Bayesian Optimisation for solving Continuous State-Action-Observation POMDPs

Philippe Morere
The University of Sydney
philippe.morere@sydney.edu.au

Roman Marchant
The University of Sydney
r.marchant@sydney.edu.au

Fabio Ramos
The University of Sydney
framos@cs.usyd.edu.au

Abstract

Decision making under uncertainty is a challenging task, especially when dealing with complex realistic scenarios. The *Partially Observable Markov Decision Process* (POMDP) framework, designed to solve this problem, was subject to much work lately. Most POMDP solvers, however, focus on planning in discrete state, action and/or observations spaces, which does not truly reflect the complexity of real word problems. This paper presents *Continuous Belief Tree Search* (CBTS), a planner for continuous state, action and observations spaces using *Bayesian Optimisation* (BO) to dynamically sample promising actions while constructing a belief tree. Dynamic action sampling allows for richer action generation, and shows promising results in simulation.

1 Introduction

Planning with incomplete knowledge of the environment is a challenging task. Agents evolving in unknown environments often need to plan when few or no data is available, and deal with continuous real world scenarios. In many robotics tasks, robots start with very little knowledge, and the data they gather improves planning quality. This improvement motivates online planning methods, which recompute policies whenever the agent executes an action. Bayesian optimisation was proposed as an online planning method for informative path planning by [1]. While choosing the immediately most informative trajectory yields acceptable results, the myopic character of BO leads to suboptimal policies. The same monitoring problem was reformulated as a POMDP by [2] to benefit from the framework's look-ahead planning capabilities, and proved to perform better than its myopic equivalent. However, the problem of non-myopic planning for monitoring was only solved for environments with discrete actions. While discretising actions is perfectly sound in engineered scenarios, it often leads to suboptimal policies in real world problems.

This paper proposes a solution to solve POMDPs in fully continuous environments based on *Monte-Carlo Tree Search* (MCTS), an approximate tree-based method proposed by [3]. MCTS-based planners handle continuous states and observations, but are limited to discrete actions. We present CBTS, an extension of MCTS for planning on continuous action spaces. It relies on dynamic action sampling, thus generating more precise actions than traditional sampling techniques. The proposed method is applied to a space modelling problem in which a robot learns an objective function by gathering noisy measurements. The robot maintains a belief over the studied function using a Gaussian Process, and monitoring behaviour is achieved by defining a specific reward function. Experiments show planning with continuous actions results in higher accumulated rewards.

The remainder of the paper is organised as follows. Section 2 presents background on POMDP solvers. Section 3 gives theory on the POMDP framework. Our CBTS planner is presented in Section 4. Simulated experiments of a monitoring problem are shown in Section 5. The paper concludes with Section 6.

2 Related work

The planning problem in POMDPs has received much attention over the last decades. Classic offline planners compute a policy before experiments start [4, 5], generally relying on sampling and trading off optimality for speed. In most realistic scenarios, new information is gathered as robots evolve in their environment, which offline methods do not take advantage of.

Many successful online POMDP planners were developed in [6, 7, 8, 3, 9, 10], efficiently handling discrete and low-cardinality states, actions and observations. Some were extended to large and continuous state and observation spaces [7, 3, 9]. Most of these techniques approximate the search of the different spaces by sampling them to compute a more compact representation. Random sampling naturally leads to expectation estimates, which allows algorithms to estimate expected states and observations. However, finding the action yielding maximum reward is not an expectation problem and therefore cannot be tackled with the same sampling techniques. POMDPs with continuous actions were addressed by [11, 12, 13, 14]. While successful in their study cases, these methods rely on diverse unrealistic assumptions: finding the best policy of a predefined class, assuming beliefs to be Gaussian distributions, assuming to receive the most likely observations, or pre-computing a candidate policy offline. In this paper, we propose an approximate and online POMDP solver for continuous state, action and observation spaces, which does not restrict the type of belief nor makes strong assumptions on the nature of observations.

3 Background

Partially observable MDPs (POMDPs) is a well-defined framework for non-myopic decision making under uncertainty when the state is not directly observable. POMDPs are defined by the tuple $< S, A, T, R, \Omega, O, \gamma >$, where S is the space of states, A is the space of actions and Ω is the space of observations. At each step t, the agent arrives at state $s' \in S$, and receives a reward $r \in \mathbb{R}$ and an observation $o \in \Omega$ for taking an action $a \in A$ in its previous state s at step $t-1$. The transition dynamics distribution T satisfies the Markov property, expressing the probability of transitioning to state s' when executing action a in state s, $T(s, a, s') = p(s'|s, a)$. Rewards r are given by a reward function R which only depends on the current state and action, $r = R(s, a)$. The observation distribution O expresses the probability of observing o when executing action a in state s, $O(o, a, s) = p(o|a, s)$. $\gamma \in [0, 1]$ is a user-defined parameter used to discount long-term rewards when planning.

Given the partial observability nature of the problem, the agent does not have access to its true state. Instead, it relies on maintaining a belief $b(s)$ over the possible current states. POMDP planners compute policies $\pi : S \rightarrow A$ reflecting the action an agent should take when in a given state. Solving a POMDP is equivalent to finding the optimal policy π^*, maximising the expected infinite sum of future discounted rewards,

$$\pi^* = \arg\max_{\pi} E[\sum_{t=0}^{\infty} \gamma^t r_t^{\pi} | b_o] \tag{1}$$

where b_0 is the initial belief, and r_t^{π} is the reward for executing policy π at time t.

Diverse techniques were proposed to solve POMDPs [8]. The method presented in this paper is based on a stochastic tree search algorithm.

4 Continuous Belief Tree Search

Monte-Carlo Tree search is an any-time method used to partially and stochastically search trees. We refer readers not familiar MCTS to the work of [3]. MCTS was first used by [3] to plan in POMDPs, building a tree in which nodes are beliefs and branches are actions. The tree represents numerous sequences of simulated actions an agent can take at step t. Accumulated rewards and visit counts are kept on each branch. Finding the branch with the maximum accumulated reward is an approximation of Equation 1, where the infinite sum is replaced by a finite one.

MCTS is only defined to solve POMDPs with finite and discrete actions. We propose to generalise the method to infinite and continuous actions introducing *Continuous Belief Tree Search* (CBTS), consequently alleviating the need to discretise the action space prior to planning. CBTS redefines the

Algorithm 1 CBTS action selection algorithm

```
1:  function $v_l$ = ActionSelection($v$)
2:      if $length(D_v) < A_{max}$ then
3:          $a \leftarrow$ Generate action with Eq. 3 and $D_v$
4:          $r \leftarrow$ Simulate $a$ and get reward.
5:          Augment data $D_v$ with $(a, r)$, and update $b_v$ with $D_v$.
6:          return $NewNode(b(f), \mathbf{p}, r)$
7:      else
8:          return $BestChild(v)$
9:      end if
10: end function
```

discrete action selection method of MCTS, used to expand tree branches. The method proposed here relies on dynamically sampling the space of actions at the most promising locations with BO. An acquisition function is used to determine which actions are promising.

When an action a is simulated from node v of the belief tree τ and yields reward r, the resulting pair $\{a, r\}$ is used to learn a mapping from actions to rewards at a node level. Each node v stores data D_v of previous action-reward pairs, which is used for generating new actions from node v. The problem of choosing a new action a^* for simulation from node v is formulated as follows:

$$a^* = \arg\max_{\Theta} h(\Theta | D_v) \tag{2}$$

where Θ is an element of the continuous action space A and h is an acquisition function. Balancing exploitation of high-reward action and exploration of unknown areas of the action space is achieved by appropriately choosing h. The Upper-Confidence Bounds (UCB) function is generally used for such balance in the BO literature. Equation 2 then becomes

$$a^* = \arg\max_{\Theta} \mu(b_v(\Theta)) + \kappa\sigma(b_v(\Theta)) \tag{3}$$

μ and σ are the mean and variance operators respectively, κ is a parameter balancing exploration and exploitation, and b_v is a belief v maintains on its action-reward mapping using a Gaussian Process trained with D_v. Algorithm 1 provides pseudo-code for the action selection procedure of CBTS.

The more often node v is visited, the more accurate the action-rewards mapping of v gets. However, the problem of when to stop generating additional actions can be challenging. In practice, one can limit the maximum number of generated actions per node to a problem-specific fixed value A_{max}. Another approach relies on stopping whenever a convergence criterion is met. For example, such criterion can be $||\Theta_{i-1} - \Theta_i|| < \epsilon$, where Θ_{i-1} and Θ_i are the previously and newly generated actions respectively, and ϵ is a user-specified distance.

Each node's belief on the action-reward mapping is implemented with a Gaussian Process, of complexity $O(N^3)$ with the number of actions generated per node. N is in practice very small, therefore leading to negligible computation time compared to the state belief update, also of complexity $O(N^3)$ with the number of observations. In practice, CBTS displays similar running times to MCTS.

5 Application to monitoring

The POMDP-for-monitoring formulation presented in this section was first proposed by [2] to solve sequential Bayesian optimisation to carry out exploration guided by an acquisition function. Let us now describe the POMDP used.

The state $s = \{f, \mathbf{p}\}$ is defined by the function to model f and the fully-observable robot's pose $\mathbf{p}$. Continuous actions a are smooth trajectories $\mathcal{T}(\Theta, \mathbf{p})$ defined by a set of parameters Θ. The deterministic transition dynamics distribution $T(\{f, \mathbf{p}\}, \Theta, \{f', \mathbf{p'}\})$ models transitioning from $\mathbf{p}$ to $\mathbf{p'}$ with action Θ. Because transitions and f are independent, T can be rewritten as $T(\{f, \mathbf{p}\}, \Theta, \{f', \mathbf{p'}\}) = \delta(\mathcal{T}(\Theta, \mathbf{p})|_{u=1} - \mathbf{p'})$ which only depends on the endpoint of $\mathcal{T}(\Theta, \mathbf{p})$. The reward for executing action Θ in pose $\mathbf{p}$ is a sum over discrete trajectory locations:

$$R(\Theta, \mathbf{p}) = \sum_{x \in \mathcal{T}(\Theta, \mathbf{p})} UCB(b(x)) + cost(\Theta, \mathbf{p})\,, \tag{4}$$

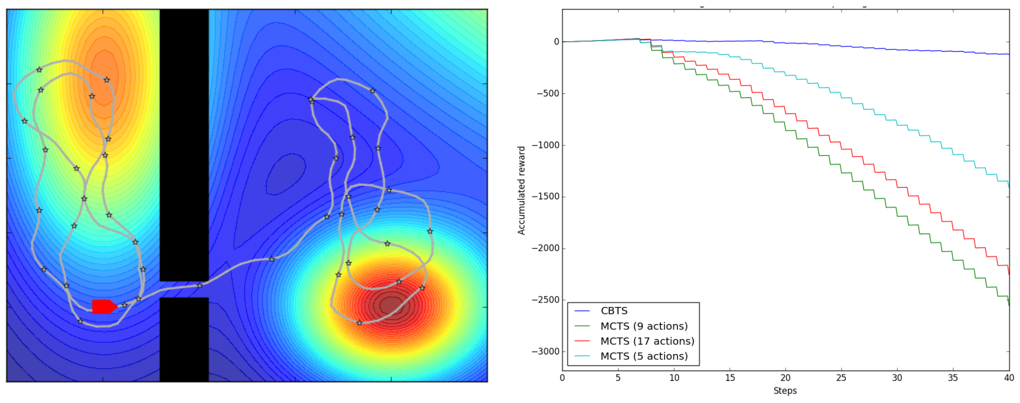

Figure 1: Space modelling domain with example of trajectory generated with CBTS (left). Black rectangles are obstacles, the red polygon represents the robot's starting pose, and background colours reflect monitored function values ranging from low (blue) to high (red). Accumulated rewards of space modelling experiment (right).

where the Upper Confidence Bounds function UCB is applied to the agent's belief, and $cost(\Theta, \mathbf{p})$ is the application specific cost of moving along $\mathcal{T}(\Theta, \mathbf{p})$. UCB is selected as a reward function for the exploration-exploitation behaviour it yields. Observations $o \in \mathbb{R}$ are noisy evaluations of f. Similarly to rewards, observations are computed on a set of locations along $\mathcal{T}(\Theta, \mathbf{p})$. The robot can simulate observations by generating noisy samples from its belief b over f, which is maintained using a Gaussian process.

5.1 Simulation results

We present experiments of a robot monitoring a fictive environmental variable by planning with CBTS. The robot starts with no data and progressively enriches its belief of the monitored function by gathering observations. The domain and fictive monitored function are shown in Figure 1, featuring two high-valued areas separated by a wall. The reward function is defined in Equation 4. Additionally, touching a wall yields a reward of -100. The robot is given full knowledge of where the obstacles are located, but only receives information about the monitored function from noisy observations. Exploration is therefore essential to receiving higher rewards in the long term.

Our experiments compare CBTS with an equivalent discrete action method (MCTS), both planning with a horizon of 3. CBTS generates a maximum of $A_{max} = 20$ actions per node. MCTS using 5, 9 and 17 actions are compared to CBTS in terms of accumulated rewards. All results are averaged over 40 experiments.

Figure 1 reports accumulated rewards. Overall, CBTS performs better than discrete actions MCTS, regardless of the number of actions used. The first 8 steps yield very similar rewards across all methods, because the robot explores the left part of the domain first. After step 8, the robot is drawn to explore the rest of the domain, and discrete actions are often not precise enough to navigate through the corridor. This experiment shows that using continuous actions planner CBTS enables robots to efficiently avoid obstacles and navigate more precisely than when planning with a discrete action planner. MCTS with 5 actions outperformed its equivalent with 9 and 17 actions because of the planner's inability to efficiently construct a belief tree with higher branching factors. Better performance could be achieved by exponentially increasing the number of MCTS iterations.

6 Conclusion

We proposed CBTS, a method using BO to solve continuous state-action-observation POMDPs. CBTS uses BO to dynamically sample continuous action spaces, thus generating more precise actions. Experiments on a monitoring problem show planning with continuous actions yields better accumulated rewards, while enabling robots to avoid obstacles more efficiently.

References

[1] R. Marchant and F. Ramos, "Bayesian optimisation for informative continuous path planning," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 6136–6143, IEEE, 2014.

[2] R. Marchant, F. Ramos, and S. Sanner, "Sequential bayesian optimisation for spatial-temporal monitoring.," in *UAI* (N. L. Zhang and J. Tian, eds.), pp. 553–562, AUAI Press, 2014.

[3] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," in *Advances in neural information processing systems*, pp. 2164–2172, 2010.

[4] R. D. Smallwood and E. J. Sondik, "The optimal control of partially observable markov processes over a finite horizon," *Operations Research*, vol. 21, no. 5, pp. 1071–1088, 1973.

[5] J. Pineau, G. Gordon, S. Thrun, *et al.*, "Point-based value iteration: An anytime algorithm for pomdps," in *IJCAI*, vol. 3, 2003.

[6] R. He, E. Brunskill, and N. Roy, "Puma: Planning under uncertainty with macro-actions.," in *AAAI*, 2010.

[7] H. Kurniawati and V. Yadav, "An online pomdp solver for uncertainty planning in dynamic environment," in *Robotics Research*, pp. 611–629, Springer, 2016.

[8] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa, "Online planning algorithms for pomdps," *Journal of Artificial Intelligence Research*, pp. 663–704, 2008.

[9] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "Despot: Online pomdp planning with regularization," in *Advances in neural information processing systems*, 2013.

[10] H. Kurniawati, D. Hsu, and W. S. Lee, "Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces.," in *Robotics: Science and Systems*, Zurich, Switzerland, 2008.

[11] A. Y. Ng and M. Jordan, "Pegasus: A policy search method for large mdps and pomdps," in *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, pp. 406–415, Morgan Kaufmann Publishers Inc., 2000.

[12] J. Van Den Berg, S. Patil, and R. Alterovitz, "Efficient approximate value iteration for continuous gaussian pomdps.," in *AAAI*, 2012.

[13] R. Platt Jr, R. Tedrake, L. Kaelbling, and T. Lozano-Perez, "Belief space planning assuming maximum likelihood observations," 2010.

[14] K. M. Seiler, H. Kurniawati, and S. P. Singh, "An online and approximate solver for pomdps with continuous action space," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.