Probabilistic Algorithms in Robotics

Sebastian Thrun
April 2000
CMU-CS-00-126

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

This article describes a methodology for programming robots known as *probabilistic robotics*. The probabilistic paradigm pays tribute to the inherent uncertainty in robot perception, relying on explicit representations of uncertainty when determining what to do. This article surveys some of the progress in the field, using in-depth examples to illustrate some of the nuts and bolts of the basic approach. Our central conjecture is that the probabilistic approach to robotics scales better to complex real-world applications than approaches that ignore a robot's uncertainty.

This research is sponsored by the National Science Foundation (and CAREER grant number IIS-9876136 and regular grant number IIS-9877033), and by DARPA-ATO via TACOM (contract number DAAE07-98-C-L032) and DARPA-ISO via Rome Labs (contract number F30602-98-2-0137), which is gratefully acknowledged. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of the United States Government or any of the sponsoring institutions.

Keywords: Artificial intelligence, bayes filters, decision theory, robotics, localization, machine learning, mapping, navigation, particle filters, planning, POMDPs, position estimation

1 Introduction

Building autonomous robots has been a central objective of research in artificial intelligence. Over the past decades, researchers in AI have developed a range of methodologies for developing robotic software, ranging from model-based to purely reactive paradigms. More than once, the discussion on what the right way might be to program robots has been accompanied with speculations concerning the very nature of intelligence per se, in animals and people.

One of these approaches, *probabilistic robotics*, has led to fielded systems with unprecedented levels of autonomy and robustness. While the roots of this approach can be traced back to the early 1960s, in recent years the probabilistic approach has become the dominant paradigm in a wide array of robotic problems. Probabilistic algorithms have been at the core of a series of fielded autonomous robots, exhibiting an unprecedented level of performance and robustness in the real world. These recent successes can be attributed to at least two developments: the availability of immense computational resources even on low-end PCs and, more importantly, fundamental progress on the basic algorithmic and theoretical levels.

So what exactly is the probabilistic approach to robotics? At its core is the idea of representing information through probability densities. In particular, probabilistic ideas can be found in *perception*, i.e., the way sensor data is processed, and *action*, i.e., the way decisions are made:

- **Probabilistic perception.** Robots are inherently uncertain about the state of their environments. Uncertainty arises from sensor limitations, noise, and the fact that most interesting environments are—to a certain degree—unpredictable. When “guessing” a quantity from sensor data, the probabilistic approach computes a probability distribution over what might be the case in the world, instead of generating a single “best guess” only. As a result, a probabilistic robot can gracefully recover from errors, handle ambiguities, and integrate sensor data in a consistent way. Moreover, a probabilistic robot knows about its own ignorance—a key prerequisite of truly autonomous robots.

- **Probabilistic control.** Autonomous robots must act in the face of uncertainty—a direct consequence of their inability to know what is the case. When making decisions, probabilistic approaches take the robot’s uncertainty into account. Some consider only the robot’s current uncertainty, others anticipate future uncertainty. Instead of considering the most likely situations only (current or projected), many probabilistic approaches strive to compute a decision-theoretic optimum, in which decision are based on all possible contingencies.

These two items are the basic characterization of the probabilistic approach to robotics.

What is the benefit of programming robots probabilistically? Our central conjecture is nothing less than the following: *A robot that carries a notion of its own uncertainty and that acts accordingly, will do better than one that does not.* In particular, probabilistic approaches are typically more robust in the face of sensor limitations, sensor noise, and environment dynamics. They often scale much better to complex environments, where the ability to handle uncertainty is of even greater importance. In fact, certain probabilistic algorithms are currently the only known working solutions to hard robotic estimation problems, such as the *kidnapped robot problem* [28], in which a mobile robot must recover from localization failure; or the problem of building accurate maps of very large environments, in the absence of a global positioning device such as GPS. Additionally, probabilistic algorithms make much weaker requirements on the accuracy of models than many classical planning algorithms do, thereby relieving the programmer from the (unsurmountable) burden to come up with accurate models. Viewed probabilistically, the *robot learning problem* is a long-term estimation problem. Thus, probabilistic algorithms provide a sound methodology for many flavors of robot learning. And finally, probabilistic algorithms are broadly applicable to virtually every problem involving perception and action in the real world.

However, these advantages come at a price. Traditionally, the two most frequently cited limitations of probabilistic algorithms are *computational inefficiency*, and *a need to approximate*. Certainly, there is some truth to these criticisms. Probabilistic algorithms are inherently less efficient than non-probabilistic ones, due to the fact that they consider entire probability densities. However, this carries the benefit of increased robustness. The need to approximate arises from the fact that most robot worlds are continuous. Computing exact posterior distributions is typically infeasible, since distributions over the continuum possess infinitely many dimensions. Sometimes, one is fortunate in that the uncertainty can approximated tightly with a compact parametric model (e.g., discrete distributions or Gaussians); in other cases, such approximations are too crude and more complex representations most be employed.

None of these limitations, however, pose serious obstacles. Recent research has led to a range of algorithms that are computationally efficient and also highly accurate. To illustrate probabilistic algorithms in practice, this article describes three such algorithms in detail, and argues that the probabilistic paradigm is unique in its ability to solve hard robotics problems in uncertain and complex worlds.

2 Mobile Robot Localization

Let us take a first, deeper look into a specific probabilistic algorithm, which solves an important problem in mobile robotics, namely that of *localization*. Localization is the problem of finding out a robot's coordinates relative to its environment, assuming that one is provided with a map of the environment. Localization is a key component in various successful mobile robot systems (see e.g., [4, 51]). Occasionally is has been referred to as " the most fundamental problem to providing a mobile robot with autonomous capabilities" [16]. Particularly challenging is the *global localization problem*, where the robot does not know its initial position and therefore has to globally localize itself.

Approached probabilistically, the localization problem is a density estimation problem, where a robot seeks to estimate a posterior distribution over the space of its poses conditioned on the available data. The term *pose*, in this article, refers to a robot's x-y-coordinates together with its heading direction θ. Denoting the robot's pose at time t by s_t, and the data leading up to time t by $d_{0...t}$, the posterior is conveniently written as

$$p(s_t|d_{0...t}, m). \tag{1}$$

Here m is the model of the world (e.g., a map). For brevity, we will denote this posterior $b_t(s_t)$, and refer to it as the robot's *belief state* at time t.

Sensor data typically comes in two flavors: Data that characterizes the momentary situation (e.g., camera images, laser range scans), and data relating to change of the situation (e.g., motor controls or odometer readings). Referring to the former as *observations* and the latter as *action data*, let us without loss of generality assume that both types of data arrives in an alternating sequence:

$$d_{0...t} \quad = \quad o_0, a_0, o_1, a_1, \ldots, a_{t-1}, o_t. \tag{2}$$

Here o_t denote the observation and a_t denotes the action data item collected at time t.

To estimate the desired posterior $p(s_t|d_{0...t}, m)$, probabilistic approaches frequently resort to a *Markov assumption*, which states that the past is independent of the future given knowledge of the current state. The Markov assumption is often referred to as the *static world assumption*, since it assumes the robot's pose is the only state in the world that would impact more than just one isolated sensor reading. Practical experience suggests, however, that probabilistic algorithms are robust to mild violations of the Markov assumption, and extensions exist that go beyond this assumption (e.g., [33]).

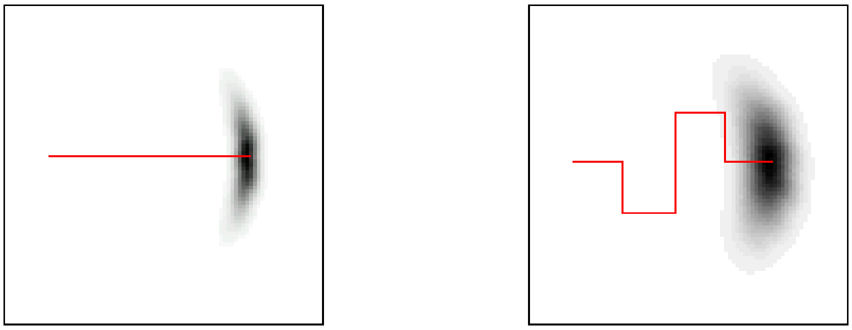

Figure 1: Probabilistic generalization of mobile robot kinematics: Each dark line illustrates a commanded robot path, and the grayly shaded shows the posterior distribution of the robot's pose. The darker an area, the more likely it is. The path in the left diagram is 40 meters and the one on the right is 80 meters long.

The desired posterior is now computed using a recursive formula, which is obtained by applying Bayes rule and the theorem of total probability to the original expression, exploiting the Markov assumption twice:

$$\begin{aligned}
b_t(s_t) \quad &= \quad p(s_t|o_0,\ldots,a_{t-1},o_t,m) \\
&\overset{\text{Bayes}}{=} \quad \eta_t\ p(o_t|o_0,\ldots,a_{t-1},s_t,m)\ p(s_t|o_0,\ldots,a_{t-1},m) \\
&\overset{\text{Markov}}{=} \quad \eta_t\ p(o_t|s_t,m)\ p(s_t|o_0,\ldots,a_{t-1},m) \\
&\overset{\text{Tot.Prob.}}{=} \quad \eta_t\ p(o_t|s_t,m) \int p(s_t|o_0,\ldots,a_{t-1},s_{t-1},m)\ p(s_{t-1}|o_0,\ldots,a_{t-1},m)\ ds_{t-1} \\
&\overset{\text{Markov}}{=} \quad \eta_t\ p(o_t|s_t,m) \int p(s_t|a_{t-1},s_{t-1},m)\ p(s_{t-1}|o_0,\ldots,o_{t-1},m)\ ds_{t-1} \\
&= \quad \eta_t\ p(o_t|s_t,m) \int p(s_t|a_{t-1},s_{t-1},m)\ b_{t-1}(s_{t-1})\ ds_{t-1}. \qquad (3)
\end{aligned}$$

Here η_t is a constant normalizer which ensures that the result sums up to 1. Within the context of mobile robot localization, the result of this transformation

$$b_t(s_t) \quad = \quad \eta_t\ p(o_t|s_t,m) \int p(s_t|a_{t-1},s_{t-1},m)\ b_{t-1}(s_{t-1})\ ds_{t-1} \qquad (4)$$

is often referred to as *Markov localization* [9, 32, 45, 49, 81, 87], but it equally represents the basic update equation in Kalman filters [48], Hidden Markov models [70], and dynamic belief networks [18, 75]. Kalman filter [48], which is historically the most popular approach for position tracking, represents beliefs by Gaussians. The vanilla Kalman filter also assumes Gaussian noise and linear motion equations; however, extensions exist that relax some of these assumptions [44, 63]. Kalman filters have been applied with great success to a range of tracking and mapping problems in robotics [58, 83]; though they tend not to work well for global localization or the kidnapped robot problem. Markov localization using discrete, topological representations for b were pioneered (among others) by Simmons and Koenig [81], whose mobile robot Xavier traveled more than 230 kilometers through CMU's hallways over a period of several years [80].

To implement Equation (4), one needs to specify $p(s_t|a_{t-1},s_{t-1},m)$ and $p(o_t|s_t,m)$. Both densities are usually time-invariant, that is, they do not depend on t, hence the time index can be omitted. The first density characterizes the effect of the robot's actions a on its pose, and can therefore be viewed as a probabilistic generalization of mobile robot kinematics; see Figure 1 for examples. The other density, $p(o|s,m)$, is a probabilistic model of perception. Figure 2 illustrates a sensor model for range finders, which uses ray-tracing and a mixture of four parametric densities to calculate $p(o|s,m)$. In our implementation, both of these probabilistic models are quite crude, using uncertainty to account for model limitations [32].

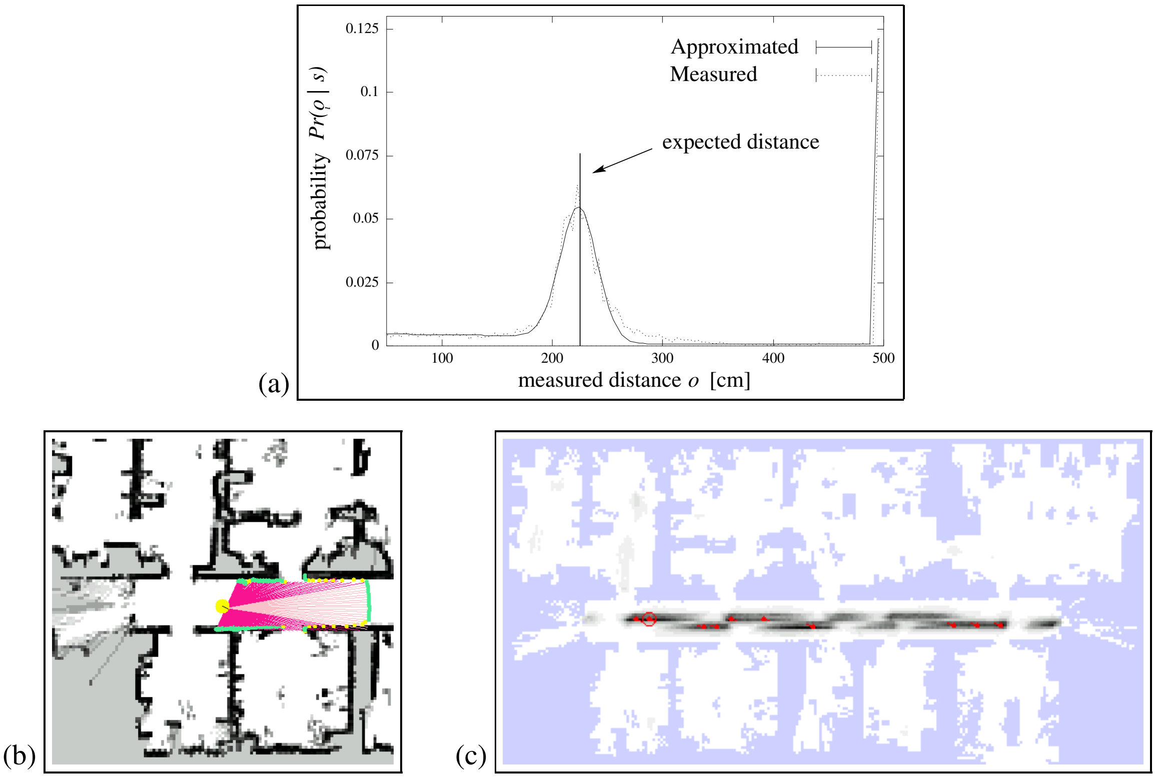

Figure 2: Probabilistic sensor model for laser range finders: (a) The density $p(o|s, m)$ relates the actual, measured distance of a sensor beam to its expected distance computed by ray tracing, under the assumption that the robot's pose is s. A comparison of actual data and our (learned) mixture model shows good correspondence. Diagram (b) shows a specific laser range scan o, for which diagram (c) plots the density $p(o|s, m)$ for different locations in the map.

Figure 3 illustrates global mobile robot localization based on sonar measurements in an office environment, using our robot Rhino. The robot's path is outlined in the first diagram, along with four reference locations. Shown there is also the initial belief, which is uniform, since the robot does not know where it is. The posterior belief after moving from the first to the second reference location is shown in Figure 3b. At this point, most of the probability mass is located in the corridor, but the robot still does not know where it is. This diagram illustrates nicely one of the features of the probabilistic approach, namely its ability to pursue multiple hypotheses, weighted by sensor evidence. After moving to the third reference position, the belief is centered around two discrete locations, as shown in Figure 3c. Finally, after moving into one of the rooms, the symmetry is broken and the robot is highly certain as to where it is (Figure 3d).

Of fundamental importance for the design of probabilistic algorithms is the choice of the representation. One of the most powerful approximations is known as *importance sampling* [74], and the resulting algorithm is known under names like *particle filters* [22, 23, 61, 68], *condensation algorithm* [42, 43] and *Monte Carlo localization* [19, 29]; here we refer to it as Monte Carlo localization (MCL). The basic idea of MCL is to approximate $b(s)$ with a weighted set of samples (particles), so that the discrete distribution defined by the samples approximates the desired one. The weighting factors are called *importance factors* [74]. The initial belief is represented by a uniform sample of size m, that is, a set of m samples drawn uniformly from the space of all poses, annotated by the constant importance factor m^{-1}. MCL implements the update equation (4) by constructing a new sample set from the current one in response to an action item a_{t-1} and an observation o_t:

1. Draw a random sample s_{t-1} from the current belief $b_{t-1}(s_{t-1})$, with a likelihood given by the importance

(a) Uniform distribution, path (b) Belief b at second pose (c) Belief b at third pose (d) Belief b at fourth pose

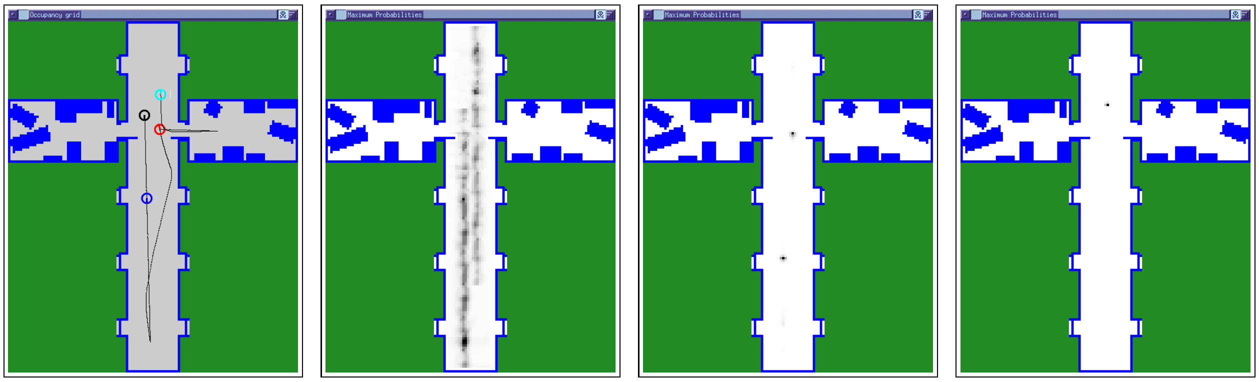

Figure 3: Example of grid-based Markov localization in a symmetric environment. (a) robot path, highlighting four robot poses, (b) to (d): belief at 2nd, 3rd, and 4th pose, respectively.

factors of the belief $b_{t-1}(s_{t-1})$.

2. For this s_{t-1}, guess a successor pose s_t, according to the distribution $p(s_t|a_{t-1}, s_{t-1}, m)$.

3. Assign a preliminary importance factor $p(o_t|s_t, m)$ to this sample and add it to the new sample set representing $b_t(s_t)$.

4. Repeat Step 1 through $3m$ times. Finally, normalize the importance factors in the new sample set $b_t(s_t)$ so that they add to 1.

Figure 4 shows MCL in action. Shown in the first diagram is a belief distribution (sample set) at the beginning of the experiment when the robot does not (yet) know its position. Each dot is a three-dimensional sample of the robot's x-y-location along with its heading direction θ. The second diagram shows the belief after a short motion segment, incorporating several sonar readings. At this point, most samples concentrate on two locations; however, the symmetry of the corridor makes it impossible to disambiguate them. Finally, the third diagram in Figure 4 shows the belief after the robot moves into one of the rooms, enabling it to disambiguate its location with high confidence.

The MCL algorithm is in fact quite efficient; slight modifications of the basic algorithms [56, 92] require as few as 100 samples for reliable localization, consuming only a small fraction of time available on low-end PC. It can also be implemented as any-time algorithm [17, 95], meaning that it can adapt to the available computational resources by dynamically adjusting the number of samples m. With slight modifications—such as sampling from the observation [92]—MCL has been shown to recover gracefully from global localization failures, such as manifested in the *kidnapped robot problem* [27], where a well-localized robot is teleported to some random location without being told. For these reasons, probabilistic algorithms like MCL are currently the best known methods for such hard localization problems.

Another feature of MCL is that its models—in particular $p(s|a, s, m)$, $p(o|s, m)$ and the map—can be extremely crude and simplistic, since probabilistic models carry their own notion of uncertainty. This makes probabilistic algorithms relatively easy to code. In comparison, traditional robotics algorithms that rely on deterministic models make much stronger demands on the accuracy of the underlying models.

Figure 4: Global localization of a mobile robot using MCL.

3 Mapping

A second area of robotics, where probabilistic algorithms have proven remarkably successful, is *mapping*. Mapping is the problem of generating maps from sensor measurements. This estimation problem is much higher-dimensional than the robot localization problem—in fact, in its pure form one could argue the problem possesses infinitely many dimensions. What makes this problem particularly difficult is its chicken-and-egg nature, which arises from the fact that position errors accrued during mapping are difficult to compensate [71]. Put differently, localization with a map is relatively easy, as is mapping with known locations. In combination, however, this problem is hard.

This section will review three major paradigms in mobile robot mapping, all of which are probabilistic and follow from the same mathematical framework. Let us begin by the most obvious idea, which is the idea of using the same approach for mapping as for localization. If we augment the state s that is being estimated via Equation (4) by the map m_t—the subscript t indicates that we allow the map to change over time—Equation (4) becomes

$$b_t(s_t, m_t) \quad = \quad \eta_t' \; p(o_t|s_t, m_t) \; \int\int p(s_t, m_t|a_{t-1}, s_{t-1}, m_{t-1}) \; b_{t-1}(s_{t-1}, m_{t-1}) \; ds_{t-1} \; dm_{t-1}. \quad (5)$$

If the map is assumed to be static—which is commonly assumed throughout the literature—the maps at times t and $t-1$ will be equivalent. This implies that $p(s_t, m_t|a_{t-1}, s_{t-1}, m_{t-1})$ is zero if $m_t \neq m_{t-1}$, and $p(s_t|a_{t-1}, s_{t-1}, m_{t-1})$ if $m_t = m_{t-1}$. The integration over maps in (5) is therefore eliminated, yielding

$$b_t(s_t, m) \quad = \quad \eta_t' \; p(o_t|s_t, m) \; \int p(s_t|a_{t-1}, s_{t-1}, m) \; b_{t-1}(s_{t-1}, m) \; ds_{t-1}. \qquad (6)$$

The major problem with (6) is complexity. The belief $b_t(s_t, m)$ is a density function in a $(N+3)$-dimensional space, where N is the number of free parameters that constitute a map (e.g., a constant times the number of landmarks), and the additional 3 parameters specify the robot's pose. N can be very large (e.g., 1,000), which makes the posterior estimation problem hard. To make matters worse, the belief $b_t(s_t, m)$ cannot easily be factorized, since the uncertainty of map items and robot poses are often highly correlated [83].

The most successful attempt to implement (6) employs Kalman filters [13, 14, 57, 58], which goes back to a seminal paper by Smith, Self, and Cheeseman [83]. Recall that Kalman filters represent beliefs by Gaussians; thus, they require $O(N^2)$ parameters to represent the posterior over an N-dimensional space. Calculating (6) involves matrix inversion, which can be done in $O(N^3)$ time [63]. This critically limits the number of features that can be mapped (see [59] for a recent attempt to escape this limitation using hierarchies of maps). In practice, this approach has been applied to mapping several hundreds of free parameters [57].

The basic Kalman filtering approach to mapping is also limited in a second, more important way. In particular, it requires that features in the environment can be uniquely identified—which is a consequence of the Gaussian noise assumption. For example, it does not suffice to know that the robot faces *a* door; instead, it must know *which* door it faces, to establish correspondence to previous sightings of the same door. This limitation is of great practical importance. It is common practice to extract a small number of identifiable features from the sensor data, at the risk of discarding all other information. Some recent approaches overcome this assumption by "guessing" the correspondence between measurements at different points in time, but they tend to be brittle if those guesses are wrong [36, 62]. However, if the assumptions are met, Kalman filters generate optimal estimates, and in particular they outperform any non-probabilistic approach.

An alternative approach, proposed in [91], seeks to estimate the *mode* of the posterior, $\operatorname{argmax}_m b(m)$, instead of the full posterior $b(m)$. This might appear quite modest a goal compared to the full posterior estimation. However, if the correspondence is unknown (and noise is non-Gaussian), this in itself is a challenging problem. To see, we note that the posterior over maps can be obtained in closed form:

$$\begin{aligned} b_t(m) &= p(m|d_{0\ldots t}) = \int b_t(s_t, m)\; ds_t \\ &= \eta_t''\, p(m) \int\int\cdots\int \prod_{\tau=0}^{t} p(o_\tau|s_\tau, m) \prod_{\tau=1}^{t} p(s_\tau|a_{\tau-1}, s_{\tau-1}, m)\; ds_1\; ds_2\; \ldots\; ds_t, \end{aligned} \tag{7}$$

where the initial pose is—somewhat arbitrarily—set to $s_0 = \langle 0, 0, 0\rangle$. This expression is obtained from (6) by integrating over s_t, followed by recursive substitution of the belief from time $t-1$ to time 0, and resorting of the resulting terms and integrals. For convenience, we will assume a uniform prior $p(m)$, transforming the problem into a maximum likelihood estimation problem. Notice that Equation (7) integrates over all possible paths, a rather complex integration. Unfortunately, we know of no way to calculate $\operatorname{argmax}_m b_t(m)$ analytically for data sets of reasonable size.

To find a solution, we notice that the robot's path can be considered "missing variables" in the optimization problem; knowing them indeed greatly simplifies the problem. The statistical literature has developed a range of algorithms for such problems, one of which is the EM algorithm [20, 64]. This algorithm computes a sequence of maps, denoted $m^{[0]}$, $m^{[1]}$, $\ldots$, which successively increasing likelihood. The superscript $^{[k]}$ is not to be confused with the time index t or the index of a particle i; all it refers to is the iteration of the optimization algorithm.

EM calculates a new map by iterating two steps, an *expectation step*, or *E-step*, and a *maximization step*, or *M-step*:

- In the E-step, EM calculates an expectation of a joint log-likelihood function of the data and the poses, conditioned on the k-th map $m^{[k]}$ (and conditioned on the data):

  $$Q[m|m^{[k]}] = E_{m^{[k]}}[\log p(s_0, \ldots, s_t, d_{0\ldots t}|m^{[k]}) \mid d_{0\ldots t})]. \tag{8}$$

  This might appear a bit cryptic, but the key thing here is that computing Q involves calculating the posterior distribution over poses $s_0, \ldots, s_t$ conditioned on the k-th model $m^{[k]}$. This is good news, as we have already seen how to estimate the posterior over poses given a map. Technically, calculating (8) involves two localization runs through the data, a forwards run and a backwards run, since *all* data has to be taken into account when computing the posterior $p(s_\tau|d_{0\ldots t})$ (the algorithm above only considers data up to time τ). We also note that in the very first iteration, we do not have a map. Thus, $Q[m|m^{[k]}]$ calculates the posterior for a "blind" robot, i.e., a robot that ignores its measurements $o_1, \ldots, o_t$.

- In the M-step, the most likely map is computed given the pose estimates obtained in the E-step. This is formally written as

  $$m^{[k+1]} = \operatorname*{argmax}_m Q[m|m^{[k]}]. \tag{9}$$

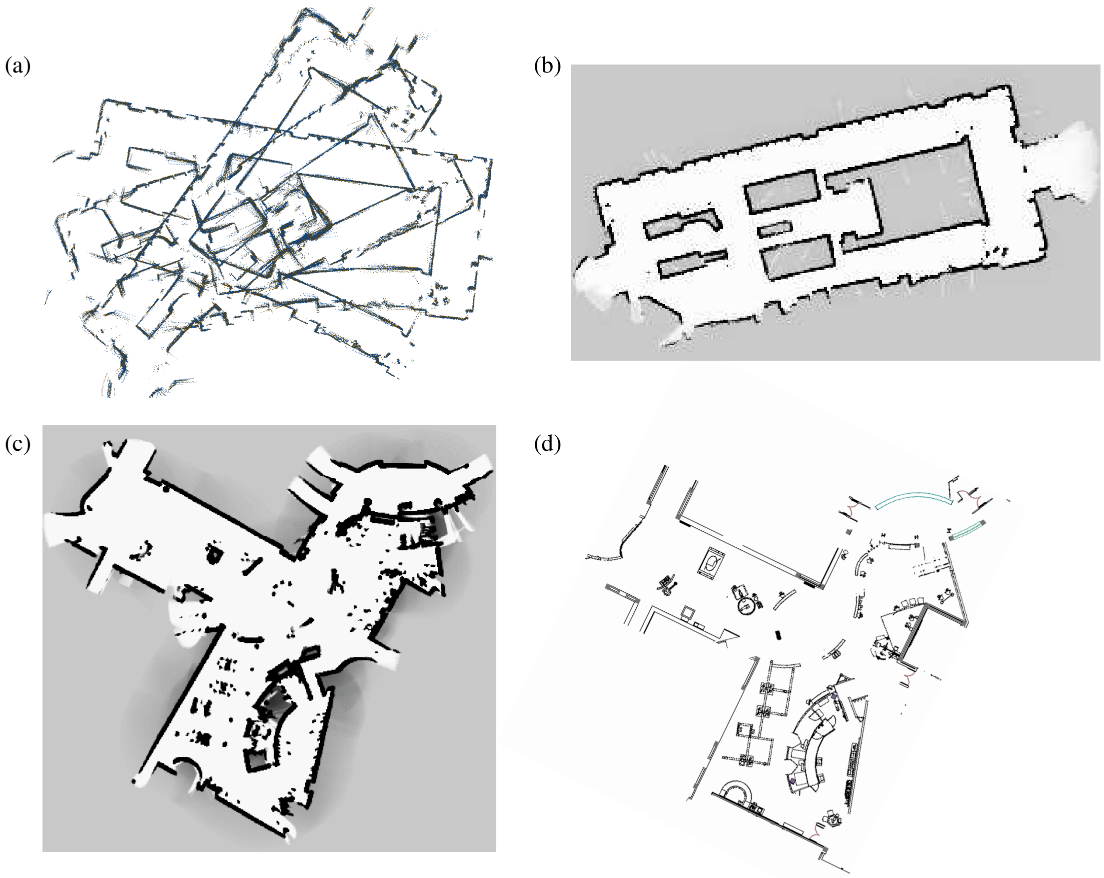

Figure 5: (a) Raw data of a large open hall (the Dinosaur Hall in the Carnegie Museum of Natural History, Pittsburgh, PA) and (b) map constructed using EM and occupancy grid mapping. (c) Occupancy grid map of another museum (The Tech Museum in San Jose, CA), and (d) architectural blueprint for comparison.

Technically, this is still a very difficult problem, since it involves finding the optimum in a high-dimensional space. However, it is common practice to decompose the problem into a collection of one-dimensional maximization problems, by stipulating a grid over the map and solving (9) independently for each grid cell. The maximum likelihood estimation for the resulting single-cell random variables is mathematically straightforward.

Iterations of both steps tends to increase the likelihood (currently we lack a proof of convergence due to the decomposition in the M-step). However, we found that this approach works very well in practice [91], solving hard mapping problems that were previously unsolved (see also [77, 78]).

The decomposition in the M-step is quite common for mapping algorithms that assume knowledge of the robot's pose. It goes back to the seminal work by Elfes and Moravec on *occupancy grid mapping* [26, 66], a probabilistic algorithm that is similar, though not identical, to the M-step above. This brings us to the third mapping algorithm. Occupancy grid mapping is currently the most widely used mapping algorithm for mobile robots [3, 25, 37, 88, 93], often augmented by ad-hoc methods for localization during mapping. It is another prime example for the success of probabilistic algorithms in robotics.

Occupancy grid mapping addresses a much simpler problem that the one above, namely the problem of estimating a map from a set of sensor measurements *given* that one already knows the corresponding poses. Let $\langle x, y\rangle$ denote a specific grid cell, and $m_t^{\langle xy\rangle}$ be the random variable the models its occupancy at time t. Occupancy is a binary concept; thus, we will write $m_t^{\langle xy\rangle}{=}1$ if a cell is occupied, and $m_t^{\langle xy\rangle}{=}0$ if it is not.

Substituting $m_t^{\langle xy\rangle}$ into Equation (4) under the consideration that this is a discrete random variable yields

$$b_t(m_t^{\langle xy\rangle}) \quad = \quad \eta_t \; p(o_t | m_t^{\langle xy\rangle}) \sum_{m_t^{\langle xy\rangle}=0}^{1} p(m^{\langle xy\rangle} | a_{t-1}, m_{t-1}^{\langle xy\rangle}) \; b_{t-1}(m_{t-1}^{\langle xy\rangle}), \tag{10}$$

which in static worlds can be simplified to

$$b_t(m^{\langle xy\rangle}) \quad = \quad \eta_t \; p(o_t | m^{\langle xy\rangle}) \; b_{t-1}(m^{\langle xy\rangle}) \quad = \quad \eta_t \; \frac{p(m^{\langle xy\rangle} | o_t) \; p(o_t)}{p(m^{\langle xy\rangle})} \; b_{t-1}(m^{\langle xy\rangle}). \tag{11}$$

The second transformation pays tribute to the fact that in occupancy grid mapping, one often is given $p(m^{\langle xy\rangle} | o_t)$ instead of $p(o_t | m^{\langle xy\rangle})$. One could certainly leave it at this and calculate the normalization factor η_t at run-time. However, for binary random variable the normalizer can be eliminated by noticing that the so-called *odds*, which is the following quotient:

$$\frac{b_t(m^{\langle xy\rangle}{=}1)}{b_t(m^{\langle xy\rangle}{=}0)} \quad = \quad \frac{p(m^{\langle xy\rangle}{=}1 | o_t)}{p(m^{\langle xy\rangle}{=}0 | o_t)} \; \frac{p(m^{\langle xy\rangle}{=}0)}{p(m^{\langle xy\rangle}{=}1)} \; \frac{b_{t-1}(m^{\langle xy\rangle}{=}1)}{b_{t-1}(m^{\langle xy\rangle}{=}0)}. \tag{12}$$

As is easily shown [88], this expression has the closed-form solution

$$b_t(m^{\langle xy\rangle}{=}1) \quad = \quad 1 - \left\{ 1 + \frac{p(m^{\langle xy\rangle}{=}1)}{1 - p(m^{\langle xy\rangle}{=}1)} \left[\prod_{\tau=0}^{t} \frac{p(m^{\langle xy\rangle}{=}1 | o_\tau)}{1 - p(m^{\langle xy\rangle}{=}1 | o_\tau)} \; \frac{1 - p(m^{\langle xy\rangle}{=}1)}{p(m^{\langle xy\rangle}{=}1)} \right] \right\}^{-1} \tag{13}$$

All three of these algorithms have shown to be highly robust and accurate in practice, and there are among the best algorithms in existence. For example, Figure 5a shows a raw data set of a large hall (approximately 50 meters wide), along with the result of first applying EM, and then occupancy grid mapping using the poses estimated with EM (Figure 5b). The map in Figure 5c has been generated using a similar probabilistic algorithm that runs on-line (unlike EM) [90] (see also [35]); Figure 5d shows an architectural blueprint for comparison. Cyclic environments are among the most difficult ones to map, since the odometry error can be very large when closing the cycle. These results illustrate that EM and occupancy grid mapping yield excellent results in practice. While the maps shown here are two-dimensional, probabilistic algorithms have also successfully been applied to build three-dimensional maps [90].

These results illustrate that probabilistic algorithms are well suited for high-dimensional estimation and learning problems; in fact, we know of no comparable algorithm that can solve problems of equal hardness but does not explicitly address the inherent uncertainty in perception. To date, the best mapping algorithms are probabilistic, and most of them are versions of the three algorithms described above. Our analysis also suggests that probabilistic algorithms are somewhat of a natural fit for problems like the ones studied here. Past research has shown that many estimation and learning problems in robotics have straightforward solutions when expressed using the language of probability theory, with mapping just being one example.

4 Robot Control

Finally, let us turn our attention to the issue of robot control. The ultimate goal of robotics is to build robots that do the right thing. As stated in the introduction, we conjecture that a robot that takes its own uncertainty into account when selecting actions will be superior to one that does not.

Unfortunately, the field of probabilistic robot control is much poorer developed than that of probabilistic perception. This is because of the enormous computational complexity of decision making. However, within AI this issue has recently received considerable attention. Even in robotics, some noticeable successes have been achieved, where probabilistic algorithms outperformed conventional, non-probabilistic algorithms [45, 81].

One such algorithm is the *coastal navigation* algorithm [72, 73] (see also [45]), a motion planning algorithm for mobile robots that takes uncertainty into account. The algorithm was originally motivated by the observation that ships that navigate through open water without GPS often stay in close proximity to the coast, to reduce the danger of getting lost. The same applies to mobile robots: The choice of control can have a profound impact on the likelihood of localization errors. The coastal navigation algorithm selects paths accordingly, explicitly considering uncertainty.

To study this algorithm, let us step back a little and consider the mathematical framework that underlies this and many other probabilistic control algorithms: POMDPs. POMDP (short for *partially observable Markov decision processes*) is a framework for acting optimally under uncertainty in sequential decision tasks. While POMDPs can be traced back to the Seventies [65, 82, 84], the AI community has only recently begun to pay attention to this framework, motived by the work by Littman, Cassandra, and Kaelbling [46, 60]. POMDPs address the problem of choosing actions so as to minimize a scalar *(immediate) cost function*, denoted $C(s)$. For example, in robot motion planning, one might set $C(s) = 0$ for goal locations s, and -1 elsewhere. Since reaching a goal location typically requires a whole sequence of actions, the control objective is to minimize the *expected cumulative cost*:

$$J \;=\; \sum_{\tau=t+1}^{t+T} E[C(s_\tau)]. \tag{14}$$

Here the expectation is taken over all future states. T may be ∞—in which case cost is often discounted over time by an exponential factor.

Many important POMDP algorithms [46, 60] are *off-line* algorithms, in the sense that they calculate a policy for action selection for arbitrary situations (i.e., belief states) in an explicit, off-line phase. The policy is denoted π and maps belief states into actions.

The most prominent approach to calculating π is *value iteration* [1, 41], a version of dynamic programming for computing the expected cumulative cost of belief states that has become highly popular in the field of reinforcement learning [47, 85]. Value iteration in belief space computes a value function, denoted by V, which in the ideal case measures the expected cumulative cost if one starts in a state s drawn according to the belief distribution b, and acts optimally thereafter. Thus, the value $V(b)$ of the belief state is the best possible cumulative costs one can expect for being in b. This is expressed as

$$V(b) \;=\; \int \sum_{\tau=t+1}^{t+T} E[C(s_\tau) | s_t = s]\; b(s)\; ds. \tag{15}$$

Following [1, 85], the value function can be computed by recursively adjusting the value of individual belief states b according to

$$V(b) \;\longleftarrow\; \min_a \int [V(b') + C(b')]\; p(b'|a, b, m)\; db', \tag{16}$$

which assigns to $V(b)$ the *expected* value at the next belief, b'. Here the immediate cost of a belief state b' is obtained by integrating over all states $C(b') = \int C(s')\; b'(s')\; ds'$. The conditional distribution $p(b'|a, b, m)$ is the belief space counterpart to the next state distribution, which is obtained as follows:

$$p(b'|a, b, m) \;=\; \int p(b'|o', a, b, m)\; p(o'|a, b, m)\; do', \tag{17}$$

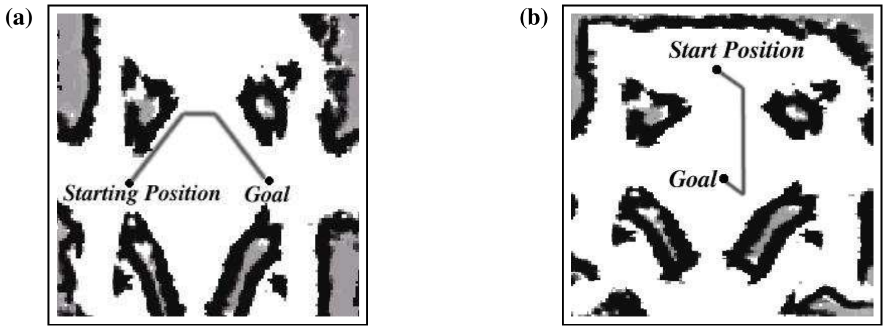

Figure 6: Coastal plans: the robot actively seeks the proximity of obstacles to improve its localization. The large open area in the center of this Smithsonian museum is approximately 20 meters wide and usually crowded with people.

where $p(b'|o', a, b, m)$ is a Dirac distribution defined through Equation (4), and

$$p(o'|a, b, m) \quad = \quad \int\int p(o'|s', m)\; p(s'|a, s, m)\; b(s)\; ds'\; ds. \tag{18}$$

Once V has been computed, the optimal policy is obtained by selecting actions that minimize the expected V-value over all available actions:

$$\pi(b) \quad = \quad \operatorname*{argmin}_a \int V(b')\; p(b'|a, b, m)\; db'. \tag{19}$$

While this approach defines a mathematically elegant and consistent way to compute the optimal policy from the known densities $p(s'|a, s, m)$ and $p(o'|s', m)$—which are in fact the exact same densities used in MCL—there are two fundamental problems. First, in continuous domains the belief space is the space of all distributions over the continuum, which is an infinitely-dimensional space. Consequently, no exact method exists for calculating V in the general case. Second, even if the state space is discrete—which is commonly assumed in the POMDP framework—the computational burden can be enormous. This is because for state spaces of size n, the corresponding belief space is a $(n-1)$-dimensional continuous space. The best known solutions, such as the *witness algorithm* [46], can only handle problems of the approximate size of 100 states, and a planning horizon of no more than $T = 5$ steps. In contrast, state spaces in robotics routinely possess orders of magnitude more states, even under crude discretizations. This makes approximating imperative.

Coastal navigation is an extension of POMDPs that relies on an approximate representation for belief states b. The underlying assumption is that the *exact* nature of the uncertainty is irrelevant for action selection; instead, it suffices to know the *degree of uncertainty* as expressed by the *entropy* of a belief state $H[b]$. Thus, coastal navigation represents belief states by the following tuple:

$$\bar{b} \quad = \quad \langle\; \operatorname*{argmax}_s b(s);\; H[b]\; \rangle. \tag{20}$$

While this approximation is coarse, it causes value iteration to scale exponentially better to large state spaces than the full POMDP solution, while still exhibiting good performance in practice.

Figure 6 shows an example trajectory calculated by the coastal navigation algorithm for a large, featureless environment: a Smithsonian museum in Washington, DC. The goal of motion is to reach a target location with high probability. By considering uncertainty, the coastal planner generates paths that actively seeks the proximity of known obstacles so as to minimize the localization error—at the expense of an increased path length when compared to the shortest path. Experimental results [73] have shown that the success rate of the coastal planner is superior to conventional shortest path planners that ignore the inherent uncertainty in robot motion.

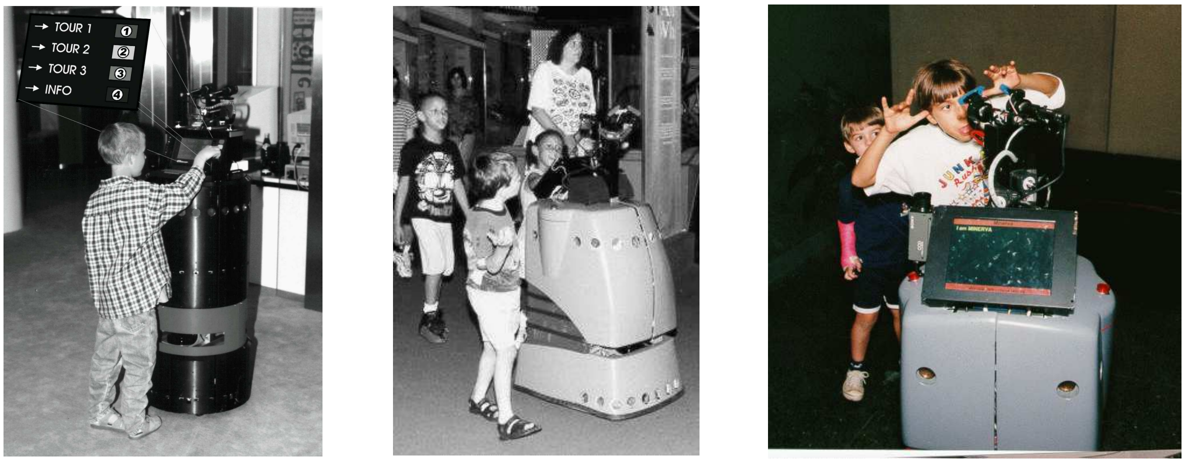

Figure 7: Probabilistic algorithms were used pervasively for the museum tour-guide robots Rhino (left image) and Minerva (right two images), which guided thousand of people through crowded museums fully autonomously.

Coastal navigation is only one out of many examples. It highlights an important principle, namely that crude approximations can go a long way when implementing probabilistic control algorithms. Recent research had led to a range of other control algorithms that rely on approximate probabilistic representations. Of particular importance are algorithms for maximizing knowledge gain, which typically rely on a single-step search horizon to generate robot control. Examples include the rich work on robot exploration, in which robots (or teams thereof) select actions so as to maximally reduce their uncertainty about their environments [24, 50, 54, 79, 88, 94]. They also include work on active localization [11, 30], where a robot moves to places that maximally disambiguate its pose. Another class of approaches rely on tree search for policy determination, such as the work on active perception and sensor planning by Kristensen [52, 53]. His approach uses models of uncertainty to select the appropriate sensors in an indoor navigation task. All these approaches have demonstrated that probabilistic algorithms lead to more robust solutions to important control problems in robotics.

5 A Case Study: Museum Tour-Guide Robots

Probabilistic algorithms have been at the core of a number of state-of-the-art robot systems (see e.g., [2, 21]), such as the Xavier robot mentioned above [80]. In fact, recently the number of publications on statistically sound algorithms for perception and control has increased dramatically at leading robotics conferences.

In our own work, we recently developed two autonomous museum tour-guide robots [8, 89] (see also [40]), which pervasively employed probabilistic algorithms for navigation and people interaction. Pictures of both robots are shown in Figure 7. The robot shown on the left, Rhino, was the world's first museum tour-guide robot, which was deployed at the Deutsches Museum in Bonn in 1997. The other robot, Minerva, led thousands of people through a crowded Smithsonian museum in Washington, DC. Both robots were developed to showcase probabilistic algorithms in complex and highly dynamic environments. They were unique in their ability to navigate safely and reliably in the proximity of people.

The task of these robots was simple: to attract people, interact with them, and guide them from exhibit to exhibit. Several factors made this problem challenging. To find their way around, the robots had to know where they were. However, large crowds of people almost permanently blocked the robots' sensors, making localization a difficult problem. In fact, people frequently sought to confuse the robot, or force it close to hazards such as downward staircases. To make matters worse, the robots' ability to sense such hazards was extremely limited. For example, the sensors consistently failed to sense glass cases put up for the protection of certain exhibits, and neither robot possessed a sensor to detect staircases. Thus, accurate localization played a prime role in avoiding collisions with such "invisible" obstacles and hazards such as staircases, whose location was modeled in the map.

To challenge the autonomy of our robots, we did not modify the environment in any way. And even though the museums were crowded, the robots had to navigate at approximate walking speeds to sustain people's interest, while avoiding collisions with people at all costs.

Detailed descriptions of the robots' software architecture and experimental findings are beyond the scope of this paper (see [8, 89]); here we remark that probabilistic algorithms were employed at virtually all levels of the software architecture. In total, both robots traversed a distance of more than 60km, with average speeds between 30cm/sec and 80cm/sec, and top speeds well above 160cm/sec. In Rhino's case, we carefully evaluated every failure, and concluded that only one major localization failure was observed over a period of six days [8]; however, this localization failure coincided with a malfunctioning of the sonar sensors. Rhino employed a probabilistic collision avoidance routine that guaranteed, with high likelihood, that the robot does not collide with "invisible" obstacles even when the robot is highly uncertain where it is [31]. In addition, Minerva employed probabilistic algorithms to learn occupancy grid maps of the museums. In other experiments, we have devised practical probabilistic algorithm for active exploration, both in pursuit of finding out where a robot is [11] and learning maps of unknown terrain [88] with teams of collaborating robots [10].

In all these cases, the probabilistic nature of the algorithms has been essential for achieving robustness in the face of uncertain and dynamic environments. And all these algorithms rely on sometimes remarkably simple approximations and shortcuts that make hard problems computationally tractable.

6 Discussion

The last few decades have led to a flurry of different software design paradigms for autonomous robots. Early work on model-based robotics often assumed the availability of a complete and accurate model of the robot and its environment, relying on planners (or theorem provers) to generate actions [12, 55, 76]. Such approaches are often inapplicable to robotics due to the difficulty of generating models that are sufficiently accurate and complete. Recognizing this limitation, some researchers have advocated model-free reactive approaches. Instead of relying on planning, these approaches require programmers to program controllers directly. A popular example of this approach is the "subsumption architecture" [7], where controllers are composed of small finite state automata that map sensor readings into control while retaining a minimum of internal state. Some advocates of this approach went as far as refusing the need for internal models and internal state altogether [7, 15]. Observing that *"the world is its own best model"* [6], behavior-based approaches usually rely on immediate sensor feedback for determining a robot's action. Obvious limits in perception (e.g., robots can't see through walls) pose clear boundaries on the type tasks that can be tackled with this approach. This leads us to conclude that while the world might well be its most *accurate* model, it is not necessarily its most *accessible* one [86]. And accessibility matters!

The probabilistic approach is somewhere between these two extremes. Probabilistic algorithms rely on models, just like the classical plan-based approach. For example, Markov localization requires a perception model $p(o|s, m)$, a motion model $p(s'|a, s)$, and a map of the environment. However, since these models are probabilistic, they only need to be approximate. This makes them much easier to implement (and to learn!) than if we had to meet the accuracy requirements of traditional approaches. Additionally, the ability to acknowledge existing uncertainty and even anticipate upcoming uncertainty in planning leads to qualitatively new solutions in a range of robotics problems, as demonstrated in this article.

Probabilistic algorithms are similar to behavior-based approaches in that they place a strong emphasis on sensor feedback. Because probabilistic models are insufficient to predict the actual state, sensor measurements play a vital role in state estimation and, thus, in determining a robot's actual behavior. However, they differ from behavior-based approaches in that they rely on planning, and in that a robot's behavior is not just a function of a small number of recent sensor readings. As an example that illustrates the importance of the latter, imagine placing a mobile robot in an a crowded place full of invisible hazards! Surely, the problem can be solved by adding more sensors; however, such an approach is expensive at best, but more often it will be plainly infeasible due to lack of appropriate sensors. Our robot Rhino, for example, was equipped with five different sensor systems—vision, laser, sonar, infrared and tactile—yet it still could not perceive all the hazards and obstacles in this fragile environment with the necessary reliability (see [8] for a discussion). Thus, it seems unlikely that a reactive approach could have performed anywhere nearly as reliably and robustly in this task.

Probabilistic robotics is closely related to a rich body of literature on uncertainty in AI ([39, 67] are good starting points, as are any recent proceedings of UAI). In fact, many of the basic algorithms in robotics have counterparts in the UAI community, the major difference being that their focus tends be on discrete spaces, whereas robots typically live in continuous spaces. Also, building real robotics systems constrains the assumptions under which one can operate. Consequently, approximations and real-time algorithms play a greater role in robotics than they play currently in mainstream AI.

One of the most exciting aspect of the probabilistic paradigm is that it opens great new opportunities for basic research in robotics and AI, with great potential for high impact in robotics and beyond. Probabilistic algorithms are still far from mainstream in robotics, and there is a range of problems that appear to be highly amenable to probabilistic solutions. We therefore conclude this article by laying out five broad areas of research which we deem to be highly important.

- **Representations.** The choice of representation is crucial in the design of any probabilistic algorithm, as it determines its robustness, efficiency, and accuracy. Recent research has already led to a range of representations for probabilistic information in continuous spaces, such as the particle representation in the example described above. However, the development of new representations is absolutely essential for scaling up to more complex problems, such as the control of highly articulated robots or multi-robot coordination.

- **Learning.** The probabilistic paradigm lends itself naturally to learning, yet very little work has been carried out on automatically learning models (or behaviors) in real-world robotic applications using probabilistic representations. Many of today's best learning algorithms are grounded in statistical theory similar to the one underlying the current approach. We conjecture that a better understanding of how to automatically acquire probabilistic models and behaviors over the lifetime of a robot has the potential to lead to new, innovative solutions to a range of hard open problems in robotics.

- **High-Level Reasoning and Programming.** Current research on probabilistic robotics predominately focuses on low-level perception and control. However, the issues raised in this paper transcend to all levels of reasoning and decision making [5]. The issue of probabilistic high-level reasoning and programming for robots remains poorly explored. Research is needed on algorithms that integrate probabilistic representations into high-level robot control (see e.g., [34, 38, 69]).

- **Theory.** The groundedness in statistical theory makes probabilistic approaches to robotics well-suited for theoretical investigation. However, existing models are often too restrictive to characterize robot interaction in complex environments. For example, little is known about the consequences of violating the Markov assumption that underlies much of today's work on localization and mapping, even though virtually all interesting environments violate this assumption. Little is also known about the effect of approximate representation on the performance of robotic controllers. A better theoretical understanding of probabilistic robotic algorithms is clearly desirable and would further our understanding of the benefits and limitations of this approach.

- **Innovative Applications.** Finally, we see a tremendous opportunity to apply probabilistic algorithms to a range of important robotic problems, including multi-robot coordination, sensor-based manipulation, and human robot interaction.

We hope that this article motivated the probabilistic approach to robotics, and stimulates new thinking in this exciting area. Ultimately, we believe that probabilistic algorithms is essential for a much broader class of embedded systems equipped with sensors and actuators.

Acknowledgments

The author likes to thank Wolfram Burgard, Frank Dellaert, Dieter Fox, Nicholas Roy, and the other members of CMU's Robot Learning Lab and the Mobile Robot Lab at the University of Bonn, whose contributions to this research were essential.

References

[1] R. E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.

[2] A. Bennett and J.J. Leonard. A behavior-based approach to adaptivefeature mapping with autonomous underwater vehicles. *IEEE Journal of Oceanic Engineering*, 25(2):213–226, 2000.

[3] J. Borenstein. *The Nursing Robot System*. PhD thesis, Technion, Haifa, Israel, June 1987.

[4] J. Borenstein, B. Everett, and L. Feng. *Navigating Mobile Robots: Systems and Techniques*. A. K. Peters, Ltd., Wellesley, MA, 1996.

[5] C. Boutilier, R. Reiter, M. Soutchanski, and S. Thrun. Decision-theoretic, high-level robot programming in the situation calculus. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Austin, TX, 2000. AAAI.

[6] R. Brooks. Elephants don't play chess. *Autonomous Robots*, 6:3–15, 1990.

[7] R. A. Brooks. A robot that walks; emergent behaviors from a carefully evolved network. *Neural Computation*, 1(2):253, 1989.

[8] W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1-2):3–55, 1999.

[9] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the absolute position of a mobile robot using position probability grids. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, Menlo Park, August 1996. AAAI, AAAI Press/MIT Press.

[10] W. Burgard, D. Fox, M. Moors, R. Simmons, and S. Thrun. Collaborative multi-robot exploration. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, 2000. IEEE.

[11] W. Burgard, D. Fox, and S. Thrun. Active mobile robot localization. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI)*, San Mateo, CA, 1997. Morgan Kaufmann.

[12] J. Canny. *The Complexity of Robot Motion Planning*. The MIT Press, Cambridge, MA, 1987.

[13] J.A. Castellanos, J.M.M. Montiel, J. Neira, and J.D. Tardós. The SPmap: A probabilistic framework for simultaneous localization and map building. *IEEE Transactions on Robotics and Automation*, 15(5):948–953, 1999.

[14] J.A. Castellanos and J.D. Tardós. *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach*. Kluwer Academic Publishers, Boston, MA, 2000.

[15] J. Connell. *Minimalist Mobile Robotics*. Academic Press, Boston, 1990.

[16] I.J. Cox. Blanche—an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on Robotics and Automation*, 7(2):193–204, 1991.

[17] T.L. Dean and M. Boddy. An analysis of time-dependent planning. In *Proceeding of Seventh National Conference on Artificial Intelligence AAAI-92*, pages 49–54, Menlo Park, CA, 1988. AAAI, AAAI Press/The MIT Press.

[18] T.L. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Computational Intelligence*, 5(3):142–150, 1989.

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

[20] A.P. Dempster, A.N. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[21] E.D. Dickmanns, R. Behringer, D. Dickmanns, T. Hildebrandt, M. Maurer, J. Schiehlen, and F. Thomanek. The seeing passenger car VaMoRs-P. In *Proceedings of the International Symposium on Intelligent Vehicles*, Paris, France, 1994.

[22] A Doucet. On sequential simulation-based methods for bayesian filtering. Technical Report CUED/F-INFENG/TR 310, Cambridge University, Department of Engineering, Cambridge, UK, 1998.

[23] A. Doucet, N.J. Gordon, and J.F.G. de Freitas, editors. *Sequential Monte Carlo Methods In Practice*. forthcoming, 2000.

[24] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. Robotic exploration as graph construction. *IEEE Transactions on Robotics and Automation*, 7(6):859–865, 1991.

[25] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, RA-3(3):249–265, June 1987.

[26] A. Elfes. *Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation*. PhD thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, 1989.

[27] S. Engelson. *Passive Map Learning and Visual Place Recognition*. PhD thesis, Department of Computer Science, Yale University, 1994.

[28] S. Engelson and D. McDermott. Error correction in mobile robot map learning. In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, pages 2555–2560, Nice, France, May 1992.

[29] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, Orlando, FL, 1999. AAAI.

[30] D. Fox, W. Burgard, and S. Thrun. Active markov localization for mobile robots. *Robotics and Autonomous Systems*, 25(3-4):195–207, 1998.

[31] D. Fox, W. Burgard, and S. Thrun. A hybrid collision avoidance method for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1998.

[32] D. Fox, W. Burgard, and S. Thrun. Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11:391–427, 1999.

[33] D. Fox, W. Burgard, S. Thrun, and A.B. Cremers. Position estimation for mobile robots in dynamic environments. In *Proceedings of the AAAI Fifteenth National Conference on Artificial Intelligence*, 1998.

[34] S. Glesner and D. Koller. Constructing flexible dynamic belief networks from first-order probabilistic knowledge bases. In Ch. Froidevaux and J. Kohlas, editors, *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pages 217–226, Berlin, 1995. Springer Verlag.

[35] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 2000.

[36] J.-S. Gutmann and B. Nebel. Navigation mobiler roboter mit laserscans. In *Autonome Mobile Systeme*. Springer Verlag, Berlin, 1997.

[37] D. Guzzoni, A. Cheyer, L. Julia, and K. Konolige. Many robots make short work. *AI Magazine*, 18(1):55–64, 1997.

[38] J. Halpern. An analysis of first-order logics of probability. *Artificial Intelligence*, 46:311–350, 1990.

[39] D. Heckerman. A tutorial on learning with bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, 1995. (revised November, 1996).

[40] I. Horswill. Polly: A vision-based artificial agent. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*. MIT Press, 1993.

[41] R. A. Howard. *Dynamic Programming and Markov Processes*. MIT Press and Wiley, 1960.

[42] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision*, pages 343–356, 1996.

[43] M. Isard and A. Blake. Condensation: conditional density propagation for visual tracking. *International Journal of Computer Vision*, 1998. In press.

[44] A.M. Jazwinsky. *Stochastic Processes and Filtering Theory*. Academic, New York, 1970.

[45] L.P. Kaelbling, A.R. Cassandra, and J.A. Kurien. Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1996.

[46] L.P. Kaelbling, M.L. Littman, and A.R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.

[47] L.P. Kaelbling, M.L. Littman, and A.W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4, 1996.

[48] R. E. Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME, Journal of Basic Engineering*, 82:35–45, 1960.

[49] S. Koenig and R. Simmons. Passive distance learning for robot navigation. In L. Saitta, editor, *Proceedings of the Thirteenth International Conference on Machine Learning*, 1996.

[50] S. Koenig and R. G. Simmons. Exploration with and without a map. In *Proceedings of the AAAI Workshop on Learning Action Models at the Eleventh National Conference on Artificial Intelligence (AAAI)*, pages 28–32, 1993. (also available as AAAI Technical Report WS-93-06).

[51] D. Kortenkamp, R.P. Bonasso, and R. Murphy, editors. *AI-based Mobile Robots: Case studies of successful robot systems*, Cambridge, MA, 1998. MIT Press.

[52] Steen Kristensen. *Sensor Planning with Bayesian Decision Analysis*. PhD thesis, Faculty of Technology and Science, Aalborg University, Aalborg, Denmark, 1996.

[53] Steen Kristensen. Sensor planning with bayesian decision theory. *Robotics and Autonomous Systems*, 19(3–4):273–286, March 1997.

[54] B. Kuipers and Y.-T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, 8:47–63, 1991.

[55] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[56] S. Lenser and M. Veloso. Sensor resetting localization for poorly modelled mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, 2000. IEEE.

[57] J. J. Leonard and H. F. Durrant-Whyte. *Directed Sonar Sensing for Mobile Robot Navigation*. Kluwer Academic Publishers, Boston, MA, 1992.

[58] J.J. Leonard, H.F. Durrant-Whyte, and I.J. Cox. Dynamic map building for an autonomous mobile robot. *International Journal of Robotics Research*, 11(4):89–96, 1992.

[59] J.J. Leonard and H.J.S. Feder. A computationally efficient method for large-scale concurrent mapping and localization. In J. Hollerbach and D. Koditschek, editors, *Proceedings of the Ninth International Symposium on Robotics Research*, Salt Lake City, Utah, 1999.

[60] M.L. Littman, A.R. Cassandra, and L.P. Kaelbling. Learning policies for partially observable environments: Scaling up. In A. Prieditis and S. Russell, editors, *Proceedings of the Twelfth International Conference on Machine Learning*, 1995.

[61] J. Liu and R. Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93, 1998.

[62] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.

[63] P.S. Maybeck. The Kalman filter: An introduction to concepts. In *Autonomous Robot Vehicles*. Springer verlag, 1990.

[64] G.J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley Series in Probability and Statistics, New York, 1997.

[65] George E Monahan. A survey of partially observable markov decision processes: Theory, models, and algorithms. *Management Science*, 28(1):1–16, 1982.

[66] H. P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, pages 61–74, Summer 1988.

[67] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers, San Mateo, CA, 1988.

[68] M. Pitt and N. Shephard. Filtering via simulation: auxiliary particle filter. *Journal of the American Statistical Association*, 1999. Forthcoming.

[69] D. Poole. Probabilistic horn abduction and bayesian networks. *Artificial Intelligence*, 64:81–129, 1993.

[70] L.R. Rabiner and B.H. Juang. An introduction to hidden markov models. In *IEEE ASSP Magazine*, 1986.

[71] W.D. Rencken. Concurrent localisation and map building for mobile robots using ultrasonic sensors. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2129–2197, Yokohama, Japan, July 1993.

[72] N. Roy, W. Burgard, D. Fox, and S. Thrun. Coastal navigation: Robot navigation under uncertainty in dynamic environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1999.

[73] N. Roy and S. Thrun. Coastal navigation for mobile robot navigation. In *Proceedings of Conference on Neural Information Processing Systems (NIPS)*, 1999. to appear.

[74] D.B. Rubin. Using the SIR algorithm to simulate posterior distributions. In M.H. Bernardo, K.M. an DeGroot, D.V. Lindley, and A.F.M. Smith, editors, *Bayesian Statistics 3*. Oxford University Press, Oxford, UK, 1988.

[75] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Englewood Cliffs, NJ, 1995.

[76] J. T. Schwartz, M. Scharir, and J. Hopcroft. *Planning, Geometry and Complexity of Robot Motion*. Ablex Publishing Corporation, Norwood, NJ, 1987.

[77] H. Shatkay. *Learning Models for Robot Navigation*. PhD thesis, Computer Science Department, Brown University, Providence, RI, 1998.

[78] H Shatkay and L. Kaelbling. Learning topological maps with weak local odometric information. In *Proceedings of IJCAI-97*. IJCAI, Inc., 1997.

[79] R. Simmons, D. Apfelbaum, W. Burgard, M. Fox, D. an Moors, S. Thrun, and H. Younes. Coordination for multi-robot exploration and mapping. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Austin, TX, 2000. AAAI.

[80] R. Simmons, R. Goodwin, K. Haigh, S. Koenig, and J. O'Sullivan. A layered architecture for office delivery robots. In *Proceedings of the First International Conference on Autonomous Agents*, Marina del Rey, CA, February 1997.

[81] R. Simmons and S. Koenig. Probabilistic robot navigation in partially observable environments. In *Proceedings of IJCAI-95*, pages 1080–1087, Montreal, Canada, August 1995. IJCAI, Inc.

[82] R.W. Smallwood and E.J. Sondik. The optimal control of partially observable markov processes over a finite horizon. *Operations Research*, 21:1071–1088, 1973.

[83] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I.J. Cox and G.T. Wilfong, editors, *Autonomous Robot Vehnicles*, pages 167–193. Springer-Verlag, 1990.

[84] E.J. Sondik. The optimal control of partially observable markov processes over the infinite horizon: Discounted costs. *Operations Research*, 26(2):282–304, 1978.

[85] R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[86] S. Thrun. To know or not to know: On the utility of models in mobile robotics. *AI Magazine*, 18(1):47–54, 1997.

[87] S. Thrun. Bayesian landmark learning for mobile robot localization. *Machine Learning*, 33(1), 1998.

[88] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.

[89] S. Thrun, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. MINERVA: A second generation mobile tour-guide robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1999.

[90] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, 2000. IEEE.

[91] S. Thrun, D. Fox, and W. Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31:29–53, 1998. also appeared in Autonomous Robots 5, 253–271.

[92] S. Thrun, D. Fox, and W. Burgard. Monte carlo localization with mixture proposal distribution. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Austin, TX, 2000. AAAI.

[93] B. Yamauchi and P. Langley. Place recognition in dynamic environments. *Journal of Robotic Systems*, 14(2):107–120, 1997.

[94] B. Yamauchi, P. Langley, A.C. Schultz, J. Grefenstette, and W. Adams. Magellan: An integrated adaptive architecture for mobile robots. Technical Report 98-2, Institute for the Study of Learning and Expertise (ISLE), Palo Alto, CA, May 1998.

[95] S. Zilberstein and S. Russell. Approximate reasoning using anytime algorithms. In S. Natarajan, editor, *Imprecise and Approximate Computation*. Kluwer Academic Publishers, Dordrecht, 1995.