Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva

S. Thrun[1], **M. Beetz**[3], **M. Bennewitz**[2], **W. Burgard**[2], **A.B. Cremers**[3], **F. Dellaert**[1]
D. Fox[1], **D. Hähnel**[2], **C. Rosenberg**[1], **N. Roy**[1], **J. Schulte**[1], **D. Schulz**[3]

[1]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA
[2]Computer Science Dept., University of Freiburg, Freiburg, Germany
[3]Computer Science Dept. III, University of Bonn, Bonn, Germany

Abstract

This paper describes Minerva, an interactive tour-guide robot that was successfully deployed in a Smithsonian museum. Minerva's software is pervasively probabilistic, relying on explicit representations of uncertainty in perception and control. During two weeks of operation, the robot interacted with thousands of people, both in the museum and through the Web, traversing more than 44km at speeds of up to 163 cm/sec in the unmodified museum.

1 Introduction

Robotics is currently undergoing a major change. While in the past, robots have predominately been employed in assembly lines and other well-structured environments, a new generation of service robots has begun to emerge, designed to assist people in everyday life [45, 81, 113, 122]. These robots must cope with the uncertainty that inherently exists in real-world application domains. Uncertainty arises from five primary sources:

1. **Environments.** Most interesting real-world environments are unpredictable. This is the case, for example, if robots operate in the proximity of people. The type environments considered in this paper are extremely dynamic, imposing significant uncertainty in the robot's internal perception of the world.

2. **Robots.** Robot hardware, too, is unpredictable. Robots are subject to wear-and-tear. Internal sensors for measuring robot actuation, such as odometry, are often only approximately correct.

3. **Sensors.** Sensors are inherently limited. The physical process that generates sensor measurements typically induces significant randomness on its outcome, making sensor measurements noisy. Moreover, range and resolution of sensors are intrinsically limited. Such limitations make it often impossible to measure important quantities when needed.

4. **Models.** Models of physical phenomena such as robots and robot environments are inherently approximate. Thus, the use of models introduces additional uncertainty, a fact that is still mostly ignored in robotics.

5. **Computation.** Robots are real-time systems, imposing limitations on the amount of computation carried out. Many of the algorithms described in this paper compute approximations, which introduces further uncertainty.

This article focuses on the probabilistic paradigm for robotics. This paradigm pays tribute to the inherent uncertainty in robot perception, relying on explicit representations of uncertainty when determining what to do. Viewed probabilistically, perception is a statistical state estimation problem, where information deduced from sensor data is represented by probability distributions. Planning and control is a utility optimization problem, in which a robot seeks to maximize expected utility (performance) under uncertainty. Our central conjecture is that the probabilistic approach is a viable solution to a large range of robot problems involving sensing in the physical world [143].

The focus of this article is a specific robot system, developed to evaluate the idea of probabilistic robotics in a complex real-world setting. *Minerva*, which is shown in Figure 2, is an interactive museum tour-guide robot. In the Fall of 1998, Minerva was deployed in one of the largest museum in the US: the Smithsonian Museum of American History in Washington, DC. Minerva operated in the center area of the museum's first floor, guiding visitors through a decade-old exhibition known as *Material World*. Figure 1 shows a panoramic view of the exhibition's main area. The robot's task involved attracting people and explaining to them the various exhibits while guiding them through the museum. The robot also enabled remote users to visit the museum through a Web link. This link allowed people to watch images collected in the museum, and to control the robot's operation. During its 14 day-long deployment, Minerva traversed more than 44 km through crowds of people, giving 620 tours to people and visiting more than 2,600 exhibits.

Operating in a museum is a challenging task, different in many aspects from more traditional operation domains of mobile robots. The museum environment can be densely crowded, with dozens of people gathering around the machine. Consequently, the robot's sensor measurements are extremely erroneous, turning simple tasks such as localization into major challenges. In fact, people often deliberately seek to compromise the system, which imposes additional challenges on the software design. We did not modify the environment in any way to facilitate the robot's operation. Thus, the robot had to rely on natural cues for its orientation. A further challenge arose from the need to operate at walking speed while at the same time avoiding collisions with people at all costs. Collisions with exhibits and other obstacles in the museum were almost equally undesirable, as many of the museum's exhibits were fragile and precious. A particular challenge was the fact that not all obstacles and hazards were "visible" to the robot's sensors. For example, the museum possessed a downward escalator in close proximity to the robot's operational area. Falling down this escalator was to be avoided; however, none of the robot's sensors were able to detect this hazard. Similarly, several obstacles were encased in glass cases; however, the robot's primary obstacle detection sensors, a pair of laser range finders, use light for measuring range and hence are unable to detect glass. The presence of such invisible hazards raised the question as to how to avoid them if they cannot even be detected.

At the same time, the museum environment creates a challenging human robot interaction problem. In the Smithsonian museum, most of the interaction took place over short periods of time, e.g., within ten minutes. People who approached the robot were typically

Figure 1: Panoramic view of the Material World Exhibition, Minerva's major operation area, which is located in the entrance area of the Smithsonian's National Museum of American History (NMAH).

inexperienced with robotic technology. However, providing visitors with complicated operation manuals was not an option. Instead, the robot had to be self-explanatory and engaging. Once leading a tour, the robot had to communicate effectively its intents and goals. People seemed to enjoy blocking the robot's path—so how can a robot effectively make progress even with dozens of people around? At other times, the challenge was to attract people, e.g., between tours when one group of people had just left. A final challenge was the design of an easy-to-use Web interface, enabling people all around the world to pay "virtual visits" to the museum. Museums are currently bound by their location when trying to attract people. The use of robots promises to open up museums to people all over the world, which could fundamentally alter the way museums operate. Minerva, thus, was a unique testbed for Internet technology using robots in public places.

As apparent from our domain description, uncertainty indeed plays a primary role in the Minerva project. Minerva's software was pervasively probabilistic, relying on explicit representation of uncertainty at various levels of perception, planning, and control. For example, Minerva employs a probabilistic algorithm for learning maps of its environment. Once a map has been learned, another probabilistic algorithm, called Markov localization, is used for localizing Minerva relative to its map. To generate motion, Minerva uses a probabilistic motion planner that anticipates future uncertainty, thereby reducing the chances of loosing track of the robot's position. The motion commands are then processed by a probabilistic collision avoidance module, which considers uncertainty when avoiding invisible hazards. Minerva also employs probabilistic learning algorithms at the user interaction level, enabling it to learn behaviors for attracting people, and to compose tours so as to meet the desired tour-length regardless of how crowded the museum is.

Minerva is a second generation tour-guide robot, following the successful example of the robot Rhino developed by the same team of researchers [19]. Rhino was deployed in the Deutsches Museum in Bonn in 1997, with many of the same probabilistic navigation algorithms. Minerva, however, went beyond Rhino in various ways, from using new probabilistic algorithms for learning maps from scratch, to a much-improved skill set for people interaction. This article describes the major software components of the Minerva robot, and compares them to those implemented on Rhino, Minerva's predecessor. We will argue throughout this article that the probabilistic nature of Minerva's primary software components was essential for its success.

2 Software Architecture Overview

Minerva's software architecture consists of approximately twenty distributed modules, which communicate asynchronously, as shown in Table 1. At the lowest level, various interface modules communicate directly with the robot's sensors and effectors (lasers, sonars, cameras, motors, pan/tilt unit, face, speech unit, touch-sensitive display, Internet server, etc.). On top

(a) (b) (c)

Figure 2: (a) Minerva. (b) Minerva gives a tour in the Smithsonian's National Museum of American History. (c) Interaction with museum visitors.

of that, various navigation modules perform functions like mapping, localization, collision avoidance, and path planning. The interaction modules determine the "emotional state" of the robot, control its head direction, and determine how to engage the people around it using sounds or speech. The Web interface consists of modules concerned with displaying information such as images and the robot's position on the Web, and with receiving Web user commands. Finally, the high-level modules perform global mission scheduling and control.

The main components of the data and control flow are as follows. All of these modules will be explained in more detail below. Sensor readings, in particular laser range scans, sonar scans, images from a camera pointed towards the ceiling, and odometry readings, are continuously broadcast across the network of modules. Off-line, before the deployment, these data are collected by the mapper, which builds a geometric map of the environment that is used by the localization module and the planning modules. On-line, during regular runtime, the map is not modified. Instead, the sensor data are sent to the localization module, which estimates the robot's pose relative to the map. The pose estimates are passed on to several modules, most notably the mission planner, the motion planner, and the reactive collision avoidance module. The mission planner monitors the user interface and the Web for user commands. It also exchanges information with the interaction modules, which control Minerva's face, voice, display, pan/tilt unit, etc. Once a tour has been chosen, it informs the motion planner of the location of the next exhibit to visit. The motion planner then generates via-points, which are passed on to the collision avoidance. The collision avoidance uses the sensor data (sonars, lasers) to "translate" the via-points into motor commands (forward and rotational velocities). Additionally, a related module uses the actual location estimates and the map to generate "virtual" obstacles that correspond to hazards in the map. These virtual measurements are also considered in collision avoidance. To accommodate changes in the robot's path that might arise from unexpected obstacles, the motion planner concurrently replans and generates new via-points as necessary.

Most of Minerva's software can adapt to the available computational resources. For example, modules that consume substantial processing time, such as the motion planner or the localization module, can produce results regardless of the time available for computation. The more processing cycles are available, however, the more accurate the result. In Minerva's software, resource flexibility is achieved by two mechanisms: selective data processing and any-time algorithms [32, 151]. Selective data processing is achieved by considering only a subset of the available data, which for example is the case in the localization routine. Other

high-level control and learning (mission planning, scheduling)
human interaction modules ("emotional" FSA, Web interface)
navigation modules (localization, map learning, path planning)
hardware interface modules (motors, sensors, Internet)

Table 1: Minerva's layered software architecture.

modules, such as the motion planning module, are any-time. That is, they can quickly draft initial solutions, which are then refined incrementally, so that an answer is available when needed.

Minerva's software does not possess a centralized clock or a centralized communication module. Synchronization of different modules is strictly decentralized, as in [47, 129]. Time-critical software (e.g., all device drivers), and software that is important for the safety of the robot (e.g., collision avoidance), are run on the robot's on-board computers. Higher-level software, such as the mission planner, is run on stationary off-board computers. This software organization has been found to yield robust behavior even in the presence of unreliable communication links (specifically the radio link which connects the on-board and off-board computers) and various other events that can temporarily delay the message flow or reduce the available computational resources. The modular, decentralized software organization eases the task of software configuration. Each module adds a certain competence, but not all modules are required to run the robot. The idea of decentralized, distributed decision making has been at the core of research on behavior-based robotics over the last decade [1, 17, 112], but there modules are typically much lower in complexity (e.g., finite state machines).

3 Mobile Robot Localization

3.1 The Localization Problem

A prime example of probabilistic computing in Minerva is *localization*. Localization is the problem of determining a robot's pose from sensor data. The term *pose* refers to the robot x-y-coordinates in the environment along with its heading direction. Localization enables the robot to find its way around the environment, and to avoid "invisible" hazards such as the downward escalator. It is therefore an essential component of Minerva's and Rhino's software architecture. The reader should notice that localization is a key component in many other successful mobile robot systems (see e.g., [12, 84, 76]). Occasionally, the localization problem has been referred to as "the most fundamental problem to providing a mobile robot with autonomous capabilities" [29].

The literature distinguishes three types of localization problems, in increasing order of difficulty:

1. **Position tracking.** Here the initial robot pose is known, and the goal of localization is to compensate small odometry error as the robot moves. Typically, the uncertainty in position tracking is bounded, making unimodal state estimators such as Kalman filters applicable [2, 57, 84, 120].

2. **Global localization.** If the robot does not know its initial pose, it faces a global localization problem. To localize itself from scratch, a robot must be able to cope with ambiguities and multiple, competing hypotheses during localization.

3. **Robot kidnapping [46].** This problem is a variant of the global localization problem in which a well-localized robot is tele-ported to some random pose *without being told.* It is harder than the global localization problem, since the robot might falsely believe it is somewhere else. Robot kidnapping simulates catastrophic failure of a localization routine and tests a robot's ability to recover from such failures—a critical ability for truly autonomous robots.

Minerva's localization algorithm can cope with all three localization problems.

3.2 Probabilistic Localization

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. Denoting the robot's pose at time t by s_t and the data leading up to time t by $d_{0\ldots t}$, the posterior is conveniently written as

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

Here m is the model of the world (e.g., a map). We will denote this posterior as $b_t(s_t)$, and refer to it as the robot's *belief state* at time t. For now we will assume the robot is given a map. Further below, we will describe our approach for learning a map from data.

Minerva uses laser range scans and images collected from a camera pointed towards the ceiling for localization. Such sensor data come in two flavors: Data that characterizes the momentary situation (e.g., camera images, laser range scans), and data relating to the change of the situation (e.g., motor controls or odometer readings). Referring to the former as *observation* and the latter as *action data*, let us without loss of generality assume that both types of data arrive in an alternated sequence:

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

Here o_i denotes the observation and a_i denotes the action data item at time i.

To estimate the desired posterior $p(s_t | d_{0\ldots t}, m)$, our approach resorts 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. Clearly, this is not the case in museums full of people. However, for now we will consider only the static case; an extension for dealing with environment dynamics will be described further below.

Armed with the necessary assumption, the desired posterior is now computed using a recursive formula, which is obtained by applying Bayes rule and the theorem of total probability. We also exploited the Markov assumption twice, as indicated:

$$\begin{aligned}
b_t(s_t) \quad &= \quad p(s_t | o_0, \ldots, a_{t-1}, o_t, m) \\
&\overset{\text{Bayes}}{=} \quad \eta_t \; p(o_t | o_0, \ldots, a_{t-1}, s_t, m) \; p(s_t | o_0, \ldots, a_{t-1}, m) \\
&\overset{\text{Markov}}{=} \quad \eta_t \; p(o_t | s_t, m) \; p(s_t | o_0, \ldots, a_{t-1}, m)
\end{aligned}$$

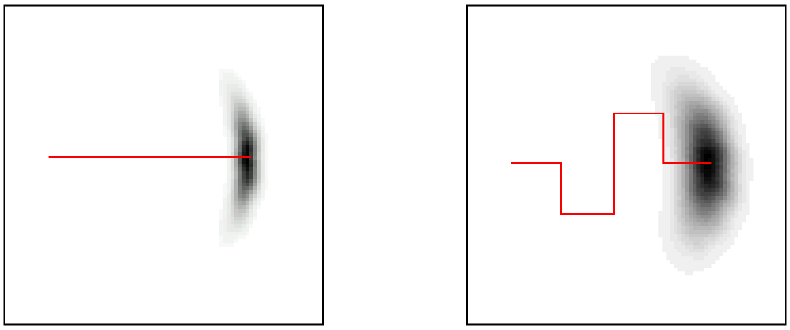


Figure 3: Probabilistic generalization of mobile robot kinematics: Each dark line illustrates a commanded robot path, and the shaded area shows the posterior distribution of the robot's pose. The darker an area, the more likely it is. The path in the left diagram is 40 meters and the one on the right is 80 meters long.

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

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* [20, 50, 68, 75, 132], but it equally represents a generalization of the basic update equation in Kalman filters [72], Hidden Markov models [108], and dynamic belief networks [33, 118]. Kalman filters [72], which are historically the most popular approach for position tracking, represent beliefs by Gaussians. The vanilla Kalman filter also assumes Gaussian noise and linear motion equations; however, extensions exist that relax some of these assumptions [67, 92]. Kalman filters have been applied with great success to a range of tracking and mapping problems in robotics [85, 134]; though they tend not to work well for global localization or the kidnapped robot problem (see [55] for an experimental comparison). Markov localization using discrete, topological representations for the map m were pioneered (among others) by Simmons and Koenig [132], whose mobile robot Xavier traveled more than 230 kilometers through CMU's hallways over a period of several years [130, 131].

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, 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 3 for examples. The other density, $p(o|s, m)$, is a probabilistic model of perception. Figure 4 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. For brevity, we omit a more detailed description of these models and instead refer the reader to [50].

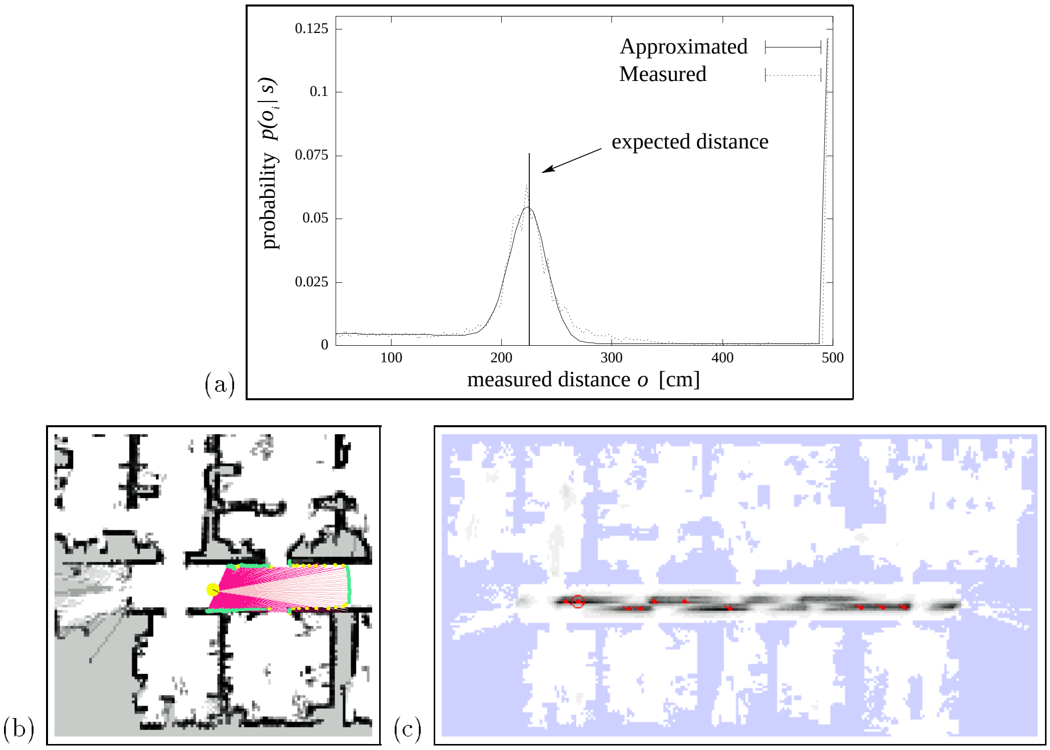


Figure 4: Probabilistic sensor model for laser range finders: (a) The density $p(o|s,m)$ relates the actual, measured distance of a sensor beam to its expected distance computed by ray tracing, under the assumption that the robot's pose is s. A comparison of actual data and our (learned) mixture model 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 5 illustrates how Minerva localizes itself from scratch (global localization). Initially, the robot does not know its pose; thus, $p(s_0)$ at time $t = 0$ is uniformly distributed. After incorporating one sensor reading according to the update rule (4), $p(s_1)$ is distributed as shown in Figure 5a. While this distribution is multi-modal, high probability mass is already placed near the correct pose. Finally, upon moving forward and subsequently incorporating another laser range measurement, the resulting posterior $p(s_2)$ at time $t = 2$ is centered on the correct pose, as shown in Figure 5b.

3.3 Monte Carlo Localization

Of fundamental importance for the design of probabilistic algorithms is the choice of the representation. During the museum exhibit, we used a piecewise constant grid-representation for representing the belief b, described in detail in [50]. More recently, we developed an alternative representation which is both more efficient than grids and more accurate. Therefore, we will describe it here.

The Monte Carlo localization algorithm (MCL) is a version of Markov localization that uses samples to approximate the belief b [34, 35, 39, 48, 83]. It is based on the SIR algorithm (SIR stands for *sampling/importance resampling*) originally proposed in [117], and is

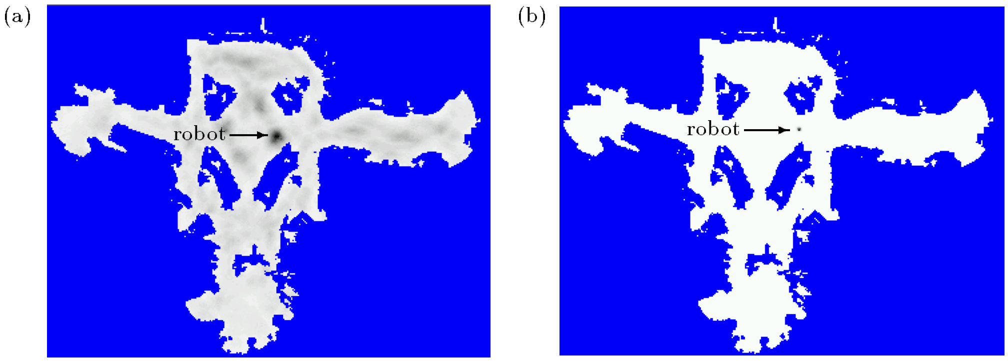


Figure 5: Global localization: (a) Pose posterior $b_t(s_t)$ after integrating a first laser scan (projected into 2D). The darker a pose, the more likely it is. (b) shows $b_t(s_t)$ after integrating a second sensor scan. Now the robot knows its pose with high certainty/accuracy.

a version of *particle filters* [40, 41, 88, 106]. Similar algorithms are known as *condensation algorithm* [64, 65] in computer vision, and *survival of the fittest* in artificial intelligence [73]. The basic idea of MCL is to approximate the belief distribution $b(s)$ with a weighted set of *samples*, also called *particles*, so that the discrete distribution defined by the samples approximates the desired one. The weighting factors are called *importance factors* [117]. The initial belief is represented by a uniform sample set of size k, that is, a set of k samples drawn uniformly from the space of all poses, annotated by the constant importance factor k^{-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 probability given by the importance factors of the belief $b_{t-1}(s_{t-1})$.

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

3. Assign the (unnormalized) 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 3 k times. Finally, normalize the importance factors in the new sample set $b_t(s_t)$ so that they sum up to 1.

Figure 6 shows MCL in action. Shown in the first diagram is a belief distribution (weighted sample set) at the beginning of the experiment, when the robot does not yet know its position. Each dot in this diagram is a three-dimensional sample of the robot's x-y-location along with its heading direction. The second diagram shows the belief after a short motion segment, incorporating several sensor measurements. At this point, most samples concentrate on the center region in the museum. However, the symmetry of this region makes it impossible to disambiguate different places in the museum. Finally, the third diagram in Figure 6 shows the belief a few moments later, where all samples focus on the correct pose. These results were obtained using a map of the museum's ceiling that is shown in Figure 10b, and that will be discussed further below.

The MCL algorithm is in fact quite efficient [34, 35, 48]; slight modifications of the basic algorithms [48, 83, 145] require only a few hundred samples for reliable localization,

(a) (b) (c)

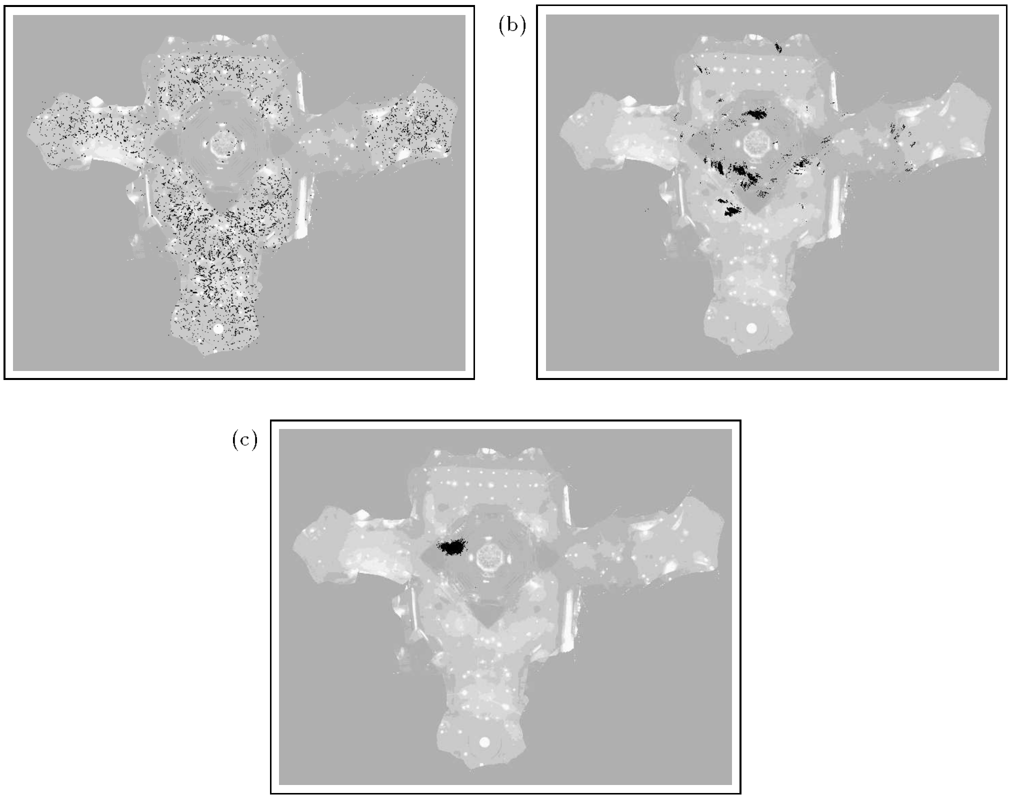


Figure 6: Global localization of a mobile robot with the MCL algorithm, using a camera pointed at the ceiling and the ceiling map shown in Figure 10b.

consuming only a small fraction of time available on a low-end PC. Our implementation is any-time [32, 151], meaning that it can adapt to the available computational resources by dynamically adjusting the number of samples k. The same extension has been shown to recover gracefully from global localization failures, such as manifested in the *kidnapped robot problem* mentioned above, where a well-localized robot is tele-ported to some random location without being told. Another feature of MCL (and Markov localization in general) is that the underlying models—in particular $p(s|a,s,m)$, $p(o|s,m)$ and the map m—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 lack a way to express model uncertainty. Hence. they make stronger demands on the accuracy of the underlying models (see the related work section for a more detailed discussion).

3.4 Finding People with Distance Filters

One of the key characteristics of the museum environment is that people populate it. At peak museum hours, we often counted more than 100 people surrounding the robot. The

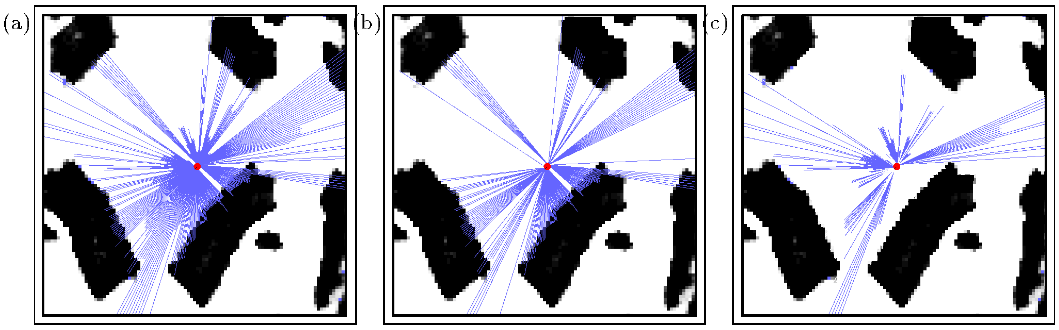


Figure 7: Distance filtering for locating people. Diagram (a) shows a laser range scan in a crowded situation, projected at the robot's most likely position. The distance filter sorts each individual measurement into two bins: "authentic" measurements, shown in (b), and measurements believe to correspond to people, shown in (c).

presence of people raises additional challenges to the robot's software. In particular, the Markov assumption in our localization algorithms requires a static environment, that is, one where the robot's pose is the only state that changes. People induce systematic noise on sensor data, invalidating the Markov assumption. While plain Markov localization (and MCL) is usually robust to small disturbances of this kind, it may easily fail when the number of nearby people is large, and if people intentionally attempt to confuse the robot—both of which frequently happened in the Smithsonian museum.

One approach for accommodating people is to include people's location in the state s that is being estimated. While such an approach is mathematically legitimate, it poses serious computational problems, since the state space is now much larger. It also requires probabilistic models of the motion of crowds, which might be difficult to obtain.

Minerva uses an alternative approach. It filters range measurements using a *distance filter* [50]. The distance filter sorts individual measurements into two bins: one that is believed to be "authentic," by which we mean that the sensor detected a known obstacle, and one that is believed to originate from a person or another unknown obstacle not part of the map.

The idea of the distance filter builds on a crucial property of range measurements: Measurement errors induced by people make range measurements shorter, not longer. Distance filters identify readings that, with high probability, are short. The probabilistic framework makes it straightforward to identify such readings. Let $o_{\alpha t}$ be the range measurement at time t, taken at an angle α relative to the robot's local coordinate system. Suppose, for a moment, that the robot pose s_t is known. Then the probability that an authentic measurement in the direction of α returns a value *larger* than $o_{\alpha t}$ is given by

$$p(o_{\alpha t}\ \text{short}|s_t, m) \quad = \quad \int_{o_\alpha > o_{\alpha t}} p(o_\alpha|s_t, m)\ do_\alpha \tag{5}$$

In other words, to compute the probability that a reading $o_{\alpha t}$ is short one simply integrates the probability of all measurements o_α that are larger than $o_{\alpha t}$. In practice, one of course does not know the exact pose s_t. Instead, all the robot is given is the belief b_t. The following expression calculates the probability that a sensor beam $o_{\alpha t}$ is short under uncertain pose information. It is obtained by integrating Expression (5) over all poses, weighted by the belief distribution b_t:

$$p(o_t \text{ short}|m) \quad = \quad \int \left[\int_{o_\alpha > o_{\alpha t}} p(o_\alpha|s_t, m)\, do_\alpha \right] b_t(s_t)\, ds_t \tag{6}$$

To accommodate people in localization, our approach simply discards measurements that with high probability (e.g., $p(o_t \text{ short}|m) > 0.99$) are short. It uses only the remaining measurements for localization. A systematic comparison and evaluation in [50] illustrates that distance filters are extremely effective in filtering our undesired sensor measurements while retaining sufficiently many authentic measurements to ensure accurate and reliable localization. Our comparison also shows that distance filters are capable of recovering from global localization failures (robot kidnapping).

An additional benefit of the distance filter arises from the fact that it aids human robot interaction. Several of Minerva's interaction strategies described below rely on the ability to find people. Assuming that people are the only unmodeled, moving objects in the museum, the distance filter can identify their locations.

4 Concurrent Mapping and Localization

We now return to the question of acquiring maps. Recall that our localization algorithm relies on a map m of the environment. In Rhino, Minerva's predecessor, the map was constructed by hand. However, manual mapping is tedious and precludes the rapid installation of a tour-guide robot. Minerva, in contrast, learns the map from sensor data.

From a statistical standpoint of view, concurrent mapping and localization is an estimation problem, similar to localization. This estimation problem is much higher-dimensional than the robot localization problem. For example, some of the grid maps shown in this paper require as many as 50,000 parameters. 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 [110]. Put differently, localization with a map is relatively easy, as is mapping with known locations. The problem of *simultaneously* localizing and mapping, however, is hard.

Currently, the best mapping algorithms are all probabilistic, following the same basic state estimation framework described above. One popular family of approaches, known as SLAM algorithms [22, 23, 84, 85, 134], employs Kalman filters [72, 92] for concurrently estimating robot poses and maps. Unfortunately, this approach requires that features in the environment can be uniquely identified—which is a consequence of the Gaussian noise assumption inherent in Kalman filters. For example, it does not suffice to know that the robot faces *a* doorway; instead, it must know *which* doorway it faces, to establish correspondence to previous sightings of the same doorway. 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 [56, 89]. In the Smithsonian museum, we know of no set of uniquely identifiable features that would give maps of the target resolution required for accurate localization.

4.1 EM Mapping

Minerva uses an alternative approach for mapping, which is based on the same mathematical framework as the Kalman filter approach above [144]. In particular, our approach seeks to estimate the *mode* of the posterior, denoted $\bar{m} = \text{argmax}_m \, p(m|d)$, instead of the full posterior $p(m|d)$. This might appear quite modest a goal compared to the full posterior estimation in the Kalman filter approach. 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:

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

where the initial pose is—somewhat arbitrarily—set to $s_0 = \langle 0, 0, 0 \rangle$. This expression is obtained from (4) 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 $\bar{m}$ 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 simplifies the problem greatly. The statistical literature has developed a range of algorithms for such problems, one of which is the EM algorithm [37, 94]. This algorithm computes a sequence of maps, denoted $m^{[0]}$, $m^{[1]}$, $\ldots$, which successively increasing likelihood. The superscript $^{[\cdot]}$ 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):

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

  The key observation is that computing Q involves calculating the posterior distribution over poses $s_0, \ldots, s_t$ conditioned on the K-th model $m^{[K]}$. We have already seen how to estimate the posterior over poses given a map, in the section on localization. Technically, calculating (8) involves two Markov 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

  $$m^{[K+1]} = \underset{m}{\text{argmax}} \, Q[m|m^{[K]}]. \qquad (9)$$

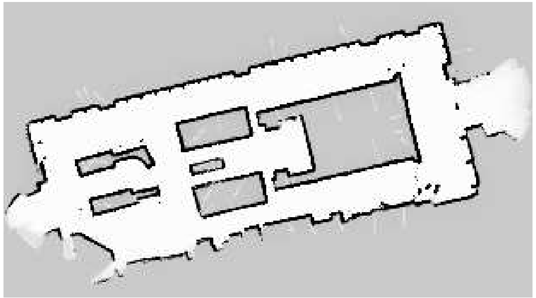
(a)  (b) 

Figure 8: (a) Raw data collected in 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.

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 [144].

Iterations of both steps tends to increase the log-likelihood. Details of the mathematical derivation and the implementation of this algorithm can be found in [144].

4.2 Occupancy Grid Maps

In a final mapping step, Minerva transforms its maps into occupancy grid maps [43, 97]. Occupancy grids are widely used in mobile robotics; see e.g., [11, 42, 58, 142, 149]. Most state-of-the-art algorithms for generating occupancy grid maps are probabilistic.

Occupancy grid mapping addresses a much simpler problem than 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 $m_t^{\langle xy\rangle}$ the random variable that 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) 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 simplifies 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 in (11) pays tribute to the fact that in occupancy grid mapping, one usually is given an inverse perceptual model, $p(m^{\langle xy\rangle} | o_t)$, instead of $p(o_t | m^{\langle xy\rangle})$ [142].

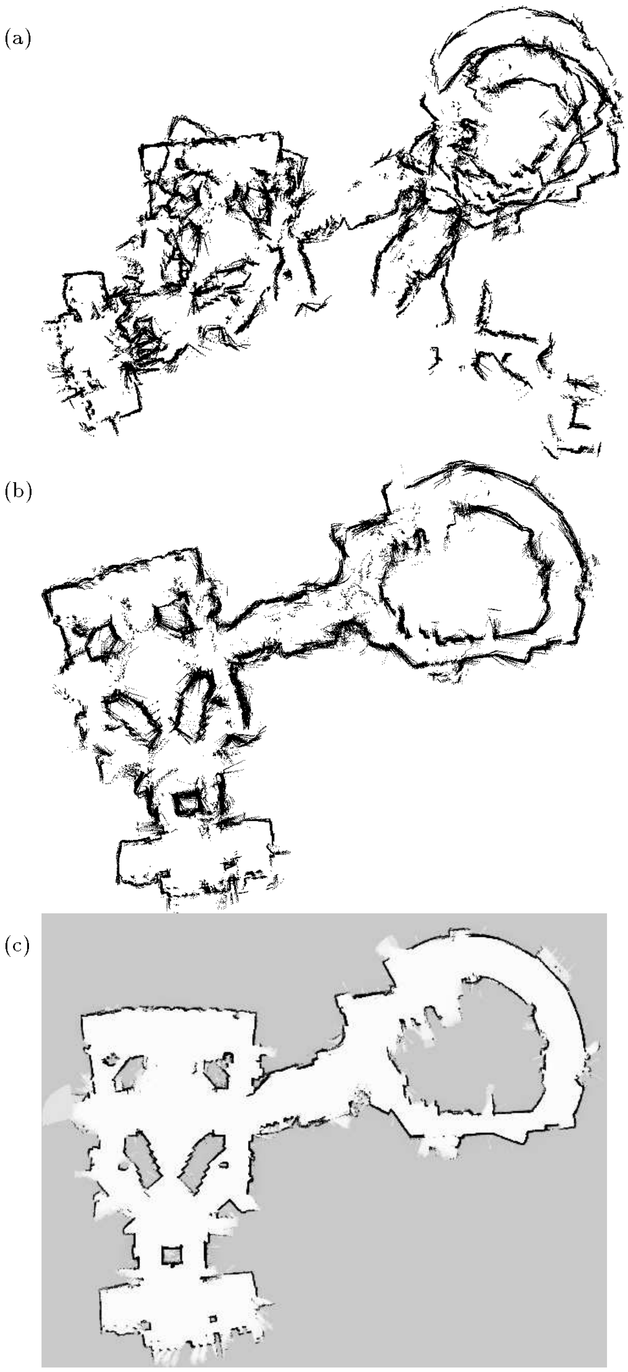


Figure 9: (a) Raw data collected in the Smithsonian museum. (b) Data after adjusting the scans using EM. (c) Final occupancy grid map. This map is approximately 110 meters wide and is the largest we ever built. However, a smaller map (constructed from a different data set) was used for navigation, due to changes in the operational area of the robot.

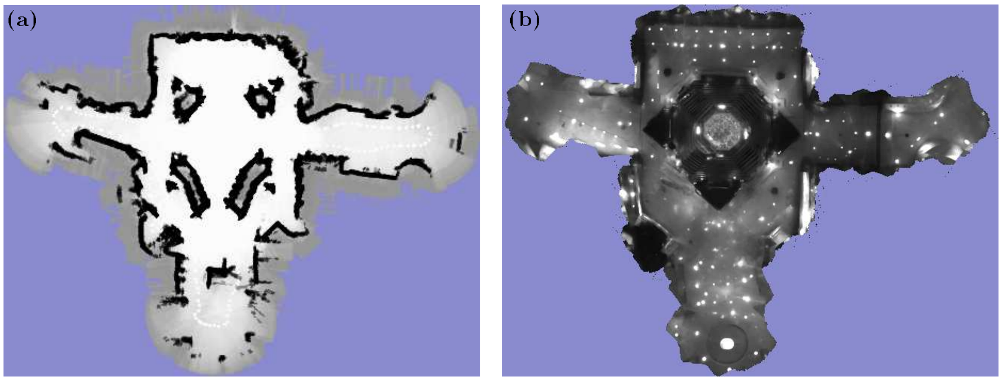


Figure 10: (a) Occupancy map of the center portion of the Smithsonian museum. (b) Mosaic of the museum's ceiling. The various bright spots correspond to lights. The center portion of the ceiling contains an opening—the lights there are approximately 15 meters higher.

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 calculating that the *odds* of the occupancy $m^{\langle xy \rangle}$, which is the quotient of (11) and its negation:

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

As is easily shown [142], this expression has the closed-form solution

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

Figure 8a 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 8b). Figure 9 shows a map built in the Smithsonian museum. These data were collected approximately six months before the exhibition, to develop and test our navigation routines. Figure 9a shows the raw data. Here the robot accrued an odometry error of 70 meters and approximately 180 degrees. Figure 9b shows the result of EM mapping. The final occupancy grid map is shown in Figure 9c. This map is over 110 meters wide. While it is geometrically somewhat inaccurate (see the upper boundary of the area on the left, which should be a straight line), it is sufficiently accurate for navigation purposes. However, this map does not cover the robot's entire operation range, which was defined after gathering these data. Thus, we collected a different data set just days before the exhibition began. The resulting map is shown in Figure 10a. This map is approximately 65 meters wide.

4.3 Ceiling Maps

Rhino, Minerva's predecessor, relied on lasers for localization. To deal with the large open spaces, Minerva additionally had a camera pointed at the ceiling, which we used approximately half of the time to augment the laser for localization. The ceiling map is a large-scale mosaic of a ceiling's texture. Such ceiling mosaics are more difficult to generate than occupancy maps. This is because the height of the ceiling is unknown, which makes it difficult to translate coordinates in the image plane into real-world coordinates.

A typical ceiling mosaic is shown in Figure 10b. Our approach uses the (previously learned) occupancy map to pre-adjust errors in the odometry. While those poses are not accurate to the precision that can be attained using the high-resolution vision sensors, they eliminate the difficult-to-solve global alignment problem. The likelihood $p(m|d)$ of the ceiling map is then maximized by searching in the space the following parameters: the pose s at which each image was taken, the height of ceiling segments, and two additional parameters per image specifying variations in lighting conditions. Our approach employs the well-known Levenberg-Marquardt algorithm [38] for optimization. As a result, the images are brought into local alignment, the ceiling height is estimated, and a global mosaic is constructed. Figure 10b shows the ceiling mosaic of the robot's operational range. A typical run for an environment of its size involves optimizing over about 3000 unknown variables, which requires approximately 30 minutes of processing time on a state-of-the-art computer. In follow-up research, we developed a probabilistic mosaicing algorithm which does not require pre-adjustment using occupancy grid maps [36].

5 Planning and Navigation

Minerva employs three modules concerned with planning and navigation: a low-level reactive collision avoidance module, a motion planner for moving from one exhibit to another, and a mission planner for scheduling tours and battery changes, which also processes user input.

5.1 Collision Avoidance

Minerva's collision avoidance module controls the momentary motion direction and velocity of the robot to avoid collisions with obstacles—people and exhibits alike. Many collision avoidance methods for mobile robots consider only the kinematics of a robot, without taking dynamics into account [13]. This is legitimate at speeds where robots can stop almost instantaneously. However, at velocities of up to 163 cm/sec, inertia and torque limits impose constraints on robot motion, which may not be neglected. To control the robot in tight run-time conditions, this module is reactive in that it considers only a small number of recent sensor readings.

Minerva's collision avoidance method, called μDWA is described in depth in [49]. It has been directly adopted from the Rhino software [19]; therefore we will only sketch it here. In essence, the input to μDWA is raw proximity sensor readings along with a desired target location, based on which μDWA sets the robot's velocity (translation and rotation). It does this by obeying a collection of constraints. Constraints come in two flavors: hard and soft. Hard constraints establish the basic safety of the robot (e.g., the robot must always be able to come to a full stop before impact) and they also express dynamic constrains (e.g., torque limits). Soft constraints are used to trade off the robot's desire to move towards the goal location, and its desire to move away from obstacles into open space. In combination, these constraints ensure safe and smooth local navigation.

A key issue in collision avoidance is invisible hazards. Recall that certain hazards, such as downward escalators, are invisible to Minerva's sensors; yet it is essential that the robot avoid them. These location of these hazardous locations known; we assume that a person marks the map accordingly. Thus, avoiding them requires the collision avoidance to translate map coordinates into local robot coordinates. At first glance, one might be tempted to perform this translation using a simple geometric transformation, which considers the robot's most

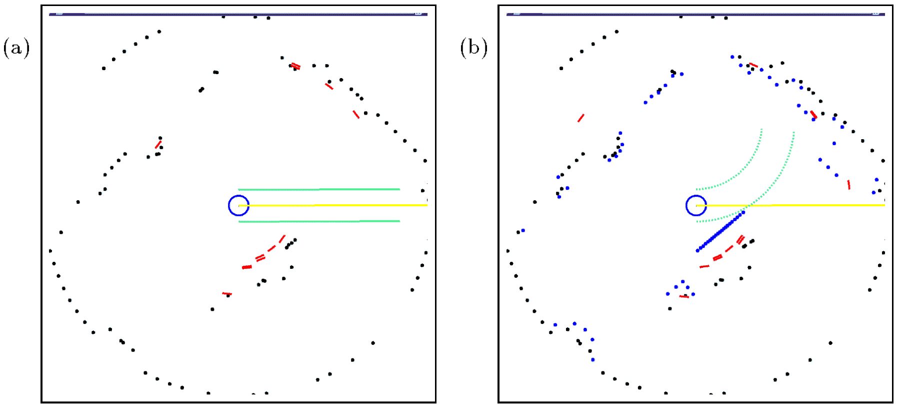


Figure 11: (a) Sensor measurements and motion direction generated by the collision avoidance module. The long straight line indicates the direction of the intermediate target point, and the shorter lines visualize the motion direction. (b) Same situation with virtual measurements. Now the robot avoids a large, invisible obstacle to its right.

likely pose $\hat{s_t} = \text{argmax}_{s_t} b_t(s_t)$ only. However, such an approach is brittle in the face of uncertainty. It would also fail to take advantage of the probabilistic nature of Minerva's localization approach.

Rather than relying on a single pose estimate for avoiding invisible hazards, Minerva employs a safer rule, which guarantees the robot's safety with high probability even if the robot is highly uncertain as to where it is. The basic idea is to avoid places that with probability > 0.01 are hazardous. This is achieved by adding "virtual" range measurements to the physical measurements, which with high probability (> 0.99) are shorter than an actual noise-free measurement of the distance to the nearest hazardous place. These virtual measurements are also fed into the collision avoidance module, along with the real measurements. As a result, the robot will be prevented from entering hazardous places with very high probability, even if it does not know exactly where it is. Obviously, such an approach is much safer in the face of uncertainty than simply relying on the maximum likelihood pose $\hat{s_t}$.

One of the advantages of the probabilistic framework is that the computation of such virtual measurements is mathematically straightforward. Let us consider the virtual measurement at angle α relative to the robot. Let $\sigma(\alpha, s_t, m)$ denote the distance to the nearest invisible hazard in the direction α, assuming that the robot's pose is s_t. Since σ assumes knowledge of the robot's pose, it is easily computed using ray tracing [90]. In practice, of course, one does not know the pose s_t; instead, all one is given is the posterior $b_t(s_t)$. The following term calculates the probability that our noise-free virtual sensor would return a measurement $o_{\alpha t}$ that is larger than a, under the belief $b_t(s_t)$:

$$p(o_{\alpha t} > a) \quad = \quad \int I_{\sigma(\alpha, s_t, m) > a}\; b_t(s_t)\; ds_t \tag{14}$$

Here I denotes the indicator function which is 1 iff its argument is true, and 0 otherwise. If we generate a virtual sensor measurement a for which $p(o_{\alpha t} > a) \geq 0.99$, we can be 99%-certain that the "true" distance to the nearest hazardous region in the direction α is larger than a. Put differently, our approach generates virtual measurements

$$\sup_a \ \{ p(o_{\alpha t} > a) \geq 0.99 \} \tag{15}$$

by maximizing a under the constraint that the true distance is underestimated with probability at least 99%. Virtual measurements are generated for all angles α, at an angular resolution of 2 degrees. Using these virtual measurements, the robot is safe with probability 99%, even though it may be uncertain as to where it is relative to the map.

Figure 11 illustrates the notion of virtual measurements. Shown in the left diagram is a set of sensor measurements (sonars, lasers) along with the robot. The direction of the next target location relative to the robot is illustrated by the long line. The collision avoidance module sends the robot in this direction, as indicated by the two shorter lines that indicate the area the robot intends to traverse. However, this control will lead to a collision with an invisible, large exhibit located to the right of the robot. Figure 11b shows the same situation, but with virtual measurements added. The robot now turns to the left, as indicated by the circular trajectory, thereby avoiding a collision with this specific exhibit. Our approach, which takes advantage of the explicit representation of uncertainty in the robot's pose estimate, was found to be essential for ensuring the robot's safety, both in the Rhino and the Minerva project [19].

5.2 Motion Planning

Minerva's motion planner computes globally consistent motion commands that guide the robot from one exhibit to the next. Uncertainty plays a major role in Minerva's motion planning algorithm. While Rhino operated in a narrow museum, always in safe proximity sufficiently many known objects to guarantee accurate localization, the Smithsonian museum contained a large, open, featureless region in its center. Here the danger of getting lost is significant, specifically at peak opening hours where this space is filled with hundreds of people. Thus, to minimize the danger of getting lost, Minerva's path planner seeks the proximity of known obstacles. Minerva's motion planner is called a *coastal planner* [114, 116]. In analogy to boats, which typically stay close to the coast to avoid getting lost (unless they are equipped with a global positioning system), Minerva's motion planner generates controls that navigate the robot close to known obstacles, to minimize the danger of getting lost.

Possibly the most general framework for probabilistic planning is known as *partially observable Markov decision processes*, or in short *POMDP* [96, 133, 135]. Recently, POMDPs have become popular in AI [69, 87]. POMDPs address the problem of choosing actions so as to minimize a scalar *cost function*, denoted $C(s)$. In robot motion planning, we use $C(s) = 0$ 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*:

$$J \;=\; \sum_{\tau=t+1}^{t+T} E[C(s_\tau)]. \tag{16}$$

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.

The basic idea of POMDPs is to construct a value function in belief space, using a generalized version of value iteration [9, 63]. A value function, denoted by V, 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

$$V(b) \quad = \quad \int \sum_{\tau=t+1}^{t+T} E[C(s_\tau)|s_t]\ b(s_t)\ ds. \tag{17}$$

Following [9, 137], the value function can be computed by recursively adjusting the value of individual belief states b according to

$$V(b) \quad \longleftarrow \quad \min_a \int [V(b') + C(b')]\ p(b'|a,b,m)\ db', \tag{18}$$

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:

$$p(b'|a,b,m) \quad = \quad \int p(b'|o',a,b,m)\ p(o'|a,b,m)\ do', \tag{19}$$

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

$$p(o'|a,b,m) \quad = \quad \int\int p(o'|s',m)\ p(s'|a,s,m)\ b(s)\ ds'\ ds. \tag{20}$$

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

$$\pi(b) \quad = \quad \operatorname*{argmin}_a \int V(b')\ p(b'|a,b,m)\ db'. \tag{21}$$

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 same densities used in mapping and localization. However, there are two fundamental problems. First, in continuous domains the belief space, which is the space of all distributions, 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* [69], 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 a POMDP algorithm 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:

$$\bar{b} \quad = \quad \langle\ \operatorname*{argmax}_s b(s);\ H[b]\ \rangle. \tag{22}$$

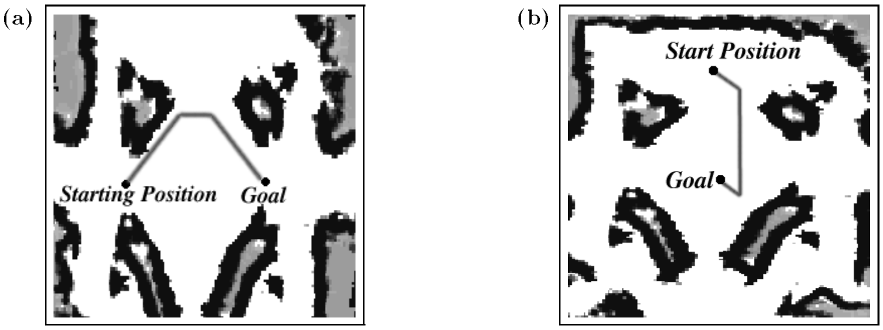


Figure 12: Coastal plans: the robot actively seeks the proximity of obstacles to improve its localization. The large open area in the center of the Smithsonian museum is approximately 20 meters wide and usually crowded with people.

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 [114, 116].

Figure 12 show two example trajectories calculated by the coastal navigation algorithm for the center region of the museum. By considering uncertainty, the coastal planner generates paths that actively seek 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 described elsewhere [116] 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.

5.3 Mission Planning

Minerva's high-level controller performs two important tasks:

1. During everyday normal operation, it schedules tours and monitors their execution. The target duration for tours was six minutes, which was determined to be the duration the average visitor would enjoy following the robot. Unfortunately, the rate of progress depends critically on the number and the behavior of the surrounding people. This makes it necessary to compose tours on-the-fly.

2. The high-level controller also has to *monitor* the execution of tours and change the course of actions when an exception occurs. Examples include the battery voltage, which, if below a critical level, forces the robot to terminate its tour and return to the charger. An exception is also triggered when the confidence of Minerva's localization routines drop below a critical level (luckily an extremely rare event), in which case the tour must temporarily be suspended to invoke a strategy for re-localization [7].

Minerva's plan-based controller is a *structured reactive controller* (SRC) [5] built on top of RPL [93]. It is a collection of concurrent, percept-driven control routines that specifies routine activities and can adapt itself to non-standard situations. Minerva executes three kinds of high-level control processes: scheduled tour plans that work well in standard situations, monitoring processes that detect non-standard situations, and plan adaptors that are responsible for managing the tour plans during their execution. Thus, Minerva carries out museum tours with the constraint that, when circumstances change, a runtime

	average	min	max
static	398 ± 204 sec	121 sec	925 sec
with learning	384 ± 38 sec	321 sec	462 sec

Table 2: This table summarizes the time spent on individual tours. In the first row, tours were pre-composed by static sequences of exhibits; in the second row, tours were composed on-the-fly, based on a learned model of travel time, successfully reducing the variance by a factor of 5.

plan adaptation process is triggered. For example, such a situation might occur when the robot suffers an unexpected delay while traveling from one exhibit to another, or when tour requests are added or modified on-line. During the 14 day-long deployment Minerva's plan-based controller performed roughly 3,200 execution time plan revisions, including the insertion of plans for new user requests, the removal of plans for accomplished requests, and tour rescheduling. The controller communicated with the rest of the software using HLI, a component of GOLEX [59]. More recently, we have extended this framework to include probabilistic representations [6, 8]; however, these extensions were not used in the Minerva project.

To meet the desired length for individual tours, Minerva' mission planner composes tours on-the-fly. To do so, it *learns* the time required for moving between pairs of exhibits, based on data recorded in the past (using the empirical mean as estimator). After an exhibit is explained, the interface chooses the next exhibit based on the remaining time. If the remaining time is below a threshold, the tour is terminated and Minerva instead returns to the center portion of the museum. Otherwise, it selects exhibits whose sequence best fit the desired time constraint.

Table 2 illustrates the effect of dynamic tour decomposition on the duration of tours. Minerva's environment contained 23 designated exhibits, and there were 77 sensible pairwise combinations between them (certain combinations were invalid since they did not fit together topic-wise). In the first days of the exhibition, all tours were static. The first row in Table 2 illustrates that the timing of those tours varies significantly (by an average of 204 seconds). The average travel time, shown in Table 3, was estimated using 1,016 examples, collected during the first days of the project. The second row in Table 2 shows the results when tours were composed dynamically. Here the variance of the duration of a tour is only 38 seconds, thanks to the dynamic tour composition using learning. Minerva's high-level interface also made the robot return to its charger periodically, so that we could hot-swap its batteries.

6 Human Robot Interaction

Interaction with people is Minerva's primary purpose. The type of interaction faced by a tour-guide robot is *spontaneous* and *short-term*: Visitors of the Smithsonian museum typically had no prior exposure to robotics technology, and they could not be instructed beforehand as to how to interact with the robot. The robot often interacts with crowds of people as well as individual visitors. In the museum, most people spent less then 10 minutes (though some spent hours and some even came on multiple days). This type of interaction is characteristic for robots that operate in public places, such as receptionists, information kiosks, and merchandising robots[44, 74, 81, 104, 122]. It differs significantly from the majority of interactive modes studied in the field, which typically assume long-term

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1		26	68	14	28															
2					23	38	13													
3				81	66	51		66						60						
4																76	22			
5											62								49	
6		41					44													
7			44	1	55			42						51						
8									44	63										
9																				
10																				
11				34								16	69							
12				61	53	69		72	32				87	55						
13												28								
14	33		39																	
15									60											
16																		46		68
17								59						13		57				
18					46	42		31	36					31						12
19				1		25		58			69		12							
20			57	62														37		
21				55	24	20		15						74						
22		208		66	46	38		38	23					56	39					
23				113	76	59		24	46					59						

Table 3: Time (in sec) it takes to move from one exhibit to another, estimated from 1,016 examples collected in the museum. These times, plus the (known) time used for explaining an exhibit, form the basis for the decision-theoretic planner.

interaction with a single subject.

To maximize Minerva's effectiveness, the robot has human-like features such as a motorized face, a neck, and a simple finite state machine emulating "emotions." It uses reinforcement learning to shape her interactive skills on-line.

6.1 The Face

Figure 13 shows Minerva's face. To engage museum visitors, we sought to present as recognizable and intuitive an interface as possible [14, 123]. Obviously, the face is only a caricature, containing only schematic features related to the expression of simple emotions. It contains those elements necessary for the degree of expression appropriate for a tour-guide robot. A fixed mask would be incapable of visually representing mood, and a highly accurate simulation of a human face would contain numerous distracting details beyond our control. A physical face was deemed more appropriate than a simulated one displayed on a computer screen, because people can view it from arbitrary angles (even from the back), letting museum visitors see it without standing directly in front of the robot. As Figure 13 documents, an iconographic face consisting of two eyes with eyebrows and a mouth is almost universally recognizable and can portray the range of simple emotions useful for tour-guide interaction.

6.2 Emotional States

When giving tours, Minerva uses its face, its head direction, and its voice to communicate with people, so as to maximize its progress and please the audience. A stochastic finite state machine shown in Figure 14 is employed to model simple *emotional states* (moods), which allow the robot to communicate its intent to visitors in a social context familiar to people from human-human interaction [14, 103, 104]. Moods range from happy to angry,

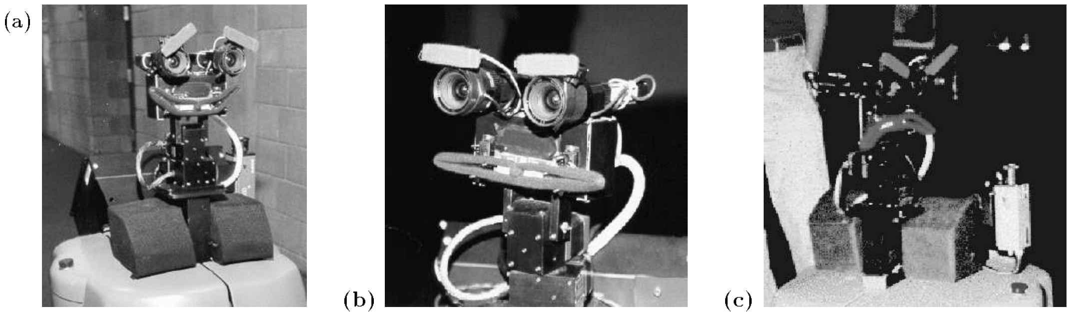


Figure 13: Minerva's face with (a) happy, (b) neutral, and (c) angry facial expressions.

depending on the persistence of the people who block its path. When happy, Minerva smiles and politely asks for people to step out of the way; when angry, its face frowns and the robot's voice sounds angry. Most museum visitors had no difficulty understanding the robot's intention and emotional state. In fact, the ability to exhibit such extremely caricatured pseudo-emotions proved to be one of the most-liked aspects of the robot.

6.3 Learning to Attract People

How can a robot attract attention? Since there is no obvious answer, we applied an on-line learning algorithm. More specifically, Minerva uses a memory-based reinforcement learning approach [137] (with no delayed reward). Minerva's behavior is conditioned on the current density of people. Reinforcement is received in proportion of the proximity of people as determined by Minerva's people finding module; coming too close, however, leads to a distinct penalty for violating Minerva's space. Possible actions include different strategies for head motion (e.g., looking at the nearest person), different facial expressions (e.g., happy, sad, angry), and different speech acts (e.g., "Come over," "Do you like robots?"). Learning occurs during one-minute-long, dedicated *mingling phases*, which take place between tours. Here the robot chooses with high probability the best-known action, so that it attracts as many people as possible. However, with small probability the robot chooses a random action to explore new forms of interaction. This approach is similar to methods for solving the exploration-exploitation dilemma studied in the k-arm bandit literature [139].

During the two weeks in the Smithsonian museum, Minerva performed 201 attraction interaction experiments, each of which lasted approximately one minute. Over time, Minerva developed a positive attitude (saying friendly things, looking at people, smiling). As shown in Figure 15, acts best associated with a positive attitude attracted the most people. For example, when grouping speech acts and facial expressions into two categories, friendly and unfriendly, we found that the former type of interaction performed significantly better than the first (with 95% confidence). However, peoples response was highly stochastic and the amount of data that we were able to collect during the exhibition is insufficient to yield statistical significance in most cases. Hence, we are unable to comment on the effectiveness of individual actions.

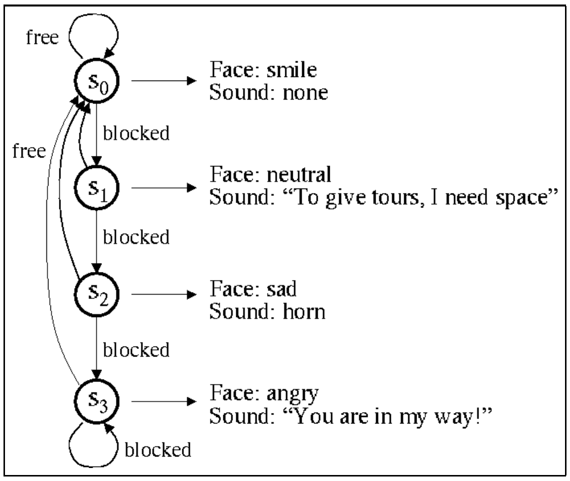


Figure 14: State diagram of Minerva's emotions during travel. "Free" and "blocked" indicate whether a person stands in the robot's path.

6.4 Web Interface

One of the goals of the project was to enable remote users to establish a virtual telepresence in the museum, using the Internet. While in the Smithsonian museum Minerva was connected to the Web at `http://www.cs.cmu.edu/~minerva`, where Web users all over the world could control Minerva and look through its eyes. In addition, a stationary zoom camera mounted on a pan/tilt unit enabled Web users to watch Minerva and nearby visitors from a distance.

While the museum was open to visitors, Minerva was controlled predominately by the visitors of the museum, who could select tours using a touch-sensitive screen mounted at Minerva's back. Every third tour, however, was selected by Web users via a voting scheme: Votes for individual tours were counted, and the most popular tour was chosen. At all times, the web page displayed current camera images recorded by Minerva and by the off-board camera, and a museum map with the robot's position. To facilitate updating the position of Minerva several times a second, Web users downloaded a robot simulator written in Java, which used TCP communication and server-push technology to communicate the position of the robot in near real time [124].

During several special scheduled Internet events, all of which took place when the museum was closed to visitors, Web users were given exclusive control of the robot. Using the interface shown in Figure 16a, they could schedule target points, which the robot approached in the order received. The number of pending target points was limited to five. All rows in Table 4 marked "Web only" correspond to times where Web users assumed exclusive control over the robot. In one case, Minerva moved at an average velocity of 73.8 cm/sec. Its maximum velocity was 163 cm/sec, which was attained frequently. Such high velocities, however, were only attained when the museum was closed. When visitors were around, the speed was reduced to less than 70 cm/sec (walking speed) to avoid people perceiving the robot as a threat.

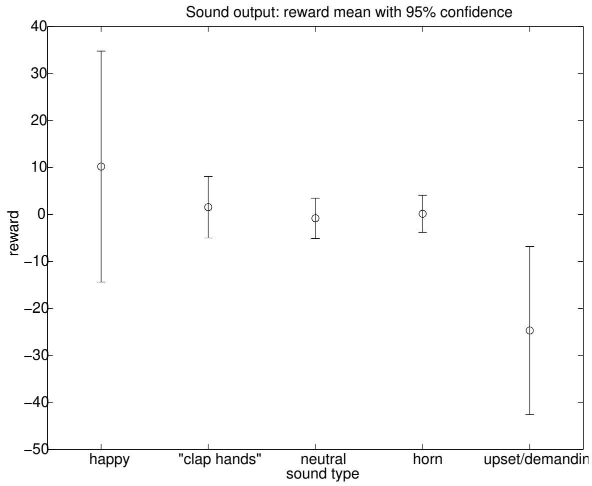


Figure 15: Statistics of people's response to different styles of interaction (from friendly on the left to upset/demanding on the right). The data were input to a reinforcement learning algorithm, which learned interaction patterns on-line.

7 Statistics

Table 4 gives the overall statistics of Minerva's 13 days-long performance. As can be seen, the robot traveled a total of 44 km, at top speed of 163 cm/sec and an average speed of 38.8 cm/sec. Minerva's speed was limited to 70 cm/sec during opening hours, but limited only by hardware limitations when the museum was closed and the robot was controlled through the Internet. Figure 16b shows the robot's path between two battery charges; a battery charge lasted approximately two hours.

Since the Rhino robot was developed by the same research team and employed many of the same basic navigation modules, a comparison between both robots seems in order. Navigation in the Smithsonian museum posed completely new challenges that were not present in the Deutsches Museum Bonn. Minerva's environment was much larger, with a particular challenge arising from the large featureless, open area in the center portion of the museum. Minerva also had to cope with many more people than Rhino.

To accommodate these difficulties, Minerva's navigation system was more sophisticated. In particular, Rhino did not use camera images for localization, and its motion planner did not consider information gain when planning paths. In addition, Rhino was supplied with a manually derived map; it lacked the ability to learn maps from scratch. We believe that these extensions were essential for Minerva's success. Rhino also lacked the ability to compose tours on-the-fly, and it was also unable to detect exceptions such as battery drain (which caused problems) [19].

In the Rhino project, we carefully counted the number of collisions with static obstacles and other failures (there were a total of six in six days). In Minerva's case, we counted two collisions in thirteen days, due to small localization errors. We also recall two occasions at which Minerva lost its position entirely, both times involving huge crowds of people that persistently blocked virtually all of the robots sensors for extended periods of time. Additionally, a misadjusted low-level motion controller in the robot's base, which was inaccessible to us, made the robot's motion a bit jerkier than that of Rhino. However, this did not affect

(a) (b)

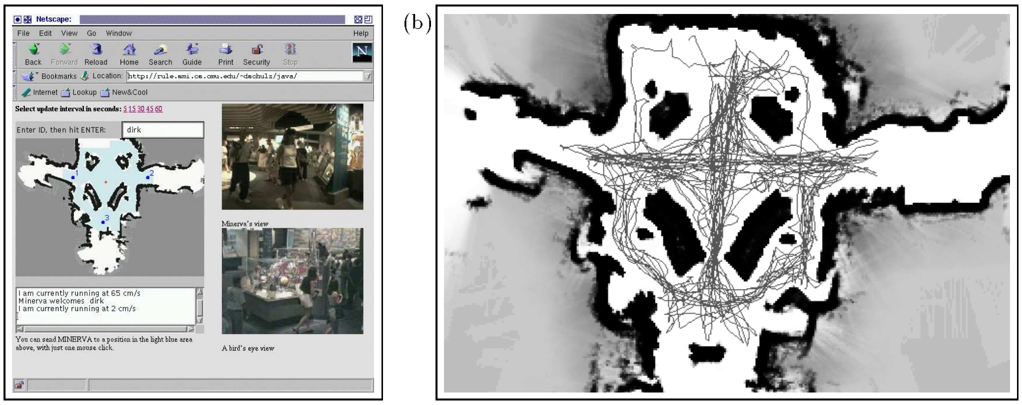


Figure 16: (a) Web control interface. Users can authenticate themselves in on the left side of the window, and subsequently specify target locations by clicking in the map. The map shows current robot position, pending target locations, and a dialogue box displays the current speed of the robot. On the right. users can watch images recorded using the robot's camera (top image) and by a stationary camera with zoom mounted on a pan/tilt unit (bottom image). (b) Multi-hour path of the robot in the museum.

Minerva's overall performance.

A key difference between both robots relates to their interactive capabilities. As mentioned above, Rhino's interaction was more rudimentary. It lacked a face, did not exhibit "emotional states," and it did not actively attract or engage people. As a result, Minerva was much more effective in attracting people and making progress. When compared to the Rhino project, we consistently observed that people cleared the robot's path much faster. We found that both robots maintained about the same average speed (Minerva: 38.8 cm/sec, Rhino: 33.8 cm/sec), despite the fact that Minerva's environment was more crowded. These numbers illustrate the effectiveness of Minerva's interactive approach to making progress.

In comparison with Rhino, people also appeared more satisfied and amused. According to a poll involving 63 people (36 male, 27 female), 93.7% liked Minerva, while the remaining 6.3% were undecided. Not a single visitor answered this question with no. When asked whether people were satisfied with the robot, 77.8% answered yes, 15.9% were undecided, and only 6.3% responded with no. 39.7% of the visitors would be willing to pay $1,000 or more, if they could purchase a robot like Minerva (with the same level of capability) for their private home. When asked what level of animal (from a list of five options) Minerva's intelligence was most comparable to, we received the following answers: human: 36.9%; monkey: 25.4%; dog: 29.5%; fish: 5.7%; amoeba:2.5%. However, due to the small sample size, it is difficult to draw strong conclusions from these responses. Unfortunately, we did not ask people the same questions at the Rhino exhibition. A similar evaluation of the effectiveness of robot emotions for robots operating in public places can be found in [103, 104].

Minerva also possessed an improved Web interface, which enabled Web users to specify arbitrary target locations instead of choosing locations from a small pool of pre-specified locations. Rhino's Web interface prescribed a small set of 13 possible target locations, which corresponded to designated target exhibits. When under exclusive Web control, Minerva was more than twice as fast as Rhino (see Table 4). In everyday operation, however, the maximum speed of both robots was limited to the same speed.

date	uptime	travel time	distance	avg. speed	tours	exhibits	mode
Aug 24	7:16:08	2:34:36	2,881.13 m	31.3 cm/sec	52	174	
Aug 25	7:41:52	2:17:05	2,725.90 m	33.1 cm/sec	55	169	
Aug 26	6:57:35	2:39:24	2,642.23 m	27.6 cm/sec	28	102	
Aug 27	5:40:58	1:33:00	1,147.12 m	31.7 cm/sec	53	203	
	1:56:21	0:50:55	1,755.98 m	57.5 cm/sec	28	104	Web only
Aug 28	6:48:59	2:08:14	2,416.15 m	31.4 cm/sec	54	192	
Aug 29	5:40:23	1:50:22	2,436.92 m	36.7 cm/sec	59	219	
Aug 30	6:42:36	2:17:58	3,305.44 m	39.9 cm/sec	66	231	
Aug 31	7:25:57	2:09:02	3,372.91 m	43.6 cm/sec	77	258	
Sept 1	7:11:54	2:22:40	3,707.19 m	43.3 cm/sec	61	230	
Sept 2	4:28:07	1:27:33	1,954.19 m	37.2 cm/sec	37	137	
Sept 3	9:56:53	3:25:08	5,332.76 m	43.3 cm/sec	54	203	
Sept 4	1:13:15	0:52:34	2,143.86 m	68.0 cm/sec		103	Web only
	6:49:35	2:04:49	2,611.71 m	34.9 cm/sec	48	168	
	2:17:04	1:17:00	3,411.41 m	73.8 cm/sec		175	Web only
Sept 5	6:15:46	1:42:34	2,173.90 m	35.3 cm/sec	49	156	
total	94:23:20	31:32:54	44,018.8m	38.8cm/sec	620	2,668	

Table 4: Summary statistics of Minerva's operation. The rows labeled "Web only" indicate times when the museum was closed to the public, and Minerva was under exclusive Web control. At all other times, Web users and museum visitors shared the control of the robot. Minerva's top speed was 163 cm/sec.

8 Related Work

Probably the first tour-guide robot was Ian Horswill's robot *Polly* [61, 62], a small mobile robot that guided visitors through a research lab. To our knowledge, Rhino was the first museum tour-guide robot [19]; it operated in the Fall of 1997. Rhino inspired Sage/Chips (the name was changed while the robot was in operation) [100, 102], which had its debut in 1998 in the Carnegie Museum of Natural History in Pittsburgh, PA (see map in Figure 8). Sage, or Chips, has now operated with interruptions for approximately two years. However, its environment has been modified significantly to aid the navigation, and it also lacks a Web interface. Others have developed proto-type robots that interact with people at fairs, trade shows, and retail stores (see e.g., [44, 74, 76, 81, 103, 104, 122]).

8.1 Probabilistic Methods

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 [21, 82, 125]. 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" [15], where controllers are composed of small finite state automata that map sensor readings directly 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 [15, 28]. Observing that *"the world is its own best model"* [16], behavior-based approaches usually rely on immediate sensor feedback for determining a robot's action. Obvious limits in perception (e.g., robots cannot see through walls) pose clear boundaries on the type of 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 [140]. 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 leads to qualitatively new solutions for 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 purely reactive mobile robot in a crowded place full of invisible hazards! Surely, adding more sensors can remedy the problem. However, such an approach is expensive at best, but more often it will be plainly infeasible due to lack of appropriate sensors. Minerva's predecessor 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 [19] for a discussion). Thus, it seems unlikely that a reactive approach could have performed anywhere nearly as reliably and robustly in this task domain.

8.2 Localization

Mobile robot localization has frequently been recognized as a key problem in robotics with significant practical importance [29]. A recent book on this topic [12] provides an excellent overview of recent work in mobile robot localization. Localization plays a key role in various successful mobile robot architectures [30, 44, 51, 60, 84, 85, 99, 105, 110, 131, 148] and various chapters in [76]. While some localization approaches, such as those described in [62, 78, 132] localize the robot relative to some landmarks in a topological map, Minerva's approach localizes the robot in a metric space, just like those methods proposed in [10, 141, 144].

The vast majority of approaches is incapable of localizing a robot globally or to recover from robot kidnapping. Instead, they are designed to track the robot's position by compensating small odometric errors [56, 57, 89, 120, 134]. Recently, several researchers proposed the *Markov localization* used by Minerva, which enables robots to localize themselves under global uncertainty [20, 35, 48, 68, 101, 132]. Minerva's and Rhino's localization algorithm goes beyond previous approaches in that it can cope with invisible hazards and highly dynamic environments.

8.3 Mapping

Several major approaches to concurrent mapping and localization were already reviewed in Section 4.1. Occupancy grids have originally been proposed by Elfes and Moravec [42, 43, 97].

Since, they have been employed in numerous robotic systems [13, 18, 58, 121, 149, 150]. Occupancy grid maps are an example of the *metric* paradigm [142]. There exists a second, major paradigm to mapping, called *topological*. Topological methods represent maps as graphs. Usually, nodes in these graphs correspond to distinct places in the environment, and arcs to actions for moving from one place to another [24, 25, 26, 27, 79, 80, 91, 98, 146, 152]. Probably the most related mapping algorithm to the one described in this paper has been developed by Shatkay and Kaelbling [126, 127]. Their algorithm generates topological maps using a version of the EM algorithm known as the Baum-Welsh algorithm [107]. To do so, it requires that appropriate landmarks can be found in the sensor data. Minerva's mapping algorithm, in contrast, utilizes all sensor data and generates fine-grained, metric maps.

8.4 Motion Planning

Robot motion planning has been subject to intense research, as documented by a large body of literature on this topic (see e.g., [21, 82, 109, 125]). The majority of work addresses more complicated problems than the one addressed in this article, such as motion planning in higher-dimensional and continuous spaces. Latombe [82] pioneered the use of randomized algorithms for motion planning in high-dimensional spaces. However, randomization is only used for search, not for representing uncertainty. The issue of uncertainty in robot motion has largely been ignored in the field of motion planning.

Within AI, the issue of uncertainty in planning has been studied extensively [118]. As discussed above, one of the most popular frameworks is known as *partially observable Markov decision processes (POMDPs)* [66, 70, 86]. Exact POMDPs are inapplicable to the robot motion planning problem due to their enormous computational complexity. Our approach is a crude approximation to POMDPs which considers uncertainty, but is unable to distinguish different types of uncertainty with the same entropy.

8.5 Human Robot Interaction

The issue of human robot interaction has recently received considerable attention. One of the earliest examples of a user interface is a natural language interface for teaching mobile robots names of places in an indoor environment [146]. Due to the lack of a speech recognition system, this interface requires the user to operate a keyboard. More recently, several researchers [3, 115] developed interfaces that integrates a speech recognition system into a phrase-based natural language interface. Others have proposed vision-based interfaces that allow people to instruct mobile robots via arm gestures [71, 77, 147]. Unfortunately, all these interfaces are somewhat inappropriate for Minerva, since they all assume knowledge on the side of the robot operator, and they all assume that the robot interacts with a single person at a time. As noticed in Section 6.2 of this article, several researchers have developed interfaces similar to Minerva's, which allow the robot to communicate intent and moods to people through facial gestures [14, 103, 104, 111]. More broadly, Minerva can be seen as a believable robotic agent, and hence is related to literature on believability in software agent research [4] and in robotics [31].

8.6 Web Interfaces

Web interfaces have received serious attention in robotics throughout the last years, since they allow people to tele-operate a robot at a distant site. Three early systems, whose interfaces were designed along these lines, are the Mercury Project [52] installed in 1994, Australia's Tele-robot on the Web [138], which came on-line nearly at the same time, and the Tele-Garden [53], which replaced the Mercury robot in 1995. While the Mercury robot and the Tele-Garden enabled Web users to perform different types of digging tasks, excavation of artifacts and watering and seeding flowers, the Tele-robot on the Web gave Web users the opportunity to build complex structures from toy blocks. The PumaPaint Project [136] enables people to draw a painting by controlling a PUMA 760 robot arm.

Minerva's Web interfaces borrow some ideas from Xavier [130, 131], one of the first interactive mobile robots controllable via the Web. Xavier can be advised by Web users to move to an office and to tell a knock-knock joke. Xavier collects requests off-line and processes them during special working hours. It informs the Web user afterwards about task completion via email. The Web interface relies on client-pull and server-push techniques to provide images taken by the robot as well as a map indicating the robot's current position in regular intervals. In contrast to Xavier, however, our robots provide status information with smooth visualizations. Our interfaces immediately react to requests and inform users instantly about the current schedule of the robot. KephOnTheWeb [119, 95], another mobile robot on the Web, allows virtual visitors to move a Khepera robot and to control several cameras, using a set of clickable maps. There is also a huge list of Web cameras, which deliver image streams to the Web. Some of these cameras can even be controlled by virtual visitors, such as the one described in [54], which is installed on a robot arm in a museum. Other Web interfaces can be found in a recent magazine issue [128].

9 Discussion

This article described the software architecture of a mobile tour-guide robot, which successfully operated for a two-week time period at the Smithsonian's National Museum of American History. During more than 94 hours of operation (31.5 hours of motion), Minerva gave 620 tours and visited 2,668 exhibits. The robot interacted with thousands of people, and traversed more than 44 km. Its average speed was 38.8 cm/sec, and its maximum speed was 163 cm/sec. The map learning techniques enabled us to develop the robot in 3 weeks, from the arrival of the base platform to the opening of the exhibition. A Web interface gave people direct control of the robot when the museum was closed to the public.

So what did we learn? Minerva's software was pervasively probabilistic. As noted in the introduction, the probabilistic paradigm pays tribute to the inherent uncertainty in robot perception, relying on explicit representations of uncertainty when determining what to do.

Our 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. Our results also show favorably performance in planning and reactive control using probabilistic algorithms. Probabilistic representations were essential for reliable localization, and the robot's ability to safely avoid downward escalators and other "invisible" hazards in the densely populated museum.

We conjecture that the probabilistic paradigm is a general, powerful approach to robotics, highly applicable to a whole range of robot applications involving real-world sensing. Sensors are inherently limited. Environments are dynamic. Models are inaccurate. Therefore, uncertainty plays a predominant role in robotics. We hope that the results described in this paper shed light onto the appropriateness of the probabilistic approach to robotics, illustrating how a range of challenging problems can be solved in a novel and mathematically consistent way.

Acknowledgments

We are deeply indebted to the Deutsches Museum Bonn and the Lemelson Center of the Smithsonian National Museum of American History for their support and enthusiasm for these projects. We also thank the Real World Interface Division of IS Robotics for lending us the Minerva hardware free of charge.

This research is sponsored in part by DARPA via AFMSC (contract number F04701-97-C-0022), TACOM (contract number DAAE07-98-C-L032), and Rome Labs (contract number F30602-98-2-0137). Additional funding was received from the EC (contract number ERBFMRX-CT96-0049) under the TMR programme, from DaimlerChrysler Research, and from Andy Rubin, all of which is gratefully acknowledged. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of DARPA, AFMSC, TACOM, Rome Labs, the United States Government, the EC, DaimlerChrysler Research, or A. Rubin.

References

[1] R. Arkin. *Behavior-Based Robotics*. MIT Press, Boston, MA, 1998.

[2] K.O. Arras and S.J. Vestli. Hybrid, high-precision localization for the mail distributing mobile robot system MOPS. In *Proceedings of the IEEE International Conference on Robotics & Automation (ICRA)*, 1998.

[3] H. Asoh, S. Hayamizu, H. Isao, Y. Motomura, S. Akaho, and T. Matsui. Socially embedded learning of office-conversant robot jijo-2. In *Proceedings of IJCAI-97*. IJCAI, Inc., 1997.

[4] J. Bates. The role of emotion in believable agents. Technical Report CMU-CS-94-136, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, April 1994.

[5] M. Beetz. Structured reactive controllers—a computational model of everyday activity. In *Proceedings of the Third International Conference on Autonomous Agents*, 1999.

[6] M. Beetz, M. Bennewitz, and H. Grosskreutz. Probabilistic, prediction-based schedule debugging for autonomous robot office couriers. In *Proceedings of the 23rd German Conference on Artificial Intelligence (KI 99), Bonn, Germany*. Springer Verlag, 1999.

[7] M. Beetz, W. Burgard, D. Fox, and A.B. Cremers. Integrating active localization into high-level robot control systems. *Journal of Robotics and Autonomous Systems*, 1999. forthcoming.

[8] M. Beetz and H. Grosskreutz. Probabilistic hybrid action models for predicting concurrent percept-driven robot behavior. In *Proceedings of the Fifth International Conference on AI Planning Systems*, Breckenridge, CO, 2000. AAAI Press.

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

[10] M. Betke and L. Gurvits. Mobile robot localization using landmarks. Technical Report SCR-94-TR-474, Siemens Corporate Research, Princeton, December 1993. will also appear in the IEEE Transactions on Robotics and Automation.

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

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

[13] J. Borenstein and Y. Koren. The vector field histogram – fast obstacle avoidance for mobile robots. *IEEE Journal of Robotics and Automation*, 7(3):278–288, June 1991.

[14] C. Breazeal (Ferrell). A motivational system for regulating human-robot interaction. In *Proceedings of AAAI'98*, pages 54–61, Madison, WI, 1998.

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

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

[17] R.A. Brooks. Intelligence without reason. In *Proceedings of IJCAI-91*, pages 569–595. IJCAI, Inc., July 1991.

[18] J. Buhmann, W. Burgard, A.B. Cremers, D. Fox, T. Hofmann, F. Schneider, J. Strikos, and S. Thrun. The mobile robot Rhino. *AI Magazine*, 16(1), 1995.

[19] 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.

[20] 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.

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

[22] 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.

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

[24] H. Choset. *Sensor Based Motion Planning: The Hierarchical Generalized Voronoi Graph*. PhD thesis, California Institute of Technology, 1996.

[25] H. Choset, I. Konuksven, and J.W. Burdick. Sensor Based Planning for a Planar Rod Robot. In *Proc. IEEE/SICE/RSJ Int. Conf. on Multisensor Fusion on Multisensor Fusion and Integration for Intelligent Systems*, Washington, DC, 1996.

[26] H. Choset, I. Konuksven, and A. Rizzi. Sensor Based Planning: A Control Law for Generating the Generalized Voronoi Graph. In *Proc. IEEE Int. Advanced Robotics*, Washington, DC, 1996.

[27] E. Chown, S. Kaplan, and D. Kortenkamp. Prototypes, location, and associative networks (plan): Towards a unified theory of cognitive mapping. *Cognitive Science*, 19:1–51, 1995.

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

[29] 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.

[30] I.J. Cox and G.T. Wilfong, editors. *Autonomous Robot Vehicles*. Springer Verlag, 1990.

[31] K. Dautenhahn. The role of interactive conceptions of intelligence and life in cognitive technology. In *Proceedings of CT-97*, pages 33–43, 1997.

[32] 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.

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

[34] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, Fort Collins, CO, 1999. IEEE.

[35] 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.

[36] F. Dellaert, C. Thorpe, and S. Thrun. Mosaicing a large number of widely dispersed, noisy, and distorted images: A bayesian approach. Technical Report CMU-RI-TR-99-34, Carnegie Mellon University, Pittsburgh, PA, 1999.

[37] 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.

[38] J.E. Dennis and R.B. Schnabel. *Numerical methods for unconstrained optimization and nonlinear equations.* Prentice-Hall, Englewood Cliffs, NJ, 1983.

[39] J. Denzler, B. Heigl, and H. Niemann. Combining computer graphics and computer vision for probabilistic self-localization. Internal Report, 1999.

[40] 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.

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

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

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

[44] H. Endres, W. Feiten, and G. Lawitzky. Field test of a navigation system: Autonomous cleaning in supermarkets. In *Proc. of the 1998 IEEE International Conference on Robotics & Automation (ICRA 98)*, 1998.

[45] G. Engelberger. Services. In Shimon Y. Nof, editor, *Handbook of Industrial Robotics*, chapter 64., pages 1201–1212. John Wiley and Sons, 2nd edition, 1999.

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

[47] C. Fedor. *TCX. An interprocess communication system for building robotic architectures. Programmer's guide to version 10.xx.* Carnegie Mellon University, Pittsburgh, PA 15213, December 1993.

[48] 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.

[49] 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.

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

[51] T. Fukuda, S. Ito, N. Oota, F. Arai, Y. Abe, K. Tanake, and Y. Tanaka. Navigation system based on ceiling landmark recognition for autonomous mobile robot. In *Proc. Int'l Conference on Industrial Electronics Control and Instrumentation (IECON'93)*, volume 1, pages 1466 – 1471, 1993.

[52] K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, and J. Wiegley. Desktop teleoperation via the World Wide Web. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1995.

[53] K. Goldberg, J. Santarromana, G. Bekey, S. Gentner, R. Morris, J. Wiegley, and E. Berger. The telegarden. In *Proc. of ACM SIGGRAPH*, 1995.

[54] S. Goldberg, G.A. Bekey, and Y. Akatsuka. DIGIMUSE: An interactive telerobotic system for remote viewing of three-dimensional art objects. In *Proceedings of the IEEE IROS'98 Workshop on Robots on the Web*, 1998.

[55] J.-S. Gutmann, W. Burgard, D. Fox, and K. Konolige. An experimental comparison of localization methods. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1998.

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

[57] J.-S. Gutmann and C. Schlegel. Amos: Comparison of scan matching approaches for self-localization in indoor environments. In *Proceedings of the 1st Euromicro Workshop on Advanced Mobile Robots*. IEEE Computer Society Press, 1996.

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

[59] D. Haehnel, W. Burgard, and G. Lakemeyer. GOLEX: Bridging the gap between logic (GOLOG) and a real robot. In *Proceedings of the 22st German Conference on Artificial Intelligence (KI 98)*, Bremen, Germany, 1998.

[60] R. Hinkel and T. Knieriemen. Environment perception with a laser radar in a fast moving robot. In *Proceedings of Symposium on Robot Control*, pages 68.1–68.7, Karlsruhe, Germany, October 1988.

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

[62] I. Horswill. Specialization of perceptual processes. Technical Report AI TR-1511, MIT, AI Lab, Cambridge, MA, September 1994.

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

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

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

[66] T. Jaakkola, S. P. Singh, and M. I. Jordan. Reinforcement learning algorithm for partially observable decision problems. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems 7*, Cambridge, MA, 1995. MIT Press.

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

[68] 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.

[69] 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.

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

[71] R.E. Kahn, M.J. Swain, P.N. Prokopowicz, and R.J. Firby. Gesture recognition using the perseus architecture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 734–741, San Francisco, CA, 1996.

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

[73] K. Kanazawa, D. Koller, and S.J. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In *Proceedings of the 11th Annual Conference on Uncertainty in AI*, Montreal, Canada, 1995.

[74] S. King and C. Weiman. Helpmate autonomous mobile robot navigation system. In *Proceedings of the SPIE Conference on Mobile Robots*, pages 190–198, Boston, MA, November 1990. Volume 2352.

[75] 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.

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

[77] D. Kortenkamp, E. Huber, and P. Bonasso. Recognizing and interpreting gestures on a mobile robot. In *Proceedings of AAAI-96*, pages 915–921. AAAI Press/The MIT Press, 1996.

[78] D. Kortenkamp and T. Weymouth. Topological mapping for mobile robots using a combination of sonar and vision sensing. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 979–984, Menlo Park, July 1994. AAAI, AAAI Press/MIT Press.

[79] B. Kuipers and Y.-T. Byun. A robust qualitative method for spatial learning in unknown environments. In *Proceeding of Eighth National Conference on Artificial Intelligence AAAI-88*, Menlo Park, Cambridge, 1988. AAAI Press / The MIT Press.

[80] 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.

[81] G. Lacey and K.M. Dawson-Howe. The application of robotics to a mobility aid for the elderly blind. *Robotics and Autonomous Systems*, 23:245–252, 1998.

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

[83] 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.

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

[85] 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.

[86] 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.

[87] 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.

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

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

[90] R.E. Maeder. Ray tracing and graphics extensions. *The Mathematica Journal*, 4(3), 1994.

[91] M. J. Matarić. A distributed model for mobile robot environment-learning and navigation. Master's thesis, MIT, Cambridge, MA, January 1990. also available as MIT AI Lab Tech Report AITR-1228.

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

[93] D. McDermott. A reactive plan language. Research Report YALEU/DCS/RR-864, Yale University, 1991.

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

[95] O. Michel, P. Saucy, and F. Mondada. Khepontheweb : An experimental demonstrator in telerobotics and virtual reality. In *Proceedings of the IEEE International Conference on Virtual Systems and Multimedia (VSMM'97)*, 1997.

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

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

[98] U. Nehmzow, T. Smithers, and J. Hallam. Location recognition in a mobile robot using self-organizing feature maps. In G. Schmidt, editor, *Information Processing in Autonomous Mobile Robots*. springer Verlag, 1991.

[99] H. Neven and G. Schöner. Dynamics parametrically controlled by image correlations organize robot navigation. *Biological Cybernetics*, 75:293–307, 1996.

[100] I. Nourbakhsh. An affective mobile robot with a full-time job. *Artificial Intelligence*, 114(1–2):95–124, 1999.

[101] I. Nourbakhsh, R. Powers, and S. Birchfield. DERVISH an office-navigating robot. *AI Magazine*, 16(2):53–60, Summer 1995.

[102] I.R. Nourbakhsh. The failures of a self-reliant tour robot with no planner. Can be obtained at `http://www.cs.cmu.edu/~illah/SAGE/index.html`, 1998.

[103] T. Ogata and S. Sugano. Between humans and robots—consideration of primitivee language in robots. In *Proceedings of the Conference on Intelligent Robots and Systems (IROS'99)*, pages 870–875, 1999.

[104] T. Ogata and S. Sugano. Emotional communication between humans and the autonomous robot which has the emotion model. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'99)*, pages 3177–3182, 1999.

[105] L. Peters, H. Surmann, S. Guo, K. Beck, and J. Huser. Moria fuzzy logik gesteuertes, autonomes fahrzeug. internal report, 1994.

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

[107] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*. IEEE, 1989. IEEE Log Number 8825949.

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

[109] J.H. Reif. Complexity of the mover's problem and generalizations. In *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*, pages 421–427, 1979.

[110] 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.

[111] C. Rosenberg and C. Angle. IT: An interactive animatronic prototype. IS Robotics Inc. internal development project, description available at: `http://www.cs.cmu.edu/~chuck/robotpg/itpg/`, December 1994.

[112] J. Rosenblatt. *DAMN: A Distributed Architecture for Mobile Navigation*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, January 1997. Technical Report CMU-RI-TR-97-01.

[113] N. Roy, G. Baltus, D. Fox, F. Gemperle, J. Goetz, T. Hirsch, D. Magaritis, M. Montemerlo, J. Pineau, J. Schulte, and S. Thrun. Towards personal service robots for the elderly. In *Proceedings of the Workshop on Interactive Robotics and Entertainment (WIRE)*, Pittsburgh, PA, 2000. Carnegie Mellon University.

[114] 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.

[115] N. Roy, J. Pineau, and S. Thrun. Spoken dialogue management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000)*, Hong Kong, 2000.

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

[117] 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.

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

[119] P. Saucy and F. Mondada. KhepOnTheWeb: One year of access to a mobile robot on the Internet. In *Proceedings of the IEEE IROS'98 Workshop on Robots on the Web*, 1998.

[120] B. Schiele and J. Crowley. A comparison of position estimation techniques using occupancy grids. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 1628–1634, San Diego, CA, May 1994.

[121] F. E. Schneider. Sensorinterpretation und Kartenerstellung für mobile Roboter. Master's thesis, Dept. of Computer Science III, University of Bonn, 53117 Bonn, December 1994. In German.

[122] R.D. Schraft and G. Schmierer. *Service Robots*. A.K. Peters, 2000.

[123] J. Schulte, C. Rosenberg, and S. Thrun. Spontaneous short-term interaction with mobile robots in public places. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 1999.

[124] D. Schulz, W. Burgard, A.B. Cremers, D. Fox, and S. Thrun. Web interfaces for mobile robots in public places. *IEEE Magazine on Robotics and Automation*, 7(1):48–57, March 2000.

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

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

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

[128] R. Siegwart and K. Goldberg (eds). Robots on the Web. *IEEE Magazine on Robotics and Automation (special issue)*, 7(1), March 2000.

[129] R. Simmons. Concurrent planning and execution for autonomous robots. *IEEE Control Systems*, 12(1):46–50, February 1992.

[130] R. Simmons. Where in the world is xavier, the robot? *Machine Perception*, 5(1), 1996.

[131] 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.

[132] 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.

[133] 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.

[134] 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.

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

[136] M.R. Stein. Painting on the world wide web: The PumaPaint project. In *Proceedings of the IEEE IROS'98 Workshop on Robots on the Web*, 1998.

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

[138] K. Taylor and J. Trevelyan. Australia's telerobot on the Web. In *Proceedings of the 26th International Symposium On Industrial Robots*, 1995.

[139] M. A. L. Thathachar and P. S. Sastry. Estimator algorithms for learning automata. In *Proceedings of the Platinum Jubilee Conference on Systems and Signal Processing*, Bengalore, India, 1986.

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

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

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

[143] S. Thrun. Probabilistic algorithms in robotics. *AI Magazine*, 2000. to appear.

[144] 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.

[145] 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.

[146] M. C. Torrance. Natural communication with robots. Master's thesis, MIT Department of Electrical Engineering and Computer Science, Cambridge, MA, January 1994.

[147] S. Waldherr, S. Thrun, R. Romero, and D. Margaritis. Template-based recognition of pose and motion gestures on a mobile robot. In *Proceedings of the AAAI Fifteenth National Conference on Artificial Intelligence*, pages 977–982, 1998.

[148] G. Weiß, C. Wetzler, and E. von Puttkamer. Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans. In *Proceedings of the International Conference on Intelligent Robots and Systems*, pages 595–601, 1994.

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

[150] 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.

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

[152] U.R. Zimmer. Robust world-modeling and navigation in a real world. *Neurocomputing*, 13(2–4), 1996.