrant-Whyte H. An efficient approach to the simultaneous localisation and mapping problem. In *Proc. ICRA*, 2002.
[22] H Shatkay and L. Kaelbling. Learning topological maps with weak local odometric information. In *Proc. IJCAI*, 1997.
[23] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proc. ICRA*, 2000.
[24] E. Trucco and A. Verri. *Introductory Techniques for 3D Computer Vision*. Prentice Hall, 1998.
[25] B. Yamauchi, P. Langley, A.C. Schultz, J. Grefenstette, and W. Adams. Magellan: An integrated adaptive architecture for mobile robots. Technical Report 98-2, ISLE, 1998.