Simultaneous Learning of Moving and Active Perceptual Policies for Autonomous Robot

Extended Abstract

Wataru Hatanaka
RICOH COMPANY,LTD
Kanagawa, Japan
wataru.hatanaka@jp.ricoh.com

Fumihiro Sasaki
RICOH COMPANY,LTD
Kanagawa, Japan
fumihiro.fs.sasaki@jp.ricoh.com

Ryota Yamashina
RICOH COMPANY,LTD
Kanagawa, Japan
ryohta.yamashina@jp.ricoh.com

Atsuo Kawaguchi
RICOH COMPANY,LTD
Kanagawa, Japan
atsuo.kawaguchi@jp.ricoh.com

ABSTRACT

Humans can move their bodies and eyes actively to perceive the state of the environment they are surrounded by. Autonomous robots are needed to learn this ability, so called active perception, to behave as humans do. In this paper, we propose a reinforcement learning algorithm to make the robots have the perceptual ability. In our algorithm, we simultaneously train two agents which control the robot and its sensor on the robot to achieve a task. We conducted experiments on navigation tasks in a 3D environment where useful information for the task achievement is partially occluded. The experimental results show that our algorithm can obtain better perceptual behavior and achieve higher success rates than conventional reinforcement learning algorithms.

KEYWORDS

Active Perception; Reinforcement Learning; Autonomous Robot

ACM Reference Format:
Wataru Hatanaka, Fumihiro Sasaki, Ryota Yamashina, and Atsuo Kawaguchi. 2021. Simultaneous Learning of Moving and Active Perceptual Policies for Autonomous Robot: Extended Abstract. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 3 pages.

1 INTRODUCTION

While autonomous robots are designed to perceive the state of the environment via various sensors, their decision making is often hindered due to sensor occlusions. Active perception is used to enable autonomous robots to actively perceive the necessary information for decision making in such a situation [1, 7]. There are some deep reinforcement learning (RL) studies that are related to active perception, however, their applicability is limited just in scanning or mapping [4, 5].

In this paper, we simultaneously train two agents which make different roles, one is a *moving agent* to behave for achieving a given task while moving the robot, the other is a *perceptual agent* to actively explore the state of the environment for better task achievement. For learning the perceptual policy, we introduce a *meta-evaluation* process which measures how much the perceptual policy improves the behavior of the moving policy. We conducted experiments on navigation tasks in a 3D simulation environment with some obstacles, and the results show that our algorithm can obtain better perceptual behavior and achieve a task with a higher success rate than conventional RL algorithms.

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

2 PROBLEM DESCRIPTION

Let a^m denote the moving agent that controls the robot's movement and a^c denote the perceptual agent that decides where to face the sensor, which is assumed to be a camera mounted on the robot in this paper. We propose a new markov decision process (MDP) $\mathcal{M} = \langle \mathcal{S}, \mathcal{U}^m, \mathcal{U}^c, \mathcal{P}^m, \mathcal{P}^c, r, p_1, \gamma \rangle$ in our problem setup, where $\mathcal{S}$ is a set of states, $\mathcal{U}^m$ and $\mathcal{U}^c$ are sets of moving and perceptual agent's actions. $\mathcal{P}^m : \mathcal{S} \times \mathcal{U}^m \times \mathcal{S} \rightarrow [0, 1]$ and $\mathcal{P}^c : \mathcal{S} \times \mathcal{U}^c \times \mathcal{S} \rightarrow [0, 1]$ are the state transition functions, $r : \mathcal{S} \times \mathcal{U}^m \rightarrow \mathbb{R}$ is a reward function, $p_1 : \mathcal{S} \rightarrow [0, 1]$ is a distribution over initial states, and $\gamma \in (0, 1)$ is a discount factor, respectively. We define a stochastic policy π as the probability distribution over actions conditioned on the current state; $\pi : \mathcal{S} \times \mathcal{U} \rightarrow [0, 1]$. For a given π, the state value is defined as $V^\pi(s) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, u_t)\right]$, where $s_0 = s$ and $u_0 \sim \pi(\cdot|s_0)$. We assume a realistic setting where the robot's movements affect which direction the camera faces whereas the camera's movements do not affect the robot's movements. The state transition in this setting is described as follows:

1. Given a state $s \in \mathcal{S}$, the moving agent a^m takes an action $u^m \in \mathcal{U}^m$ according to the moving policy $\pi^m : \mathcal{S} \times \mathcal{U}^m \rightarrow [0, 1]$.
2. The perceptual agent a^c receives a state $\bar{s} \in \mathcal{S}$ induced by the state transition function $\mathcal{P}^m(\cdot|s, u^m)$, and takes an action $u^c \in \mathcal{U}^c$ given the state $\bar{s}$ according to the perceptual policy $\pi^c : \mathcal{S} \times \mathcal{U}^c \rightarrow [0, 1]$.
3. The moving agent a^m receives the next state s' induced by the other state transition function $\mathcal{P}^c(\cdot|\bar{s}, u^c)$.

Based on the transition, the joint distribution $p^{m,c}$ of a state trajectory $\mathbf{s}_\tau = \{s_1, \bar{s}_1, \cdots, \bar{s}_{T-1}, s_T\}$ induced by rollouts following π^m,

π^c, $\mathcal{P}^m$ and $\mathcal{P}^c$ is thus written as follows:

$$\begin{aligned} p^{m,c}(\mathbf{s}_\tau) &= p_1(s_1)\prod_{t=1}^{T}\mathcal{P}^c(s_{t+1}|\overline{s_t},u_t^c)\pi^c(u_t^c|\overline{s_t}) \\ &\qquad \mathcal{P}^m(\overline{s_t}|s_t,u_t^m)\pi^m(u_t^m|s_t) \\ &= p_1(s_1)\prod_{t=1}^{T}\tilde{\mathcal{P}}(s_{t+1}|s_t,u_t^m)\pi^m(u_t^m|s_t), \end{aligned} \tag{1}$$

where $\tilde{\mathcal{P}}(s_{t+1}|s_t,u_t^m) = \mathcal{P}^c(s_{t+1}|\overline{s_t},u_t^c)\pi^c(u_t^c|\overline{s_t})\mathcal{P}^m(\overline{s_t}|s_t,u_t^m)$ is a state transition function that involves π^c.

3 OPTIMIZATION OF POLICIES

In our setting, we prepare two same environments : $\mathcal{E}_1$ and $\mathcal{E}_2$. In the environment $\mathcal{E}_1$, the moving policy π_θ^m runs with a perceptual policy $\pi_{\phi_1}^c$ while the value function approximator V_{w_1} estimates the expected cumulative rewards the moving agent a^m earns. In the environment $\mathcal{E}_2$, another perceptual policy $\pi_{\phi_2}^c$ and the value function approximator V_{w_2} make the same role of $\pi_{\phi_1}^c$ and V_{w_1}, respectively. The policies π_θ^m, $\pi_{\phi_1}^c$ and $\pi_{\phi_2}^c$ are parameterized by θ, ϕ_1 and ϕ_2, the value function approximators V_{w_1} and V_{w_2} are parameterized by w_1 and w_2, respectively. We define a *meta-evaluator* $\mathbb{V}$ which measures how much perceptual policy $\pi_{\phi_2}^c$ is better suited for the moving agent a^m than $\pi_{\phi_1}^c$ by calculating the difference of the average values $\overline{V}_{w_2}$ and $\overline{V}_{w_1}$ given the trajectories in each environment. The average value $\overline{V}_{w_e}$, which is calculated by the value function approximator V_{w_e} updated in the environment $\mathcal{E}_e$, summarizes how much cumulative rewards the moving policy π_θ^m can obtain with a perceptual policy $\pi_{\phi_e}^c$ in the environment $\mathcal{E}_e$ for $\forall e \in \{1,2\}$. The meta-evaluator $\mathbb{V}$ can be calculated as follows:

$$\mathbb{V}(\tau_k) = \overline{V}_{w_2}(\tau_k) - \frac{1}{K}\sum_{\tau_{k'}\in D_1}\overline{V}_{w_1}(\tau_{k'}), \tag{2}$$

where $\tau_k = \{s_t, u_t^m, r_t, \bar{s}_t, u_t^c\}_{t=1}^T$ is a trajectory for $\forall k \in \{1,2,\cdots,K\}$ and K is the number of trajectories. $D_1 = \{\tau_k\}_{k=1}^K$ denotes a set of trajectories obtained by rollouts of the policies in the environment $\mathcal{E}_1$. We update $\pi_{\phi_2}^c$ using the framework of REINFORCE [8] by replacing the cumulative rewards with the meta-evaluator $\mathbb{V}$. To update the $\pi_{\phi_1}^c$ to approximate what $\pi_{\phi_2}^c$ is used to be, we employ the *soft update* rule as follows:

$$\phi_1 = \alpha\phi_2 + (1-\alpha)\phi_1 \tag{3}$$

where α controls rate of the updates and we set $\alpha = 0.05$. After the soft update, we employ the general actor-critic algorithm to update both the moving policy π_θ^m and the value function approximator V_{w_1} by using D_1 is sampled again by a rollout of π_θ^m with $\pi_{\phi_1}^c$. Since the perceptual policy $\pi_{\phi_1}^c$ is expected to be better suited for the moving policy π_θ^m than the one before update in (3), the moving policy π_θ^m is updated so that π_θ^m gets further more rewards with the better perceptual policy $\pi_{\phi_1}^c$. The process of updating policies is completed by copying w_1 to w_2 for the next update. In the training process, we empirically found that the ϵ-greedy strategy is beneficial to learn better perceptual policies. We sample the action u_t^{c2} of the perceptual agent in the environment $\mathcal{E}_2$ as follows:

$$u_t^{c2} = \begin{cases} \arg\max \pi_{\phi_2}^c(\bar{s}_t) & \text{with probability } 1-\epsilon \\ \text{a random action} & \text{with probability } \epsilon \end{cases}.$$

After training, we employ $\pi_{\phi_1}^c$ as the actual perceptual policy with which the moving policy π_θ^m performs at the test time.

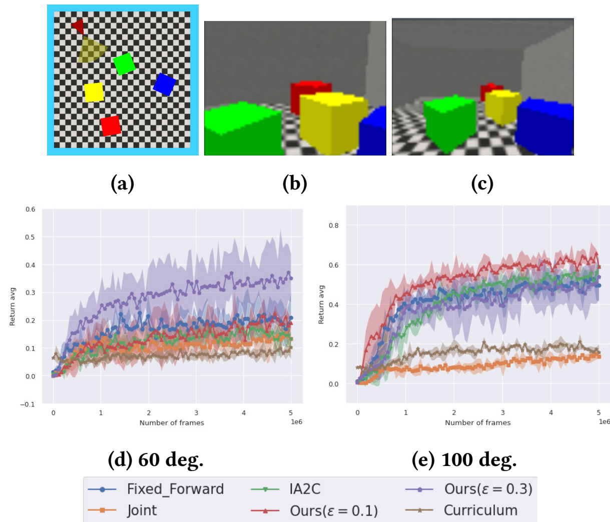

Figure 1: (a) is bird's-eye view of the map, (b) and (c) are examples of observations of the agent with the field of view 60 and 100 degrees, (d) and (e) are average returns in each setting.

4 EXPERIMENTS AND RESULTS

We build the environment by using the Mini-world package[3]. It has a single room where multiple boxes (green, yellow, blue) are placed between the agent and the goal (red box) as obstacles. We perform navigation tasks and evaluate our algorithm ($\epsilon = 0.1, 0.3$) against the following four baselines: **Fixed_Forward** (only the moving policy is trained with the camera is fixed in the forward direction), **Joint** (a single agent with the joint action space $\mathcal{U}^m \times \mathcal{U}^c$), **IA2C** ($a^m$ as well as a^c are trained separately by using the cumulative reward), **Curriculum** (inspired by [2], after the **Joint** agent is pre-trained in the environment without any obstacles, its parameters are used as initial values for training in the environment with the obstacles). The moving agent a^m for all methods are trained by advantage actor-critic (A2C) [6].

Figure 1 shows the experimental results with two horizontal fields of view 60 (Figure 1(d)) and 100 (Figure 1(e)) degrees during training. In the setting with 100 degrees, some baselines obtain large returns. Even in this setting, our algorithm with $\epsilon = 0.1$ yields the best result. On the other hand, although all methods deteriorate the average returns in the setting with 60 degrees, our algorithm with $\epsilon = 0.3$ achieves the highest success rate. These results show that our algorithm makes autonomous robots learn perceptual behaviors for better task achievement, and are consistent with our intuition – the less information we have, the more we are going to look at our surroundings.

REFERENCES

[1] Ruzena Bajcsy, Yiannis Aloimonos, and John K Tsotsos. 2018. Revisiting active perception. *Autonomous Robots* 42, 2 (2018), 177–196.

[2] Ricson Cheng, Arpit Agarwal, and Katerina Fragkiadaki. 2018. Reinforcement Learning of Active Vision for Manipulating Objects under Occlusions. In *Conference on Robot Learning*. 422–431.

[3] Maxime Chevalier-Boisvert. 2018. gym-miniworld environment for OpenAI Gym. https://github.com/maximecb/gym-miniworld.

[4] Gokhan Uzunbas M. Nam Lim S. Devrim Kaba, M. 2017. A reinforcement learning approach to the view planning problem.. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6933—-6941.

[5] Wei Jing, Chun Fan Goh, Mabaran Rajaraman, Fei Gao, Sooho Park, Yong Liu, and Kenji Shimada. 2018. A computational framework for automatic online path generation of robotic inspection tasks via coverage planning and reinforcement learning. *IEEE Access* 6 (2018), 54854–54864.

[6] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*. 1928–1937.

[7] Steven D. Whitehead and Dana H. Ballard. 1990. Active Perception and Reinforcement Learning. In *Machine Learning Proceedings 1990*, Bruce Porter and Raymond Mooney (Eds.). Morgan Kaufmann, San Francisco (CA), 179 – 188. https://doi.org/10.1016/B978-1-55860-141-3.50025-0

[8] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3-4 (1992), 229–256.