POMDP Library Optimizing Over Exploration and Exploitation in Robotic Localization, Mapping, and Planning

Joyce Anderson Annan, Akram Alghanmi, Marius Silaghi

Florida Institute of Technology

ABSTRACT

Localization, mapping, and planning are critical in autonomous robots operating in uncertain environments and in continuous and discrete domains. High-quality probabilistic models for a complex robot depend heavily on details from its environment, involving multiple parameters. However, there is a lack of accurate probabilistic models for existing robots that can handle reasonably the challenges posed by real applications. For most robots, actions are highly non-deterministic. Furthermore, there is a lack of general software packages applicable to new scenarios. Specifically, we propose a POMDP library for optimal planning and localization given new available models, and dedicated to optimize over exploration and exploitation tradeoffs.

Keywords: Exploration, Exploitation, Localization, Mapping, Planning, Uncertainty, POMDP

1. Introduction

A humanoid robot such as NAO has made significant advances in various tasks such as emotional support, sensing, and manipulation. A utility-driven robot typically operates in a structured environment while performing clearly defined tasks to increase the expected utility, which is influenced by the likelihood of obtaining various possible rewards for reaching goal states. Well known successes are with tasks that are often repetitive and are carried out by a series of simple, predefined subroutines.

However, unexpected problems can arise when performing more complex tasks while making real-time decisions in unknown environments with imperfect observations. As a result, a NAO robot that encounters uncertainty about how the world works and its current status must engage in optimal exploration to resolve the world's unknown by choosing to discover high-reward strategies that require policies conducing to a temporally extended sequence of complex behaviours that, individually, may not be rewarding. Alternatively, consider whether to try to exploit known behaviours in order to achieve higher rewards.

Despite its limited knowledge of its surroundings, the NAO robot is expected by users to perform reliably due to its detailed humanoid aspect. Its actions, however, are not entirely dependable and accurate. The NAO robot occasionally lingers in its neutral state or advances beyond its predicted location when it aims to move. Additionally, when it attempts to turn, it occasionally misses the action. It has the same problems with its sensors. Sometimes its vision generates more false identifications; at times it generates inferences from observations that heavily contradict the current state estimate [1]. How do we principally model exploration and exploitation when robot localization, motion planning and mapping are prone to error? We address this issue with a new software library that helps users to easier model worlds in principled formal ways.

In the following we present related work, background, and describe a proposed architecture.

2. Background

Partially Observable Markov Decision Process (POMDPs) is a problem formulation that describes a sequential decisions process in an uncertain environments. This model extends a well-studied Markov Decision Process (MDPs) framework to real-world situations in which an agent cannot accurately identify its inherent environment or determine where it will end after a sequence of actions due to its partial observability states, and nondeterministic actions. As a result, the true state is unknown, and reasoning can only use a belief about the true state using observation.

A POMDP is expressed as a set of tuple $\langle S, A, \Omega, T, O, R, \lambda \rangle$ that consists of a set of states (S), actions (a), observations Ω, transitions (T), observation function (O), rewards (R), and discount factor (γ) [2].

The tuple $\langle s, a, t, r, \gamma \rangle$ describes the Markov Decision Process whereas the Observation space (Ω) and the Observation function (O) are the only differences brought by POMDPs compared to the MDP. The Observation space (Ω) is a finite set of observations an agent can experience in its world and an Observation function (O) provides a probability distribution for each observation in each state. For every transition, an agent is expected to receive a reward R(s,a,s$'$) based on an action a taken from a given state s to reach a possible state s'. The reward might be positive or negative, but it commonly is assumed to be within a certain range Rmax (0,1). POMDP approaches assume an initial belief b while the initial state is unknown because of the probability distribution over the state. When an action is performed by the agent and an observation is received, the belief is updated.

POMDPs are particularly adept at dealing with partially observable, stochastic environments and can provide wise policies in the face of unexpected evidence. An exact solution to a POMDP yields the optimal action for each possible belief over the world states. The optimal action maximizes the expected reward (or minimizes the cost) of the agent over a possibly infinite horizon. The mapping of optimal actions to beliefs is known as the optimal policy of the agent for interacting with its environment.

3. Related Work

There are just a few frameworks that can solve decision-making problems in partially observable environments, particularly when it comes to optimizing exploration and exploitation tradeoffs. However, in stochastic, partially observable contexts, these frameworks only yield nearly optimal results and do not deliver satisfactory or accurate solutions.

Approximate POMDP Planning (APPL) Toolkit implements SARSOP, DESPOT and MCVI algorithms for solving discrete, online and continuous POMDPs respectively and output a policy that approximates the optimal policy for a given problem [3]. The ZMDP software solves, benchmarks and evaluates several heuristic search algorithms to output policy for a search strategy and problem [4]. SolvePOMDP solves POMDPs optimally by the use of incremental prunning while executing randomized point base value iteration to provide both exact and approximate methods [5]. The R pomdp package provides an interface for various exact and approximate solution algorithms [6]. pyPOMDP a (PO)MDP toolbox (simulator, solver, learner, file reader) for Python [7]. Finite-state Controllers using branch-and-bound an exact POMDP solver for policies of a bounded size [8]. AI-Toolbox [9] also provides a set of known algorithms.

POMDPs.jl is a Julia and Python interface for defining and solving MDPs and POMDPs with a variety of solvers aimed at simplifying problem writing by offering a set of basic functions. It provides expressiveness by allowing users to prototype new code and remove the problem specification. The extensible nature of the interface also allows new algorithms to be implemented easily by allowing algorithm writers to access any component of the model with simple function calls [10].

Although POMDPs.jl and other mentioned solvers provides simplicity, expressiveness, extensibility to all POMDPs problem, however the existing packages require the user to manually design the exploitation/exploration problem and do not give more support through any extension of pomdp.

Exploration and exploitation are often addressed utilizing particular algorithms that include an exploring strategy and an updating strategy. The update approach is primarily determined by the knowledge representation used, ranging from a simple weighted mean for punctual estimations [11] to Bayesian updates [12] for probability distributions. Among the various exploration strategies are e-greedy [13], optimistic initialisation [14], higher confidence bound [15], Thompson sampling [16] and reinforcement learning [17]. Nonetheless, despite the fact these strategies provide a typical environment in which the exploration-exploitation tradeoff occurs, it should be noted that there is no accurate algorithm that solves this tradeoff optimally.

4. Proposed Architecture

The goal of this project is to provide a general software package for optimal planning and localization that can be applied to new scenarios and is dedicated to support choices in the tradeoff between exploration and exploitation. We introduce an automated modeling capability for exploration and exploitation tradeoffs, approach that can use "greedy in the limit of infinite exploration" to prevent a limited probability of missing an optimal action an unbounded number of times. We provide support by extending existing pomdps models, as well to optimize computational efficiency.

Furthermore, help users model unknown rewards as known values. Assuming that the probability distributions are uniform in unknown spaces, we can reduce the problem to the automatic modeling of a pomdp via our library.

As an example, take a robot in a labyrinth problem in consideration. It starts from a point/state where he notices two unfamiliar doors on the right and left sides of a room and is given the task of leaving the room. So as he walks around the room, he realizes he needs to exit through one of the doors, but he has no idea which door leads to the exit. As a result, the probability density function for getting out will be distributed evenly across the areas it sees. After that, he will go on an exploration mission while mapping and exploiting. The probability distribution can exploit heuristics based on the observation that ramification in the section is a key point with a higher likelihood of taking a map at points previously seen.

Thus, in order to transform the problem into an exploration and exploration pomdp, the user only needs to specify the *s,a,r* and based on this specify another parameter. Then we convert it to a pomdp automatically.

5. State of The Work

We have designed the high level requirements of the library and are designing the user interface; the current research focus being on model conversion.

6. Conclusions

Despite availability of powerful and efficient models and algorithms, robot and problem modelling remains challenging, as it requires deep familiarity of the user with advanced concepts. This is a problem we plan to address through the development of new models, patterns, and model conversions supported in appropriate software packages. We have introduced our research goals, the design principles and the current state of the work.

References

[1] B. Coltin and M. Veloso, "Multi-observation sensor resetting localization with ambiguous landmarks," *Autonomous robots*, vol. 35, no. 2, pp. 221–237, 2013.

[2] S. Russell and P. Norvig, "Artificial intelligence: A modern approach. third edit," *Upper Saddle River, New Jersey*, vol. 7458, 2010.

[3] 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*, vol. 2008, Citeseer, 2008.

[4] T. Smith, "Zmdp software for pomdp and mdp planning," *GitHub. https://github. com/trey0/zmdp, Son erisim tarihi*, vol. 1, 2007.

[5] E. Walraven and M. Spaan, "Accelerated vector pruning for optimal pomdp solvers," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[6] "Infrastructure for partially observable markov decision processes (pomdp) [r package pomdp version 1.0.1]," Mar 2022.

[7] K. Zheng and S. Tellex, "pomdp_py: A framework to build and solve pomdp problems," *arXiv preprint arXiv:2004.10099*, 2020.

[8] P. Poupart and C. Boutilier, "Bounded finite state controllers," *Advances in neural information processing systems*, vol. 16, 2003.

[9] E. Bargiacchi, D. M. Roijers, and A. Nowé, "Ai-toolbox: A c++ library for reinforcement learning and planning (with python bindings).," *J. Mach. Learn. Res.*, vol. 21, pp. 102–1, 2020.

[10] M. Egorov, Z. N. Sunberg, E. Balaban, T. A. Wheeler, J. K. Gupta, and M. J. Kochenderfer, "Pomdps. jl: A framework for sequential decision making under uncertainty," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 831–835, 2017.

[11] O. A. M. Rangel, *Active Localization for Bearing-Only Sensor Using Stochastic Control Techniques*. PhD thesis, Centro de Investigación en Matemáticas, 2012.

[12] S. Ross, B. Chaib-draa, and J. Pineau, "Bayesian reinforcement learning in continuous pomdps with application to robot navigation," in *2008 IEEE International Conference on Robotics and Automation*, pp. 2845–2851, IEEE, 2008.

[13] H. Y. Lee, H. Kamaya, and K. Abe, "Labeling q-learning for non-markovian environments," in *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 99CH37028)*, vol. 5, pp. 487–491, IEEE, 1999.

[14] D. Szer, F. Charpillet, and S. Zilberstein, "Maa*: A heuristic search algorithm for solving decentralized pomdps," *arXiv preprint arXiv:1207.1359*, 2012.

[15] Y. Xiong, N. Chen, X. Gao, and X. Zhou, "Sublinear regret for learning pomdps," *arXiv preprint arXiv:2107.03635*, 2021.

[16] D. J. Russo, B. Van Roy, A. Kazerouni, I. Osband, Z. Wen, *et al.*, "A tutorial on thompson sampling," *Foundations and Trends® in Machine Learning*, vol. 11, no. 1, pp. 1–96, 2018.

[17] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.