Merging probabilistic models of navigation: the Bayesian Map and the Superposition operator

Julien Diard, Pierre Bessiere, Emmanuel Mazer

2005 IEEE/RSJ International Conference on Intelligent Robots and Systems

Merging probabilistic models of navigation: the Bayesian Map and the Superposition operator*

Julien Diard, Pierre Bessière and Emmanuel Mazer
Laboratoire GRAVIR / IMAG – CNRS, INRIA Rhône-Alpes, France
Julien.Diard@free.fr

Abstract— **This paper deals with the probabilistic modeling of space, in the context of mobile robot navigation. We define a formalism called the *Bayesian Map*, which allows incremental building of models, thanks to the *Superposition* operator, which is a formally well-defined operator. Firstly, we present a syntactic version of this operator, and secondly, a version where the previously obtained model is enriched by experimental learning. In the resulting 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.**

I. Introduction and related work

In the domain of mobile 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 mobile robot needs internal models (maps) for solving 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 [10], Markov Localization models [15], (Partially or Fully) Observable Markov Decision Processes [1], and Hidden Markov Models [13]. 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 select automatically the partition of the state-space (see for instance [5], [7], or browse through the references in [12]). More exceptionnally, one can find hierarchical POMDPs, as in [12]. Some hierarchical approaches outside of the MDP community include Hierarchical HMMs and their variants (see [11] 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 [14], or a MDP [8].

*This work is supported by the BIBA european project (IST-2001-32115).

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 [12]). 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 [6].

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 [3]. This operator is formally well-defined: given a set of Bayesian Maps, 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 (state variable). We consider the case where each map describes the same environment using different state variables. This makes our work different from most approaches to sensor fusion, where usually, each model uses the same state variable. Moreover, we will also consider the effects of actions in each map, and so, go beyond the framework of sensor fusion.

There are two variants of our operator, depending on whether the actions indicated by the combined maps can be combined or not: in this paper, 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 [2]).

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, some probabilistic conditional independence 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.

II. Bayesian Robot Programming

The work we present here is based on BRP, a Bayesian Robot Programming methodology. We summarize it here, but still invite the reader to refer to [9] for all the details.

In the BRP formalism, a bayesian robotic program is a structure (see Fig. 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 $P(Searched \mid Known)$. 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

III. Bayesian Maps

In our previous work, we have applied the BRP method to the problem of localization and mapping for a mobile robot [3], [2]. 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.

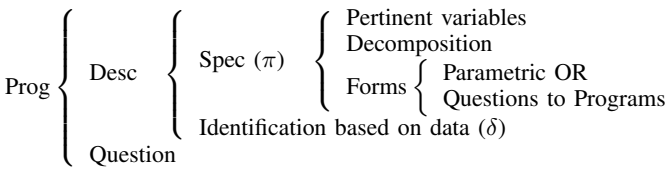

Fig. 1. Structure of a Bayesian Robotic Program.

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 than 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 here that all these variables have finite domains.

The choice of decomposition is not constrained: any probabilistic dependency structure can therefore be chosen here. 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.

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

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

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

IV. Superposition of 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*: the two maps describe the same part of the environment of the robot. But they do so in different terms (the values of L_t^1 and L_t^2). They can also select actions, in the domains of A^1 and 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. 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 in c^1 *or* apply a behavior from c^2. Therefore, we construct the action variable A of the superposed map as follows: the domain 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 $A = A^1 \oplus A^2$.

A. $A = A^1 \oplus A^2$, 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 IV-B 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 Fig. 3, and is commented in the rest of this section.

Program
- Description
  - Specification
    - Pertinent variables:
      - $P = P^1 \wedge P^2$
      - $L_t = L_t^1 \wedge L_t^2; L_{t'} = L_{t'}^1 \wedge L_{t'}^2$
      - $A = A^1 \oplus A^2 : \mathcal{D}_A = \mathcal{D}_{A^1} \cup \mathcal{D}_{A^2}$
    - Decomposition:
      - $P(P\ L_t\ L_{t'}\ A) =$
        $P(L_t^1)P(L_t^2)P(P^1 \mid L_t^1)P(P^2 \mid L_t^2)$
        $P(A)P(L_{t'}^1 \mid A\ L_t^1)P(L_{t'}^2 \mid A\ L_t^2)$
    - Parametric forms:
      - $P(L_t^1) =$ Uniform
      - $P(L_t^2) =$ Uniform
      - $P(P^1 \mid L_t^1) = P(P^1 \mid L_t^1\ c^1)$
      - $P(P^2 \mid L_t^2) = P(P^2 \mid L_t^2\ c^2)$
      - $P(A) =$ Uniform
      - $P(L_{t'}^1 \mid [A = a]\ L_t^1) =$
        - $P(L_{t'}^1 \mid [A^1 = a]\ L_t^1\ c^1)$ if $a \in \mathcal{D}_{A^1}$
        - Uniform otherwise
      - $P(L_{t'}^2 \mid [A = a]\ L_t^2) =$
        - $P(L_{t'}^2 \mid [A^2 = a]\ L_t^2\ c^2)$ if $a \in \mathcal{D}_{A^2}$
        - Uniform otherwise
  - Identification: no free parameters to identify
- Questions:
  - $P(L_t \mid P) \propto P(P^1 \mid L_t^1)P(P^2 \mid L_t^2)$
  - $P(L_{t'} \mid A\ L_t) \propto P(L_{t'}^1 \mid A\ L_t^1)P(L_{t'}^2 \mid A\ L_t^2)$
  - $P(A \mid L_t\ L_{t'}) \propto P(L_{t'}^1 \mid A\ L_t^1)P(L_{t'}^2 \mid A\ L_t^2)$

Fig. 3. The superposition operator in the $A = A^1 \oplus A^2$ variant, no information added.

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 conjunctions 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 c by choosing the following decomposition:

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

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 [15].

We now turn to the definition of the forms for each term appearing in (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, by defining their forms as being 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 $a \in \mathcal{D}_{A^1}$, then the term $P(L_{t'}^1 \mid [A = a]\ L_t^1)$

can be extracted from map c^1 by a question to this map. When $[A = a]$ and $a \notin \mathcal{D}_{A^1}$, *i.e.*, $a \in \mathcal{D}_{A^2}$, then the term $P(L^1_{t'} \mid [A = a]\ L^1_t)$ is set to a uniform distribution: the map c^1 does not encode how applying an action a defined in c^2 will affect the variable L^1_t. The definition of $P(L^2_{t'} \mid A\ L^2_t)$ 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^1_t and $L^1_{t'}$, and for L^2_t and $L^2_{t'}$. In order to conclude this proof, we have to examine the three questions of localization, prediction and control. We can easily show (derivations omitted) that the answers to these questions only involve terms extracted from the underlying maps c^1 and c^2:

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

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 informative. This concludes the proof that the superposed map we have defined is indeed a Bayesian Map.

B. $A = A^1 \oplus A^2$, 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 (1).

We have defined the terms $P(L^1_{t'} \mid A\ L^1_t)$ and $P(L^2_{t'} \mid A\ L^2_t)$ partly by questions to the underlying maps c^1 and c^2, and partly by uniform distributions. However, recall that c^1 and c^2 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 does 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: $P(L^1_{t'} \mid A\ L^1_t)$, for values of A that come from map c^2, can be learned 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 $P(L^1_{t'} \mid A\ L^1_t)$ and $P(L^2_{t'} \mid A\ L^2_t)$, and their conditional independence hypotheses.

$P(L^1_{t'} \mid A\ L^1_t)$ states for example that the probability distribution on $L^1_{t'}$ is independent of the knowledge of L^2_t, provided we know A and L^1_t. Relaxing this hypothesis yields the term $P(L^1_{t'} \mid A\ L^1_t\ L^2_t)$. In a symmetrical manner we obtain $P(L^2_{t'} \mid A\ L^1_t\ L^2_t)$. 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 terms of dimension 1 (*i.e.* they only have one variable on their left hand side), that are projections of the joint space $L^1_{t'} \wedge L^2_{t'}$. We can revert to that joint space by relaxing the conditional independence hypothesis between $L^1_{t'}$ and $L^2_{t'}$. The two terms become a single term: $P(L^1_{t'}\ L^2_{t'} \mid A\ L^1_t\ L^2_t)$.

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

$$\begin{aligned} P(P\ L_t\ L_{t'}\ A) \;=\; & P(L^1_t)P(L^2_t)P(P^1\ P^2 \mid L^1_t\ L^2_t) \\ & P(A)P(L^1_{t'}\ L^2_{t'} \mid A\ L^1_t\ L^2_t). \end{aligned} \tag{2}$$

We define the terms of (2) with the following forms. $P(L^1_t)$, $P(L^2_t)$ and $P(A)$ are uniform distributions. By hypothesis, all variables have finite domains: $P(P^1\ P^2 \mid L^1_t\ L^2_t)$ and $P(L^1_{t'}\ L^2_{t'} \mid A\ L^1_t\ L^2_t)$ 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.

C. Example

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

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 has an array of light sensors, that 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 a wall following task: the proximity sensors help compute a distance to the wall (the potential value) and an angle towards it (the gradient direction). In this example, we use the light sensing robot scenario.

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: $L^1_t = Lum_t \wedge \alpha_t$, 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: $L^1_{t'} = Lum_{t'} \wedge \alpha_{t'}$. Finally, the

action variable is a set of basic capabilities that allow the robot to ascent or descent the gradient: $A^1 = \{ascent^1, descent^1\}$.

For simplicity, we will assume in the following that the potential variable only has three values: 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 "centered" on zone 1; we note this $P(L^1_{t'} \mid [A = ascent^1]\ [L^1_t = 2]) = \boldsymbol{\delta}_1(L^1_{t'})$.

This Bayesian Map therefore defines the joint distribution $P(\vec{s}\ Lum_t\ \alpha_t\ Lum_{t'}\ \alpha_{t'}\ A^1 \mid c^1)$. 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 have another 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 Fig. 4).

2) Superposition of c^1 and c^2, without learning: We now apply the superposition operator on c^1 and c^2. We further assume that the equipotential lines defining the zones of maps c^1 and c^2 are perpendicular (see Fig. 4). Intuitively, this example will show that superposing these two maps, that only have three locations each, enables the robot to create a model which has nine locations. Moreover, experimental learning gives the robot a "good" model of how the nine zones relate.

We apply our operator: the new action variable is: $A = A^1 \oplus A^2 = \{ascent^1, ascent^2, descent^1, descent^2\}$. The rest of the superposed map is obtained automatically, following the definition of Fig. 3.

3) Superposition of c^1 and c^2, with learning: For 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 $P(L^2_{t'} \mid A\ L^2_t)$. Recall that, without learning, these terms are partly defined by questions to the underlying maps, and partly by uniform distributions. The simplest version of the learning variant is to replace these uniforms by learned histograms. Suppose the robot navigates in the environment shown Fig. 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 Fig. 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 thus 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 $ascent^2$):

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

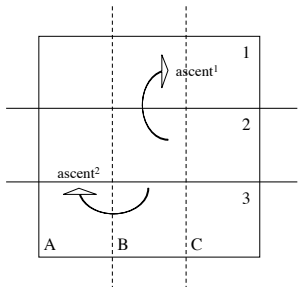

Fig. 4. Superposition of maps c^1 and c^2. The zones 1, 2 et 3, A, B et C actually create nine locations of interest.

Note that as the data accumulates, these histograms tend toward Dirac distributions, whereas the "no information added" version had uniform distributions for the same terms. Even this simple example shows that there is a lot of meaningful information that can be captured and identified in the superposed map. Let us now turn to relaxing the conditional independence hypotheses concerning the transition terms.

Experimentally identifying 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 $\langle C, 2\rangle$ and if it applies $[A = ascent^1]$, the probability that it ends in zone $\langle C, 1\rangle$ is very high. From the same location, if it applies $[A = ascent^2]$, then the location $\langle B, 2\rangle$ can be identified as the most probable arrival point. We show Fig. 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, if this action leads with high probability to the node j.

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 [1]. Moreover, the learned model captures the real topological structure of the space of possible locations (compare Fig. 4 and Fig. 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.

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 that actions had guaranteed results, *i.e.*, when the robot is in one zone and acts, it can predict with

[1]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.

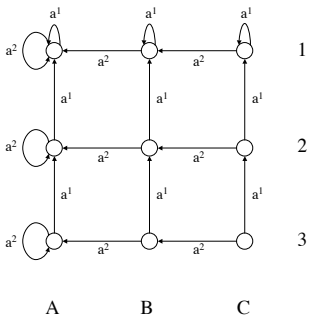

Fig. 5. Graph extracted from the learned transition model in the superposed map c. For readability purposes, we only show edges labeled by actions $ascent^1$ and $ascent^2$ (denoted here a^1 and a^2).

certainty its arrival point. We have also assumed that the zone were defined by straight boundaries. These assumptions led to learning histograms that converged towards 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 Fig. 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 [2]). 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, this redundancy can be detected, for example by analyzing the similarity of the learned distributions. 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 any angle, or even with varying angles over the environment: sometimes perpendicular, sometimes running parallel. The probabilities learned will numerically reflect the uncertainties that result from the ignorance of the angle between the equipotential lines.

V. 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 scenario of fusion of sensor modalities. We have presented a syntactic version of this operator which does not add information in the process of superposition. We have shown how simplifying assumptions made by this operator can be relaxed in a learning phase. We have shown, on an example, that much meaningful information can 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 [4]), other ongoing works aim at applying our operator to more specific models of the literature, for obtaining Superpositions of Kalman Filters, Superpositions of Particle Filters, Superpositions of Markov Localization models, etc. or combinations thereof.

Obtaining such combinations 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.

References

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

[2] 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.

[3] J. Diard, P. Bessière, and E. Mazer. Hierarchies of probabilistic models of navigation: the bayesian map and the abstraction operator. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA04)*, pages 3837–3842, New Orleans, LA, USA, 2004.

[4] J. Diard, P. Bessière, and E. Mazer. A survey of probabilistic models, using the bayesian programming methodology as a unifying framework. In *The 2^{nd} Int. Conf. on Computational Intelligence, Robotics and Autonomous Systems (CIRAS 2003)*, Singapore, December 2003.

[5] 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.

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

[7] 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.

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

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

[10] 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.

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

[12] 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.

[13] 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.

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

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