Continuous-State POMDPs with Hybrid Dynamics

Emma Brunskill, Leslie Kaelbling, Tomas Lozano-Perez, Nicholas Roy
Computer Science and Artificial Laboratory
Massachusetts Institute of Technology
Cambridge, MA
emma,lpk,tlp,nickroy@csail.mit.edu

Abstract

Continuous-state POMDPs provide a natural representation for a variety of tasks, including many in robotics. However, existing continuous-state POMDP approaches are limited by their reliance on a single linear model to represent the world dynamics. We introduce a new switching-state (hybrid) dynamics model that can represent multi-modal state-dependent dynamics. We present a new point-based POMDP planning algorithm for solving continuous-state, discrete-observation POMDPs using this dynamics model and approximate the value function as a mixture of a bounded number of Gaussians. We compare our hybrid dynamics model approach to a linear dynamics continuous-state planner and a discrete-state POMDP planner and show that in some scenarios we can outperform such techniques.

Introduction

Partially observable Markov decision processes (POMDPs) (Sondik 1971) provide a rich framework for describing a number of planning problems that arise in situations with hidden state and stochastic actions. Most prior work has focused on solving POMDPs with discrete states, actions and observations.

However, in many applications, such as navigation or robotic grasping, the world is most naturally represented using continuous states. Though any continuous domain can be described using a sufficiently fine grid, the number of discrete states grows exponentially with the underlying state space dimensionality. Existing discrete state POMDP algorithms can only scale up to the order of a hundred thousand states, beyond which they become computationally infeasible (Pineau, Gordon, and Thrun 2006; Spaan and Vlassis 2005; Smith and Simmons 2005). Therefore, approaches for dealing efficiently with continuous-state POMDPs are of great interest.

Previous work on planning for continuous-state POMDPs has typically modeled the world dynamics using a single linear Gaussian model[1] to describe the effects of an action(Brooks et al. 2006; Porta et al. 2006; Thrun 2000). Unfortunately, this model is not powerful enough to represent the multi-modal state-dependent dynamics that arise in a number of problems of interest such as robotic traversal over varying terrain. Though such dynamics are easily represented in discrete-state environments using the standard transition matrices, a single linear Gaussian continuous-state model will be insufficient to adequately model these multi-modal state-dependent dynamics. In this paper we present a hybrid dynamics model that can represent a stochastic distribution over a set of different linear dynamic models.

We develop a new point-based approximation algorithm for solving such hybrid-dynamics POMDP planning problems that builds on Porta et al.'s continuous-state point-based approach (2006). We first develop our algorithm under the assumption that the observations take on a finite set of discrete values. Within our algorithm we provide an heuristic projection method to maintain a bounded value function representation. We compare our approach to using a simpler linear Gaussian dynamics model in a continuous-state planner, and to a discrete-state planner, and show that in some scenarios our approach outperforms both other techniques. We also present preliminary work on extending our algorithm to handle a continuous observation model where observations are a noisy Gaussian function of the true state.

POMDPs

Partially observable Markov decision processes (POMDPs) are a popular model for decision making under uncertainty in artificial intelligence (Kaelbling, Littman, and Cassandra 1998). A POMDP consists of: a set of states S; a set of actions A; a set of observations Z; a dynamics model that represents the probability of making a transition to state s' after taking action a in state s, $p(s'|s,a)$; an observation model describing the probability of receiving an observation z given the state s' and prior action a , $p(z|s',a)$; a reward model that specifies the reward received from being in state s and taking action a, $R(s,a)$; the discount factor to trade off the value of immediate and future rewards, γ; and an initial belief state distribution, b_o.

A belief state b_t is used to summarize the probability of the world being in each state given the past history of observations and actions $(o_{1:t}, a_{1:t})$. A policy $\pi : b \rightarrow a$ maps belief states to actions. The goal of POMDP planning techniques is to construct a policy that maximizes the (possibly

Copyright © 2008, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

[1] In some prior work(Brooks et al. 2006; Thrun 2000) a special exception is included to encode boundary conditions such as obstacles in a robotic task.

discounted) expected sum of rewards $E[\sum_{t=1}^{T} \gamma^t R(s_t, a_t)]$ over an action sequence of length T. The policy is often found by computing this expected reward using a value function over the space of belief states. When S, A, and Z are discrete, Sondik (1971) showed that the optimal finite horizon value function can be represented by a set of α-vectors ($\alpha(s)$) and is piecewise linear and convex (PWLC).

In exact POMDP solutions, the number of α vectors required to represent the value function often grows exponentially with the length of the horizon. One of the most successful classes of approximation techniques is point-based value iteration (Pineau, Gordon, and Thrun 2006; Spaan and Vlassis 2005; Zhang and Zhang 2001). Point-based techniques estimate the value function at only a small set of N chosen belief points $\tilde{B}$, resulting in a value function represented by at most N α-vectors. This representation is constructed by iteratively computing an approximately optimal t-step value function V_t from the previously-computed $(t-1)$-step value function V_{t-1} by backing up the value function at beliefs $b \in \tilde{B}$ using the Bellman equation:

$$V_t(b) = \max_{a \in A} \sum_{s \in S} R(s,a)b(s) + \dots$$

$$\gamma \sum_{z \in Z} \max_{\alpha_{t-1} \in V_{t-1}} \sum_{s \in S} \sum_{s' \in S} p(s'|s,a)p(z|s')\alpha_{t-1}(s')b(s)$$

$$= \max_{a \in A} \left[\sum_{s \in S} R(s,a)b(s) + \gamma \sum_{z \in Z} \max_{\alpha_{azj}} \langle \alpha_{azj}, b \rangle \right] \quad (1)$$

where α_{azj} is the vector associated with taking action a, receiving observation z and then following the $(t-1)$ policy associated with α_j. This approximate representation is guaranteed to be a lower bound on the optimal value function.

Switching State-Space Dynamics Models

Our interest lies in using the rich framework of POMDPs to handle continuous-state problems directly without converting to a discrete representation. One critical representational issue is how to flexibly represent the dynamics in a continuous-state system. In general the dynamics of robotic grasping and many other problems of interest are highly complex and nonlinear. However we can approximate such dynamics using a variant of a "switching state space" model (SSM). SSMs (also known as hybrid models and jump-linear systems) are a popular model in the control community for approximating systems with complex dynamics (see for example (Ghahramani and Hinton 2000)). A SSM will allow us to both represent actions that result in multi-modal stochastic distributions over the state space, and succinctly represent any shared dynamics among states. See Figure 1 for an illustration of a typical SSM and the particular one we use in this work.

In order to model systems involving multi-modal, state-dependent dynamics (such as dynamics on sand vs concrete), our SSM model conditions discrete mode transitions based on the previous continuous state, and may be expressed as

$$p(s'|s,a) = \sum_{h} p(s'|s, m' = h)p(m' = h|s,a) \quad (2)$$

where s, s' are the continuous states at time t and $t+1$ respectively, a is the discrete action taken at time t, and m' is the discrete mode at time $t+1$. In this paper we will assume that for each action a the hidden mode m can take on one of H values. Each mode value h and action a is associated with a linear Gaussian model $\mathcal{N}(s'; \zeta_{ha}s + \beta_{ha}, \sigma^2_{ha})$. For mathematical convenience we model the conditional probability of a mode taking on a particular value h given the previous state s and action a using a weighted sum of F Gaussians

$$p(m' = h|s,a) = \sum_{f=1}^{F} w_{fha} \mathcal{N}(s; \mu_{fha}, \sigma^2_{fha}). \quad (3)$$

Note that for finite H it is impossible to select the parameters $w_{fha}, \mu_{fha}, \sigma^2_{fha}$ such that the sum of probability of the next mode state m' taking on any value for a given state s, $\sum_h p(m' = h|s,a)$, sums to 1 for all states s. Therefore in practice we will choose models that approximately sum to 1 over all the states of interest in a particular experimental domain. We choose to use this slightly awkward representation rather than normalizing the distribution across modes (such as by using a softmax function) because it allows closed form updates of the belief state and value function.

Substituting equation 3 into equation 2, the full dynamics model is a sum of HF Gaussian products:

$$p(s'|s,a) = \sum_{h=1}^{H} \mathcal{N}(s'; \zeta_{ha}s + \beta_{ha}, \sigma^2_{ha}) \sum_{f=1}^{F} w_{fha} \mathcal{N}(s; \mu_{fha}, \sigma^2_{fha}).$$

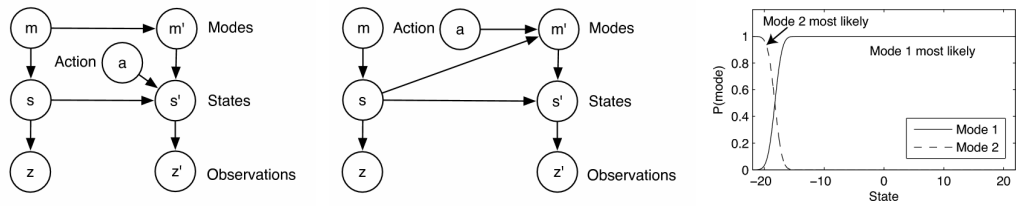
An added benefit of this model is that it can flexibly represent relative transitions (transitions that are an offset from the current state, by setting $\zeta \neq 0$) and absolute transitions (transitions that go to some arbitrary global state, by setting $\zeta = 0$ and $\beta \neq 0$). This allows the model to compactly represent domains in which many states share the same relative or absolute transition dynamics.

POMDP Planning with Hybrid Models

We now describe a new planning algorithm for POMDPs with hybrid dynamics models. Recently Porta et al. (2006) showed that for a continuous-state space S and discrete actions A and observations Z, the optimal finite horizon value function is PWLC and may be represented by a finite set of α-functions[2]. Therefore point-based approaches to continuous state POMDPs that exactly represent the α-functions will also provide a lower bound on the value function. Porta et al.'s algorithm provides an approximation of this lower bound: our algorithm is inspired by theirs and handles multi-modal state-dependent dynamics.

For clarity we will explain the mathematics for a one-dimensional state space, but it is easily extended to higher dimensions. We assume the reward function $r(s,a)$ is expressed as a sum of G Gaussian components for each action a, $r(s,a) = \sum_{g=1}^{G} w_{ag} \mathcal{N}(s; \mu_{ag}, \sigma^2_{ag})$, and each discrete observation $z \in Z$ is expressed as a sum of L Gaussian components $p(z|s) = \sum_{l=1}^{L} w_{zl} \mathcal{N}(s; \mu_{zl}, \sigma^2_{zl})$ such

[2]The expectation operator $\langle f, b \rangle$ is a linear function in the belief space and the value function can be expressed as the maximum of a set of these expectations: for details see (Porta et al. 2006).



(a) Typical SSM (b) Our Dynamics Model (c) Example of mode model

Figure 1: Switching State-Space Models

that $\forall s \sum_z p(z|s) = 1$. Here we have assumed an observation model very similar to the mode representation (equation 3) and the same comments made for that choice apply here to the observation model. We choose to represent the belief states b and α-functions using weighted sums of Gaussians. Each belief state b is a sum of D Gaussians $b(s) = \sum_{d=1}^{D} w_d \mathcal{N}(s; \mu_d, \sigma_d^2)$, and each α_j, the value function of policy tree j, is represented by a set of K Gaussians $\alpha_j(s) = \sum_k w_k \mathcal{N}(s; \mu_k, \sigma_k^2)$. Recall that for each action a, there are H modes and F Gaussian components per mode.

We do not lose expressive power by choosing this representation because a weighted sum of a sufficient number of Gaussians can approximate any continuous function on a compact interval (and our domains of interest are closed and bounded and therefore fulfill the criteria of being compact). But, of course, we will be effectively limited in the number of components we can employ, and so in practice our models and solutions will both be approximations.

Belief Update and Value Function Back Ups

Point-based POMDP planners must include a method for backing up the value function at a particular belief b (as in equation 1) and for updating the belief state b after a new action a is taken and a new observation z is received. Choosing a weighted sum of Gaussians representation allows both computations to be performed in closed form.

The belief state is updated using a Bayesian filter:

$$\begin{aligned} b^{a,z=i}(s) &= p(s'|z=i,a,b) \\ &\propto p(z=i|s',a,b)p(s'|a,b) \\ &= p(z=i|s')p(s'|a,b) \end{aligned}$$

where the last equality holds due to the Markov assumption. Substituting in the dynamics and observation models, and normalizing such that $\int_s b^{a,z}(s)ds = 1$, we get

$$b^{a,z=i}(s) = \sum_{dfhl} d_{dfhl} N(s|\mu_{dfhl}, \sigma^2_{dfhl})$$

where d_{dfhl} is the (scalar) weight of the $dfhl$-th component. Hence the representation of the belief as a mixture of Gaussians is closed under belief update: note however that the number of components to represent the belief has increased by a factor of FHL.

The other key computation is to be able to back up the value function for a particular belief. Since we also use discrete actions and observations, this can be expressed as a slight modification to equation 1 by replacing sums with integrals and writing out the expectation:

$$\begin{aligned} V_t(b) =& \max_{a\in A} \int_{s\in S} R(s,a)b(s)ds + \gamma \sum_{z\in Z} \max_{\alpha_{azj}} \int_s \alpha_{azj} b(s) ds \\ =& \langle \max_{a\in A} R(s,a) + \gamma \sum_{z\in Z} \max_{\alpha_{azj}} \alpha_{azj}, b \rangle \end{aligned}$$

where we have used the inner-product operator $\langle f, b\rangle$ as shorthand for expectation to obtain the second equality. As stated previously, α_{azj} is the α-function for the conditional policy corresponding to taking action a, receiving observation z and then following the previous $t-1$-step policy tree α_j, and can here be expressed as

$$\alpha_{azj}(s) = \int_{s'} \alpha_{j,t-1}(s') p(z|s') p(s'|s,a) ds'.$$

Substituting in all the chosen representations yields

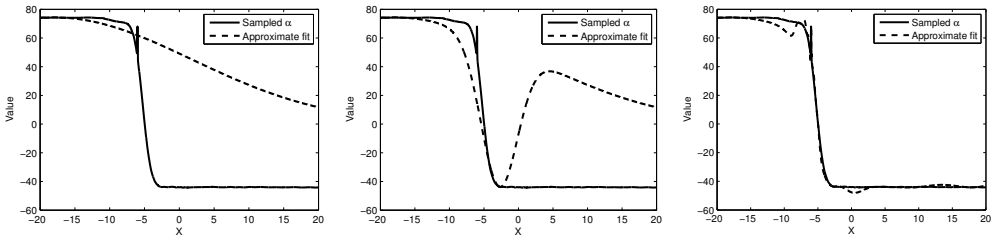
$$\begin{aligned} \alpha_{azj}(s) =& \int_{s'} \sum_{k=1}^{K} w_k \mathcal{N}(s'; \mu_k, \sigma_k^2) \sum_{l=1}^{L} w_l \mathcal{N}(s'; \mu_l, \sigma_l^2) \dots \\ & \sum_{h=1}^{H} \mathcal{N}(s'; \zeta_{ha} s + \beta_{ha}, \sigma^2_{ha}) \sum_{f=1}^{F} w_{fha} \mathcal{N}(s; \mu_{fha}, \sigma^2_{fha}) ds' \\ =& \sum_{f=1}^{F} \sum_{h=1}^{H} \sum_{k=1}^{K} \sum_{l=1}^{L} w_{fha} w_k w_l \mathcal{N}(s; \mu_{fha}, \sigma^2_{fha}) \dots \\ & \int_{s'} \mathcal{N}(s'; \mu_k, \sigma_k^2) \mathcal{N}(s'; \mu_l, \sigma_l^2) \mathcal{N}(s'; \zeta_{ha} s + \beta_{ha}, \sigma^2_{ha}) ds'. \end{aligned}$$

To perform the integral we combine the three Gaussians inside the integrand into a single Gaussian which is a function of s' and other terms that are independent of s'. Integrating over s' yields

$$\alpha_{azj} = \sum_{f=1}^{F} \sum_{h=1}^{H} \sum_{k=1}^{K} \sum_{l=1}^{L} w_{fhkl} \mathcal{N}(s; \mu_{fhkl}, \sigma^2_{fhkl})$$

where

$$\begin{aligned} w_{fhkl} =& w_{fha} w_k w_l \mathcal{N}(s_l; s_k, \sigma_k^2 + \sigma_l^2) \cdot \dots \\ & \mathcal{N}(\mu_{fha}; \frac{c - \beta_{ha}}{\gamma_{ha}}, \sigma^2_{fha} + \frac{C + \sigma^2_{ha}}{\gamma^2_{ha}}), \\ C = \frac{\sigma_l^2 \sigma_k^2}{\sigma_l^2 \sigma_k^2},\ \mu_{fhkl} =& \frac{(C + \sigma^2_{ha})\mu_{fha} + \sigma^2_{fha} \gamma_{ha} (c - \beta_{ha})}{\sigma^2_{fha}(C + \sigma^2_{ha})}, \\ \sigma^2_{fhkl} =& \frac{\sigma^2_{fha}(C + \sigma^2_{ha})}{C + \sigma_h^2 + \sigma^2_{fha} \gamma^2_{ha}},\ c = \frac{\mu_k \sigma_l^2 + \sigma_k^2 \mu_l}{\sigma_l^2 \sigma_k^2}. \end{aligned}$$



(a) 1 component (b) 2 components (c) 15 components

Figure 2: Approximating the α-functions. We use a simple residual fitting technique. First the function is sampled. Then a Gaussian component is added to reduce the highest component. This process is repeated for a set number of iterations.

The new α_{azj} now has $F \times H \times K \times L$ components, compared to the step $t-1$ α_j, policy tree, which only had K components. To finish the value function backup, we substitute α_{azj} back into equation 4 and choose the policy tree α that maximizes the future expected reward $\langle \alpha, b \rangle$ of belief b

$$\begin{aligned}\alpha(s) &= \max_a R(s,a) + \gamma \sum_{z=1}^{|Z|} \max_{\alpha_{azj}} \alpha_{azj} \\ &= \max_a \left[\sum_{g=1}^{G} N(s|\mu_g, \sigma_g^2) + \gamma \sum_{z=1}^{|Z|} \max_{\alpha_{azj}} \alpha_{azj} \right]\end{aligned}$$

Since all elements in this result are weighted sums of Gaussians, the α function stays in closed form. Note that had we utilized a softmax distribution for the observation model or mode probabilities that it would not be possible to iteratively perform the integrals in closed form.

Unfortunately, the number of Gaussian components in a single α function has greatly increased: from K to $G + |Z|FHKL$ components. Compared to previous approaches (Porta et al. 2006) the new dynamics model has introduced an extra factor of FH components. The number of components therefore scales exponentially in the time horizon with base $|Z|FHL$.

Approximating α Functions

It is not computationally feasible to maintain all components over multiple backups. Instead, by carefully combining the components generated after each backup, we maintain a bounded set of α functions. Since α functions represent the value of executing a particular policy tree over the entire belief space, it is important to make the approximation as close as possible throughout the belief space.[3] To reduce the number of components used to represent the belief states and α functions, Porta et al. use a slight variant of Goldberger and Roweis's method (2005) that minimizes the Kullback-Leibler (KL) distance between the original model $f(x)$ and the approximation $\tilde{f}(x)$

$$D_{KL}(f||\tilde{f}) = \int_x f(x) \log \frac{f(x)}{\tilde{f}(x)} dx. \quad (4)$$

[3]Ideally we could restrict this approximation to the reachable belief space; however analyzing the reachable belief space in continuous-state POMDPs will be an area of future work.

However, the KL distance is not particularly appropriate as a distance measure for quantities that are not probability distributions. It also can result in poor approximations in parts of the space where the original function has small values since if $f(x)$ is zero then regardless of the value of $\tilde{f}(x)$ the distance for that x is always zero.

Ideally we would like to find the projection that minimizes the max norm error $||\tilde{\alpha} - \alpha||_\infty$: minimizing this norm would allow us to bound the worst approximation error we induce by approximating the α-functions by a smaller number of components. This in turn would enable us to bound the overall error in the final value function due to performing projections potentially at every backup.

Unfortunately there is no closed form expression for the max norm between two Gaussian distributions. Instead we employ a fast heuristic method which seeks to minimize the max norm at a subset set of points lying along the value function. We first sample the α-function at regular intervals along the state space. These samples are considered potential residuals r. First the largest magnitude residual (either positive or negative) is found and a Gaussian is placed at this point, with a height equal to the residual and variance approximated by estimating the nearby slope of the function. This Gaussian is added to the set of new components. Then a new set of residuals is computed by estimating this Gaussian along the same intervals of state space, and subtracting its value from the original residuals: $r = r - w_i \mathcal{N}(s|\mu_i, \Sigma_i)$. This process is repeated until the number of new Gaussians reaches the fixed size of our representation for the α-functions. See Figure 2 for an illustration of this process.

Note that at each round during this process we are adding a Gaussian component at the location which currently has the max norm error at the sampled points. Therefore this algorithm takes a greedy approach to minimizing the max norm between the approximate function $\tilde{\alpha}$ and the original α-function along the set of sampled points. The sampling prevents any formal guarantees on the actual max norm between the original and approximated function unless further assumptions are made on the minimum variance of any potential α components. However, a small max norm error on the sampled points in practice often indicates that there is a small max norm between α and $\tilde{\alpha}$. Experimentally, this algorithm was fast and provided good approximations.

Planning

We now have the major components necessary to apply a point-based approach to POMDP planning. Inspired by results in discrete-state planners (Smith and Simmons 2005; Shani, Brafman, and Shimony 2007) we create a belief set B by sampling belief states in a set of short trajectories, though in our case we simply perform random actions from an initial starting belief state b_0. We initialize the value function as a single α-function. Starting with this initial α-function, we iteratively perform value function backups. At each round, we select a trajectory from our belief set and back up the beliefs in that trajectory in reverse order. This approach was found to be superior to randomly backing up belief points.

Power Outlet Localization

We demonstrate our approach on a small problem with state-dependent dynamics. Though it is a small problem, a straight forward implementation of a linear Gaussian continuous-state planner (such as Porta et al.'s approach) fails in this domain since its dynamics models do not vary according to the state of the agent. It is also interesting because HSVI2, a leading discrete POMDP planner, struggles with this domain since a fine discretization is required in parts of the state space to find good policies.

In this experiment a robot must navigate a long corridor ($s \in$ [-21,21]) to find a power socket located at -16.2. The robot's actions are almost deterministic and involve moving left or right using small or large steps that transition it over by 0.1 or 5.0 respectively (with standard deviation 0.01). However, if the robot is too close to the left or right walls (located at -21 and 21) it will bump into the wall. Therefore the robot's dynamics are state-dependent, since the result of an action depends on the robot's initial location. Plugging in keeps the robot at the same location. All movement actions receive a reward of 0.05. If the robot plugs itself in at the right location it receives a large positive reward (modeled by a highly peaked Gaussian); otherwise it receives a lesser reward of 5.8. The power supply lies beneath the robot sensors so the robot is effectively blind: however, since the robot knows the environment map and dynamics model, it can take actions to effectively localize itself and then navigate to the power outlet.

We compared three planning approaches for this task: Smith and Simmons(2005)'s state of the art discrete-state planner HSVI2, a linear-model continuous-state planner, and our hybrid-model continuous-state planner. The two continuous-state planners were run on 100 trajectories (leading to a total of 1000 belief points) gathered by starting in a random state $s \in [-21, 21]$ with a Gaussian approximately uniform belief and acting randomly for episodes of 10 steps. The policies produced were evaluated by the average sum of rewards received over 100 episodes of 50 steps/episode using the computed policy. See Table 1 for results.

Our hybrid dynamics model finds a good policy that involves taking steps to the left until the agent knows it is very likely to have hit the left wall, and then taking a few more steps to reach the power socket. The linear dynamics model continuous-state POMDP runs faster but fails to find a good policy since it cannot model that the dynamics change when the robot is near the wall, and since there are no unique observations, it does not have a way of localizing the agent. The discrete-state solutions does poorly at coarse granularities since the $PlugIn$ reward gets washed out by averaging over the width of a too wide state. At fine state granularities the discrete approach finds a good policy but requires more time: our continuous-state planner finds a solution much faster than the coarsest discrete-state planner that can find a good solution. It is also important to note that it is hard to determine *a priori* what level of discretization is required for a discrete-state planner to find a good policy, and a conservatively fine discretization can result in a significantly longer planning time.

Models	Continuous-state		Discretized (Number of States)		
	Linear	Hybrid	840	1260	2100
Time(s)	44	**548**	577	**23262**	75165
Reward	290	465	290	484	484

Table 1: Power Supply Experiment Results

Continuous Observations

One motivation for investigating continuous-state models is the desire to scale to higher dimensional problems. Our current approach still struggles with finding a compact representation of the dynamics and observation models; indeed, so far the simplest method employed to ensure these models are proper probability models requires effectively tiling the space with uniformly spaced Gaussian components. This method will naturally fail to scale gracefully to higher dimensional problems.

However, we may gain some traction by moving to a continuous observation model. In particular, we choose to examine the situation where observations are a noisy Gaussian function of the true underlying state. This is a departure from prior work by Porta et al. (2006) who used kernel smoothing over a set of training observation-state tuples to define their continuous observation model. By using a simpler model we hope to gain scalability, while still using a representation that has been widely and successfully used in the robotics community. Our model also draws from Hoey and Poupart's work (2005) on solving POMDPs with continuous observations in a discrete-state domain.

This new representation makes only a small change to a belief update of a belief state b given an action a and observation z. The observation model $p(z|s')$ is simply a single multi-dimensional Gaussian, versus in our earlier discrete-observation representation, this $p(z|s')$ was represented by a weighted sum of Gaussians. The calculations therefore are largely identical, with the exception that the number of components needed to represent the belief state grows slower since now only the dynamics model representation causes an increase in the number of belief state components.

However, we also need to be able to back up the value function, and this is more challenging since there is now an infinite number of possible observations. We therefore approximate this integral by performing sampling (as did Hoey

and Poupart (2005) in discrete-state spaces). We now derive a new α-function backup for our continuous observation model, and then discuss its properties.

First recall the value function backup equation

$$V(b) = \max_a \int_s R(s)b(s)ds + ... \\ \gamma \int_s \int_z \max_{\alpha_{n-1}} \int_{s'} p(s'|s,a)p(z|s')\alpha_{n-1}(s')ds'dzb(s)ds$$

We next multiply and divide by $p(z|a,b) \equiv \int_{s'''} p(z|s''') \int_{s''} p(s'''|s'',a)b(s'')ds''ds'''$ and switch the order of integration:

$$V(b) = \max_a \int_s R(s)b(s)ds + \gamma \int_{s'} \int_z \max_{\alpha_{n-1}} p(z|b,a) \cdot ... \\ \frac{\alpha_{n-1}(s')p(z|s') \int_s b(s)p(s'|s,a)p(z|s')ds}{p(z|b,a)} dzds'.$$

We can condense this expression by recognizing that part of it is simply the belief update equation ($b^{a,z}(s') = p(z|s') \int_s b(s)p(s'|s,a)ds/p(z|b,a)$):

$$V(b) = \max_a \int_s R(s)b(s)ds + ... \\ \gamma \int_{s'} \int_z \max_{\alpha_{n-1}} p(z|a,b)\alpha_{n-1}(s')b^{az}(s')dzds'.$$

We will approximate the expectation over all observations by sampling a set of observations from the distribution $p(z|a,b)$and average over the value of the resulting samples. Sampling provides an estimation of $p(z = z_i|a,b) = \frac{1}{N_z} \sum_{z_i} \delta_{z_i}(z)$ where N_z is the number of samples.

$$V(b) = \max_a \int_s R(s)b(s)ds + \frac{\gamma}{N_z} \int_{s'} \sum_{z_i} \max_{\alpha_{n-1}} \alpha_{n-1}(s')b^{az_i}(s')ds'.$$

Note that the $p(z|a,b)$ term has disappeared due to the sampling approximation of the expectation.

Now re-express b^{az} as the belief update of b and then reverse the orders of integration and summation:

$$V(b) = \max_a \int_s b(s)[R(s) + \\ \frac{\gamma}{N_z} \sum_{z_i} \max_{\alpha_{n-1}} \frac{\int_{s'} \alpha_{n-1}(s')p(z_i|s')p(s'|s,a)ds'}{p(z_i|b,a)}]ds.$$

We define the term in the brackets as a backed up α-function

$$\alpha_n^c \equiv R(s) + \frac{\gamma}{N_z} \sum_{z_i} \max_{\alpha_{n-1}} \frac{\int_{s'} \alpha_{n-1}(s')p(z_i|s')p(s'|s,a)ds'}{p(z_i|b,a)}$$

where we have used α_n^c to denote that this function is associated with continuous observations.

There are a couple interesting things to note about this expression. First is that due to our chosen representations (Gaussian observation model, weighted sum of Gaussians for the dynamics model, reward model, and belief states) this expression is, like in our discrete observation model, an integral over a weighted sum of Gaussians, and can therefore be computed in closed form. In addition, recall that in discrete-observation α-function backups, the observation model increased the number of components in the backed up α function by a factor of$|Z|$*number of components/observation where this second term typically scales exponentially with the state space dimension. In contrast, by using a Gaussian observation model the number of components created in a backup increases by a factor of the number of samples used $|Z_i|$. The critical issue is how many samples are necessary to get a good approximation. We suspect that the number of samples needed may scale better with increasing dimensionality than the number of components needed to specify a naive discrete observation model. We are currently investigating this issue.

Conclusion

We have presented a novel algorithm for solving continuous-state POMDPs and demonstrated its ability to handle problems involving multi-modal dynamics models. In the future we intend to investigate how the continuous-observation approach scales to higher dimensional problems.

References

Brooks, A.; Makarenko, A.; Williams, S.; and Durrant-Whyte, H. 2006. Parametric POMDPs for planning in continuous state spaces. *Robotics and Autonomous Systems*.

Ghahramani, Z., and Hinton, G. 2000. Variational learning for switching state-space models. *Neural Computation* 12:831–864.

Goldberger, J., and Roweis, S. 2005. Hierarchical clustering of a mixture model. In *NIPS*.

Hoey, J., and Poupart, P. 2005. Solving pomdps with continuous or large discrete observation spaces. In *IJCAI*.

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

Pineau, J.; Gordon, G.; and Thrun, S. 2006. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research* 27:335–380.

Porta, J.; Spaan, M.; Vlassis, N.; and Poupart, P. 2006. Point-based value iteration for continuous POMDPs. *Journal of Machine Learning Research* 7:2329–2367.

Shani, G.; Brafman, R.; and Shimony, S. 2007. Forward search value iteration for POMDPs. In *IJCAI*.

Smith, T., and Simmons, R. 2005. Point-based pomdp algorithms: Improved analysis and implementation. In *UAI*.

Sondik, E. J. 1971. *The Optimal Control of Partially Observable Markov Processes*. Ph.D. Dissertation, Stanford University.

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

Thrun, S. 2000. Monte carlo POMDPs. In *NIPS*.

Zhang, N., and Zhang, W. 2001. Speeding up the convergence of value iteration in partially observable markov decision processes. *Journal of Artificial Intelligence Research* 14:29–51.