A survey of probabilistic models, using the Bayesian Programming methodology as a unifying framework

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 presents a survey of the most common probabilistic models for artefact conception. We use a generic formalism called Bayesian Programming*, which we introduce briefly, for reviewing the main probabilistic models found in the literature. Indeed, we show that Bayesian Networks, Markov Localization, Kalman filters, etc., can all be captured under this single formalism. We believe it offers the novice reader a good introduction to these models, while still providing the experienced reader an enriching global view of the field.*

1 Introduction

We think that over the next decade, probabilistic reasoning will provide a new paradigm for understanding neural mechanisms and the strategies of animal behaviour at a theoretical level, and will raise the performance of engineering artefacts to a point where they are no longer easily outperformed by the biological examples they are imitating.

Rational reasoning with incomplete and uncertain information is quite a challenge for artificial systems. The purpose of probabilistic inference is precisely to tackle this problem with a well-established formal theory. During the past years a lot of progress has been made in this field both from the theoretical and applied point of view. The purpose of this paper is to give an overview of these works and especially to try a synthetic presentation using a generic formalism named *Bayesian Programming* (BP).

[*] Julien Diard is currently with the Laboratoire de Physiologie de la Perception et de l'Action, Collège de France, Paris and the Department of Mechanical Engineering of the National University of Singapore. He can be reached at Julien.Diard@free.fr.

Several authors have already been interested in the relations between some of these models. Smyth, for instance, expresses the Kalman Filters (KFs) and the Hidden Markov Models (HMMs) in the Bayesian Network (BN) framework [1, 2]. Murphy replaces KFs, HMMs and a lot of their variants in the Dynamic Bayesian Network (DBN) formalism [3, 4] . Finally, Ghahramani focuses on the relations between BNs, DBNs, HMMs, and some variants of KFs [5, 6, 7].

The current paper includes the above, and adds Recursive Bayesian Estimation, Bayesian Filters (BFs), Particle Filters (PFs), Markov Localization models, Monte Carlo Markov Localization (MCML), and (Partially Observable) Markov Decision Processes (POMDPs and MDPs). This paper builds upon previous works, where more models are presented: Bessière *et al.* treat Mixture Models, Maximum Entropy Approaches, Sensor Fusion, and Classification in [8], while Diard treats Markov Chains and develops Bayesian Maps in [9].

All these models will be presented by their rewriting into the Bayesian Programming formalism, which will allow to use a *unique notation and structure* throughout the paper. This gives the novice reader an efficient first introduction to a wide variety of models and their relations. This also, hopefully, brings other readers the "big picture". By making the hypotheses made by each model explicit, we also bring to perspective all their variants, and shed light upon the possible variatons that are yet to be explored.

We will present the different models following a general-to-specific ordering. The "generality" measure we have chosen takes into account the number of hypotheses made by each model: the less hypotheses made by a model, the more general it is. However, not all hypotheses are comparable: for example, specialising the semantics of variables or specialising the dependency structure of the probabilistic models are

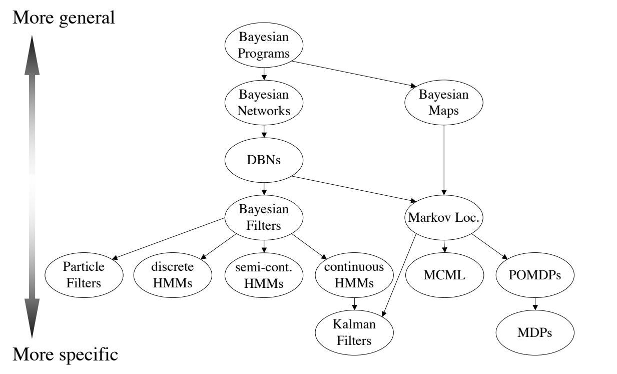

Figure 1: Probabilistic modelling formalisms treated in this paper and their general-to-specific partial ordering.

not easily compared. Therefore, we obtain a partial ordering over the space of all possible models, which can be seen Figure 1.

Of course, generality allows a bigger power of expression (more models will be instances of general formalisms). Whereas specialization allows for efficient solutions to inference and learning issues. For instance, the Baum-Welch algorithm, and the closed form solution of the state estimation problem in the case of KFs, both come from and justify the hypotheses made by HMMs and KFs, respectively.

The rest of the paper is organized as follows. Section 2 introduces the Bayesian Programming formalism. Each subsequent section briefly presents one of the formalisms of Figure 1, traversing the tree in an almost depth-first manner. For each of these formalisms, we will rewrite them in the BP framework and notations, describe their most salient features, and provide references for the interested reader. Sections 2 to 5 present general purpose models, where the modelling choices are made independently of any specific knowledge about the phenomenon, while Sections 6 to 9 present problem oriented models, where problem dependent knowledge are exploited; these are taken from the domain of robotics.

2 Bayesian Programming

We now introduce BP, a Bayesian Programming methodology. We briefly summarize it here, but still invite the interested reader to refer to Lebeltel *et al.* for all the details about this methodology and its use in robotics [10] .

As this formalism is only based on the inference

$$\text{Prog}\begin{cases}\text{Desc}\begin{cases}\text{Spec }(\pi)\begin{cases}\text{Pertinent variables}\\ \text{Decomposition}\\ \text{Forms}\begin{cases}\text{Parametric OR}\\ \text{Programs}\end{cases}\end{cases}\\ \text{Identification based on Data }(\delta)\end{cases}\\ \text{Question}\end{cases}$$

Figure 2: Structure of a bayesian program.

rules needed for probability calculus, it is very general. Indeed, this formalism will be used in the rest of this paper, to present all the other models we consider.

In the BP formalism, a bayesian program is a structure (see Figure 2) 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, applying Bayes' rule, 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 $P(\text{Searched} \mid \text{Known})$, where Searched and Known are conjunctions of variables appearing in disjoint subsets of the n variables. 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 modelling issues, we will assume that the inference problems are solved and implemented in an efficient manner by the programmer or by an inference engine [1].

[1]The one we use to tackle these problems has been described elsewhere [8].

3 Bayesian Networks

Bayesian Networks, first introduced by Pearl [11], have emerged as a primary method for dealing with probabilistic and uncertain information. They are the result of the marriage between the theory of probabilities and the theory of graphs. They are defined by the Bayesian Program of Figure 3.

$$
\text{P}\left\{\begin{array}{l}
\text{D}\left\{\begin{array}{l}
\text{S}\left\{\begin{array}{l}
\text{Variables} \\
\quad X^1, \ldots, X^N \\
\text{Decomposition} \\
\quad P(X^1 \ldots X^N) = \prod_{i=1}^{N} P(X^i |\ Pa^i), \\
\quad \text{with } Pa^i \subseteq \{X^1, \ldots, X^{i-1}\} \\
\text{Forms: any}
\end{array}\right. \\
\text{Identification: any}
\end{array}\right. \\
\text{Question: } P(X^i \mid Known)
\end{array}\right.
$$

Figure 3: The BN formalism rewritten in BP.

The pertinent variables are not constrained and have no specific semantics.

The decomposition, on the contrary, is specific: it is a product of distributions of one variable X^i, conditioned by a conjunction of other variables, called its "parents", Pa^i, $Pa^i \subseteq \{X^1, \ldots, X^{i-1}\}$. This assumes that variables are ordered, and ensures that applying Bayes' rule correctly defines the joint distribution. Also note that X^i denotes one and only one variable. Therefore, the following model, which can fit into a BP, does not into a BN:

$$P(A\ B\ C\ D) = P(A\ B) P(C \mid A) P(D \mid B).$$

In a BN, if A and B are to appear together on the left hand side of a term, as in $P(A\ B)$, they have to be merged into a single variable $\langle A, B\rangle$, and cannot be subsequently separated, as in $P(C \mid A)$ and $P(D \mid B)$.

An obvious bijection exists between joint probability distributions defined by such a decomposition and directed acyclic graphs: nodes are associated to variables, and oriented edges are associated to conditional dependencies. Using graphs in probabilistic models leads to an efficient way to define hypotheses over a set of variables, an economic representation of a joint probability distribution and, most importantly, an easy and efficient way to do probabilistic inference.

The parametric forms are not constrained theoretically, but in BN commercial softwares they are very often restricted to probability tables (as in Netica), or tables and constrained Gaussians (as in Hugin [2]).

[2] `http://www.norsys.com` and `http://www.hugin.com`.

Very efficient inference techniques have been developped to answer questions of the form $P(X^i \mid \text{Known})$, where Known is a subset of the other variables of the BN. However, some difficulties appear for more general questions (*i.e.* with more than one variable on the left hand side).

Readings on BNs should start by the books by Pearl [11], Lauritzen [12], Jordan [13] and Frey [14].

4 Dynamic Bayesian Networks

To deal with time and to model stochastic processes, the framework of Bayesian Networks has been extended to Dynamic Bayesian Networks [15]. Given a graph representing the structural knowledge at time t, supposing this structure to be time-invariant and time to be discrete, the resulting DBN is the repetition of the first structure from a start time to a final time. Each part at time t in the final graph is named a time slice. The DBNs are defined by the Bayesian Program of Figure 4.

$$
\text{P}\left\{\begin{array}{l}
\text{D}\left\{\begin{array}{l}
\text{S}\left\{\begin{array}{l}
\text{Variables} \\
\quad X_0^1, \ldots, X_0^N, \ldots, X_T^1, \ldots, X_T^N \\
\text{Decomposition} \\
\quad P(X_0^1 \ldots X_T^N) \\
\qquad = P(X_0^1 \ldots X_0^N) \prod_{t=0}^{T} \prod_{i=1}^{N} P(X_t^i |\ R_t^i) \\
\text{Forms: any}
\end{array}\right. \\
\text{Identification: any}
\end{array}\right. \\
\text{Question: } P(X_T^i \mid Known)
\end{array}\right.
$$

Figure 4: The DBN formalism rewritten in BP.

In Figure 4, R_t^i is a conjunction of variables taken in the set $\{X_t^1, \ldots, X_t^{i-1}\} \bigcup \{X_{t-1}^1, \ldots, X_{t-1}^N\}$. This means that X_t^i depends only on its parents at time t ($\{X_t^1, \ldots, X_t^{i-1}\}$), as in a regular BN, and on some variables from the previous time slice ($\{X_{t-1}^1, \ldots, X_{t-1}^N\}$).

$\prod_{i=1}^{N} P(X_t^i |\ R_t^i)$ defines a graph for time slice t, and all time slices are identical when the time index t is changing.

These hypotheses are very often made: the fact that a time slice only depends on the previous one is commonly called the *first order Markov assumption*. The fact that all time slices are identical is the *stationarity hypothesis*. In this case, the model defined by a time slice is called the *local model*, and is said to be *time-invariant*, or even *homogeneous*.

As can easily be seen on Figure 4, a DBN as a whole, "unrolled" over time, may be considered as a regular BN. Consequently the usual inference techniques applicable to BN are still valid for such "unrolled" DBNs.

The best introduction, survey and starting point on DBNs is Murphy's Ph.D. thesis [4]. The interested reader can also refer to papers by Kanazawa *et al.* [16], or to Ghahramani for the learning aspects in DBNs [5].

5 Recursive Bayesian Estimation

5.1 Bayesian Filtering, Prediction and Smoothing

Recursive Bayesian Estimation is the generic denomination for a very largely applied class of numerous different probabilistic models of time series. They are defined by the Bayesian Program of Figure 5.

$$
\text{P}\left\{\begin{array}{l}
\text{D}\left\{\begin{array}{l}
\text{S}\left\{\begin{array}{l}
\text{Variables}\\
\quad S_0,\dots,S_T,O_0,\dots,O_T\\
\text{Decomposition}\\
\quad P(S_0\dots S_T\; O_0\dots O_T) =\\
\qquad P(S_0)P(O_0\mid S_0)\\
\qquad \prod_{i=1}^{T}\left[P(S_i\mid S_{i-1})P(O_i\mid S_i)\right]\\
\text{Forms: any}
\end{array}\right.\\
\text{Identification: any}
\end{array}\right.\\
\text{Question: } P(S_{t+k}\mid O_0\dots O_t)\\
\quad k=0 \;\equiv\; \text{Filtering}\\
\quad k>0 \;\equiv\; \text{Prediction}\\
\quad k<0 \;\equiv\; \text{Smoothing}
\end{array}\right.
$$

Figure 5: Recursive Bayesian estimation in BP.

Variables $S_0,\dots,S_T$ are a time series of "state" variables considered on a time horizon ranging from 0 to T. Variables $O_0,\dots,O_T$ are a time series of "observation" variables on the same horizon.

The decomposition is based on three terms. $P(S_i \mid S_{i-1})$, called the "system model" or "transition model", formalizes the knowledge about transitions from state at time $i-1$ to state at time i. $P(O_i \mid S_i)$, called the "observation model", expresses what can be observed at time i when the system is in state S_i. Finally, a prior $P(S_0)$ is defined over states at time 0.

The question usually asked to these models is $P(S_{t+k} \mid O_0\dots O_t)$: what is the probability distribution for state at time $t+k$ knowing the observations made from time 0 to $t, t \in 1,\dots,T$? The most common case is Bayesian Filtering where $k = 0$, which means that one searches for the present state knowing the past observations. However it is also possible to do "prediction" ($k > 0$), where one tries to extrapolate future state from past observations, or to do "smoothing" ($k < 0$), where one tries to recover a past state from observations made either before or after that instant. However, some more complicated questions may also be asked (see Section 5.2).

Bayesian Filters ($k = 0$) have a very interesting recursive property which contributes largely to their interest. Indeed, $P(S_t \mid O_0\dots O_t)$ may be simply computed from $P(S_{t-1} \mid O_0\dots O_{t-1})$ with the following formula (derivation omitted):

$$
\begin{aligned}
&P(S_t \mid O_0\dots O_t) =\\
&\quad P(O_t \mid S_t)\\
&\quad \sum_{S_{t-1}}\left[(P(S_t \mid S_{t-1})P(S_{t-1} \mid O_0\dots O_{t-1})\right]
\end{aligned}
\tag{1}
$$

In the case of prediction and smoothing, the imbricated sums for solving the same question can quickly become a huge computational burden.

Readers interested by Bayesian Filtering should refer to [17].

5.2 Hidden Markov Models

Hidden Markov Models are a very popular specialization of Bayesian Filters. They are defined by the Bayesian Program of Figure 6.

$$
\text{P}\left\{\begin{array}{l}
\text{D}\left\{\begin{array}{l}
\text{S}\left\{\begin{array}{l}
\text{Variables}\\
\quad S_0,\dots,S_T,O_0,\dots,O_T\\
\text{Decomposition}\\
\quad P(S_0\dots S_T\; O_0\dots O_T) =\\
\qquad P(S_0)P(O_0\mid S_0)\\
\qquad \prod_{i=1}^{T}\left[P(S_i\mid S_{i-1})P(O_i\mid S_i)\right]\\
\text{Forms}\\
\quad P(S_0) \;\equiv\; \text{Matrix}\\
\quad P(S_i\mid S_{i-1}) \;\equiv\; \text{Matrix}\\
\quad P(O_i\mid S_i) \;\equiv\; \text{Matrix}
\end{array}\right.\\
\text{Identification: Baum-Welch algorithm}
\end{array}\right.\\
\text{Question: } P(S_0\; S_1\dots S_t-1 \mid S_t\; O_0\dots O_t)
\end{array}\right.
$$

Figure 6: The HMM formalism rewritten in BP.

Variables are supposed to be discrete. Therefore, the transition model $P(S_i \mid S_{i-1})$ and the observation model $P(O_i \mid S_i)$ are both specified using probability matrices (or CPTs for *conditional probability tables*). Variants exist about this particular point: when the observation variables are continuous, the formalism becomes known as "semi-continuous HMMs" [18, 4]. In this case, the observation model is associated either with a Gaussian form, or a Mixture of Gaussian form.

The most popular question asked to HMMs is $P(S_0\, S_1 \ldots S_{t-1} \mid S_t\, O_0 \ldots O_t)$: what is the most probable series of states that leads to the present state knowing the past observations? This particular question may be answered with a specific and very efficient algorithm called the "Viterbi algorithm".

Finally, the "Baum-Welch" algorithm is a specific learning algorithm that has been developped for HMMs: it computes the most probable observation and transition models from a set of experimental data.

Two nice entry points into the huge HMM literature are the tutorial by Rabiner [18] and the chapter 6 of his book *Fundamentals of Speech Recognition* [19].

5.3 Kalman Filters

The very well known Kalman Filters [20] are another specialization of Bayesian Filters. They are defined by the Bayesian Program of Figure 7.

$$
P \left\{ \begin{array}{l}
D \left\{ \begin{array}{l}
S \left\{ \begin{array}{l}
\text{Variables} \\
\quad S_0, \ldots, S_T, O_0, \ldots, O_T \\
\text{Decomposition} \\
\quad P(S_0 \ldots S_T\ O_0 \ldots O_T) = \\
\qquad P(S_0) P(O_0 \mid S_0) \\
\qquad \prod_{i=1}^{T} \left[P(S_i \mid S_{i-1}) P(O_i \mid S_i) \right] \\
\text{Forms} \\
\quad P(S_0) \ \equiv\ \mathbf{G}(S_0, \mu, \sigma) \\
\quad P(S_i \mid S_{i-1}) \ \equiv\ \mathbf{G}(S_i, A \cdot S_{i-1}, Q) \\
\quad P(O_i \mid S_i) \ \equiv\ \mathbf{G}(O_i, H \cdot S_i, R)
\end{array} \right. \\
\text{Identification}
\end{array} \right. \\
\text{Question: } P(S_t \mid O_0 \ldots O_t)
\end{array} \right.
$$

Figure 7: The KF formalism rewritten in BP.

Variables are continuous. The transition model $P(S_i \mid S_{i-1})$ and the observation model $P(O_i \mid S_i)$ are both specified using Gaussian laws with means that are linear functions of the conditioning variables.

Due to these hypotheses, and using the recursive Equation 1, it is possible to analytically solve the inference problem to answer the usual $P(S_t \mid O_0 \ldots O_t)$ question. This leads to an extremely efficient algorithm that explains the popularity of Kalman Filters and the number of their everyday applications.

When there is no obvious linear transition and observation models, it is still often possible, using a first order Taylor's expansion, to consider that these models are locally linear. This generalization is commonly called the Extended Kalman Filter (EKF). Sometimes KFs also include action variables, and thus become specializations of Markov Localization models (see Section 6).

A nice tutorial by Welch and Bishop may be found on the internet [21]. For a more complete mathematical presentation one should refer to a report by Barker *et al.* [22], but these are only two entries to a huge literature concerning the subject.

5.4 Particle Filters

The fashionable Particle Filters (PFs) may be seen as a specific implementation of Bayesian Filters.

The distribution $P(S_{t-1} \mid O_0 \ldots O_{t-1})$ is approximated by a set of N particles having weights proportional to their probabilities. The recursive Equation 1 is then used to inspire a dynamic process that produces an approximation of $P(S_t \mid O_0 \ldots O_t)$. The principle of this dynamical process is that the particles are first moved according to the transition model $P(S_t \mid S_{t-1})$, and then their weights are updated according to the observation model $P(O_t \mid S_t)$.

See the tutorial by Arulampalam *et al.* for a start [23].

6 Markov Localization

A possible alternative to Bayesian Filters is to add control variables $A_0, \ldots, A_{t-1}$ to the model. This extension is sometimes called input-output HMM [24, 25, 6, 26], or sometimes Bayesian Filters still, but, in the field of robotics, it has received more attention under the name of Markov Localization (ML) [27, 28]. In this field, such an extension is natural, as a robot can observe its state by sensors, but can also influence its state via motor commands. ML models are defined by the Bayesian Program of Figure 8.

$$
P \left\{ \begin{array}{l}
D \left\{ \begin{array}{l}
S \left\{ \begin{array}{l}
\text{Variables} \\
\quad S_0, \ldots, S_T, A_0, \ldots, A_{T-1}, O_0, \ldots, O_T \\
\text{Decomposition} \\
\quad P(S_0 \ldots S_T\ A_0 \ldots A_{T-1}\ O_0 \ldots O_T) = \\
\qquad P(S_0) P(O_0 \mid S_0) \\
\qquad \prod_{i=1}^{T} \left[\begin{array}{l} P(A_{i-1}) P(S_i \mid A_{i-1}\ S_{i-1}) \\ P(O_i \mid S_i) \end{array} \right] \\
\text{Forms: Matrices or Particles}
\end{array} \right. \\
\text{Identification}
\end{array} \right. \\
\text{Question: } P(S_t \mid A_0 \ldots A_{t-1}\ O_0 \ldots O_t)
\end{array} \right.
$$

Figure 8: The ML formalism rewritten in BP.

Starting from a Bayesian Filter structure, the control variable is used to refine the transition model $P(S_i \mid S_{i-1})$ into $P(S_i \mid A_{i-1}\ S_{i-1})$, which is then called the action model. The rest of the dependency structure is unchanged.

The forms can be defined in several ways: they are commonly matrices, but when they are implemented using Particles (in a similar manner as the one presented Section 5.4), the model takes the name of Monte Carlo Markov Localization (MCML). Among the forms, $P(A_{i-1})$ is almost always assumed to be uniform (and thus disappears of Equation 2).

The resulting model is used to answer the question $P(S_t \mid A_0 \ldots A_{t-1}\ O_0 \ldots O_t)$, which estimates the state of the robot, given past actions and observations: when this state represents the position of the robot in its environment, this amounts to localization. This question is similar to the Bayesian Filtering question, and can also be solved in a recursive manner:

$$\begin{aligned} & P(S_t \mid A_0 \ldots A_{t-1}\ O_0 \ldots O_t) = \\ & \quad P(A_{t-1}) P(O_t \mid S_t) \\ & \quad \sum_{S_{t-1}} \left[\begin{array}{l} P(S_t \mid A_{i-1}\ S_{t-1}) \\ P(S_{t-1} \mid A_0 \ldots A_{t-2}\ O_0 \ldots O_{t-1}) \end{array} \right] \end{aligned} \tag{2}$$

A reference paper to ML and its use in robotics is the survey by Thrun [29].

7 Decision Theoretic Planning

Partially Observable Markov Decision Processes (POMDPs) and Markov Decision Processes (MDPs) are used in robotics to model a robot that has to plan and to execute a sequence of actions. A complete review of POMDPs and MDPs by Boutilier *et al.* [30] is an interesting starting point.

7.1 Partially Observable Markov Decision Processes

Formally, POMDPs use the same probabilistic model than Markov Localization except that it is enriched by the definition of a *reward (and/or cost) function.*

This reward function R models which states are good for the robot, and which actions are costly. In the most general notation, it therefore is a function that associates, for each couple state - action, a real valued number: $R : S_i, A_i \to \mathbb{R}$.

The reward function helps driving the planning process. Indeed, the aim of this process is to find an optimal plan in the sense that it maximizes a certain measure based on the reward function. This measure is most frequently the expected discounted cumulative reward, $\langle \sum_{t=0}^{\infty} \gamma^t R_t \rangle$, where γ is a discount factor (less than 1), R_t is the reward obtained at time t, and $\langle \cdot \rangle$ is the mathematical expectation. Given this measure, the goal of the planning process is to find a optimal mapping from probability distributions over states to actions (a *policy*). This planning process, which leads to intractable computations, is sometimes approximated using iterative algorithms called *policy iteration* or *value iteration.* These algorithms start with random policies, and improve them at each step until some numerical convergence criterion is met. Unfortunately, state-of-the-art implementations of these algorithms still cannot cope with state spaces of more than a hundred states [31].

An introduction to POMDPs is proposed by Kaelbling *et al.* [32].

7.2 Markov Decision Processes

Another class of approach for tackling the intractability of the planning problem in POMDPs is to suppose that the robot knows what state it is in. The state becomes observable, therefore the observation variable and model are not needed anymore: the resulting formalism is called a (Fully Observable) Markov Decision Process (MDP), and is summed up by the Bayesian Program of Figure 9.

$$P \left\{ \begin{array}{l} D \left\{ \begin{array}{l} S \left\{ \begin{array}{l} \text{Variables} \\ \quad S_0, \ldots, S_T, A_0, \ldots, A_{T-1} \\ \text{Decomposition} \\ \quad P(S_0 \ldots S_T\ A_0 \ldots A_{T-1}) = \\ \qquad P(S_0) \prod_{i=1}^{T} [P(A_{i-1}) P(S_i \mid A_{i-1}\ S_{i-1})] \\ \text{Forms: Matrices} \end{array} \right. \\ \text{Identification} \end{array} \right. \\ \text{Question: } P(A_0 \ldots A_{t-1} \mid S_t\ S_0) \end{array} \right.$$

Figure 9: The MDP formalism rewritten in BP.

MDPs can cope with planning in state-spaces bigger than POMDPs, but are still limited to some hundreds of states. Therefore, many recent research efforts are aimed toward hierarchical decomposition of the planning and modelling problems in MDPs, especially in the robotic field, where their full observability hypothesis makes their practical use difficult [33, 34, 31].

8 Bayesian Robot Programming

Lebeltel *et al.* applied the BP methodology to mobile robotics [35, 10]. The adequacy of the method as a robotic programming tool has been demonstrated through a succession of increasingly complex experiments: learning of simple behaviours, behaviour combination, sensor fusion, hierarchical behaviour composition, situation recognition and temporal sequencing. This series of experiments were also integrated for solving a complex robotic task, showing the possibility of incremental development.

9 Bayesian Maps

One of the most crucial problems in robotics is the modelling by the robot of its environment. Most common approaches rely either on ML models, or variants of KFs (for example, that include action variables). However, as can be easily seen Figure 8, the only question asked to such models is a *localization* question, of the form $P(S_t \mid A_0 \ldots A_{t-1}\ O_0 \ldots O_t)$. Therefore, the action variables are merely used as input variables, as the model is not concerned with computing probability distributions over actions.

As noted by Diard [9] and Thrun [36], such a separation between the localization and control models is not satisfying. This remark, among others, led to the definition of the *Bayesian Map* model, which is a generalization of ML models. The local model (see Section 4) of a Bayesian Map is defined by the Bayesian Program of Figure 10.

$$\text{P}\left\{\begin{array}{l} \text{D}\left\{\begin{array}{l} \text{S}\left\{\begin{array}{l} \text{Variables}\\ \quad S_t, S_{t'}, A_t, O_t\\ \text{Decomposition: any}\\ \text{Forms: any} \end{array}\right.\\ \text{Identification: any} \end{array}\right.\\ \text{Question: behaviour definition}\\ \quad P(A^i \mid X), A^i \subseteq A, X \subseteq \left(\{P, L_t, L_{t'}, A\} \setminus A^i\right) \end{array}\right.$$

Figure 10: The Bayesian Map formalism in BP.

A Bayesian Map is a model that includes four variables: an observation, an action, and a state variable at time t, and a state variable at a later time t'. The decomposition is not constrained (if the four variables are atomic, there are already 1015 decompositions to choose from – Attias recently exploited one of these [37]), nor are the forms or the identification phase.

On the contrary, the use of this model is strongly constrained, as we require that the model generates *behaviours*, which are questions of the form $P(A^i \mid X)$: what is the probability distribution over (a subset of) the action variable(s), knowing some other variables? This constraint ensures that the model will be, *in fine*, used for controlling the robot. That the model is also used for explicitly computing a distribution over state variables (*i.e., localization*) is an optional goal, that can be convenient for human-robot communication purposes, for example.

The interested reader can refer to Diard *et al.* for more details concerning the Bayesian Map formalism, and in particular for the definition of the Abstraction and Superposition Operators, that allow for building hierarchies of Bayesian Maps [9, 38].

10 Conclusion

In this paper, we have presented a large class of probabilistic modelling formalisms. They all have been rewritten in the Bayesian Programming framework, which proves its generality. Moreover, the exploring of the relations between the various formalisms was greatly eased, because BP forces to express explicitly all the hypotheses made, from the choice of variables to the use of the model (questions).

We think that this studying of the different hypotheses can be very fruitful, especially for designing new formalisms, as we have briefly illustrated by some of the reasoning that led to the Bayesian Map model.

Acknowledgements

This work has been supported by the BIBA european project (IST-2001-32115). The authors would also like to thank Pr Marcelo. H. Ang Jr.

References

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

[2] P. Smyth. Belief networks, hidden markov models, and markov random fields: a unifying view. *Pattern Recognition Letters*, 1998.

[3] K. Murphy. An introduction to graphical models. Technical report, University of California, Berkeley, May 2001.

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

[5] Z. Ghahramani. Learning dynamic Bayesian networks. In C. Lee Giles and Marco Gori, editors, *Adaptive Processing of Sequences and Data Structures*, number 1387 in Lecture Notes in Artificial Intelligence, LNAI, pages 168–197. Springer-Verlag, 1998.

[6] Z. Ghahramani. An introduction to hidden markov models and bayesian networks. *Journal of Pattern Recognition and Artificial Intelligence*, 15(1):9–42, 2001.

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

[8] 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.

[9] J. Diard. *La carte bayésienne – Un modèle probabiliste hiérarchique pour la navigation en robotique mobile.* Ph.D. thesis, Institut National Polytechnique de Grenoble, Grenoble, France, Janvier 2003.

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

[11] J. Pearl. *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, Ca, 1988.

[12] S. L. Lauritzen. *Graphical Models.* Clarendon Press, Oxford, 1996.

[13] M. Jordan. *Learning in Graphical Models.* MIT Press, 1998.

[14] B. J. Frey. *Graphical Models for Machine Learning and Digital Communication.* MIT Press, 1998.

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

[16] K. Kanazawa, D. Koller, and S. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In Philippe Besnard and Steve Hanks, editors, *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence (UAI'95)*, pages 346–351, San Francisco, CA, USA, August 1995. Morgan Kaufmann Publishers.

[17] A. H. Jazwinsky. *Stochastic Processes and Filtering Theory.* New York : Academic Press, 1970.

[18] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. of the IEEE Trans. on ASSP*, 77(2):257–285, February 1989.

[19] 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.

[20] R. E. Kalman. A new approach to linear filtering and predictive problems. *Transactions ASME, Journal of basic engineering*, 82:34–45, 1960.

[21] G. Welch and G. Bishop. An introduction to the kalman filter. Technical Report TR95-041, Computer Science Department, Univerity of North Carolina, Chapel Hill, 1995.

[22] A. L. Barker, D. E. Brown, and W. N. Martin. Bayesian estimation and the Kalman Filter. Technical Report IPC-94-02, Institute for Parallel Computation, University of Virginia, August, 5 1994.

[23] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions of Signal Processing*, 50(2):174–188, February 2002.

[24] Y. Bengio and P. Frasconi. An input/output HMM architecture. In G. Tesauro, D.S. Touretzky, and T.K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 427–434. MIT Press, Cambridge, MA, 1995.

[25] T. W. Cacciatore and S. J. Nowlan. Mixtures of controllers for jump linear and non-linear plants. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 719–726. Morgan Kaufmann Publishers, Inc., 1994.

[26] M. Meila and M. I. Jordan. Learning fine motion by markov mixtures of experts. In D. Touretzky, M. C. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages –. Morgan Kaufmann Publishers, Inc., 1996.

[27] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the absolute position of a mobile robot using position probability grids. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, pages 896–901, Menlo Park, August, 4–8 1996. AAAI Press / MIT Press.

[28] S. Thrun, W. Burgard, and D. Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning and Autonomous Robots (joint issue)*, 31/5:1–25, 1998.

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

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

[31] 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.

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

[33] M. Hauskrecht, N. Meuleau, L. P. Kaelbling, T. Dean, and C. Boutilier. Hierarchical solution of Markov decision processes using macro-actions. In Gregory F. Cooper and Serafín 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.

[34] 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.

[35] O. Lebeltel. *Programmation Bayésienne des Robots.* Ph.D. thesis, Institut National Polytechnique de Grenoble, Grenoble, France, Septembre 1999.

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

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

[38] 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) (to appear)*, 2003.