Combining probabilistic models of space for mobile robots: the Bayesian Map and the Superposition operator

Julien Diard, Pierre Bessiere, Emmanuel Mazer

HAL Id: inria-00182076

https://inria.hal.science/inria-00182076

Submitted on 24 Oct 2007

Combining probabilistic models of space for mobile robots: the Bayesian Map and the Superposition operator

Julien Diard*, Pierre Bessière, and Emmanuel Mazer
Laboratoire GRAVIR / IMAG – CNRS
INRIA Rhône-Alpes, 655 avenue de l'Europe
38330 Montbonnot Saint Martin FRANCE
Julien.Diard@free.fr

Abstract

This paper deals with the probabilistic modeling of an environment that a robot has to navigate in. We use a method for the probabilistic modeling of space called the Bayesian Map *formalism. This formalism allows incremental building of models: we define the* Superposition *operator, which is a formally well-defined operator. We present first a syntactic version of this operator, and second, a version where the previously obtained model is refined and enriched by experimental learning. In the resulting superposed map, locations are the conjunction of underlying possible locations, which allows for more precise localization and more complex tasks. A theoretical example validates the concept, and hints at its usefulness for realistic robotic scenarios.*

1 Introduction and related work

In service robotics, modeling the environment that a robot has to face is a crucial problem. Whether it is a robotic personal assistant operating indoors (e.g. in a hospital, factory or airport) or a robotic carlike vehicle operating outdoors (e.g. in a parking lot or campus), any service robot needs internal models (maps) for solving almost all navigation tasks. This problem has received a lot of attention in the community, the most promising approaches relying on the probability calculus, especially for its capacity to handle incomplete models and uncertain information. These approaches include – but are far from limited to – Kalman Filters [13], Markov Localization models [20], (partially or fully) observable Markov Decision Processes [3, 8], and Hidden Markov Models [16]. For references that present several of them at once, giving unifying pictures, see [2, 6, 14, 17, 18]. We will

here assume that the reader has some familiarity with these approaches.

In this domain of probabilistic modeling for robotics, hierarchical solutions are currently flourishing – while still representing a very small part of the literature. The more active domain in this regard is decision theoretic planning: one can find variants of MDPs that accomodate hierarchies or that select automatically the partition of the state-space (see for instance [7, 10], or browse through the references in [15]). More exceptionnally, one can find hierarchical POMDPs, as in [15]. Some hierarchical approaches outside of the MDP community include Hierarchical HMMs and their variants (see [14] and references therein), which, unfortunately, rely on the notion of final state of the automata, which is inconvenient in a purely probabilistic approach, as we are pursuing here. Another class of approaches that rely on deterministic notions are based on the extraction of a graph from a probabilistic model, like for example a Markov Localization model [19], or a MDP [11].

However, the main philosophy used by the hierarchical approaches is to try to extract, from a very complex but intractable model, a hierarchy of smaller models (structural decomposition, see [15]). Automatically selecting the right decomposition is of course a very difficult problem. Moreover, even obtaining in the first place the initial, complex model, is still a difficult challenge in the general case [9].

We pursue here an alternate route, investigating how, starting from a set of simple probabilistic models, one can combine them for building more complex models. We have developed a new formalism for building models of the space in which a robot has to navigate (the Bayesian Map model). We will briefly introduce this model here. This model is particularly well suited to the definition of operators in order to combine maps together. For instance, we have defined the Abstraction operator, which combines Bayesian Maps in a hierarchical manner [5]. This operator is formally well-defined: given a set of Bayesian Maps,
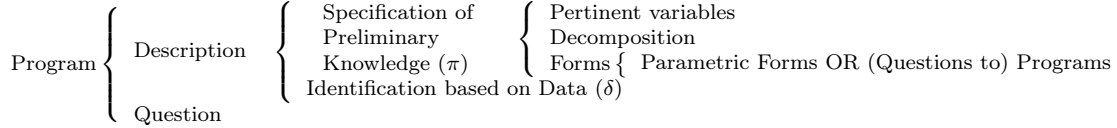
Figure 1: Structure of a Bayesian Robotic Program.

this operator outputs a new Bayesian Map which is a hierarchical composition of the given maps.

The current paper enriches the previous work, by defining another well-defined formal operator: the Superposition operator. It combines together several maps that describe the same geographical space. In this paper, we present the binary version of this operator. Intuitively, it puts together two Bayesian Maps by superposing each map's set of possible locations. Therefore the robot localizes itself in the two models simultaneously, giving it a richer "vocabulary" for describing its environment. Building complex models out of simpler ones is often needed, for example when different sensory modalities are used by a robot, and have to be mixed.

There are two variants of this operator, depending on whether the actions indicated by the combined maps can be combined or not. In this paper, we present the variant where we suppose that we do not know how to combine actions defined in the underlying maps: the robot can use a behavior defined either in the first map, or the second map, but is not able to apply both simultaneously (or, in other words, applying both can give unforeseen results). The other variant supposes the contrary, *i.e.* the robot knows a method for combining behaviors coming from each map in a meaningful manner (see [4]).

We will first present the simpler case where the combination does not add knowledge to the obtained Bayesian Map, with respect to the initial maps. We then proceed to a more interesting and complicated scenario, where the combined map is enriched by further knowledge, so as to investigate the interplay between the combined models. This supplementary knowledge takes two forms: in the first one, some parametric forms are tuned by experimental learning, and in the second one, the probabilistic conditional independency hypotheses are relaxed.

We exemplify our Superposition operator by a theoretical example, whose simplicity is tailored toward pedagogical purposes, while still illustrating the interest of our approach.

2 Bayesian Robot Programming

The work we present here is based on BRP, a Bayesian Robot Programming methodology. We briefly summarize it here, but still invite the interested reader to refer to [12] for all the details about this methodology.

In the BRP formalism, a bayesian robotic program is a structure (see Figure 1) made of two components.

The first is a *declarative* component, where the user defines a **description**. The purpose of a description is to specify a method to compute a joint distribution over a set of relevant variables $\{X_1, X_2, \ldots, X_n\}$, given a set of experimental data δ and preliminary knowledge π. This joint distribution is denoted $P(X_1 \ X_2 \ \ldots \ X_n \mid \delta \ \pi)$. To specify this distribution, the programmer first lists the pertinent variables (and defines their domains), then decomposes the joint distribution as a product of simpler terms (possibly stating conditional independence hypotheses so as to simplify the model and/or the computations), and finally, assigns forms to each term of the selected product (these forms can be parametric forms, or recursive questions to other bayesian programs). If there are free parameters in the parametric forms, they have to be assessed. They can be given by the programmer (*a priori* programming) or computed on the basis of a learning mechanism defined by the programmer and some experimental data δ.

The second component is of a *procedural* nature, and consists of using the previously defined description with a **question**, *i.e.* computing a probability distribution of the form $P(\text{Searched} \mid \text{Known})$. Answering a "question" consists in deciding a value for the variable *Searched* according to $P(Searched \mid Known)$. Different decision policies are possible, in our robotic experiments we usually choose to draw a value at random according to that distribution. It is well known that general Bayesian inference is a very difficult problem, which may be practically intractable. But, as this paper is mainly concerned with modeling issues, we will assume that the inference problems are solved and implemented in an efficient manner by the programmer [1].

3 Bayesian Maps

In our previous work, we have applied the BRP method to the problem of localization and mapping

---

[1]The inference engine we use to tackle these problems has been described elsewhere [2].

$$
\text{Program}
\begin{cases}
\text{Description}
\begin{cases}
\text{Specification}
\begin{cases}
\begin{array}{l}
\text{Pertinent variables} \\
\quad P \ : \text{perception variable} \\
\quad L_t \ : \text{location variable at time } t \\
\quad L_{t'} \ : \text{location var. at time } t', \ t' > t \\
\quad A \ : \text{action variable}
\end{array} \\
\text{Decomposition: any} \\
\text{Parametric forms: any}
\end{cases} \\
\text{Identification: any}
\end{cases} \\
\text{Questions (required):} \\
\quad P(A^i \mid X), A^i \subseteq A, X \subseteq \left(\{P, L_t, L_{t'}, A\} \setminus A^i\right) \ : \\
\quad \text{elementary behaviors} \\
\text{Questions (guides):} \\
\quad P(L_t \mid P) \ : \text{localization} \\
\quad P(L_{t'} \mid A \ L_t) \ : \text{prediction} \\
\quad P(A \mid L_t \ L_{t'}) \ : \text{control}
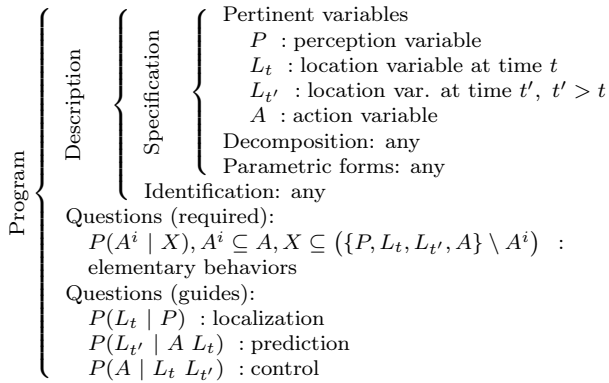\end{cases}
$$

Figure 2: The Bayesian Map model definition expressed in the BRP formalism.

for a mobile robot [4, 5]. We have developed a new formalism for building models of the space in which a robot navigates, by constraining the general BRP framework. This formalism is called the Bayesian Map model and is defined below.

A Bayesian Map c is a description that defines a joint distribution $P(P \ L_t \ L_{t'} \ A)$, 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 on this variable). For simplicity, we will assume that all these variables are discrete variables.

The choice of decomposition is not constrained. For example, if we assume the variables P, L_t, $L_{t'}$ and A atomic, there are already 1015 decompositions to choose from (see Chapter 7 of [4]). See a recent paper for an example of how this leverage can lead to interesting new models [1]. Finally, the definition of forms and the learning mechanism (if any) are not constrained, either.

For a Bayesian Map to be useable in practice, we need the description to be rich enough to generate *behaviors*. We call *elementary behavior* any question of the form $P(A^i \mid X)$, where A^i is a subset of A, and X a subset of the other variables of the map (*i.e.*, not in A^i). A behavior can be not elementary, for example if it is a sequence of elementary behaviors, or, in more general terms, if it is based on elementary behaviors and some other knowledge (which need not be expressed in terms of maps).

For a Bayesian Map to be interesting, we will also require that it generates *several* behaviors – otherwise, defining just a single behavior instead of a map is enough. Such a map is therefore a ressource, 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 behaviors, is to check if 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 have to be informative, in the sense that their entropy is "far enough" of 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 means to generate behaviors. Checking that the answers to these questions are informative is a first step to evaluate the quality of a Bayesian Map with respect to solving a given task.

Figure 2 is a summary of the definition of the Bayesian Map formalism.

4 Superposing Bayesian Maps

Having defined the Bayesian Map concept, we now turn to defining the Superposition operator for putting Bayesian Maps together. We develop here the binary version of this operator: let c^1 and c^2 be two Bayesian Maps, that deal respectively with variables P^1, L_t^1, $L_{t'}^1$, A^1 and P^2, L_t^2, $L_{t'}^2$, A^2, and that can compute in a satisfactory manner the three questions of localization $P(L_t^1 \mid P^1)$, of prediction $P(L_{t'}^1 \mid A^1 \ L_t^1)$, and of control $P(A^1 \mid L_t^1 \ L_{t'}^1)$ (respectively $P(L_t^2 \mid P^2)$, $P(L_{t'}^2 \mid A^2 \ L_t^2)$ and $P(A^2 \mid L_t^2 \ L_{t'}^2)$).

Let us assume that c^1 and c^2 cover approximately the *same physical space*. In other words, the two maps both describe the same part of the environment of the robot. But they do so in different terms, or different vocabularies (L_t^1 and L_t^2). Finally, they can also select actions, in the domains of variables A^1 et A^2 respectively, in order to realize behaviors.

We define the *superposed* Bayesian Map c, obtained by applying the superposition operator to the given underlying maps c^1 and c^2. The obtained map deals with variables P, L_t, $L_{t'}$ and A.

As previously stated, there are two variants of the superposition operator, with different choices for the action variable A in the superposed map. In this paper, we suppose that we do not know how to mix the actions coming from c^1 and c^2: the robot can either apply a behavior defined from c^1 *or* apply a behavior from c^2.

As we do not suppose that combining behaviors from c^1 and c^2 is possible, we construct the action variable A of the superposed map as follows: the do-

$$\text{Program}\begin{cases}\text{Description}\begin{cases}\text{Specification}\begin{cases}\begin{array}{l}\text{Pertinent variables}\\ \quad P = P^1 \wedge P^2\\ \quad L_t = L_t^1 \wedge L_t^2\\ \quad L_{t'} = L_{t'}^1 \wedge L_{t'}^2\\ \quad A = A^1 \oplus A^2 : \mathcal{D}_A = \mathcal{D}_{A^1} \cup \mathcal{D}_{A^2}\\ \text{Decomposition}\\ \quad P(P\ L_t\ L_{t'}\ A) = P(L_t^1)P(L_t^2)P(P^1\mid L_t^1)P(P^2\mid L_t^2)P(A)P(L_{t'}^1\mid A\ L_t^1)P(L_{t'}^2\mid A\ L_t^2)\\ \text{Parametric forms}\\ \quad \begin{array}{lcl} P(L_t^1) & = & \mathbf{U}(L_t^1)\\ P(L_t^2) & = & \mathbf{U}(L_t^2)\\ P(P^1\mid L_t^1) & = & P(P^1\mid L_t^1\ c^1)\\ P(P^2\mid L_t^2) & = & P(P^2\mid L_t^2\ c^2)\\ P(A) & = & \mathbf{U}(A)\\ P(L_{t'}^1\mid [A=a]\ L_t^1) & = & \begin{cases} P(L_{t'}^1\mid [A^1=a]\ L_t^1\ c^1) \text{ if } a \in \mathcal{D}_{A^1}\\ \mathbf{U}(L_{t'}^1) \text{ otherwise}\end{cases}\\ P(L_{t'}^2\mid [A=a]\ L_t^2) & = & \begin{cases} P(L_{t'}^2\mid [A^2=a]\ L_t^2\ c^2) \text{ if } a \in \mathcal{D}_{A^2}\\ \mathbf{U}(L_{t'}^2) \text{ otherwise}\end{cases}\end{array}\end{array}\end{cases}\\ \text{Identification: no free parameters to identify}\end{cases}\\ \text{Questions:}\\ \quad\begin{array}{lcl} P(L_t\mid P) & = & \frac{1}{Z_1}P(P^1\mid L_t^1)P(P^2\mid L_t^2)\\ P(L_{t'}\mid A\ L_t) & = & \frac{1}{Z_2}P(L_{t'}^1\mid A\ L_t^1)P(L_{t'}^2\mid A\ L_t^2)\\ P(A\mid L_t\ L_{t'}) & = & \frac{1}{Z_3}P(L_{t'}^1\mid A\ L_t^1)P(L_{t'}^2\mid A\ L_t^2). \end{array}\end{cases}$$
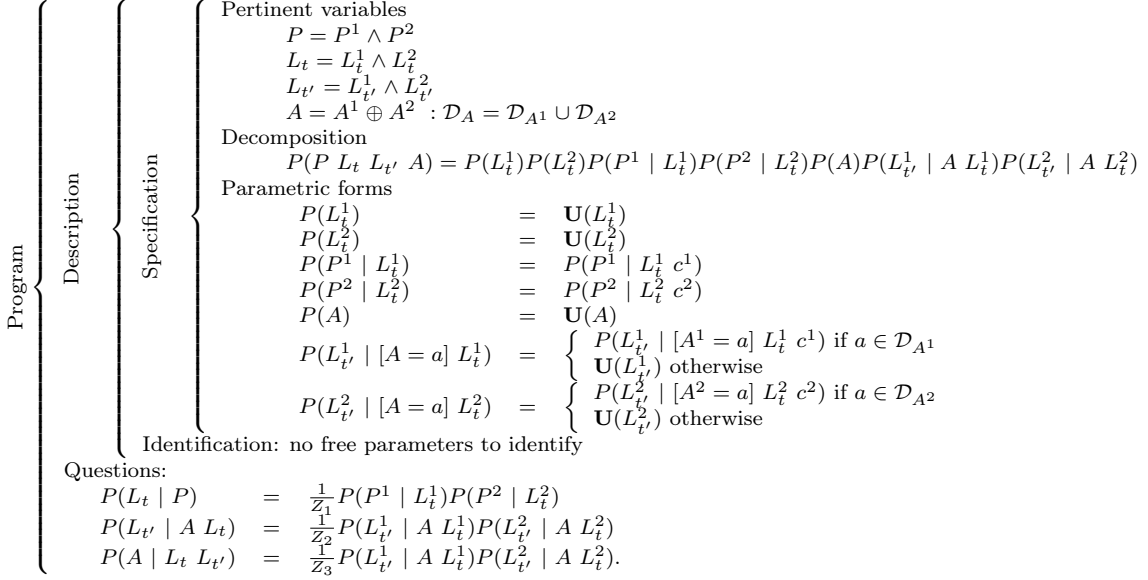
Figure 3: The superposition operator in the $A = A^1 \oplus A^2$ variant, no information added. The $\mathbf{U}$ notation refers to the uniform probability distribution.

main of A is the union of the domains of A^1 and A^2 ($\mathcal{D}_A = \mathcal{D}_{A^1} \cup \mathcal{D}_{A^2}$). We note $A = A^1 \oplus A^2$.

4.1 $\quad A = A^1 \oplus A^2$, no information added

We choose here to superpose c^1 and c^2 without adding supplementary information, *i.e.* we create the superposed map c so that it only contains parametric forms taken from c^1 or c^2, or uniform distributions. We will see Section 4.2 how to use this simple map as a basis for further enriching it through a learning phase.

This first version of the superposition operator is defined Figure 3, and is commented in the rest of this section.

As previously mentioned, the new action variable A is defined as $A = A^1 \oplus A^2$. As for the other variables, they are just the list of the variables that appear in the underlying maps: $P = P^1 \wedge P^2$, $L_t = L_t^1 \wedge L_t^2$, and $L_{t'} = L_{t'}^1 \wedge L_{t'}^2$.

We now define the joint distribution $P(P\ L_t\ L_{t'}\ A)$ for the superposed map by choosing the following decomposition:

$$\begin{aligned} &P(P\ L_t\ L_{t'}\ A)\\ =\ & P(P^1\ P^2\ L_t^1\ L_t^2\ L_{t'}^1\ L_{t'}^2\ A)\\ =\ & P(L_t^1)P(L_t^2)P(P^1\mid L_t^1)P(P^2\mid L_t^2) \quad (1)\\ & P(A)P(L_{t'}^1\mid A\ L_t^1)P(L_{t'}^2\mid A\ L_t^2). \end{aligned}$$

This decomposition particularizes the direct sensor models $P(P^1\mid L_t^1)$ and $P(P^2\mid L_t^2)$, and the transition models $P(L_{t'}^1\mid A\ L_t^1)$ and $P(L_{t'}^2\mid A\ L_t^2)$, in the same manner as Markov Localization models [20].

We now turn to the definition of parametric forms for each term appearing in Equation 1. $P(L_t^1)$, $P(L_t^2)$ and $P(A)$ are defined as uniform distributions, because we choose not to express any preference neither for any particular position in space, nor for any action to execute. The direct sensor models, $P(P^1\mid L_t^1)$ and $P(P^2\mid L_t^2)$, are respectively extracted from the underlying maps c^1 and c^2. This is done by defining their parametric forms as probabilistic questions: $P(P^1\mid L_t^1) = P(P^1\mid L_t^1\ c^1)$. However, this cannot so easily be done for the last two terms, $P(L_{t'}^1\mid A\ L_t^1)$ and $P(L_{t'}^2\mid A\ L_t^2)$, because the domain for A is bigger than the domains of A^1 and A^2. So we define these terms in two different manners, depending on the value a taken by A. Consider the term $P(L_{t'}^1\mid A\ L_t^1)$. When $[A=a]$ and $a \in A^1$, then the term $P(L_{t'}^1\mid [A=a]\ L_t^1)$ can actually be extracted from map c^1 by a question to this map. When $[A=a]$ and $a \notin A^1$, then the term $P(L_{t'}^1\mid [A=a]\ L_t^1)$ is set to a uniform distribution (the map c^1 does not encode how applying action a will affect the variable L_t). The definition of $P(L_{t'}^2\mid A\ L_t^2)$ follows a symmetrical argument.

We now want to prove that the superposed map is indeed a Bayesian Map. It obviously has one perception variable, one action variable, and the construction of L_t and $L_{t'}$ ensures that they have the same domains, since by hypothesis this was true for L_t^1 and $L_{t'}^1$ on the one hand, and for L_t^2 and $L_{t'}^2$ on the other hand. In order to conclude this proof, we have to examine the three questions of localization, prediction and control. We can easily show (derivation omitted)

that the answers to these questions only involve terms extracted from the underlying maps c^1 and c^2:

$$
\begin{aligned}
P(L_t \mid P) &\propto P(P^1 \mid L_t^1)P(P^2 \mid L_t^2), \\
P(L_{t'} \mid A\ L_t) &\propto P(L_{t'}^1 \mid A\ L_t^1)P(L_{t'}^2 \mid A\ L_t^2), \\
P(A \mid L_t\ L_{t'}) &\propto P(L_{t'}^1 \mid A\ L_t^1)P(L_{t'}^2 \mid A\ L_t^2).
\end{aligned}
$$

By hypothesis, c^1 and c^2 are Bayesian Maps, which means that the terms used in these computations are informative. Therefore the results of these questions in the superposed maps are also going to be informative. This concludes the proof that the superposed map we have defined is indeed a Bayesian Map. The superposition operator therefore is an internal operator in the space of all possible Bayesian Maps, and the map it outputs can safely be used as an input to another operator.

4.2 $A = A^1 \oplus A^2$, with learning

In this Section, we examine how a map obtained by the superposition operator can be enriched by experimental learning. We begin by the simple case of parametric form identification, then turn to relaxing some simplifying assumptions made by the decomposition (Equation 1).

We have seen that the terms $P(L_{t'}^1 \mid A\ L_t^1)$ and $P(L_{t'}^2 \mid A\ L_t^2)$ were partly defined by questions to the underlying maps c^1 and c^2, and partly by uniform distributions. However, recall that c^1 and c^2 actually cover the same space, but represent it using different location variables. In a pratical scenario, it is unlikely that applying a behavior from map c^1 will not change the location of the robot in map c^2. When the robot applies a behavior from c^1, the values for the location variable in c^2 can be recorded, and used for computing a learned histogram. Of course, the situation is symmetrical: the term $P(L_{t'}^1 \mid A\ L_t^1)$, for values of A that come from map c^2 can also be identified experimentally in the same manner.

We now understand why, in practice, it may be interesting to relax several simplifying assumptions previously made, in order to identify the particular interplay between maps c^1 and c^2. Let us focus on the transition terms $P(L_{t'}^1 \mid A\ L_t^1)$ and $P(L_{t'}^2 \mid A\ L_t^2)$, and their conditional independence hypotheses.

$P(L_{t'}^1 \mid A\ L_t^1)$ states for example that the probability distribution on $L_{t'}^1$ is independent of the knowledge of L_t^2, provided we know A and L_t^1. Relaxing this hypothesis yields the term $P(L_{t'}^1 \mid A\ L_t^1\ L_t^2)$. In a symmetrical manner we obtain $P(L_{t'}^2 \mid A\ L_t^1\ L_t^2)$. These terms break the main independence between the two maps c^1 and c^2: it is now possible to identify experimentally how the position given by one map influences the prediction of the location in the other map. These terms allow for more precise models, and

the reduction of their uncertainties. It is possible to go one step further, by noticing that they are mono-dimensional terms (*i.e.* they only have one variable on their left hand side), that are projections of the joint space $L_{t'}^1 \wedge L_{t'}^2$. We can revert to that joint space by relaxing the conditional independence hypothesis between $L_{t'}^1$ and $L_{t'}^2$. The two terms become a single term: $P(L_{t'}^1\ L_{t'}^2 \mid A\ L_t^1\ L_t^2)$.

If we follow a similar reasoning for the sensor models $P(P^1 \mid L_t^1)$ and $P(P^2 \mid L_t^2)$, we obtain the term $P(P^1\ P^2 \mid L_t^1\ L_t^2)$. Therefore, the final decomposition we choose for this version of the superposition operator is:

$$
\begin{aligned}
&P(P\ L_t\ L_{t'}\ A) \\
&= P(L_t^1)P(L_t^2)P(P^1\ P^2 \mid L_t^1\ L_t^2) \qquad (2) \\
&\quad P(A)P(L_{t'}^1\ L_{t'}^2 \mid A\ L_t^1\ L_t^2).
\end{aligned}
$$

We associate to the terms of Equation 2 the following parametric forms. $P(L_t^1)$, $P(L_t^2)$ and $P(A)$ are defined as uniform distributions. As we have assumed that all variables would be discrete, $P(P^1\ P^2 \mid L_t^1\ L_t^2)$ and $P(L_{t'}^1\ L_{t'}^2 \mid A\ L_t^1\ L_t^2)$ can be defined as learned histograms. This concludes the definition of the superposition operator in the variant $A = A^1 \oplus A^2$, with learning.

Let us summarize the method for defining such a superposed map: 1) build two bayesian maps c^1 and c^2; 2) superpose c^1 and c^2 to obtain $c_{without}$, using the operator that adds no information; 3) use $c_{without}$ to navigate in the environment and collect experimental data Δ; 4) superpose c^1 and c^2 to obtain c, using the operator that integrates learning; 5) compute the learned histograms of c using data Δ.

4.3 Example

We now develop an example of the application of the superposition operator. We choose a formal and very simplistic example, in order to illustrate clearly our operator and method.

4.3.1 Bayesian Maps c^1 and c^2

We first describe the two Bayesian Maps c^1 and c^2. Both are based on computing potentials and using their gradients to navigate. This is typically the case for example, when a robot is equipped with an array of light sensors, that are used to measure both a light intensity (the potential value) and an angle toward the light source (the gradient direction). This is also the case to a lesser extent for proximity sensors, for example assuming an obstacle avoidance task: the proximity sensors help compute a distance to the obstacle (the potential value) and an angle towards it
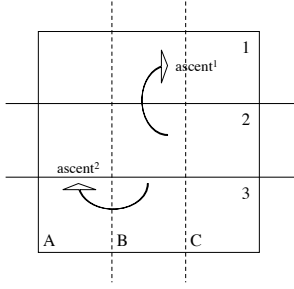
Figure 4: Superposition of maps c^1 and c^2. The zones 1, 2 et 3, A, B et C actually create nine zones.

(the gradient direction). We will use the light sensing robot scenario as a basis for this example.

The perception variable P^1 of map c^1 is $P^1 = \vec{s}$, where $\vec{s}$ is the vector of light sensors readings. The location variable at time t, L^1_t, is the conjunction of the potential and gradient direction variables: $L^1_t = Lum_t \times \alpha_t$, where Lum_t and α_t are the light intensity value and the angle of the light source at time t. By definition of the Bayesian Map model, this also defines the location variable at time t', $L^1_{t'}$, since L^1_t and $L^1_{t'}$ must have the same domains: $L^1_{t'} = Lum_{t'} \times \alpha_{t'}$, with the same variables instantiated at time t'. Finally, the action variable of the map is a set of basic capabilities that allow the robot to ascent the gradient, or descent it: $A^1 = \{ascent^1, descent^1\}$.

For simplicity, we will assume in the following that the potential variable only has three values, so that the environment is divided in three zones (labeled "1", "2" and "3"). We will also assume that the equipotential lines that define these zones are straight lines. Again, for simplicity of the presentation, we assume that the actions have almost guaranteed effects: if the robot is in zone 2, applying $ascent^1$ always brings the robot to zone 1. If it is in zone 2, applying $descent^1$ always brings it to zone 3. This means, for example, that the probability distribution of the possible next locations given that the action is $ascent^1$, and given that the starting point is zone 2, is a Dirac distribution "centred" on zone 1; we note this $P(L^1_{t'} \mid [A = ascent^1] [L^1_t = 2]) = \delta_1(L^1_{t'})$.

This Bayesian Map therefore defines the joint distribution $P(\vec{s}\ Lum_t\ \alpha_t\ Lum_{t'}\ \alpha_{t'}\ rot\ trans \mid c^1)$. We will asssume it can be used for defining several behaviors, like going back to the light source, or going to hide in shadows, for example.

We suppose that we also have the Bayesian Map c^2, defined in a similar fashion, and based on some other gradient (which defines three zones labeled "A", "B" and "C", see Figure 4).

4.3.2 Superposition of c^1 and c^2, without learning

We now turn toward applying the superposition operator on c^1 and c^2. In this example, we assume that the equipotential lines defining the zones of maps c^1 and c^2 are perpendicular (see Figure 4). Intuitively, this example will show that superposing these two maps, that only have three zones to consider each, enables the robot to create a model which has nine zones. Moreover, experimental learning will give the robot a "good" model of how these nine zones relate.

We apply the superposition operator: we first construct the new action variable: $A = A^1 \oplus A^2 = \{ascent^1, ascent^2, descent^1, descent^2\}$. We also obtain the obtain the superposed map as defined Figure 3. For the rest of this example, we will focus on the transition terms: $P(L^1_{t'} \mid A^1\ L^1_t)$ for map c^1, $P(L^2_{t'} \mid A^2\ L^2_t)$ for map c^2, and, in the obtained map c, these terms become $P(L^1_{t'} \mid A\ L^1_t)$ and $P(L_{t'} \mid A\ L^2_t)$. Recall that, without learning, these terms are partly defined by questions to the underlying maps, and partly by uniform distributions.

4.3.3 Superposition of c^1 and c^2, with learning

The simplest version of the learning variant is to replace these uniforms by learned histograms. Suppose the robot navigates in the environment Figure 4, collecting experimental data about its position with respect to both maps at the same time. Imagine the robot in zone B, applying the behavior $ascent^2$, and monitoring the values over time of the variable L^1_t. We see Figure 4 that, whatever its position with respect to map c^1, behavior $ascent^2$ will leave it unchanged, because it moves the robot parallel to the equipotential lines of c^1. For example, starting from cell $\langle B, 1 \rangle$, applying $ascent^2$ will always lead the robot to cell $\langle A, 1 \rangle$. The experimental data we gather are of the form $\langle l, ascent^2, l \rangle$, for every value of l in the domain of L^1_t. If we see k such data, computing the histograms for $P(L^1_{t'} \mid [A = ascent^2]\ L^1_t)$, for all values of L^1_t will lead to (with a^2 for $ascent^2$):

$$P([L^1_{t'} = l] \mid [A = a^2] [L^1_t = l]) = \frac{k+1}{k+3}$$

$$\forall j \neq l, P([L^1_{t'} = j] \mid [A = a^2] [L^1_t = l]) = \frac{1}{k+3}.$$

Note that as the data accumulates, these histograms tend toward Dirac distributions, while the "no information added" version had uniform distributions for the same terms. Even this simplistic example shows that there is a lot of meaningful information that can be captured and identified in the superposed map. Let us now go one step further by relaxing the condi-
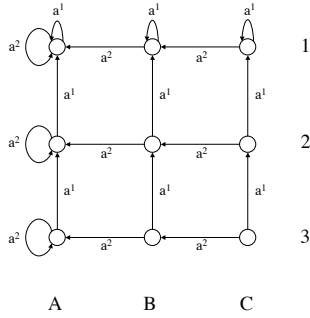
Figure 5: Graph extracted from the learned transition model in the superposed map c.

tional independence hypotheses concerning the transition terms.

Indeed, identifying experimentally the term $P(L^1_{t'} \ L^2_{t'} \mid A \ L^1_t \ L^2_t)$ will also lead to computing histograms that tend towards Dirac distributions. For example, if the robot is in location $\langle 2, C \rangle$ and if it applies $[A = ascent^1]$, the probability that it ends in zone $\langle 1, C \rangle$ is very high. If, from the same location, it applies $[A = ascent^2]$, then the location $\langle 2, B \rangle$ can be identified as the most probable arrival point. We show Figure 5 a graph representing the resulting learned probability distributions: there is a node for every possible cell of the space $L^1_t \wedge L^2_t$, and an edge from node i, labeled by an action a, whenever this action leads with high probability to the node j. We only show on this figure the edges labeled by actions $ascent^1$ and $ascent^2$.

4.3.4 Discussion

Note that the resulting distributions encode meaningful information: the robot has learned a model which deals with an internal space of nine locations, instead of having two models, each with three locations. As the new map covers the same physical space as the underlying maps with more locations, it follows that the superposed map is less prone to the *perceptual aliasing* problem [2]. Moreover, the learned model captures the real topological structure of the space of possible locations (compare Figures 4 and 5). This obtained model permits new reasoning about the environment, like planning, or solving a task like *reach_cell_* $\langle C, 1 \rangle$, which had no means to be encoded in c^1 or in c^2, but is represented in the learned superposed map c.

---

[2]The fact that two different locations in the environment are represented as the same location for the robot; this effect is critical for the localization process.

4.3.5 Simplicity of the example

For pedagogical purposes, we have chosen an example that made several simplifying assumptions. We now discuss more realistic scenarios.

We have assumed in our example that actions had guaranteed results, *i.e.*, when the robot is in one zone and acts, it can predict with certainty its arrival point. We have also assumed that the zone where defined by straight boundaries. These assumptions led to learning histograms that were almost like Dirac distributions. Of course, these assumptions do not hold in a realistic scenario; but the probabilistic framework will be able to capture the resulting uncertainties, and learn histograms that represent the real probabilities of a given situation.

We have also assumed that the equipotential lines that defined the zones where perpendicular. This led to an optimal result, where the nine resulting zones where naturally well structured by the set of available actions (see Figure 5 – in a sense, the two maps superposed can be said to be "informationally" orthogonal). Again, in a more realistic scenario, this does not hold. However, the effect of non-perpendicular equipotential lines can be analyzed (see [4]). In the worst extreme case, the two maps we superpose actually use the same gradients: their equipotential lines define exactly the same zones. In this case, superposing the maps will not yield any benefit, as the information given by each map is redundant. However, in this case, an analysis of the learned distributions obtained after superposition will show for example that some actions have the same effects on the location of the robot. For instance, applying $ascent^1$ and $ascent^2$ displace the robot in the same manner, therefore these actions are redundant.

A realistic scenario will likely be somewhere in between these extreme cases: the equipotentials are neither identical, nor perpendicular. They can be at a $45°$ angle, or with varying angles over the environment: sometimes perpendicular, sometimes running parallel. Again, the probabilities learned will numerically reflect the uncertainties that result from the ignorance of the angle between the equipotential lines.

5 Conclusion

We have presented a formal operator for combining Bayesian Maps, called the *Superposition* operator. It permits to put together and reason with different models that deal with the same physical space, which is useful in a fusion of sensor modalities scenario. We have presented a syntactic version of this operator which does not add information in the process of superposition. We have shown how simplifying assump-

tions made by this operator can be relaxed and used in a learning phase. We have shown, on an example, that much meaningful information could be captured in the superposed map, reducing localization ambiguities and allowing more complex tasks to be performed.

The implementation of the examples used in the paper are part of the current work. Moreover, as our Bayesian Map model is a generalization of most probabilistic approaches found in the literature (see [6]), other ongoing works aim at applying our operator to more specific models of the literature, for obtaining Superpositions of Kalman Filters, Superposition of Particle Filters, Superposition of Markov Localization models, etc. or combination thereof.

Obtaining such combination of sensorimotor models is also relevant to biologically inspired models, as it appears that no single metric model can account alone for large scale navigation capacities of animals; the relevance of our approach with respect to biological findings is the last aspect of our current research.

Although we are only at the beginning of this research track, we do believe it is a promising one, and hope it will spark some interest in the community as well.

References

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

[2] P. Bessière, J.-M. Ahuactzin, O. Aycard, D. Bellot, F. Colas, C. Coué, J. Diard, R. Garcia, C. Koike, O. Lebeltel, R. LeHy, O. Malrait, E. Mazer, K. Mekhnacha, C. Pradalier, and A. Spalanzani. Survey: Probabilistic methodology and techniques for artefact conception and development. Technical report rr-4730, INRIA Rhône-Alpes, Montbonnot, France, 2003.

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

[4] 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.

[5] J. Diard, P. Bessière, and E. Mazer. Hierarchies of probabilistic models of space for mobile robots: the bayesian map and the abstraction operator. In *Reasoning with Uncertainty in Robotics (IJCAI'03 Workshop)*, Acapulco, Mexico, August 2003.

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

[7] 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 Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 220–229, San Francisco, July, 24–26 1998. Morgan Kaufmann.

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

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

[10] 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.

[11] T. Lane and L. P. Kaelbling. Nearly deterministic abstractions of markov decision processes. In *Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, 2002.

[12] O. Lebeltel, P. Bessière, J. Diard, and E. Mazer. Bayesian robot programming. *Autonomous Robots (accepted for publication)*, 2003.

[13] 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.

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

[15] 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.

[16] 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.

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

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

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

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