Reinforcement Learning of Active Vision for Manipulating Objects under Occlusions

Ricson Cheng, Arpit Agarwal, Katerina Fragkiadaki

2019

Bardia Mojra
February 24, 2021
Robotic Vision Lab
University of Texas at Arlington

Introduction

- A RL monocular RGB grasping system where camera pose is controlled through visual feedback to reduce occlusions.
- The authors pose the question of learning manipulation policies under occlusions and propose agents capable of hand-eye movement coordination with various distractors present in the scene.
- They introduce a modular actor-critic network architecture based on HER.
- They highlight the importance of Curriculum Learning methods.

Multi-Goal POMDP

- Multi-goal Partially Observable Markov Decision Process
- States, Goals, Action-Gripper, Action-Camera
- Action Space A: A^c X A^g
- Reward: r_g: S x A → R
- Policy: pi: O x G → A
- Results in reward r_t = r_g(o_t, a_t)
- Low dimensional embedding for object frame appearance, f_t
- Estimation target object pose, \hat{o_t}

Multi-Goal POMDP

- Multi-goal Partially Observable Markov Decision Process
- States, Goals, Action-Gripper, Action-Camera
- Action Space A: A^c X A^g
- Reward: r_g: S x A → R
- Policy: pi: O x G → A
- Results in reward r_t = r_g(o_t, a_t)
- Low dimensional embeddings for
  - object frame appearance, f_t
  - Estimation target object pose, \hat{o_t}
  - and known gripper position h_t

RL Networks

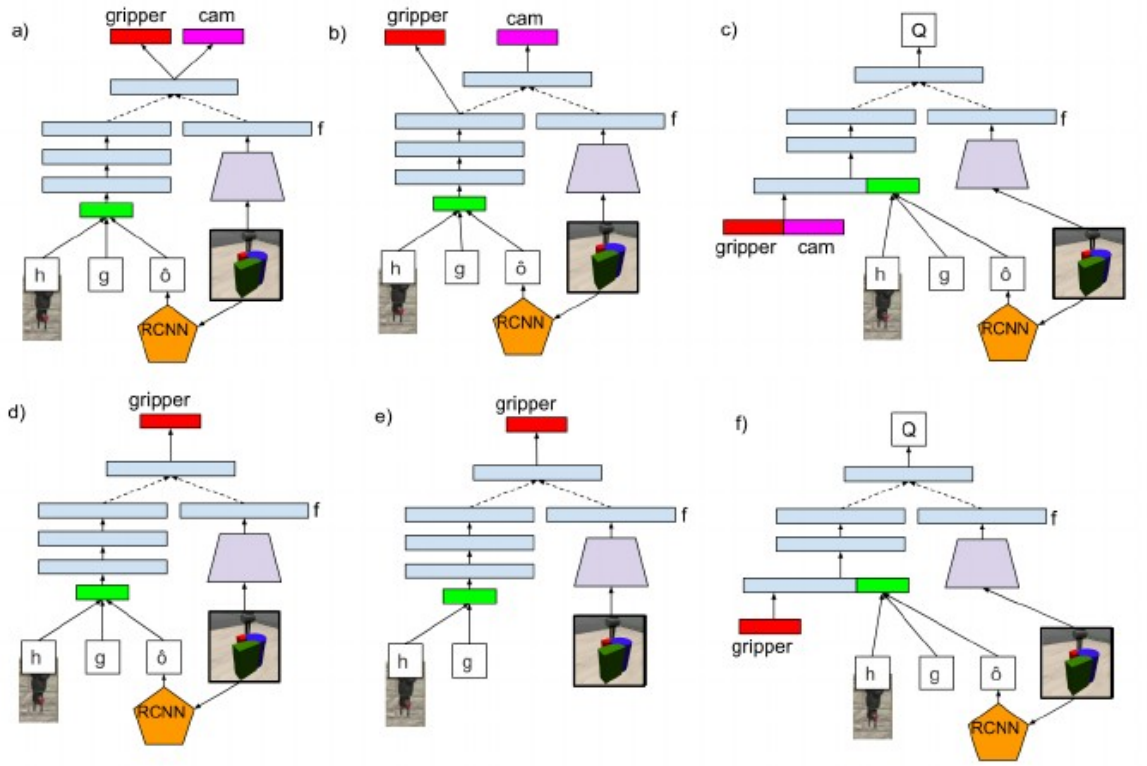

Figure 1: **Actors and critic architectures for active** *(top row)* **and static** *(bottom row)* **camera agents**. a) Actor network for *cam-active-full*. The same state representation is fed as input to hand and eye actor networks. b) Actor network for *cam-active-abstr*. The gripper actor responsible for pushing the object does not receive direct information from the RGB input, but only the 3D location of the object. c) Critic network for all active camera agents. d) Actor network for *cam-static*, the agent can only control its gripper, not its camera. e) Actor network for *cam-static-image*. The pretrained object detector subnetwork is omitted. f) Critic network for *cam-static*. Dotted lines denote elementwise addition. Trapezoids denote convolutional neural sub-networks, and blue rectangles denote fully connected layers.

Curriculum Learning

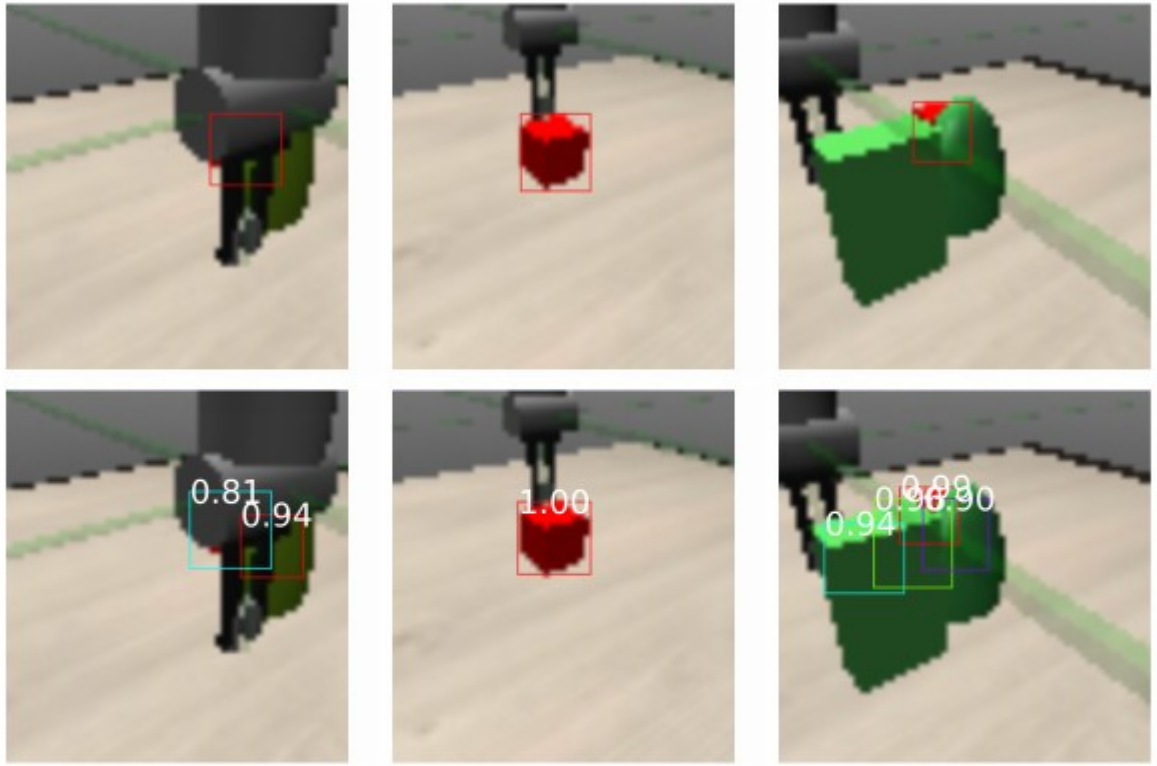

Figure 2: Sample predictions made by the trained RCNN. The RCNN is trained using *amodal* boxes. First row: ground truth, Second row: predictions, with confidence scores shown. Note that the RCNN is capable of accurately inferring the amodal boxposition of the object even when it is highcompletely occluded, but not when it is completely occluded.

Curriculum Learning

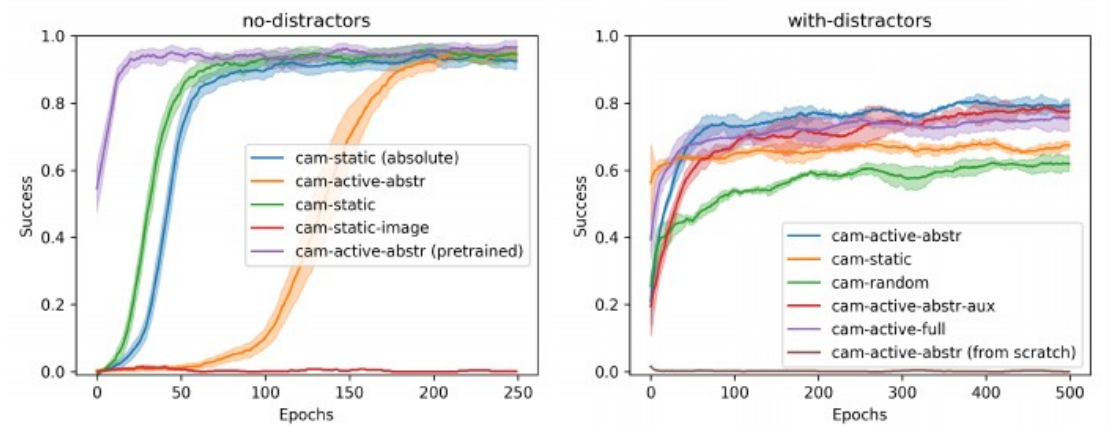

Figure 3: *Left:* **Environments without distractors.** Hand-eye policies train slower (*cam-active-abstr*), yet all architectures achieve good asymptotic performance. Hand-eye policies can be effectively pretrained from hand only policies (*cam-active-abstr* (pretrained)). Object-centric state encoding is beneficial (*cam-static* outperforms *cam-static (absolute)*). Finally, ignoring the location of the object of interest provided by the detector, and rather using only frame-centric appearance encoding does not result in successful behaviour (*cam-static-image*). *Right:* **Environments with distractors.** Active vision helps to handle occlusions from distractors (*cam-active-abstr* outperforms *cam-static*). State abstraction helps for the hand actor policy (*cam-active-abstr* outperforms *cam-active-full*). Training directly in the environment with distractors, without pretraining on the easier environment does not result in successful behaviours (*cam-active-abstr* (from scratch)). Auxiliary visibility reward is not helpful (*cam-active-abstr-aux*)necessary. A learned camera policy is superior to a random camera policy (*cam-random*). Shaded area shows 1 standard error on the mean fraction of episodes which ended with success during training. We took the mean and computed the error over 20 episodes in each of 5 training runs using different seeds.

Curriculum Learning

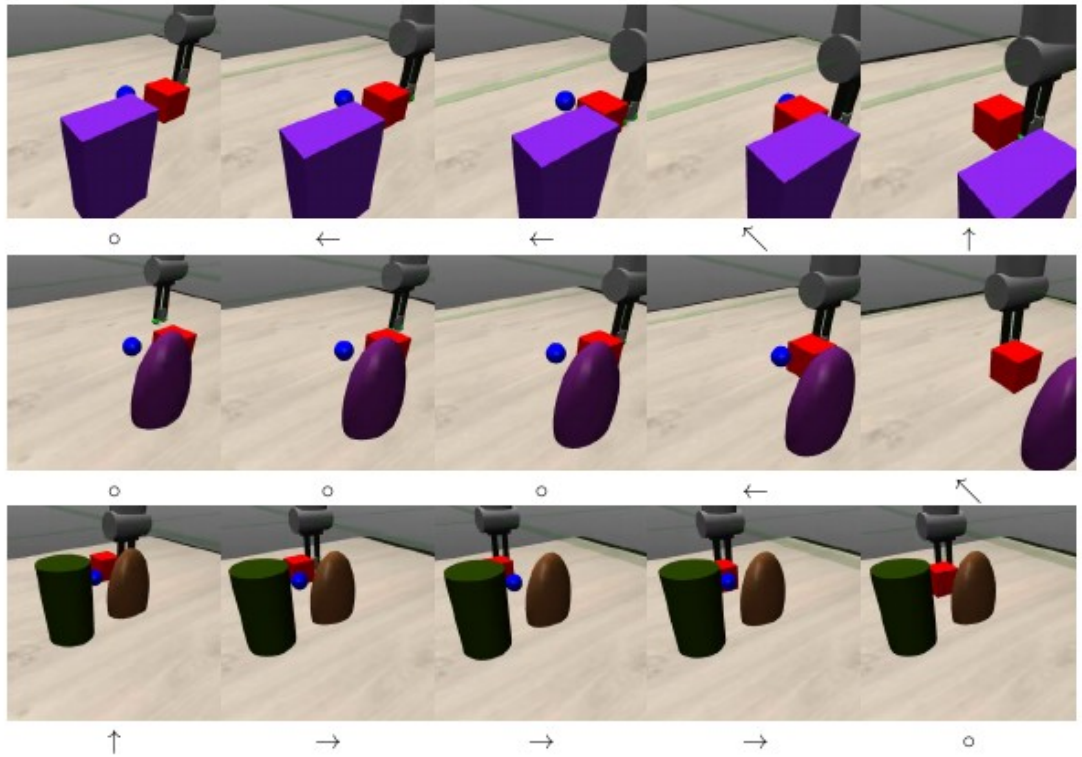

Figure 4: **Learned hand-eye control policies.** Each row corresponds to one episode, we show every other step of the episode. Since it is hard to tell the direction of camera movement from still frames, we draw arrows beneath each image showing the approximate direction of camera movement. In the top row, we see that the camera moves left and upwards to look over the obstacle. In the middle row, the robotic arm pushes the cube so that the left half is visible, and the camera moves left in order to expose the entire cube. In the bottom row, the cube is initially pushed leftward, so that if the camera is still, the cube would end up occluded by the cylinder. However, the camera moves right to compensate, so that the cube remains visible throughout the entire episode.

Visual Reward

camera	abstr	vis reward	success
active	yes	no	$71.0\% \pm 2.2\%$
active	yes	yes	$68.4\% \pm 1.5\%$
active	no	no	$64.6\% \pm 1.5\%$
random	yes	no	$55.7\% \pm 5.7\%$
static	–	no	$55.8\% \pm 6.6\%$

Table 1: Success rate of the final policies at test time, averaged across 100 episodes from each of 5 training runs using different random seeds in environments with distractors. Hand-eye control policies with state abstraction for the gripper actor and no visibility auxiliary reward perform best.

› Thank you!