The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Bayesian Reinforcement Learning in Continuous POMDPs with Gaussian Processes

Patrick Dallaire, Camille Besse, Stephane Ross and Brahim Chaib-draa

Abstract— **Partially Observable Markov Decision Processes (POMDPs) provide a rich mathematical model to handle real-world sequential decision processes but require a known model to be solved by most approaches. However, mainstream POMDP research focuses on the discrete case and this complicates its application to most realistic problems that are naturally modeled using continuous state spaces. In this paper, we consider the problem of optimal control in continuous and partially observable environments when the parameters of the model are unknown. We advocate the use of Gaussian Process Dynamical Models (GPDMs) so that we can learn the model through experience with the environment. Our results on the blimp problem show that the approach can learn good models of the sensors and actuators in order to maximize long-term rewards.**

I. INTRODUCTION

In the past few decades, Reinforcement Learning (RL) has emerged as an elegant and popular technique to handle decision problems when the model is unknown. Reinforcement learning is a general technique that allows an agent to learn the best way to behave, i.e. to maximize expected return, from repeated interactions with the environment. One of the most fundamental open questions in RL is that of exploration-exploitation: namely, how should the agent choose actions during the learning phase, in order to both maximize its knowledge of the model as needed to better achieve the objective later (to *explore*), and maximize current achievement of the objective based on what is already known about the domain (to *exploit*). Under some (reasonably general) conditions on the exploratory behavior, it has been shown that RL eventually learns the optimal action-selection behavior. However, these conditions do not specify how to optimally trade-off between exploration and exploitation such as to maximize utilities throughout the life of the agent, including during the learning phase, as well as beyond.

Model-based Bayesian RL (BRL) is a recent extension of RL that has gained significant interest from the AI community as it can jointly optimize performance during learning and beyond, within the standard Bayesian inference paradigm. In this framework, prior information about the problem (including uncertainty) is represented in parametric form, and Bayesian inference is used to incorporate any new information about the model. Thus, the exploration-exploitation problem can be handled as an explicit sequential decision problem, where the agent seeks to maximize future expected return with respect to its current uncertainty on the model. An important limitation of this approach is that making decision is significantly more complex since it involves reasoning about all possible models *and* courses of action. In addition, most work to date on this framework has been limited to cases where full knowledge of the agent's state is available at every time step [1], [2], [3], [4].

Mainstream frameworks for model-based BRL for POMDPs [5], [6] are only applicable in domains described by finite sets of states, actions and observations, where the Dirichlet distribution is used as a posterior over discrete distributions. This is an important limitation in practice as many practical problems are naturally described by continuous states, actions and observations spaces. For instance, in robot navigation problems, the state is usually described by continuous variables such as the robot's position, orientation, velocity and angular velocity; the choice of action controls the forward and angular acceleration, which are both continuous; and the observations provide a noisy estimate of the robot's state based on its sensors, which would also be continuous.

One recent framework for model-based BRL in continuous POMDP is the Bayes-Adaptive Continuous POMDP [7], which extends the previously proposed Bayes-Adaptive POMDP model for discrete domains to continuous domains. However, this framework assumes that the parametric form of the transition and observation functions are known and that only the mean vector and covariance matrix of the noise random variables are unknown. Hence, this framework has limited applicability when the transition and observation functions are completely unknown.

This paper aims to investigate a model-based BRL framework that can handle domains that are both partially observable and continuous without assuming any parametric form for the transition, observation and reward function. To achieve that, we use Gaussian Process priors to learn these functions, and then propose a planning algorithm which selects the actions that maximize long-term expected rewards under the current model.

II. RELATED WORK

In the case of continuous POMDPs, the literature is relatively sparse. In general, the common approach is to assume a discretization of the state space and therefore, be an incomplete model of the underlying system.

A first approach to continuous-state POMDPs is undoubtedly Thrun's approach [8], where a belief is viewed as a set of weighted samples, which can be considered as a

P. Dallaire, Camille Besse and B. Chaib-draa are with Department of Computer Science, Laval University, Quebec, Canada

S. Ross is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA

978-1-4244-3804-4/09/$25.00 ©2009 IEEE

particular case (a degenerative case) of Gaussian mixture representation. The author presents a Monte Carlo algorithm for learning to act in POMDPs with real-valued state and action spaces, paying thus tribute to the fact that a large number of real-world problems are continuous in nature. A reinforcement learning algorithm value iteration is used to learn the value function over belief states. Then, a sample version of nearest neighbors is used to generalize across states.

A second approach to continuous-state POMDPs has been proposed by Porta et al. [9]. In this paper, the authors show that the optimal finite-horizon value function over the infinite POMDP belief space is piecewise linear and convex. This value function is defined by a finite set of α-functions, expressed as linear combinations of Gaussians, assuming that the sensors, actions and rewards models are also Gaussian-based. Then, they show it for a fairly general class of POMDP models in which all functions of interest are modeled by Gaussian mixtures. All belief updates and value iteration backups can be carried out analytically and exactly. Experimental results, using the previously proposed randomized point-based value iteration algorithm PERSEUS [10], have been demonstrated with a simple robot planning task in a continuous domain. The same authors have extended their work to deal with continuous action and observation sets by designing effective sampling approaches [11]. The basic idea is that general continuous POMDPs can be casted in the continuous-state POMDP paradigm via sampling strategies. This work, contrary to the approach presented in this paper, assumes the POMDP model is known which is very restrictive for the type of real applications we consider.

To sum up, and as previously stated, the common approach to continuous POMDPs assumes a discretization of the state space which can be a poor model for the underlying system. This remains true if we cast the general continuous POMDP in the continuous-state POMDP paradigm as Porta and colleagues did [11]. Our approach avoids this, by proposing a "real" continuous POMDPs where (i) the state space; (ii) the action space and (iii) the observation space; are all continuous and potentially multidimensional.

Another approaches related to our work turn around the modeling of time series data using dynamical systems [12]. In the case of nonlinear time series analysis, Wang and colleagues introduced Gaussian Process Dynamical Models (GPDMs) so that they can learn models of human pose and motion from high-dimensional motion capture data. Their GPDM is a latent variable model comprising a low dimensional latent space with associated dynamics, as well as a map from the latent space to an observation space. The authors marginalized out the model parameters in closed form by using Gaussian process priors for both the dynamical and observation mappings. It results in a nonparametric model for dynamical systems. Thus, their approach is sustained by a continuous hidden Markov model and does not include actions and rewards.

III. CONTINUOUS POMDP

We consider a continuous POMDP to be defined by the tuple $(S, A, Z, T, O, R, b_1, \gamma)$:

- $S \subset \mathbb{R}^m$: The state space, which is continuous and potentially multidimensional.
- $A \subset \mathbb{R}^n$: The action space, which is continuous and potentially multidimensional. It is assumed here that A is a closed subset of $\mathbb{R}^n$, so that the optimal control, as defined below, exists.
- $Z \subset \mathbb{R}^p$: The observation space, which is continuous and potentially multidimensional.
- $T : S \times A \times S \rightarrow [0, \infty]$: The transition function which specifies the conditional probability density $T(\mathbf{s}, \mathbf{a}, \mathbf{s}') = p(\mathbf{s}'|\mathbf{s}, \mathbf{a})$ of moving to state $\mathbf{s}'$, given the current state $\mathbf{s}$ and the action $\mathbf{a}$ performed by the agent.
- $O : S \times A \times Z \rightarrow [0, \infty]$: The observation function which specifies the conditional probability density $O(\mathbf{s}', \mathbf{a}, \mathbf{z}') = p(\mathbf{z}|\mathbf{s}', \mathbf{a})$ of observing observation $\mathbf{z}'$ when moving to state $\mathbf{s}'$ after doing action $\mathbf{a}$.
- $R : S \times A \times \mathbb{R} \rightarrow [0, \infty]$: The reward function which specifies the conditional probability density $R(\mathbf{s}', \mathbf{a}, r') = p(r'|\mathbf{s}', \mathbf{a})$ of getting reward r', given that the agent performed action $\mathbf{a}$ and reached state $\mathbf{s}'$.
- $b_1 \in \Delta S$: The initial state distribution.
- γ : The discount factor.

The posterior distribution over the current state $\mathbf{s}$ of the agent, denoted b and refered to as the belief, can be maintained via Bayes' rule as follows:

$$b^{a\mathbf{z}}(\mathbf{s}') \propto O(\mathbf{s}', \mathbf{a}, \mathbf{z}') \int_S T(\mathbf{s}, \mathbf{a}, \mathbf{s}') b(\mathbf{s}) d\mathbf{s} \qquad (1)$$

The agent's behavior is determined by its policy π which specifies the probability of performing action a in belief state b. The optimal policy π^* is the one that maximizes the expected sum of discounted rewards over the infinite horizon and is obtained by solving Bellman's equation:

$$V^*(b) = \max_{\mathbf{a} \in A} \left[g(b, \mathbf{a}) + \gamma \int_Z f(\mathbf{z}|b, \mathbf{a}) V^*(b^{a\mathbf{z}}) d\mathbf{z} \right] \qquad (2)$$

where V^* is the value function of the optimal policy and is the fixed point of Bellman's equation. Also,

$$g(b, \mathbf{a}) = \int_S b(\mathbf{s}) \int_S T(\mathbf{s}, \mathbf{a}, \mathbf{s}') \int_{\mathbb{R}} r R(\mathbf{s}', \mathbf{a}, r) dr d\mathbf{s}' d\mathbf{s}$$

is the expected reward when performing $\mathbf{a}$ in belief state b and

$$f(\mathbf{z}|b, \mathbf{a}) = \int_S O(\mathbf{s}', \mathbf{a}, \mathbf{z}) \int_S T(\mathbf{s}, \mathbf{a}, \mathbf{s}') b(s) d\mathbf{s} d\mathbf{s}'$$

is the conditional probability density of observing $\mathbf{z}$ after doing action $\mathbf{a}$ in belief b. The next belief state obtained by performing a Bayes' rule update of b with action $\mathbf{a}$ and observation $\mathbf{z}$ is denoted by $b^{a\mathbf{z}}$.

IV. GP-POMDP

In this paper, we consider the problem of optimal control in such continuous POMDP where T, O and R are unknown a priori. To consider making optimal decisions, the model is learned from sequence of action-observation through Gaussian Processes (GPs) modeling [13], [14]. GPs are a class of probabilistic models which focuses on points where a function is instantiated, through a Gaussian distribution over the function space. Usually a Gaussian distribution is parameterized by a mean vector and a covariance matrix, and in the case of GPs, these two parameters are functions of the space on which the process operates.

For our problem of optimal control, we propose to use a Gaussian Process Dynamical Model (GPDM) [12] to learn the transition, observation and reward functions and then propose a planning algorithm which selects action that maximizes long-term expected rewards under the current model.

In order to use GPs to learn the transition, observation and reward model, we first assume that the dynamics can be expressed in the following form:

$$\begin{array}{rcl} \mathbf{s}_t & = & T'(\mathbf{s}_{t-1}, \mathbf{a}_{t-1}) + \epsilon_T \\ \mathbf{z}_t & = & O'(\mathbf{s}_t, \mathbf{a}_{t-1}) + \epsilon_O \\ r_t & = & R'(\mathbf{s}_t, \mathbf{a}_{t-1}) + \epsilon_R \end{array} \tag{3}$$

where ϵ_T, ϵ_O and ϵ_R are zero-mean Gaussian white noise. The functions T', O' and R' are respectively unknown deterministic function that returns the next state, observation and reward. We seek to learn this model and maintain a maximum likelihood estimate of the state trajectory by using GPDM with optimization methods.

Let us denote $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_{N+1}]^\top$ the state sequence matrix, $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_N]^\top$ the action sequence matrix, $\mathbf{Z} = [\mathbf{z}_2, \mathbf{z}_3, \ldots, \mathbf{z}_{N+1}]^\top$ the observation sequence matrix and $\mathbf{r} = [r_2, r_3, \ldots, r_{N+1}]^\top$ the reward sequence vector.

A. Gaussian Process Dynamical Model

A GPDM consists of a mapping from a latent space, which is assumed to follow first-order Markov dynamics, to an observation space. This probabilistic mapping, for the purpose of POMDPs, is defined as $S \times A \rightarrow Z \times R$ and represents the observation-reward function where the actions are fully observable and the states are latent variables. The dynamic mapping in latent space is $S \times A \rightarrow S$ and corresponds to the transition function. Both mappings are defined as linear combinations of nonlinear basis functions:

$$\begin{array}{rcl} \mathbf{s}_t & = & \sum_i \mathbf{b}_i \phi_i(\mathbf{s}_{t-1}, \mathbf{a}_{t-1}) + \mathbf{n}_s \\ \mathbf{y}_t & = & \sum_j \mathbf{c}_j \psi_j(\mathbf{s}_t, \mathbf{a}_{t-1}) + \mathbf{n}_y \end{array} \tag{4}$$

Here, $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \ldots]$ and $\mathbf{C} = [\mathbf{c}_1, \mathbf{c}_2, \ldots]$ are weights for basis function ϕ_i and ψ_j, $\mathbf{n}_s$ and $\mathbf{n}_y$ are zero-mean time invariant white Gaussian noise. The joint observation-space is denoted as $Y = Z \times R$ and therefore $\mathbf{y}_t$ is the observation vector $\mathbf{z}_t$ augmented with the reward r_t. According to Bayesian methodology, the unknown parameters of the model should be marginalized out. This can be done in closed form [15], [16] by specifying an isotropic Gaussian prior on the columns of $\mathbf{C}$ to yield a multivariate Gaussian likelihood

$$p(\mathbf{Y} \mid \mathbf{S}, \mathbf{A}, \bar{\alpha}) = \frac{|\mathbf{W}|^N}{\sqrt{(2\pi)^{N(p+1)} |\mathbf{K}_Y|^{(p+1)}}} \exp\left(-\frac{1}{2}\mathrm{tr}(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{W}^2\mathbf{Y}^\top)\right) \tag{5}$$

where $\mathbf{Y} = [\mathbf{y}_2, \mathbf{y}_3, \ldots, \mathbf{y}_{N+1}]^\top$ is a sequence of joint observations-rewards, $\mathbf{K}_Y$ is a kernel matrix computed with hyperparameters $\bar{\alpha} = \{\alpha_1, \alpha_2, \ldots, \mathbf{W}\}$. The matrix $\mathbf{W}$ is diagonal and contains $p+1$ scaling factor to account for different variances in observed data dimensions. Using a unique kernel function for both states and actions, the element of the kernel matrix are $(\mathbf{K}_Y)_{i,j} = k_Y([\mathbf{s}_i, \mathbf{a}_{i-1}], [\mathbf{s}_j, \mathbf{a}_{j-1}])$. Note that for the observation case, the action is the one done at the previous time step. The kernel used for this mapping is, with $\mathbf{x}_i = [\mathbf{s}_i, \mathbf{a}_{i-1}]$, the common radial basis function:

$$k_Y(\mathbf{x}, \mathbf{x}') = \alpha_1 \exp\left(-\frac{\alpha_2}{2}\|\mathbf{x} - \mathbf{x}'\|^2\right) + \alpha_3^{-1}\delta_{\mathbf{x}\mathbf{x}'} \tag{6}$$

where hyperparameter α_1 represents the output variance, α_2 is the inverse width of the RBF which represents the smoothness of the function and α_3^{-1} is the variance of the additive noise $\mathbf{n}_y$.

For the dynamic mapping in the latent space, similar methods are applied but needs to account for the Markov property. By specifying an isotropic Gaussian prior on the columns of $\mathbf{B}$, the marginalization can be done in closed form. The resulting probability density over latent trajectories is:

$$p(\mathbf{S} \mid \mathbf{A}, \bar{\beta}) = \frac{p(\mathbf{s}_1)}{\sqrt{(2\pi)^{N(m+n)} |\mathbf{K}_X|^{m+n}}} \exp\left(-\frac{1}{2}\mathrm{tr}(\mathbf{K}_X^{-1}\mathbf{S}_{out}\mathbf{S}_{out}^\top)\right) \tag{7}$$

Here, $p(\mathbf{s}_1)$ is the initial state distribution b_1 and assumed isotropic Gaussian, $\mathbf{S}_{out} = [\mathbf{s}_2, \ldots, \mathbf{s}_{N+1}]^\top$ is the matrix of latent coordinate that represent the unobserved states. The $N \times N$ kernel matrix $\mathbf{K}_X$ is constructed from $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top$ where $\mathbf{x}_i = [\mathbf{s}_i, \mathbf{a}_i]$ with $1 \leq i \leq t-1$. The kernel function for the dynamic mapping is:

$$k_X(\mathbf{x}, \mathbf{x}') = \beta_1 \exp\left(-\frac{\beta_2}{2}\|\mathbf{x} - \mathbf{x}'\|^2\right) + \beta_3 \mathbf{x}^\top\mathbf{x}' + \beta_4^{-1}\delta_{\mathbf{x}\mathbf{x}'} \tag{8}$$

where the additional hyperparameter β_3 represents the output scale of the linear term. Since all hyperparameters are unknown and following [17], uninformative priors are applied so that $p(\bar{\alpha}) \propto \prod_i \alpha_i^{-1}$ and $p(\bar{\beta}) \propto \prod_i \beta^{-1}$. This results in a probabilistic interpretation of action-observation sequence:

$$p(\mathbf{Z}, \mathbf{r}, \mathbf{S}, \bar{\alpha}, \bar{\beta} | \mathbf{A}) = p(\mathbf{Y}|\mathbf{S}, \mathbf{A}, \bar{\alpha}) p(\mathbf{S}|\mathbf{A}, \bar{\beta}) p(\bar{\alpha}) p(\bar{\beta}) \tag{9}$$

To learn the GPDM, it has been proposed to minimize the joint negative log-posterior of the unknowns

$-\ln p(\mathbf{S}, \bar{\alpha}, \bar{\beta}, \mathbf{W}|\mathbf{Z}, \mathbf{r}, \mathbf{A})$ which is given by:

$$\begin{aligned}\mathcal{L} = {} & \frac{m+n}{2}\ln|\mathbf{K}_X| + \frac{1}{2}\mathrm{tr}(\mathbf{K}_X^{-1}\mathbf{S}_{out}\mathbf{S}_{out}^{\top}) + \frac{1}{2}\mathbf{s}_1^{\top}\mathbf{s}_1 \\ & - N\ln|\mathbf{W}| + \frac{p+1}{2}\ln|\mathbf{K}_Y| + \frac{1}{2}\mathrm{tr}(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{W}^2\mathbf{Y}^{\top}) \\ & + \sum_i \ln\alpha_i + \sum_i \ln\beta_i + C\end{aligned} \tag{10}$$

For more details on GPDM learning methods, see [12]. The resulting MAP estimates of $\{\mathbf{S}, \bar{\alpha}, \bar{\beta}, \mathbf{W}\}$ is then used to estimate the transition and observations-reward function:

$$\begin{aligned}\mathbf{s}_{t+1} &= \mathbf{k}_X([\mathbf{s}_t, \mathbf{a}_t])^{\top}\mathbf{K}_X^{-1}\mathbf{S}_{out} \\ \mathbf{y}_{t+1} &= \mathbf{k}_Y([\mathbf{s}_{t+1}, \mathbf{a}_t])^{\top}\mathbf{K}_Y^{-1}\mathbf{Y}\end{aligned} \tag{11}$$

Here $\mathbf{k}_X$ and $\mathbf{k}_Y$ denote the vectors of covariance between test point and their respective training set w.r.t. to hyperparameters $\bar{\alpha}$ and $\bar{\beta}$ respectively. Since $\mathbf{S}$ is part of a MAP estimation, the states are assumed to be the most probable under the current model and therefore the last element is considered as the current state. This model estimate combined with the latent trajectory $\mathbf{S}$ correspond to our complete belief conditioned on observations, actions and rewards.

B. Online Planning

By using the MAP estimate to update the belief of the agent conditioned on observations in the environment, we are able to address the question of how should the agent act given its current belief. To achieve this, we adopt an online planning algorithm which predicts future trajectories of actions, observations and rewards to estimate the expected return for each sampled immediate action. Each prediction is made with the corresponding Gaussian Process' mean. Therefore, we approximate the value function with a finite horizon for each predicted states. Afterward, the agent simply performs the sampled immediate action which has highest estimate. In fact, algorithm 1 has for only purpose the evaluation of the learned GP-POMDP's model:

Algorithm 1 PLANNING($\mathbf{b}$, M, E, d)

```
if $d = 0$ then Return $0, null$
$r^* \leftarrow -\infty$
for $i = 1$ to $M$ do
    Sample $\mathbf{a} \in A$ uniformly
    Predict $\mathbf{z}|\mathbf{b}, \mathbf{a}$ and $r|\mathbf{b}, \mathbf{a}$
    $\mathbf{b}' \leftarrow$ LEARNGPDM$(\mathbf{b}, \mathbf{a}, \mathbf{z}, r)$
    $r', \mathbf{a}' \leftarrow$ PLANNING$(\mathbf{b}', E, E, d-1)$
    $r \leftarrow r + \gamma r'$
    if $r > r^*$ then $(r^*, \mathbf{a}^*) \leftarrow (r, \mathbf{a})$
end for
Return $r^*, \mathbf{a}^*$
```

At time t, the agent would call the function PLANNING$(\mathbf{b}_t, M, E, D)$, where $\mathbf{b}_t$ is its current belief, M is the number of actions to sample at the first level (selection) of the tree, E is the number of actions to sample for the next levels (evaluation) of the tree and d is the depth of the tree, i.e. the planning horizon. Then, the agent would execute the action $\mathbf{a}_t = \mathbf{a}^*$ in the environment and then compute its new belief $\mathbf{b}_{t+1}$ using the function LEARNGPDM$(\mathbf{b}_t, \mathbf{a}_t, \mathbf{z}_{t+1}, r_{t+1})$ after observing $\mathbf{z}_{t+1}$ and r_{t+1}. As described earlier, learning the GPDM is done by optimizing equation (10) which returns the maximum a posteriori state sequence and hyperparameters. Then, the planning algorithm would be called again with the new belief $\mathbf{b}_{t+1}$ to choose the next action to perform, and so on.

V. EXPERIMENTS

In this paper, we investigate the problem of learning to control the height of an autonomous blimp online, without knowing its pre-defined physical models. We have opted for blimps because, in comparison to other flying vehicles, they have the advantage of operating at relatively low speed, and they keep their altitude without necessarily moving. Furthermore, they are not sensitive to control errors as in the case of helicopters for instance [18]. In fact, the problem of controlling a blimp has been studied intensively in the past, particularly in the control community. Zhang and colleagues [19] have introduced a PID controller combined to a vision system to guide a blimp. For their part, Wyteh and Barron [20] have used a reactive controller; whereas Rao et al. [21] a controller using fuzzy logic. Some other researchers have used the non-linear dynamics to control several flight phases [22], [23].

All these approaches assume prior knowledge about the dynamics or pre-defined models. Our approach does not assume such prior knowledge[1] and aims to learn the control policy directly, without passing through a priori information about the payload, the temperature, or the air pressure. In this context, the approach that is most closely to the one described here has recently been presented by Rottmann et al. [18]. They applied model-free Monte Carlo reinforcement learning and used Gaussian processes for dealing with the continuous state-action space. More precisely, they assumed a completely observable continuous state-action space and used filtered state estimates to approximate the Q-function with a Gaussian process. Conversely, our model-based approach aims to learn the model by using a Gaussian process prior and approximate the value function with the learned posterior. In fact, the blimp application is an illustrative example that sustains the applicability of the Bayesian Reinforcement Learning in Continuous POMDPs with Gaussian Processes; an idea that certainly can be used in any complex realistic application.

The goal of our experiments is to validate the use of GPDMs for online model identification and state estimation combined with a planning algorithm for online decision-making. The agent aims to maintain a target height equals to 0 using as less energy as possible. Each episode starts at zero height and zero velocity and is run for 100 time steps. The time discretization is 1 second according to the

[1] The prior is a zero mean Gaussian process.

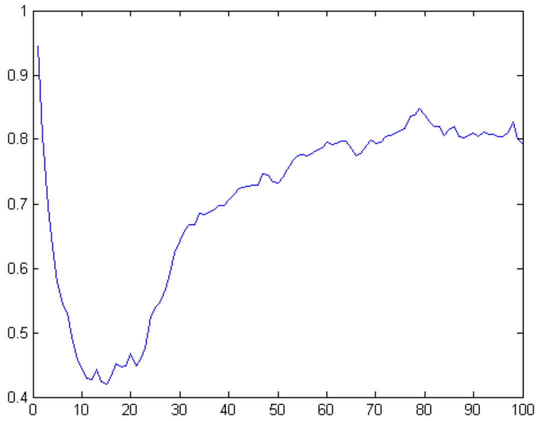

Fig. 1. Averaged received reward

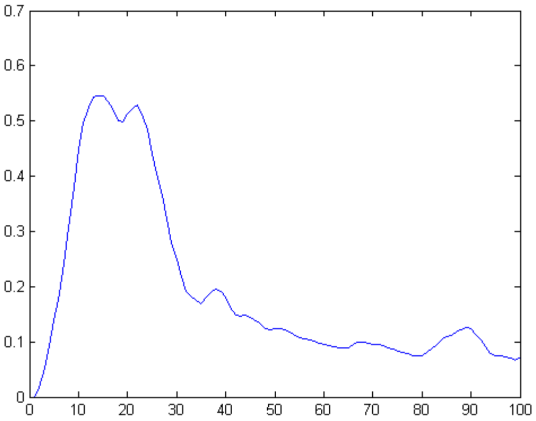

Fig. 2. Averaged distance from the target height

simulator dynamics. Observations are the height(m) and the velocity(m/s) with an additive zero-mean Gaussian noise with $1cm$ standard deviation. Rewards are corrupted with the same Gaussian noise. The blimp dynamics are simulated using a deterministic transition function where the resulting state, comprising height and velocity, is then degraded by a zero-mean isotropic Gaussian noise with $0.5cm$ standard deviation. The continuous action set is defined as $A = [-1, 1]$ where the bounds represent maximum downward and upward actions.

To learn and plan from observations and rewards, we trained a GPDM at each time step providing the planning algorithm a model and associated believed trajectory. In order to ensure a good tradeoff between exploration and exploitation, we randomly sampled actions according to a time-decreasing probability. The planning parameters were set to $M = 25$ samples for action selection level, $E = 10$ samples for state evaluation levels, $d = 3$ as tree depth and the discount factor $\gamma = 0.9$.

The reward function was crucial for our experiment since the agent has only few steps to learn and the search depth for the planning phase is limited. Consequently, the chosen reward function is set as the sum of two Gaussians. The first Gaussian having standard deviation of 1 meter gives half point for being roughly around the goal. The second one with $5cm$ standard deviation gives another half point if the blimp is only a few centimeters apart its goal. A cost for actions is also incurred to force the blimp to reach its goal with minimal energy usage. All observed rewards are corrupted by a Gaussian noise with 0.1 variance.

Figure V shows the averaged reward received by the agent. We notice that the curve stabilize around a return of 0.8. This value corresponds to the received reward when a blimp is $5cm$ appart the target height while not doing any significant action. In Figure V, we observe that the blimp averaged distance from height 0 seems to stabilize around $10cm$. Figure V shows the prediction error on the observations-reward sequence using equation (11) with the last state estimate. We defined the error as the sum of absolute errors on predicting noisy observations and rewards. Figure V shows that most trajectories have large variance at the beginning and then the variance decrease as the agent gets more observations. Each box represent the 25th and 75th percentiles, the central mark is the median and the whiskers extend to non-outlier range. A comparison with the random policy, which is not reported, showed that this policy rapidly diverge to over 3 meters from the goal.

VI. Discussion

Making decision in stochastic and partially observable environments with unknown model is a very important problem because a parametric model is often difficult to specify. To address this problem, we proposed a continuous formulation of the POMDP where the model's functions are assumed to be Gaussian processes. This assumption allowed us to use the work on Gaussian Process Dynamical Model [12] by adding the actions and rewards parts to fulfil POMDP's requirement. Although the agent had no prior information, our experimental results on the blimp problem show that the learned model is able to provide useful predictions for our online planning algorithm. Considering the chosen noises' variance, most agents achieved reasonable performance by receiving good averaged rewards over time.

However, comparing our approach to previous ones on the basis of control performance is hard, since the model is learned and only very few data are used to do so. To compare the approaches under control performances criterions, two points should be considered before. First, learning from larger data sets using sparsification techniques would surely improve prediction performance. Second, once the model is learned, adding filtering methods instead of the maximum a posteriori estimate will also help in the decision process.

Indeed, the maximum a posteriori estimation of the state trajectory limits the learning efficiency. While using Gaussian distribution on state trajectory could provide better results, it also entails learning with uncertain data sets which map Gaussian inputs to Gaussian outputs. Such learning

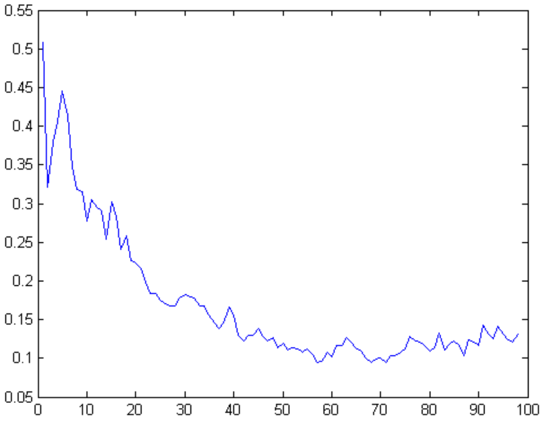

Fig. 3. Averaged prediction error

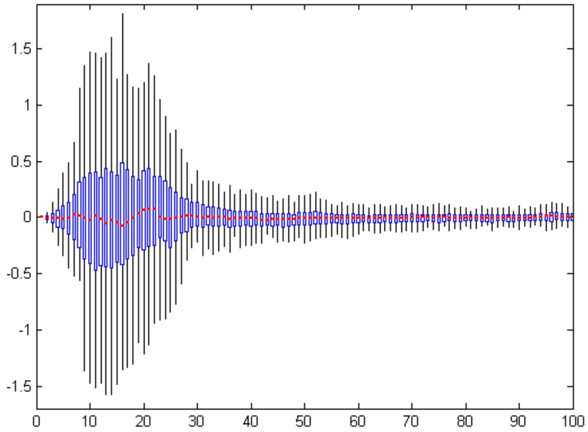

Fig. 4. Box plot of the heights distributions

procedure is one of the interesting research avenue. Moreover, as we proposed a reinforcement learning algorithm which use bayesian learning for the model, its uncertainty should be used in the planning phase to get an exploration-exploitation trade-off. Further improvements include learning from different trajectories as well as finding better and separate kernel functions for state and actions, especially when the transition function is stochastic while actions are noise-free, as in our example. At last, it is possible to use the learning procedure for model evaluation by changing the Gaussian processes' prior mean by the functions to evaluate.

To conclude, results on the blimp control task are promising. Further investigations on bayesian reinforcement learning in continuous POMDPs with Gaussian processes are currently in progress. A harder control problem is currently considered using a Gaussian approximation instead of a simple point estimates for the state trajectory estimation. The posterior distribution over the models' functions is thus computed from infered sequences of belief states.

References

[1] M. Duff, "Optimal Learning: Computational procedures for Bayes-adaptive Markov decision processes," Ph.D. dissertation, University of Massachusetts Amherst, Amherst, MA, 2002.

[2] T. Wang, D. Lizotte, M. Bowling, and D. Schuurmans, "Bayesian sparse sampling for on-line reward optimization," in *Proceedings of the 22nd international conference on Machine learning (ICML)*, 2005, pp. 956–963.

[3] P. S. Castro and D. Precup, "Using linear programming for bayesian exploration in markov decision processes," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, 2007, pp. 2437–2442.

[4] P. Poupart, N. Vlassis, J. Hoey, and K. Regan, "An analytic solution to discrete bayesian reinforcement learning," in *Proceedings of the 23rd international conference on Machine learning (ICML)*, 2006, pp. 697–704.

[5] F. Doshi, J. Pineau, and N. Roy, "Reinforcement learning with limited reinforcement: using Bayes risk for active learning in POMDPs," in *Proceedings of the 25th international conference on Machine learning*. ACM New York, NY, USA, 2008, pp. 256–263.

[6] S. Ross, B. Chaib-draa, and J. Pineau, "Bayes-adaptive pomdps," in *Advances in Neural Information Processing Systems 20*, 2008, pp. 1225–1232.

[7] ——, "Bayesian reinforcement learning in continuous pomdps with application to robot navigation," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, 2008, pp. 2845–2851.

[8] S. Thrun, "Monte Carlo POMDPs," *Advances in Neural Information Processing Systems*, vol. 12, pp. 1064–1070, 2000.

[9] J. Porta, M. Spaan, and N. Vlassis, "Robot planning in partially observable continuous domains," *Robotics: Science and Systems, MIT, Cambridge, MA*, 2005.

[10] M. Spaan and N. Vlassis, "Perseus: Randomized point-based value iteration for POMDPs," *Journal of Artificial Intelligence Research*, vol. 24, pp. 195–220, 2005.

[11] J. Porta, N. Vlassis, M. Spaan, and P. Poupart, "Point-Based Value Iteration for Continuous POMDPs," *The Journal of Machine Learning Research*, vol. 7, pp. 2329–2367, 2006.

[12] J. Wang, D. Fleet, and A. Hertzmann, "Gaussian Process Dynamical Models for Human Motion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 283–298, 2008.

[13] A. O'Hagan, "Some Bayesian numerical analysis," *Bayesian Statistics*, vol. 4, pp. 345–363, 1992.

[14] C. Williams and D. Barber, "Bayesian classification with Gaussian processes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 12, pp. 1342–1351, 1998.

[15] D. J. C. Mackay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[16] R. M. Neal, *Bayesian Learning for Neural Networks (Lecture Notes in Statistics)*, 1st ed. Springer, August 1996.

[17] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models." in *Advances in Neural Information Processing Systems 18*. The MIT Press, 2006, pp. 1441–1448, proc. NIPS'05.

[18] A. Rottmann, C. Plagemann, P. Hilgers, and W. Burgard, "Autonomous blimp control using model-free reinforcement learning in a continuous state and action space," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 2007, pp. 1895–1900.

[19] H. Zhang and J. Ostrowski, "Visual servoing with dynamics: control of an unmanned blimp," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 1, 1999.

[20] G. Wyeth and I. Barron, "An Autonomous Blimp," in *Proc. IEEE Int Conf. on Field and Service Robotics*, 1997, pp. 464–470.

[21] J. Rao, Z. Gong, J. Luo, and S. Xie, "A flight control and navigation system of a small size unmanned airship," in *Mechatronics and Automation, 2005 IEEE International Conference*, vol. 3, 2005.

[22] S. Gomes and J. Ramos Jr, "Airship dynamic modeling for autonomous operation," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 4, 1998.

[23] E. Hygounenc, I. Jung, P. Soueres, and S. Lacroix, "The Autonomous Blimp Project of LAAS-CNRS: Achievements in Flight Control and Terrain Mapping," *International Journal of Robotics Research*, vol. 23, no. 4, pp. 473–511, 2004.