Imitation Learning in POMDPs with Contacts

Hai Nguyen, Xinchao Song, Christopher Amato, Robert Platt

***Abstract*— We are interested in manipulation tasks that embrace contacts as a necessary means to perform information gathering. We formulate the problem as a partially observable decision process (POMDP) and solve it using imitation learning. We assume access to a simulator that provides all information needed for learning the tasks. Leveraging this privileged information, we train a POMDP expert that solves the task while performing informative actions using contacts. We then train an agent that acts on partial information by cloning this expert's behavior. We test our method on a novel robotics domain and set up an experiment with a real robot.**

I. Introduction

While humans use contacts as important sensory feedback, modern robots often avoid contacts due to potential structural damages. However, in many real-life manipulation tasks such as searching for items in a pocket or a drawer, contacts are ideal because other sensory modalities such as vision might be unavailable or insufficient.

To solve these tasks, we assume that during training we have a physics simulator that can provide access to all information needed for the tasks. We want to leverage this access *during training* to train an intelligent agent that makes optimal decisions in the real world in which such full state observability is often infeasible.

To exploit true states in simulations during training, previous approaches use Asymmetric Actor Critic [1] [2] in which the actor (agent) acts on observations (images) while its policy is criticized by a critic having access to true states. Imitation learning is also used in [3] [4] to behavior-clone an expert that has access to true states. These papers, however, assume a one-to-one mapping between each true state and each observation, which often does not hold in partially observable decision processes (POMDPs). By contrast, we instead train a POMDP expert and then train an agent to imitate this expert. Empirical results in a robotics domain show that this way, the trained agent can nearly reach the expert's performance reliably, while other baselines have lower and inconsistent performances.

II. Background

A POMDP is defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \Omega, \mathcal{T}, \mathcal{O}, \mathcal{R} \rangle$ where $\mathcal{S}, \mathcal{A}, \Omega$ are the state, action, and the observation spaces; $\mathcal{T}(s, a, s') = p(s'|s, a)$ is the state transition function; $\mathcal{O}(s', a, o) = p(o|s', a)$ is the observation model; and $\mathcal{R}(s, a)$ is the reward function. In POMDPs, the agent only observes $o \in \Omega$ not the true state $s \in \mathcal{S}$ at any timestep. Therefore, it has to rely on the history of actions and observations $\{o, a\}$ to make decisions optimally. However, the history would grow exponentially over time. Alternatively, the agent keeps track of a belief state b, which is defined as a probability distribution over $\mathcal{S}$. Optimal policies are those that maximize the expected discounted future reward given an initial belief. It is shown that the belief state is a sufficient statistic of history and one can construct optimal policies from the belief.

Khoury College of Computer Sciences, Northeastern University, Boston, MA 02115. Corresponding author: Hai Nguyen, email: nguyen.hai1@husky.neu.edu

III. Approach

A. Experts

First, we train a POMDP expert in the belief space. With the assumption of known transition and observation functions during training, we can update belief states using the following equation [5].

$$b'(s') = \frac{\mathcal{O}(s', a, o) \sum_{s \in \mathcal{S}} \mathcal{T}(s, a, s') b(s)}{p(o|a, b)} \tag{1}$$

A belief update is performed after the agent takes an action a, enters a next state s', and observes a new observation o. Using Eq. 1, we learn an expert, which is trained on the belief space using the actor-critic method [6]. The expert consists of two parts: the actor selects actions given the current belief $\pi(a|b_t)$ and the critic estimates $V^{\pi}(b_t)$, which is used to update the policy in a policy gradient fashion.

To compare with learning from this POMDP expert, we also train an MDP expert, which is an actor-critic agent that has access to the full state. This expert knows all information needed, therefore, it does not have to explore. For tasks that require information gathering actions, we show later that imitating such an expert, the agent will end up with a sub-optimal policy since it receives no guidance on how to explore efficiently.

B. Imitation Learning

We use DAgger [7] to learn from the above experts. The idea is to imitate experts to choose actions over the distribution of visited states. The first phase is collecting data when the agent interacts with the environment for a number of episodes (roll-outs). Next, the agent asks an expert for a correct action in each encountered state. Visited states and corresponding expert actions form a supervised set data to train the agent's policy for several epochs. After that, the updated policy is used to collect data and the whole process repeats.

In our POMDP setting, where true states are partially unknown to the agent, it has to act on n-timestep histories $\{o_i, a_i\}_{i=t-n}^{t-1}$. Also, different inputs are used to query different experts for actions: the current belief state b_t for the POMDP expert and the true state s_t for the MDP expert.

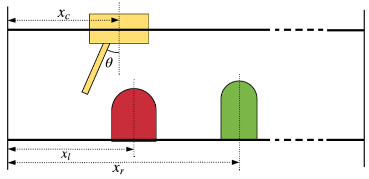


Fig. 1: The Finger domain with state $s = (\theta, x_c, x_l, x_r)$, observation $o = (\theta, x_c)$. The agent controls the moving direction of the cart (left/right) and the stiffness (0/1) of the finger. The agent has to push the right bump (green) to the right without moving the left bump (red).

IV. Experiments and Results

A. The Finger Domain

We designed the Finger domain which consists of a single finger on a movable cart (yellow), the left bump (red), and the right bump (green) (Fig. 1). A state s includes the finger's angle θ, the positions of the cart x_c, the left bump x_l, and the right bump x_r with $x_c, x_l, x_r \in \{0, \ldots, 40\}$ and $\theta \in \{-1 \text{ (pointing left)}, 0 \text{ (pointing down)}, 1 \text{ (pointing right)}\}$. At any given time, s is partially observable to the agent. It however can fully observe $o = (\theta, x_c)$. We also have a real robot setup for this domain (Fig. 3).

The agent controls the moving direction of the cart (left/right) and the stiffness (0/1) of the finger's joint. At $t = 0$, the cart and the bumps are randomly positioned such that $x_c \in \{0, \ldots, 40\}$, $x_l, x_r \in \{2, \ldots, 38\}$, $2 \leq x_r - x_l \leq 20$, and $\theta = 0$. When the cart moves, the finger can make a contact with a bump. In that case, if the finger is compliant, the cart can pass the bump without it. By contrast, when the finger is stiff, the cart can cause a displacement to the collided bump. The collision can also cause a change to θ.

The agent's goal is to move the right bump (green) to the right without moving the left one (red). It is rewarded 1 if it succeeds and 0 otherwise. The episode is terminated whenever either bump is moved. To perform the task optimally, the agent has to use contacts to "feel" and localize the right bump to be able to push it.

B. Experts

- POMDP expert: We train an asynchronous advantage actor critic (A3C) [8] agent in the belief space over unknown bumps' positions (x_l, x_r) and known observations $o = (\theta, x_c)$.
- MDP expert: We train an A3C agent on full state $s = (\theta, x_c, x_l, x_r)$.

After being trained, these two experts can solve the task nearly 100% of the time.

C. Baselines

We compared the performance with 2 baselines:

- Deep Recurrent Q-Network agent [9]: It is not easy to use true states in this case so we let the agent use $\{o_i, a_i\}_{i=t-n}^{t-1}$ to make decisions.
- Asym A2C agents [1]: The actor takes in $\{o_i, a_i\}_{i=t-n}^{t-1}$ histories, while the critic can use true states or beliefs.

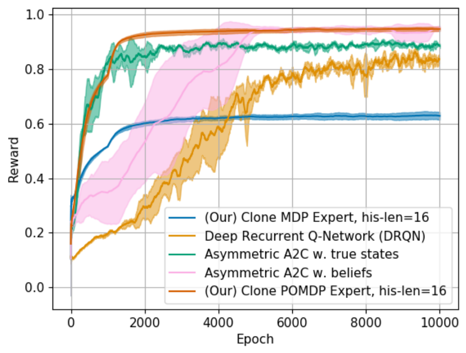


Fig. 2: The learning curve of our agent with 16 timestep histories as input compared with baselines.

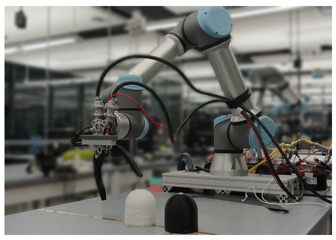


Fig. 3: The Finger domain set-up with a UR5e arm.

D. Results

We parameterize all agents as LSTM-based neural networks. Our trained DAgger agent, when cloning the POMDP expert, can reach about 95% of the expert's performance (Fig. 2). Asym A2C agent uses beliefs have comparable but more varied performance and it is more sample-inefficient. Asym A2C agent uses true states have slightly worse performance possibly because it cannot capture and exploit the past as good as beliefs do. DRQN, that relies only on observation-action histories perform worse, can reach 82% of the maximum performance. The worst performance belongs to our DAgger agent when imitating the MDP expert. The agent does not learn how to explore efficiently and the learning plateaus at sub-optimal performance after a while.

V. Conclusion and Future Work

In this paper, we use the access to full states during training and imitation learning to train an intelligent agent that uses contacts to explore to solve a manipulation task. Preliminary results have shown the potential of our approach. Our agent yields higher and more consistent learning performance than that of existing methods. Moreover, our approach can easily extended to other POMDP domains. For future work, we want to explore different types of imitation learning methods. Specifically, we want to experiment with learning from expert demonstrations [10] or imperfect ones [11]. It is also interesting to work with more complex domains where we have to learn the dynamics of the environment instead of knowing them beforehand.

References

[1] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," *arXiv preprint arXiv:1710.06542*, 2017.

[2] M. Wilson and T. Hermans, "Learning to manipulate object collections using grounded state representations," *arXiv preprint arXiv:1909.07876*, 2019.

[3] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," *arXiv preprint arXiv:1912.12294*, 2019.

[4] X. Guo, S. Singh, H. Lee, R. L. Lewis, and X. Wang, "Deep learning for real-time atari game play using offline monte-carlo tree search planning," in *Advances in neural information processing systems*, 2014, pp. 3338–3346.

[5] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[6] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[7] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 627–635.

[8] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

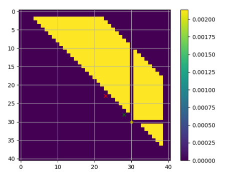
[9] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *2015 AAAI Fall Symposium Series*, 2015.

[10] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband *et al.*, "Deep q-learning from demonstrations," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

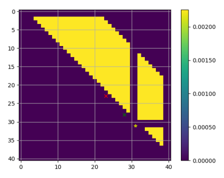
[11] Y. Gao, H. Xu, J. Lin, F. Yu, S. Levine, and T. Darrell, "Reinforcement learning from imperfect demonstrations," *arXiv preprint arXiv:1802.05313*, 2018.

Appendix

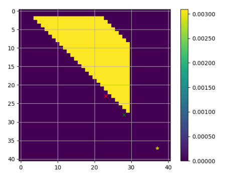
The change of the belief space (over the bumps' position) in a trajectory, which is generated by the POMDP expert, is visualized in Fig. 4.



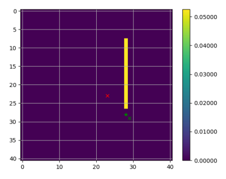
(a) T=0: The cart with a compliant finger (⋆) starts on the right side of the right bump (X). No bump is at this location therefore cells that are at the same horizontal and vertical level are dark.



(b) T=1: The agent goes right diagonally with a compliant finger (⋆) to explore. No bump is detected, belief updates filter out impossible cells.



(c) T=10: The agent finishes exploring the right side without making contact with any bump. It reaches the right limit and has to head back to find the bumps.



(d) T=29: Many cells are dark due to a bump contact. The collided bump is the right one because no bump was found on the right side. The agent then pushes this bump.

Fig. 4: The belief space in a trajectory generated by a POMDP expert. It is represented by a 41×41 grid (X - the right bump's position, Y - the left bump's position). The color of each cell represents the value of the belief $b(x_l, x_r)$. Many cells are dark ($b = 0$) initially because $x_l, x_r \in \{2, \ldots, 38\}$ and $2 \leq x_r - x_l \leq 20$. Two bumps (left bump - X, right bump - X) and the cart (⋆: with a stiff finger or ⋆: with a compliant finger) move on the diagonal of the square.