Bayesian Maps: probabilistic and hierarchical models for mobile robot navigation

Julien Diard, Pierre Bessiere

Bayesian Maps: probabilistic and hierarchical models for mobile robot navigation

Julien Diard[1] and Pierre Bessière[2]

[1] Laboratoire de Psychologie et NeuroCognition CNRS UMR 5105,
Université Pierre Mendès France, BSHM, BP 47,
38040 Grenoble Cedex 9, France,
Julien.Diard@upmf-grenoble.fr
[2] Laboratoire GRAVIR / IMAG - CNRS INRIA Rhône-Alpes,
655 avenue de l'Europe, 38330 Montbonnot Saint Martin, France,
Pierre.Bessiere@imag.fr

1 Introduction

Imagine yourself lying in your bed at night. Now try to answer these questions: Is your body parallel or not to the sofa that is two rooms away from your bedroom? What is the distance between your bed and the sofa? Except for some special cases (like rotating beds, people who actually sleep on their sofas, or tiny apartments), these questions are usually nontrivial, and answering them requires abstract thought. If pressed to answer quickly, so as to forbid the use of abstract geometry learned in high school, the reader will very probably give wrong answers.

However, if people had the same representations of their environment that roboticians usually provide to their robots, answering these questions would be very easy. The answers would come quickly, and they would certainly be correct. Indeed, robotic representations of space are usually based on large-scale, accurate, metric Cartesian maps. This enables judgment of parallelism and estimations of distances to be straightforward.

On the other hand, even though humans have difficulties with these questions, they usually have no trouble navigating from the sofa to the bed, or learning to do so in a new apartment. Robots have more difficulties in the same situation. In most robotic mapping approaches, the acquisition of a precise, and more importantly, accurate map of the environment is a prerequisite to solving navigation tasks. This is still a difficult and open issue in the general case.

Therefore, there appears to be a discrepancy in representations of space between the ones we usually provide to the robots we build and program, and the representations of space humans or animals use. Indeed, the nature, number, and possible interplay of the spatial representations involved in human or an-

imal navigation processes are still an open question in the life sciences. There is also a discrepancy in the difficulty of navigation tasks currently solved by state-of-the-art robots and the navigation tasks solved very easily by humans or animals.

We believe studying the difference between robotics and life sciences models of navigation can be very fruitful, both for modelling better robots and understanding animal navigation better. That is the topic of this chapter.

We first offer a quick overview of navigation models, both in robotics and in biology. We will first focus, more precisely, on probabilistic approaches to navigation and mapping in robotics. These approaches include – but are far from limited to – Kalman filters [Leonard et al., 1992], Markov localization models [Thrun, 2000], (partially or fully) observable Markov decision processes [Kaelbling et al., 1998], and hidden Markov models [Rabiner and Juang, 1993]. We will here assume that the reader has some familiarity with these approaches. We will show how these methods differ from most models of human or animal navigation. Indeed, whereas robotics approaches mostly rely on large-scale monolithic representations of space, models of animal navigation, right from the start, assume hierarchies of representations. We thus then describe hierarchical approaches to robotic mapping.

Indeed, in this domain of probabilistic modelling for robotics, hierarchical solutions are currently flourishing. However, we will argue that the main philosophy used by all these approaches is to try to extract, from a very complex but intractable model, a hierarchy of smaller models. Of course, *automatically* selecting the relevant decomposition of a problem into subproblems is quite a challenge – this challenge being far from restricted to the domain of navigation for robots facing uncertainties.

We propose to pursue an alternative route. We investigate how, starting from a set of simple probabilistic models, one can combine them to build more complex models. The goal of this paper is therefore to present a new formalism for building models of the space in which a robot must navigate (the *Bayesian Map* model), and a method for combining such maps together in a hierarchical manner (the *Abstraction operator*). This formalism allows for a new representation of space, in which the final program is built upon many inter-related models, each of them deeply rooted in lower-level sensorimotor relationships.

For brevity, we will discuss neither the learning methods that can be included in Bayesian Maps [Simonin et al., 2005], nor other operators for merging Bayesian Maps (such as the Superposition operator [Diard et al., 2005]). The foundation of the present work was created in Diard's PhD thesis [Diard, 2003].

The rest of this chapter is organized as follows. Section 2 gives a quick overview of the most prominent models of navigation and representation of large-scale space, first from a robotics point of view, then from a life sciences point of view. Section 3 offers a comparison of the main characteristics of the models, and an analysis of their strengths and weaknesses, and argues

in favour of the need for hierarchical and modular probabilistic models of navigation. We then introduce our contribution to the domain, the Bayesian Map formalism (Section 4), and one of the operators we defined for combining Bayesian Maps, the abstraction operator (Section 5). Finally, we report in Section 6 a series of robotic experiments in which we apply the Bayesian Map model and the abstraction operator on a Koala mobile robot, in a proof-of-concept experiment.

2 Navigation models in robotics and biology

We focus this brief review on existing models of navigation skills, in both robotics and life sciences. Because the literature in robotics concerning the representation of space is so large, we focus here on probabilistic approaches to mapping. In the life sciences, we describe some of the more prominent theoretical models of large-scale navigation in humans and animals, focusing on their hierarchical nature.

2.1 Probabilistic models of navigation and mapping

There is currently a wide variety of models in the domain of probabilistic mobile robot programming. These approaches include Kalman Filters (KF, [Leonard et al., 1992]), Markov Localization models (ML, [Thrun, 2000]), (Partially or fully) Observable Markov Decision Processes (POMDP, MDP, [Boutilier et al., 1999]), Hidden Markov Models (HMM, [Rabiner and Juang, 1993]), Bayesian Filters (BFs), and even Particle Filters (PFs). The literature covering these models is huge: for references that present several of them at once, giving unifying pictures, see [Murphy, 2002, Roweis and Ghahramani, 1999, Smyth et al., 1997]. Some of these papers define taxonomies of these approaches, by proposing some ordering that helps to classify them into families. One such taxonomy is presented in Fig. 1 (from [Diard et al., 2003]). It is based on a general-to-specific ordering: for example, it shows that Dynamic Bayesian Networks (DBNs) are a specialization of the Bayesian network formalism, tailored to take time series into account. In Fig. 1, subtrees that correspond to different specialization strategies can be identified. In the remainder of this section, we will focus on the Markov localization subtree, which corresponds to specializing DBNs using a four-variable model.

The ML model is basically an HMM with an additional action variable. It seems especially relevant in the robotic programming domain, because obviously robots can affect their states via motor commands. The stationary model of an HMM is basically the decomposition

$$P(P^t\ L^t\ L^{t-1}) = P(L^{t-1})P(L^t \mid L^{t-1})P(P^t \mid L^t), \tag{1}$$

where P^t is a perception variable, and L^t and L^{t-1} are state variables or, more precisely in our navigation case, location variables at time t and $t-1$.

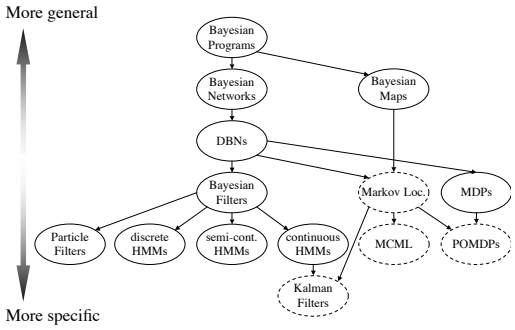

Fig. 1. Some common probabilistic modelling formalisms and their general-to-specific partial ordering (adapted from [Diard et al., 2003]). The ML subtree, which results from specializing DBNs, is highlighted (dashed nodes).

$P(L^t \mid L^{t-1})$ is commonly called the transition model, and $P(P^t \mid L^t)$ is referred to as the observation model. Starting from this structure, the action variable A^t is used to refine the transition model $P(L^t \mid L^{t-1})$ into $P(L^t \mid A^t\ L^{t-1})$, which is called the action model. Thus, the ML model is sometimes referred to as the input–output HMM model. Because of the generality of the BRP formalism, the model for Markov Localization can be cast into a BRP program. This is shown Fig. 2.

Program
- Description
  - Specification
    - Relevant Variables:
      - P^t : perception variable
      - L^t : discrete location variable at time t
      - L^{t-1} : discrete location variable at time $t-1$
      - A^t : action variable
    - Decomposition:
      - $P(A^t\ L^t\ L^{t-1}\ L^t) =$
      - $P(L^t \mid A^t\ L^{t-1})P(P^t \mid L^t)P(A^t)P(L^{t-1})$
    - Parametric Forms:
      - usually, matrices or particles
  - Identification:
    - any
  - Question:
    - localization $P(L^t \mid A^t\ P^t)$

Fig. 2. The Markov Localization stationary model expressed as a BRP.

The ML model is mostly used in the literature to answer the question $P(L^t \mid A^t\ P^t)$, which estimates the state of the robot, given the latest motor commands and sensor readings. When this state represents the position of the

robot in its environment, this amounts to localization. When this stationary ML model is replicated over time to estimate the positions $L^{0:T}$ of the robot over a long series of time steps, it can be shown that an iterative localization procedure exists that localizes the robot simply by updating the last position estimate in view of the latest motor commands A^t and sensor readings P^t. This justifies, in this presentation, the focus on the stationary model.

The ML model has been successfully applied in a range of robotic applications, the most notable examples being the Rhino ([Thrun et al., 1998, Burgard et al., 1999]) and Minerva ([Thrun et al., 1999a,b]) robotic guides. The most common application of the ML model is the estimation of the position of a robot in an indoor environment, using a fine-grained metric grid as a representation. In other words, in the model of Fig. 2, the state variable is very frequently the pose of the robot, i.e. a pair of x, y discrete Cartesian coordinates for the position, and an angle θ for the orientation of the robot. Assuming a grid cell size of 50 cm, an environment of 50 $m \times 50$ m, and a 5° angle resolution entails a state space of 720,000 states.

Using some specialized techniques and assumptions, it is possible to make this memory-consuming model tractable.

For example, the forms of the probabilistic model can be implemented using sets of particles. These approximate the probability distributions involved in Fig. 2, which leads to an efficient position estimation. This specialization is called the Monte Carlo Markov Localization model (MCML, [Fox et al., 1999]).

Another possibility is to use a Kalman filter as a specialization of the ML model, in which variables are continuous. The action model $P(L^t \mid A^t\ L^{t-1})$ and the observation model $P(P^t \mid L^t)$ are both specified using Gaussian laws with means that are linear functions of the conditioning variables. With these hypotheses, it is possible to solve the inference problem analytically to answer the localization question. This leads to an extremely efficient algorithm that explains the popularity of Kalman filters.

2.2 Biologically inspired models

All the approaches mentioned in the preceding section are based on the classical view of robotic navigation, which is inherited from marine navigation. In this view, solving a navigation task basically amounts to answering sequentially the questions of Levitt and Lawton: "Where am I?", "Where are other places with respect to me?", and "How do I get to other places from here?" [Levitt and Lawton, 1990]. These are formulated somewhat similarly in the works of Leonard and Durrant-Whyte: "Where am I?", "Where am I going?", and "How should I get there?" [Leonard and Durrant-Whyte, 1991].

While they are a valid first decomposition of the navigation task into subtasks, these questions have usually led to models that require a *global model of the environment*, which allows the robot to localize itself (the first questions), to infer spatial relationships between the current recognized location

and other locations (the second questions), and to plan a sequence of actions to move within the environment (the third questions). These skills amount to the first two phases of the "perceive, plan, act" classical model of robotic control.

Very early in their analysis, biomimetic models of navigation disputed this classical view of robotic navigation. Indeed, when studying living beings, the existence of such a unique and global representation that would be used to solve these three questions is very problematic. This seems obvious even for simple animals like bees and ants. For instance, the outdoor navigation capabilities of the desert ant *Cataglyphis*, which have been widely studied, rely on the use of the polarization patterns of the sky [Lambrinos et al., 2000]. It is clear that such a strategy is useless for navigating in a nest; this calls for another navigation strategy, and another internal model. The existence of a unique representation is also doubtful for humans. The navigation capabilities of humans are based on internal models of their environment (*cognitive maps*), but their nature, number and complexity are still largely debated (see for instance [Berthoz, 2000, Redish and Touretzky, 1997, Wang and Spelke, 2002], for entry points into the huge literature associated with this domain).

As a consequence, biomimetic approaches assume from the start the existence of multiple representations, most often articulated in a *hierarchical* manner. We now give a brief review of some theories from that domain, focusing on their hierarchical components.

Works by Redish and Touretzky

Works by Redish and Touretzky address the issue of the role of the hippocampus and parahippocampal populations in rodent navigation, focusing on the well-studied place cells and head direction cells. They proposed a conceptual model [Touretzky and Redish, 1996] and discussed its anatomical plausibility [Redish and Touretzky, 1997]. Their hierarchical conceptual model consists of four spatial representations (place code, local view, path integrator and head direction code), supplemented by two components called the reference frame selection subsystem and the goal subsystem.

Place codes are local representations tied to one or several landmarks or geometric features of the environment. When the environment of the animal becomes large or structured, several place codes may be used to describe this environment, each place code representing a part of the environment. For instance, Gothard et al. [Gothard et al., 1996] found different place codes for a rat navigating in an environment containing a goal and a starting box. They identified three independent place codes: one tied to the room, one to the goal, and one to the box. These effectively provide representations of sections of the environment: cells tuned to the box frame were only active when the rat was in or around the box, cells tuned to the goal only responded when the rat was near the goal, cells tuned to the room were active at other times (i.e. when the rat was not near the box or the goal).

The reference frame selection component is responsible for selecting the appropriate place code for navigating in the environment. In the above example, this means that it is responsible for selecting, at any given time, which place code should be active.

This theory thus proposes an account of the low-level encoding of space in central nervous systems of animals using a two-layer hierarchy of models. The low-level layer consists of a series of place codes describing portions of the animal environment, under the hierarchical supervision of a larger-space model.

Computational models of the low-level component of this hierarchy, (i.e. place cells and head-direction cells) abound in the literature (e.g. [Hartley and Burgess, 2002]), whereas the reference frame selection component, to the best of our knowledge, has yet to be mathematically defined.

Works by Jacobs and Schenk

Jacobs and Schenk proposed a new theory of how the hippocampus encodes space [Jacobs and Schenk, 2003, Jacobs, 2003]. This theory is called the Parallel Map Theory (PMT), and it defines a hierarchy of navigation representations made of three components and two layers.

The bearing map is the first, low-level, component. This is a single map based on several directional cues such as intersecting gradients. It provides a large-scale two-dimensional reference frame, enabling large-scale navigation skills, simply using gradient ascent or descent.

The sketch maps are the second component of the low-level layer of the hierarchy. They encode small-scale fine-grained representations of the relationship of landmarks that are close to each other (positional cues). This creates local representations, which can be used for precise control of the position, and thus for solving precise, small-scale navigation tasks.

Finally, the integrated map is the third, high-level, component. This is constructed from the bearing map and several sketch maps. It consists of a unified map of large-scale environments, where the local sketch maps are cast into the large-scale reference frame of the bearing map. This provides the means to infer large-scale spatial relationship between the local, metric representations of the sketch maps, thus allowing computation of large-scale shortcuts and detours.

To the best of our knowledge, the papers by Jacobs and Schenk do not provide computational models of these different components. Instead, they mainly focus on the anatomical and phylogenetic plausibility of their conceptual model. This provides many experimental predictions concerning possible impairments resulting from lesions.

Works by Wang and Spelke

These authors dispute the idea that enduring, allocentric, large-scale representations of an environment should be the main theoretical tool used for

investigating navigation in humans and animals. Indeed, the cognitive map concept, introduced by Tolman in 1948, is still controversial [Tolman, 1948]. Instead, Wang and Spelke argue that many navigation capabilities in animals can be explained by dynamic, egocentric representations that cover a limited portion of the environment [Wang and Spelke, 2000, 2002]. Such representations can be studied in animals that are far simpler than humans, such as desert ants [Lambrinos et al., 2000].

Studies on these animals have identified three subsystems: a path integration system, a landmark-based navigation system, and a reorientation system. This last component is not hierarchically related to the other two, as it is mainly responsible for resetting the path integration system when the animal becomes disoriented. However, the first two components show a strong hierarchical relation. Indeed, it has been shown that the landmark-based strategy is hierarchically higher in the cognitive mechanisms of insects and rodents. It also appears that, in the sudden absence of landmarks after learning a path, animals rely on the path integration encoding as a "backup" [Stackman et al., 2002, Stackman and Herbert, 2002].

This model is somewhat different from the previous studies, as it focuses on defining a hierarchy of skills of navigation, instead of hierarchies of representations of space, as in the PMT or studies of the hippocampal and parahippocampal areas.

Works by Kuipers, Franz, and Trullier

The hierarchies of models proposed in the biomimetic robotic literature ([Kuipers, 1996, Trullier et al., 1997, Franz and Mallot, 2000, Kuipers, 2000, Victorino and Rives, 2004]) have several aspects: they are hierarchies of increasing navigation skills, but also of increasing scale of the represented environment, of increasing time scale of the associated movements, and of increasing complexity of representations. This last aspect means that topologic representations, which are simple, come at a lower level than global metric representations, which are arguably more complex to build and manipulate. This ordering stems from the general observation that animals that are able to use shortcuts and detours between two arbitrary encoded places (skills that require global metric models) are rather complex animals, like mammalians. These skills seem to be mostly absent from simpler animals, like invertebrates.

The resulting proposed hierarchies show a striking resemblance. We present the salient and common features of these hierarchies by summarizing the Spatial Semantic Hierarchy (SSH) proposed by Kuipers [Kuipers, 1996, 2000]. It is, to the best of our knowledge, the only biomimetic approach that has been applied to obtain a complete and integrated robotic control model.

The SSH essentially consists of four hierarchical levels: the *control* level, the *causal* level, the *topological* level, and the *metric* level.

The control level is a set of reactive behaviours, which are control laws deduced from differential equations. These behaviours describe how to move

the robot to reach an extremum of some gradient measure. This extremum can be zero-dimensional (a point in the environment), in which case it is called a locally distinctive state. The associated behaviour is called a hill-climbing law. The extremum can also be one-dimensional (a line or curve in the environment), in which case the behaviour is called a trajectory-following law. Provided that any trajectory-following law guarantees arriving at a place where a hill-climbing law can be applied, then alternating laws of both types displace the robot in a repetitive fashion. This solves the problem of the accumulation of odometry errors. The control level is also referred to as the guidance level [Trullier et al., 1997, Franz and Mallot, 2000].

Given the control level, the environment can be structured and summarized by the locations of locally distinctive states and the trajectories used to go from one such state to another. This abstraction takes place in the causal level, which is the second level of the hierarchy of representations. Unlike the control level, it allows the robot to memorize relationships between places that are outside the current perceptive horizon (this is part of the way-finding capabilities in other terminologies [Trullier et al., 1997, Franz and Mallot, 2000]). To do so, Kuipers abstracts locally distinctive places as views V, the application of lower-level behaviours as actions A, and defines *schemas* as tuples $\langle V, A, V' \rangle$ (expressed as first-order logic predicates). The schemas have two meanings. The first is a procedural meaning: "when the robot is at V, it must apply action A." This aspect of a schema is equivalent to the recognition-triggered response level of the other hierarchies [Trullier et al., 1997, Franz and Mallot, 2000], or to the potential field approaches, or to other goal-oriented methods. However, the second meaning of schemas is a declarative one, where $\langle V, A, V' \rangle$ stands for: "applying action A from view V eventually brings the robot to view V'." This allows using the schemas for prediction of future events, or in a planning process, for example.

The goal of the topological level is to create a globally consistent representation of the environment, as structured by places, paths and regions. These are extracted from lower-level schemas by an abduction process that creates the minimum number of places, paths and regions to maintain consistency with the known schemas. Places are zero-dimensional parts of the environment that can be abstractions of lower-level views, or abstractions of regions (for higher-level topological models). Paths are one-dimensional, oriented, and can be built upon one or more schemas. Finally, regions are two-dimensional subspaces, delimited by paths. It must be noted that, because the problem of accumulation of odometry error was dealt with at the control level, building a globally consistent topological representation (i.e. solving the global connectivity problem) is much easier. To do so, Kuipers proposes an exploration strategy, the rehearsal procedure, which, unfortunately, requires a bound on the exploration time and is not well suited to dynamic environments. The places and paths of the topological representation obtained can be used for solving planning queries using classical graph-searching algorithms.

The last level is the metric level, in which the topological graph is cast into a unique global reference frame. For reasons outlined above (and detailed in [Kuipers, 2000]), this level is considered as a possibility, not a prerequisite for solving complex navigation tasks. If the sensors are not good enough to maintain a good estimation of the Cartesian coordinates of the position, for instance, it is still possible to use the topological model to act in the environment – although shortcuts and detours are not possible. Indeed, few robotics systems implementing biomimetic models include the metric level [Franz and Mallot, 2000, Trullier et al., 1997].

3 Theoretical comparison

3.1 Which mathematical formalism?

A major drawback of the biomimetic models presented previously is that they are seldom defined as computational models: they give conceptual frameworks for understanding animal navigation but lack complete mathematical definitions that would make simulations of these models possible. The notable exception is the SSH model, which not only defines layers in a hierarchy of space representations but also defines each of them mathematically.

However, the SSH model uses a variety of formalisms for expressing knowledge at different layers of the hierarchy: differential equations and their solutions for the control level, and first-order logic and deterministic algorithms for higher-level layers of the hierarchy. This makes it difficult to justify the consistency and correctness of the mechanisms for communication between the layers of the hierarchy theoretically. In some cases, it even limits and constrains the contents of the layers: for instance, the SSH model *requires* that the behaviours of the control level *guarantee* that the robot reaches the neighbourhood of a given locally distinctive state. In our view, this constraint is barely acceptable for any kind of realistic robotic scenario. Consider dynamic environments: how can we guarantee that a robot will reach a room if a door on the route can be closed?

We assume as a starting point for our analysis that the best formalism for expressing incomplete knowledge and dealing with uncertain information is the probabilistic formalism [Bessière et al., 1998, Jaynes, 2003]. This gives us a clear and rigorous mathematical foundation for our models. The probability distributions are our unique tool for the expression and manipulation of knowledge, and in particular, for communication between submodels. We will thus argue in favour of hierarchical probabilistic models.

3.2 Hierarchical probabilistic models

This idea is not a breakthrough: in the domain of probabilistic modelling for robotics, hierarchical solutions are currently flourishing. The more active domain in this regard is decision-theoretic planning: one can find variants of

MDPs that accommodate hierarchies or that select automatically the partition of the state space (see, for instance, [Hauskrecht et al., 1998, Lane and Kaelbling, 2001], or browse through the references in [Pineau and Thrun, 2002]). More exceptionally, one can find hierarchical POMDPs [Pineau and Thrun, 2002]. The current work can also be related to Thrun's object mapping paradigm [Thrun, 2002], in particular concerning the aim of transferring some of the knowledge the programmer has about the task to the robot.

Some hierarchical approaches outside the MDP community include hierarchical HMMs and their variants (see [Murphy, 2002] and references therein), which, unfortunately, rely on the notion of the final state of the automaton. Another class of approaches relies on the extraction of a graph from a probabilistic model such as a Markov localization model [Thrun, 1998], or an MDP [Lane and Kaelbling, 2002]. Using such deterministic notions is inconvenient in a purely probabilistic approach, such as we are pursuing here.

Moreover, the main philosophy used by all the previous approaches is to try to extract, from a very complex but intractable model, a hierarchy of smaller models (structural decomposition, see [Pineau and Thrun, 2002]).

Again, this comes from the classical robotic approach, where the process of perception (in particular, localization) is assumed to be independent of the processes of planning and action. A model such as the ML model (Fig. 2) is only concerned with localization, not control; therefore, its action variable A^t is actually only used as an *input* to the model. In this view, a pivotal representation is used between the perception and planning subproblems. It is classically assumed that the more precise this pivotal model, the better. Unfortunately, when creating integrated robotic applications, dealing with both the building of maps and their use is necessary. Some authors have realized that their global metric maps were too complex to be easily manipulated. Therefore, they have tried to *degrade* their maps, which were so difficult to obtain initially, for instance, by extracting graphs from their probabilistic models [Thrun, 1998]. This problem is also at the core of the robotic planning domain, where the given description of the environment is assumed to be an infinitely precise geometrical model. The difficulty is to discretize this intractable, continuous model into a finite model, typically, in the form of a graph [Latombe, 1991, Kavraki et al., 1996, Mazer et al., 1998, Svestka and Overmars, 1998].

3.3 Modular probabilistic models: towards Bayesian Maps

We propose to pursue an alternative route, investigating how, starting from a set of simple models, one can combine them to build more complex models. Such an incremental development approach allows us to depart from the classical "perceive, plan, act" loop, considering instead hierarchies built upon many inter-related models, each of them deeply rooted in lower-level *sensory and motor* relationships.

The Bayesian robotic programming methodology offers exactly the formal tool that can transfer information from one program to another in a theoretically rigorous fashion. Indeed, in Bayesian robotic programs, terms appearing in a description c^1 can be defined as a probabilistic question to another description c^2. This creates a link between the two descriptions, one being used as a resource by another. Depending on the way questions are used to link subprograms, several different operators can be created, each with specific semantics: for instance, in the framework of behaviour-based robotics, Lebeltel has defined behaviour combination, hierarchical behaviour composition, and behaviour sequencing and sensor model fusion operators. He has also applied these successfully to realize a complex watchman robot behaviour using a control architecture involving four hierarchical levels [Lebeltel et al., 2004].

Thus, we can solve a global robotic task problem by first decomposing it into subproblems, then writing a Bayesian robot program for each subproblem, and finally combining these subprograms. This method makes robot programming similar to structured computer programming. So far in our work, we have let the programmer do this analysis: relevant intermediary representations can be imagined, or copied from living beings. We propose to apply this strategy to the map-based navigation of mobile robots. The submodels can be submaps, in the spatial sense (i.e. covering a part of the global environment), or in the subtask sense (i.e. modelling knowledge necessary for solving part of the global navigation task), or even in less familiar senses (e.g. modelling partial knowledge from part of the sensorimotor apparatus).

Our approach is therefore based on a formalism for building models of the space in which the robot must navigate, called the *Bayesian Map* model, that allows us to build submodels that provide behaviours as resources. We also define *operators* for combining such maps in a hierarchical manner.

4 The Bayesian Map formalism: definition

4.1 Probabilistic definition

A Bayesian Map c is a description that defines a joint distribution

$$P(P\ L^t\ L^{t'}\ A \mid c),$$

where:

- P is a perception variable (the robot reads its values from physical sensors or lower-level variables);
- L^t is a *location* variable at time t;
- $L^{t'}$ is a variable having the same domain as L^t, but at time t' (without loss of generality, let us assume $t' > t$); and
- A is an action variable (the robot writes commands to this variable).

For simplicity, we will assume here that all these variables have finite domains.

The choice of decomposition is not constrained: any probabilistic dependency structure can therefore be chosen here; see [Attias, 2003] for an example of how this can lead to interesting new models. Finally, the definition of forms and the learning mechanism (if any) are also not constrained.

For a Bayesian Map to be usable in practice, the description must be rich enough to generate *behaviours*. We describe as *elementary behaviour* any question of the form $P(A_i \mid X)$, where A_i is a subset of A, and X is a subset of the other variables of the map (i.e. of those not in A_i). A typical example consists of the probabilistic question $P(A \mid [P = p]\ [L^{t'} = l])$: compute the probability distribution over actions, given the current sensor readings p and the goal l to reach in the internal space of possible locations.

A behaviour can be *not elementary*, for example, if it is a sequence of elementary behaviours, or, in more general terms, if it is based on elementary behaviours and some other knowledge (which may be expressed in terms other than maps).

For a Bayesian Map to be interesting, we will also require that it generates *several* behaviours – otherwise, defining just a single behaviour instead of a map is enough. Such a map is therefore a resource, based on a location variable relevant enough to solve a class of tasks; this internal model of the world can be reified.

A "guide" one can use to "make sure" that a given map will generate useful behaviours is to check whether the map answers in a relevant manner the three questions $P(L^t \mid P)$ (localization), $P(L^{t'} \mid A\ L^t)$ (prediction) and $P(A \mid L^t\ L^{t'})$ (control).

By "relevant manner", we mean that these distributions must be informative, in the sense that their entropy is "far enough" from its maximum (i.e. the distribution is different from a uniform distribution). This constraint is not formally well defined, but it seems intuitive to focus on these three questions. Indeed, the skills of localization, prediction and control are well identified in the literature as ways of generating behaviours. Checking that the answers to these questions are informative is a first step to evaluating the quality of a Bayesian Map with respect to solving a given task.

Figure 3 is a summary of the definition of the Bayesian Map formalism.

4.2 Generality of the Bayesian Map formalism

We now invite the reader to verify that the Markov localization model is indeed a special case of the Bayesian Map model by comparing Fig. 2 and Fig. 3. Recall that Kalman filters and particle filters are special cases of Markov localization, as they add hypotheses over the choice of dependency structure made by the Markov localization model. This implies that Kalman filters and particle filters are also special cases of Bayesian Maps.

Bayesian Maps can therefore accommodate many different forms, depending on the need or information at hand: for example, a Bayesian Map can be

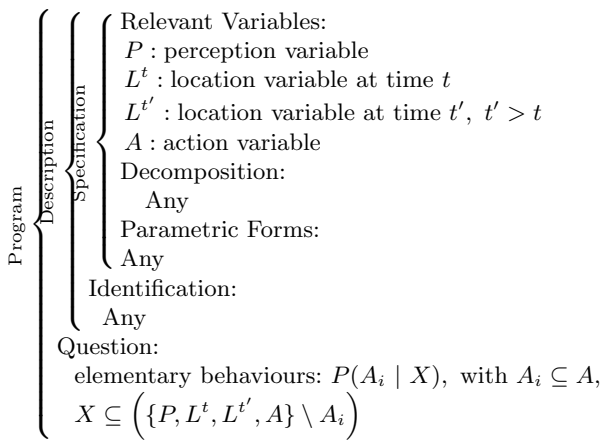

Fig. 3. The Bayesian Map model definition expressed in the BRP formalism.

structured like a real-valued Kalman filter for tracking the angle and distance to some feature when it is available. If that feature is not present, or if the linearity hypotheses fail, we can use another Bayesian Map, which may not be a Kalman filter (for example, based on a symbolic variable).

Hierarchies of Bayesian Maps (via the abstraction operator) can thus be hierarchies of Markov localization models or hierarchies of Kalman filters, and so on. Moreover, heterogeneous hierarchies of these models can be imagined: ML over KFs, or even n KFs and one ML model. In our view, this would be a potential alternative to the solution of Tomatis et al. [Tomatis et al., 2001, 2003].

5 Combining Bayesian Maps: definition of the abstraction operator and example

Having defined the Bayesian Map concept, we now turn to defining operators for combining Bayesian Maps. The one we present here is called the abstraction of maps. It is defined in Fig. 4 and discussed in the rest of this section.

As stressed above, in a Bayesian Map, the semantics of the location variable can be very diverse. The main idea behind the abstraction operator is to build *a Bayesian Map c containing locations that are other Bayesian Maps* $c_1, c_2, \ldots, c_n$. The location variable of the abstract map will therefore take n possible symbolic values, one for each underlying map c_i. Each of these maps will be "nested" in the higher-level abstract map, which justifies the use of the term "hierarchy" in our work. Recall that Bayesian Maps are designed for generating behaviours.

Program
Description
Specification

Relevant Variables:

$P = \bigwedge_{i=1}^{n} \left(P_i \wedge L_i^t \wedge L_i^{t'} \wedge A_i \right)$

$L^t : \mathcal{D}_{L^t} = \{c_1, c_2, \ldots, c_n\}, k_{L^t} = n$

$L^{t'} : \mathcal{D}_{L^{t'}} = \{c_1, c_2, \ldots, c_n\}, k_{L^{t'}} = n$

$A : \mathcal{D}_A = \{a_1, a_2, \ldots, a_k\}, k_A = k$

Decomposition:

$$P(P\ L^t\ L^{t'}\ A) = P(L^t)P(L^{t'})P(A \mid L^t\ L^{t'}) \prod_{i=1}^{n} P(P_i\ L_i^t\ L_i^{t'}\ A_i \mid L^t)$$

Parametric Forms:

$P(L^t) =$ Uniform

$$P(P_i\ L_i^t\ L_i^{t'}\ A_i \mid [L^t = c]) = \begin{cases} \text{if } c = c_i \text{ then } P(P_i\ L_i^t\ L_i^{t'}\ A_i \mid c_i) \\ \text{else Uniform} \end{cases}$$

$P(L^{t'}) =$ Uniform

$P(A \mid L^t\ L^{t'}) =$ Table

Identification:

a priori programming or learning of $P(A \mid L^t\ L^{t'})$

Question:

$$P(L^t \mid P) \propto \prod_{i=1}^{n} P(P_i\ L_i^t\ L_i^{t'}\ A_i \mid L^t)$$

$$P(L^{t'} \mid A\ L^t) \propto P(A \mid L^t\ L^{t'})$$

$$P(A \mid L^t\ L^{t'}) = P(A \mid L^t\ L^{t'}).$$

Fig. 4. The abstraction operator definition expressed as a Bayesian Map.

Let us denote $a_1, a_2, \ldots, a_k$, the k behaviours defined in the n underlying maps, with $k \geq n$. In the abstract map, these behaviours can be used for linking the locations c_i. The action variable of the abstract map will therefore take k possible symbolic values, one for each behaviour of the underlying maps. To build an abstract map having n locations, the programmer will require n previously defined lower-level maps, which generate k behaviours. The numbers n and k are therefore small, and so the abstract map deals with a small internal space, having retained from each underlying map only a symbol, and having "forgotten" all their details. This justifies the use of the name "abstraction" for this operator. However, this "summary mechanism" has yet to be described: that is what the perception variable P of the abstract map will be used for, as it will be the list of all the variables appearing in the underlying maps: $P = P_1, L_1^t, L_1^{t'}, A_1, \ldots, P_n, L_n^t, L_n^{t'}, A_n$.

Given the four variables of the abstract map, we define its joint distribution with the following decomposition:

$$P(P\ L^t\ L^{t'}\ A) = P(L^t)P(L^{t'})P(A \mid L^t\ L^{t'}) \prod_{i=1}^{n} P(P_i\ L_i^t\ L_i^{t'}\ A_i \mid L^t). \quad (2)$$

In this decomposition, $P(L^t)$ and $P(L^{t'})$ are defined as uniform distributions. All the terms of the form $P(P_i\ L_i^t\ L_i^{t'}\ A_i \mid [L_t = c])$ are defined as follows: when $c \neq c_i$, the probabilistic dependency between the variables P_i, L_i^t, $L_i^{t'}$, A_i of the map c_i is supposed unknown, and therefore defined by a uniform distribution. When $c = c_i$, however, this dependency is exactly what the map c_i defines. Therefore this term is a question to the description c_i but a question that includes the whole subdescription by asking for the joint distribution it defines. Because the last term, $P(A \mid L^t\ L^{t'})$, only includes symbolic variables that have a small number of values, it makes sense to define it as a table, which can be easily programmed *a priori* or learned experimentally.

The abstract Bayesian Map is now fully defined, and, given the n underlying maps, can be built automatically. The last step is to verify that it generates useful behaviours. We will examine the guiding questions of localization, prediction and control.

The localization question leads to the following inference (derivation omitted): $P(L^t \mid P) \propto \prod_{i=1}^{n} P(P_i\ L_i^t\ L_i^{t'}\ A_i \mid L^t)$. The interpretation of this result will be explained with an example in Section 6. The derivations for solving the prediction $P(L^{t'} \mid A\ L^t)$ and control $P(A \mid L^t\ L^{t'})$ questions are also straightforward, given Fig. 4.

Recall that the final goal of any Bayesian Map is to provide behaviours. In the abstract map, this is done by answering a question like $P(A \mid [L^{t'} = c_i]\ [P = p])$: what is the probability distribution over lower-level behaviours, knowing all values p of the variables of the lower level, and knowing that we want to "go to map c_i" (more formally, "reach some location recognized as the lower-level map c_i")? Answering this question thus allows selection of the most relevant underlying behaviour to reach a given high-level goal. The computation is as follows:

$$P(A \mid L^{t'}\ P) \propto \sum_{L^t} \left(\prod_{i=1}^{n} P(P_i\ L_i^t\ L_i^{t'}\ A_i \mid L^t) \right) P(A \mid L^t\ L^{t'}). \qquad (3)$$

This computation includes the localization question, by weighing the probabilities given by the control model $P(A \mid L^t\ L^{t'})$. In other words, the distribution over the action variable A includes all localization uncertainties. Each underlying model is used, even when the robot is located at a physical location that this model is not made for. As a direct consequence, there is no need to *decide* what map the robot is in, nor to *switch* from map to map: the computation considers all possibilities and weighs them according to their (localization) probabilities. Therefore the underlying maps are not necessarily "mutually exclusive" in a geographical sense.

6 Experimental validation

We report here an experiment made on the well-known Koala mobile robot platform (K-team company). To keep as much control as possible over our

experiments and the different effects we observe, we simplify the sensorimotor system and its environment. We only use the 16 proximeters $Px = Px_0 \wedge \ldots \wedge Px_{15}$ of our robot, and we keep two degrees of freedom of motor control, via the rotation and translation speeds $Vrot$ and $Vtrans$. The environment we use is a 5 m $\times$ 5 m area made of movable planks (a typical configuration is shown in Fig. 5). The goal of this experiment is to solve a navigation task: we want the robot to go and hide in any corner, as if the empty space in the middle of the area were dangerous.

The first programming step is to analyse this task into subtasks. Three particular situations are relevant for solving the task: the robot can either be near a wall, and it should follow it in order to reach the nearest corner, or the robot can be in a corner, and it should stop, or finally it could be in empty space, and should therefore go straight, so as to leave the exposed area as quickly as possible.

6.1 Low-level Bayesian Maps

Given this analysis, the second programming step is to define one Bayesian Map for each of the three situations. They all use the same perception variable $P = Px$ and the same action variable $A = Vrot \wedge Vtrans$.

The first map, c_{wall}, describes how to navigate in the presence of a single wall, using a location variable $L^t = \theta \wedge Dist$: the phenomenon "wall" is summarized by an angle θ and a distance $Dist$. Therefore, c_{wall} defines $P(Px\ \theta^t\ Dist^t\ \theta^{t'}\ Dist^{t'}\ Vrot\ Vtrans \mid c_{wall})$. We have implemented this map using 12 possible angle values and three different distances. This leads to a compact model that is still accurate enough to solve the subtasks. The dependency structure we choose is (c_{wall} on right hand sides omitted):

$$
\begin{aligned}
&P(Px\ \theta^t\ Dist^t\ \theta^{t'}\ Dist^{t'}\ Vrot\ Vtrans)\\
&= P(\theta^t\ Dist^t) \prod_i \left[P(Px_i \mid \theta^t\ Dist^t)\right] P(\theta^{t'}\ Dist^{t'})\\
&\qquad P(Vrot \mid \theta^t\ Dist^t\ \theta^{t'}\ Dist^{t'}) P(Vtrans \mid \theta^t\ Dist^t\ \theta^{t'}\ Dist^{t'}).
\end{aligned}
$$

$P(\theta^t\ Dist^t)$ and $P(\theta^{t'}\ Dist^{t'})$ are uniform probability distributions. Each term of the form $P(Px_i \mid \theta^t\ Dist^t)$ is a set of bell-shaped probability distributions[3] that were identified experimentally in a supervised learning phase: we physically put the robot in all 36 possible situations with respect to the wall and recorded proximeter values so as to compute experimental means and standard deviations. Finally, the two control terms $P(Vrot \mid \theta^t\ Dist^t\ \theta^{t'}\ Dist^{t'})$ and $P(Vtrans \mid \theta^t\ Dist^t\ \theta^{t'}\ Dist^{t'})$ were programmed "by hand": given the current angle and distance, and the angle and distance to be reached, what should the motor commands be?

[3] Bell-shape probability distributions approximate Gaussian probability distributions on finite discretized ranges.

This map successfully solves navigation tasks like "follow wall right", "follow wall left", "go away from wall", "stop", using behaviours of the same name. For example, "follow wall right" is defined by the probabilistic question $P(Vrot\, Vtrans \mid Px\, [L^{t'} = \langle 90, 1\rangle])$: compute the probability distribution on motor variables knowing the sensory input and knowing that the location to reach is $\theta = 90$, $Dist = 1$ (wall on the right at medium distance).

This map is an instance where a Kalman filter-based Bayesian Map could have been used instead, for example, if we had required a more accurate angle and distance to the wall using continuous variables. The coarse-grained set of values we used were sufficient for our experiments.

The two other Bayesian Maps we define are the following. 1) c_{corner} describes how to navigate in a corner, using a symbolic location variable that can take four values: $FrontLeft$, $FrontRight$, $RearLeft$ and $RearRight$. This is enough for solving tasks like "quit corner and follow right", "away from both walls", "stop". 2) $c_{emptyspace}$describes how to navigate in empty space, i.e. when the sensors do not see anything. The behaviours defined here are "straight ahead" and "stop". For brevity, these two Bayesian Maps are not described further here; the interested reader can find details in Diard's PhD thesis [Diard, 2003].

6.2 Abstract Bayesian Map

Given these three maps, the third and final programming step is to apply the abstraction operator to them. We obtain a map c with a location variable $L^t = \{c_{wall}, c_{corner}, c_{emptyspace}\}$. The action variable lists the behaviours defined by the low-level maps: $A = \{\text{follow wall right}, \text{go away from wall}, \ldots\}$. The rest of the abstract map is in accordance with the schema of Fig. 4.

We now discuss the localization question. Let us assume that the robot is in empty space: all its sensors read zero. Let us also assume that the robot is currently applying the "straight ahead" behaviour, which sets $Vrot$ and $Vtrans$ near 0 (no rotation) and 40 (fast forward movement), respectively, using sharp bell-shaped distributions.

Let us consider the probability of being in location $c_{emptyspace}$ (with w standing for $wall$, c for $corner$ and e for $emptyspace$).

$$
\begin{aligned}
&P([L^t = c_{emptyspace}] \mid P) \\
&\propto \begin{pmatrix} P(P_w\ L_w^t\ L_w^{t'}\ A_w \mid [L^t = c_{emptyspace}]) \\ P(P_c\ L_c^t\ L_c^{t'}\ A_c \mid [L^t = c_{emptyspace}]) \\ P(P_e\ L_e^t\ L_e^{t'}\ A_e \mid [L^t = c_{emptyspace}]) \end{pmatrix}.
\end{aligned}
$$

Of the three terms of the product, two have uniform distributions, and one is the joint distribution given by $c_{emptyspace}$. This joint distribution gives a very high probability for the current situation, because describing the phenomenon "going straight ahead in empty space" basically amounts to favouring sensory

readings of 0 and motor commands near 0 and 40 for $Vrot$ and $Vtrans$, respectively. The situation is quite the opposite for $P([L^t = c_{wall}] \mid P)$: for example, c_{wall} does not favour this sensory situation at all. Indeed, the phenomenon "I am near a wall" is closely related to the sensors actually sensing something. The probability of seeing nothing on the sensors knowing that the robot is near a wall is very low: $P([L^t = c_{wall}] \mid P)$ will be very low. The reasoning is similar for $P([L^t = c_{corner}] \mid P)$.

This computation can thus be interpreted as the *recognition of the most pertinent underlying map* for a given sensorimotor situation. Alternatively, it can be seen as a *measure of the coherence of the values of the variables* of each underlying map, or even as a *Bayesian comparison of the relevance of models*, as assessed by the numerical value of the joint distributions of each lower-level model. Because these distributions include (lower-level) location and action variables, the maps are recognized not only from sensory patterns but also from what the robot is currently doing.

The localization question can therefore be used to assess the "validity zones" of the underlying maps, i.e. the places in the environment where the hypotheses of each model hold. Experimentally, we make the robot navigate in the environment, and we ask at each time step the localization question. We can visually summarize the answer, for example, by drawing values for L^t, and reporting the drawn value on a Cartesian map of the environment. A simplified but readable result is shown in Fig. 5. As can be seen, the robot correctly recognizes each situation for which it has a model. Note that the resulting zones are not contiguous in the environment: for example, all the corners of the environment are associated with the same symbol, namely, c_{corner}. This effect is known as *perceptual aliasing*. However, this very simple representation is sufficient for solving the task that was given to the robot: the behaviour "go hide in any corner" is indeed generated by the abstract map.

Using the abstract Bayesian Map we have programmed in this way, the robot can solve the task of reaching corners. A typical trajectory for the robot, starting from the middle of the arena, is to start by going straight ahead. As soon as a couple of forward sensors sense something, the "empty space" situation is no longer relevant, and the robot applies the best model it has, depending on the correlation between what the sensors see: if it looks like a wall and continues to do so as the robot moves, then the probability for the "wall" model is high; on the other hand, if it instead feels like a corner, then the corner model wins the probabilistic competition. Suppose the robot is near a wall and starts to follow it until a corner is reached. In our first version, the corner model was designed too independently of the wall model: the validity zone of the c_{corner} map was too small and seldom visited by the robot as it passed the corner using the "follow wall right" behaviour, defined by c_{wall}. The robot would then miss the first corner and stop at another one. This shows that the decomposition of the task gives independent subtasks only as a first approximation. We solved the problem by modifying the "corner"

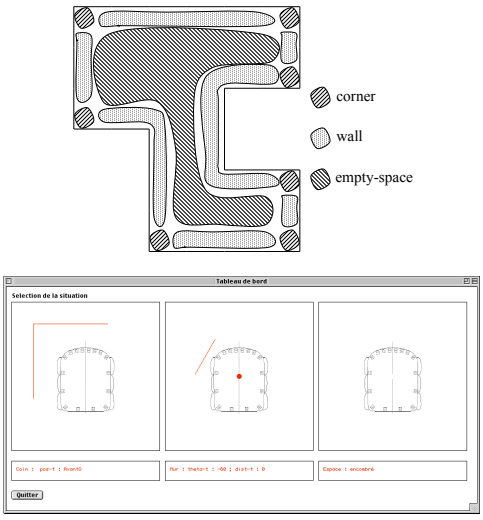

Fig. 5. 2D projection of the estimated "validity zones" of the maps c_{wall}, c_{corner} and $c_{emptyspace}$. The bottom part of the figure is a screenshot of the localization module of the abstract map: it shows the "comparison" and competition between the underlying models. The winner is marked by the central dot: in this case, the robot was near a wall.

model, so that it would recognize a corner on a typical "follow wall right" trajectory.

7 Conclusion

We have presented the Bayesian Map formalism: it is a generalization of most probabilistic models of space found in the literature. Indeed, it drops the usual constraints on the choice of decomposition, forms, or implementation of the probability distributions. We have also presented the abstraction operator, for building hierarchies of Bayesian Maps.

The experiments we presented are of course to be regarded only as "proofs of concept". Their simplicity also served didactic purposes. However, these experiments, in our view, are a successful preliminary step towards applying our formalism. Part of the current work is of course aimed at enriching these experiments, in particular with respect to the *scaling up* capacity of the formalism.

Moreover, because each map of the hierarchy is a full probabilistic model, it is potentially very rich. Possible computations based on these maps include questions like the prediction question $P(L^{t'} \mid A\ L^t)$, which can form the basis of *planning* processes. Hierarchies of Bayesian Maps can therefore be considered as model-based approaches rather than as purely reactive approaches.

Exploiting such knowledge by integrating a planning process in our Bayesian Map formalism is also part of the ongoing work.

Acknowledgements

This work has been supported by the BIBA European project (IST-2001-32115). The authors wish to thank Emmanuel Mazer, Alain Berthoz and Panagiota Panagiotaki for many discussions on this work, and Carla Koike for invaluable comments on drafts of this chapter.

References

H. Attias. Planning by probabilistic inference. In *Ninth International Workshop on Artificial Intelligence and Statistics Proceedings*, 2003.

A. Berthoz. *The Brain's Sense of Movement*. Harvard University Press, Cambridge, MA, 2000.

P. Bessière, E. Dedieu, O. Lebeltel, E. Mazer, and K. Mekhnacha. Interprétation vs. description I : Proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs. *Intellectica*, 26-27:257–311, 1998.

C. Boutilier, T. Dean, and S. Hanks. Decision theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 10:1–94, 1999.

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

J. Diard. *La carte bayésienne – Un modèle probabiliste hiérarchique pour la navigation en robotique mobile*. Thèse de doctorat, Institut National Polytechnique de Grenoble, Grenoble, France, Janvier 2003.

J. Diard, P. Bessière, and E. Mazer. A survey of probabilistic models, using the bayesian programming methodology as a unifying framework. In *The Second Int. Conf. on Computational Intelligence, Robotics and Autonomous Systems (CIRAS)*, Singapore, December 2003.

J. Diard, P. Bessière, and E. Mazer. Merging probabilistic models of navigation: the bayesian map and the superposition operator. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS05)*, pages 668–673, 2005.

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

M. Franz and H. Mallot. Biomimetic robot navigation. *Robotics and Autonomous Systems*, 30:133–153, 2000.

E. Gilet, J. Diard, and P. Bessière. Incremental learning of bayesian sensorimotor models: from low-level behaviors to the large-scale structure of the

environment. In *International Conference Spatial Cognition 06 (submitted to)*, 2006.

K. Gothard, W. Skaggs, K. Moore, and B. McNaughton. Binding of hippocampal CA1 neural activity to multiple reference frames in a landmark-based navigation task. *Journal of Neuroscience*, 16(2):823–835, 1996.

T. Hartley and N. Burgess. *Encyclopedia of Cognitive Science (in press)*, chapter Models of spatial cognition, page 369. Macmillan, 2002.

M. Hauskrecht, N. Meuleau, L. P. Kaelbling, T. Dean, and C. Boutilier. Hierarchical solution of Markov decision processes using macro-actions. In G. F. Cooper and S. Moral, editors, *Proceedings of the 14th Conf. on Uncertainty in Artificial Intelligence (UAI-98)*, pages 220–229, San Francisco, July, 24–26 1998. Morgan Kaufmann.

L. F. Jacobs. The evolution of the cognitive map. *Brain, behavior and evolution*, 62:128–139, 2003.

L. F. Jacobs and F. Schenk. Unpacking the cognitive map: the parallel map theory of hippocampal function. *Psychological Review*, 110(2):285–315, 2003.

E. T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, June 2003.

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

L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. on Robotics and Automation*, 12(4):566–580, 1996.

B. J. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119 (1–2):191–233, 2000.

B. J. Kuipers. A hierarchy of qualitative representations for space. In *Working Papers of the Tenth International Workshop on Qualitative Reasoning (QR-96)*, 1996.

D. Lambrinos, R. Möller, T. Labhart, R. Pfeifer, and R. Wehner. A mobile robot employing insect strategies for navigation. *Robotics and Autonomous Systems (Special issue on Biomimetic Robotics)*, 30:39–64, 2000.

T. Lane and L. P. Kaelbling. Toward hierarchical decomposition for planning in uncertain environments. In *Proceedings of the 2001 IJCAI Workshop on Planning under Uncertainty and Incomplete Information*, Seattle, WA, August 2001. AAAI Press.

T. Lane and L. P. Kaelbling. Nearly deterministic abstractions of markov decision processes. In *18th Nat. Conf. on Artificial Intelligence*, 2002.

J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, 1991.

O. Lebeltel, P. Bessière, J. Diard, and E. Mazer. Bayesian robot programming. *Autonomous Robots (in press)*, 16(1), 2004.

J. Leonard, H. Durrant-Whyte, and I. Cox. Dynamic map-building for an autonomous mobile robot. *The International Journal of Robotics Research*, 11(4):286–298, 1992.

J. J. Leonard and H. F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(3): 376–382, June 1991.

T. S. Levitt and D. T. Lawton. Qualitative navigation for mobile robots. *Artificial Intelligence*, 44(3):305–360, 1990.

E. Mazer, J.-M. Ahuactzin, and P. Bessière. The Ariadne's clew algorithm. *Journal of Artificial Intelligence Research (JAIR)*, 9:295–31, 1998.

K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. Ph.D. thesis, University of California, Berkeley, Berkeley, CA, July 2002.

J. Pineau and S. Thrun. An integrated approach to hierarchy and abstraction for POMDPs. Technical Report CMU-RI-TR-02-21, Carnegie Mellon University, August 2002.

L. R. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*, chapter Theory and implementation of Hidden Markov Models, pages 321–389. Prentice Hall, Englewood Cliffs, New Jersey, 1993.

A. D. Redish and D. S. Touretzky. Cognitive maps beyond the hippocampus. *Hippocampus*, 7(1):15–35, 1997.

S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 11(2):305–345, February 1999.

É. Simonin, J. Diard, and P. Bessière. Learning Bayesian models of sensorimotor interaction: from random exploration toward the discovery of new behaviors. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS05)*, pages 1226–1231, 2005.

P. Smyth, D. Heckerman, and M. I. Jordan. Probabilistic independence networks for hidden markov probability models. *Neural Computation*, 9(2): 227–269, 1997.

R. Stackman and A. Herbert. Rats with lesions of the vestibular system require a visual landmark for spatial navigation. *Behavioural Brain Research*, 128:27–40, 2002.

R. Stackman, A. Clark, and J. Taube. Hippocampal representations require vestibular input. *Hippocampus*, 12:291–303, 2002.

P. Svestka and M. Overmars. Probabilistic path planning. In J.-P. Laumond, editor, *Lecture Notes in Control and Information Sciences 229*. Springer, 1998.

S. Thrun. Probabilistic algorithms in robotics. *AI Magazine*, 21(4):93–109, 2000.

S. Thrun. Robotic mapping: A survey. Technical Report CMU-CS-02-111, Carnegie Mellon University, February 2002.

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

S. Thrun, A. Bücken, W. Burgard, D. Fox, T. Fröhlinghaus, D. Hennig, T. Hofmann, M. Krell, and T. Schmidt. Map learning and high-speed navigation in RHINO. In D. Kortenkamp, R. Bonasso, and R. Murphy, editors, *AI-based Mobile Robots: Case Studies of Successful Robot Systems*, pages 580–586. MIT Press, 1998.

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

S. Thrun, M. Bennewitz, W. Burgard, A. B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. MINERVA: A tour-guide robot that learns. In W. Burgard, T. Christaller, and A. B. Cremers, editors, *Proceedings of the 23rd Annual German Conference on Advances in Artificial Intelligence (KI-99)*, volume 1701 of *LNAI*, pages 14–26, Berlin, September, 13–15 1999b. Springer.

E. Tolman. Cognitive maps in rats and men. *Psychol. Rev.*, 55:189–208, 1948.

N. Tomatis, I. Nourbakhsh, K. Arras, and R. Siegwart. A hybrid approach for robust and precise mobile robot navigation with compact environment modeling. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation (ICRA'01)*, pages 1111–1116, 2001.

N. Tomatis, I. Nourbakhsh, and R. Siegwart. Hybrid simultaneous localization and map building: a natural integration of topological and metric. *Robotics and Autonomous Systems*, 44:3–14, 2003.

D. S. Touretzky and A. D. Redish. A theory of rodent navigation based on interacting representations of space. *Hippocampus*, 6:247–270, 1996.

O. Trullier, S. Wiener, A. Berthoz, and J.-A. Meyer. Biologically-based artificial navigation systems: Review and prospects. *Progress in Neurobiology*, 51:483–544, 1997.

A. C. Victorino and P. Rives. An hybrid representation well-adapted to the exploration of large scale indoors environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA04)*, pages 2930–2935, New Orleans, LA, USA, 2004.

R. F. Wang and E. S. Spelke. Updating egocentric representations in human navigation. *Cognition*, pages 215–250, 2000.

R. F. Wang and E. S. Spelke. Human spatial representation: insights from animals. *TRENDS in Cognitive Science*, 6(9):376–382, 2002.