Sim2Real Learning of Vision-Based Obstacle Avoidance for Robotic Manipulators

Kefang Zhang, Jiatao Lin, Lv Bi, Tan Zhang*

***Abstract*—Obstacle avoidance for robotic manipulator operating in an unstructured environment is a challenging task for deep reinforcement learning. It requires an accurate scene simulator to achieve model adaption from simulation to the real world. However, realistic scene modeling is difficult and time-consuming. To improve such sim-to-real model adaption, we propose a unified representation of obstacles and targets in both simulated and real worlds based on 3D bounding boxes. Such abstract representation is invariant to the shape and appearance of objects, thus unifying the scene representation for both worlds. Consequently, models trained in the simulated world can be effectively generalized to unseen scenes and unseen objects in the real world. This allows us to design a vision-based obstacle avoidance method that trains a Soft-Actor Critic (SAC) model in a simulator and directly applies the learned control policy in the real world. Our method achieves better performance and generation to unseen targets and dynamic scenes compared to state-of-the-art techniques.**

I. Introduction

Motion planning for robotic manipulators in open-world environments addresses the navigation of the end-effector to the desired target while avoiding obstacles. The recent advances in deep reinforcement learning (DRL) enables the development of adaptive vision-based controllers that can operate in dynamic environments. Such models can be learned end-to-end, mapping observations directly to actions. A critical issue is that such models can hardly be extended to the real-world scenarios, given that the trial-and-error search of a control policy through physical agent-environment interaction is prohibitively time-consuming [3], [4].

There are two types of Sim2Real approaches. The first builds a high-quality 3D environment with realistic rendering to train control policy to be used in the real world environments [6], [7], [9]. The second type is to manually create physical scenes similar to the simulated environments [2], or automatically reconstruct 3D models from real environments [8]. However, the domain discrepancy between simulated and real-world still exists.

To improve the sim-to-real model adaption, a unified representation for both simulated and real environments is highly desired. To achieve this, we opt for the shared representation learning inspired by [1], which proposed the concept of successor representation for RL to improve its generalization. In particular, we propose to use 3D bounding boxes to represent the obstacles and targets in both virtual and real worlds. Such abstract representation is invariant to the shape and appearance of objects, thus unifying the scene representation for both worlds. Consequently, models trained in the simulated world can be effectively generalized to unseen scenes and unseen objects in the real world.

*Corresponding author (tandy.zhang@gmail.com)
The authors are with Shenzhen University.

II. Architecture

An overview of the proposed obstacle avoidance method for robotic manipulators is illustrated in Figure 1. The World Model generates the 3D bounding boxes of targets and obstacles. The DRL model uses these bounding boxes and the robot's states to determine the robot's next action for reaching the target. Robot actions are represented by a set of joint angles that are fed into the robot low-level controller. Training is performed entirely in simulation, where it is easy to detect if the predicted actions cause collisions. By randomizing the simulated environment, we train a model that generalizes effectively to unseen scenes.

A. World Model

We train our model in a robotics simulator. The general idea is that the information of all objects with different geometries and positions in the physics engine is streamed to the RL framework, and the RL framework issues control commands based on the object information and send them back to the robot in the physics engine.

We demonstrate the performance of our model in a real dynamic setting using a real robot. In the real world, we obtain the 3D bounding boxes of all objects by using MaskFusion that can generate geometries and locations of the obstacles and use the trained model to control the robot to reach the target while avoiding obstacles.

B. Training

For the exploration module, we adopted the Soft Actor-Critic (SAC). SAC is an off-policy algorithm that optimizes a stochastic policy. It can be used for continuous actions. It is composed of three neural networks including a State value network to obtain the value of the next state, a Critic network to obtain Q value of actions from the actor network, an actor network to generate policy, which can also be called Policy network.

we define the reward function that allows the manipulator to reach the target directly and be penalized when any collision occurs. The reward function is defined as the sum of the weights of three items, i.e., action, collision situation, and the distance between the end-effector and the target destination:

$$R = w_1 R_a + w_2 R_c + w_3 R_d, \tag{1}$$

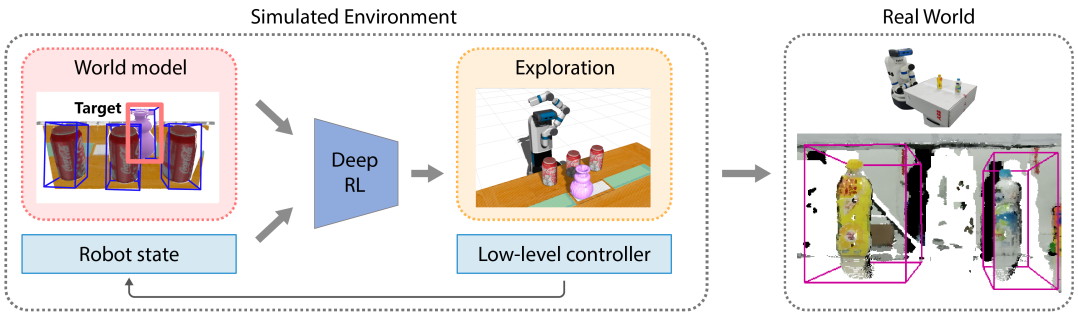

Fig. 1. An overview of our vision-based manipulator obstacle avoidance system.

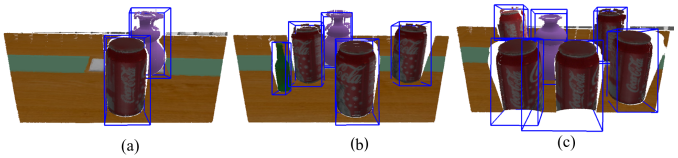

Fig. 2. The training scenes, where the vase is set as the target and other objects are obstacles.

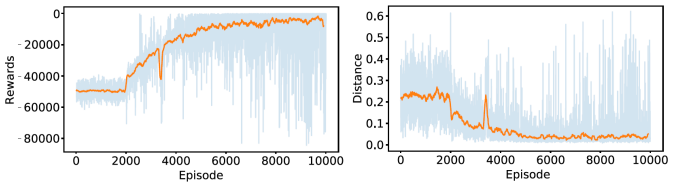

Fig. 3. The total distance and reward per episode for the scene shown in Figure 2(c).

where w_1, w_2, w_3 are the weights of the action R_a, collision situation R_c, and the distance between the end-effector and the target destination R_d, respectively. The distance R_d is calculated using the Huber loss function:

$$R_d = \begin{cases} 0.5 * d^2, & \text{for } d < \delta \\ \delta(|d| - 0.5 * \delta), & \text{otherwise} \end{cases} \quad (2)$$

where d is the Euclidean distance between the end-effector of the manipulator to the target, and the parameter δ determines the smoothness.

III. Experiments

A. Performance in Simulated Environments

We used open-source Gazebo as a simulator and combined with ROS (Robot Operating System) [5] to implement the manipulator obstacle avoidance training. We used a single-arm robot Fetch. Figure 3 shows the progress of the training of the scene in Figure 2(c), which shows the reward initially increasing and distance decreasing, and then both parameters converging.

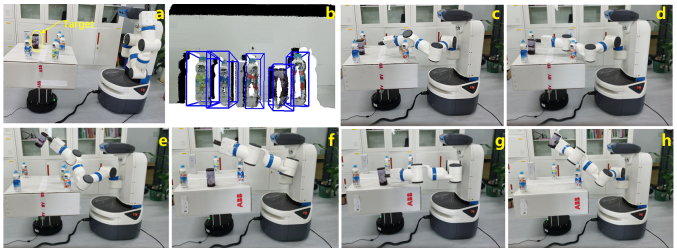

Fig. 4. The real experiment setup (a) and the real-time reconstruction (b) of the scene. (c-h) Video frame sequences when the robot reaches the target.

B. Experiments in Real Environments

To validate the generalization of the model to different real-world settings, we tested our trained model in dynamic scenes where objects have different geometries and keep changing positions. To achieve this, we put obstacles and targets on a box mounted on Turtlebot 2; see Figure 4. The Turtlebot rotated 360 degrees clockwise, which drives the obstacles and targets rotate 360 degrees. A new scene was generated each time the Turtlebot rotated 30 degrees to a new position. We used the depth sensor on the Fetch robot to obtain the 3D bounding boxes of objects in the environment. Figure 4(b) shows the result of the scene model.

We trained a model in a similar way as the proposed approach, using geometries and locations as observation instead of 3D bounding boxes. In the training scenario, we used all box-shaped objects. We then applied the new model to scenes where objects are represented as bounding boxes. Twelve scenes in both simulated and real environments were tested. A video of the robot demonstrating the final experiment by reaching targets in different real-world scenes is available at https://youtu.be/7_d6nu0iV70.

It was found that the robot was able to reach the target in real-world for nine times. There were three times the robot collided the obstacles when two consecutive scenes require a larger magnitude of the actions. It can be seen that the proposed method outperforms in transferring the model from simulation to the real world. This demonstrates the effectiveness of the unified representation in learning real-world interactions and shows a robust generalization from simulation to the real world.

REFERENCES

[1] Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

[2] Mihai Duguleana, Florin Grigore Barbuceanu, Ahmed Teirelbar, and Gheorghe Mogan. Obstacle avoidance of redundant manipulators using neural networks based reinforcement learning. *Robotics and Computer-Integrated Manufacturing*, 28(2):132–146, 2012.

[3] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436, 2018.

[4] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016.

[5] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[6] Fereshteh Sadeghi and Sergey Levine. Cad2rl: Real single-image flight without a single real image. *arXiv preprint arXiv:1611.04201*, 2016.

[7] Linhai Xie, Sen Wang, Andrew Markham, and Niki Trigoni. Towards monocular vision based obstacle avoidance through deep reinforcement learning. *arXiv preprint arXiv:1706.09829*, 2017.

[8] Shichao Yang and Sebastian Scherer. Cubeslam: Monocular 3-d object slam. *IEEE Transactions on Robotics*, 35(4):925–938, 2019.

[9] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.