UROP projects for learning to reason for task-and-motion planning problems

Beomjoon Kim

General instruction

We have two concrete projects, but I am open to pursue other projects that is relevant for this topic. If you are interested in pursuing one of the two concrete projects, then please read the section for the corresponding project. If you are interested in proposing your own topic, then please proceed to Topic 3. If you are interested in one of these topics, or have any question, please email me at beomjoon@mit.edu.

Topic 1. Learning a value function and a policy from past planning experience

Since TAMP problems involve a hybrid action space, we need to sample actions at each search node. If we can sample promising actions and decide which node to expand first, then we would be able to compute a solution more efficiently. In our paper recent paper (link), we have devised a method for learning a policy from planning experience and showed that it can indeed improve planning efficiency. In this project, we aim to explore this idea further and devise a planning algorithm that improves with experience.

Questions

Read the papers

- Guiding Search in Continuous State-action Spaces by Learning an Action Sampler from Off-target Search Experience (link)
- Mastering the game of Go without human knowledge (link)

Please answer the following questions.

1. In the first paper, what is the purpose of the action sampler?
2. In the second paper, what is the purpose of value v and p?

3. In the first paper, learning is applied to robotics domains, and in the second paper, it is applied to the game of Go, both to improve planning efficiency. Since robotics domains involve continuous action space, we need to sample actions, unlike the game of Go. In planning, we need to choose a search node to expand. What additional concerns do we have in choosing the next node to expand when we have a continuous action space?

4. Do the programming exercises at the end of this document.

Topic 2. Representation learning for geometric reasoning

The main bottleneck for task-and-motion planning problems is answering if a particular robot configuration is reachable from current configuration amid a set of obstacles. This is can be done by calling a motion planner, such as rapidly-exploring random tree, but it could get very slow if the configuration is unreachable. We would like to devise a faster way to approximately answer the reachability questing using machine learning. The challenge is that we do not yet have a canonical method to represent a scene with different number of obstacles with varying shapes with a vector. This project aims to devise an effective representation for inferring a reachability.

Questions

Read the paper, Rapidly-exploring random trees: progress and prospects. (link) Now suppose that a robot accumulates a set of path planning experience that consists of a set of initial and goal configurations, and collision-free paths between them, for different poses and shapes of obstacles and the robot in a certain environment.

Please answer the following questions that will guide you in formulating a classification problem for inferring reachability that maps a set of obstacles, their poses, initial robot configuration, and goal robot configuration to the range [0,1], where 1 indicates reachable and 0 indicates otherwise.

1. What is your input to the classifier? Justify your choice

2. What is your objective function? Justify your choice.

3. What is the architecture of your network? Justify your choice.

4. What other information, if any, do you need to predict reachability?

5. Do the programming exercises at the end of this document.

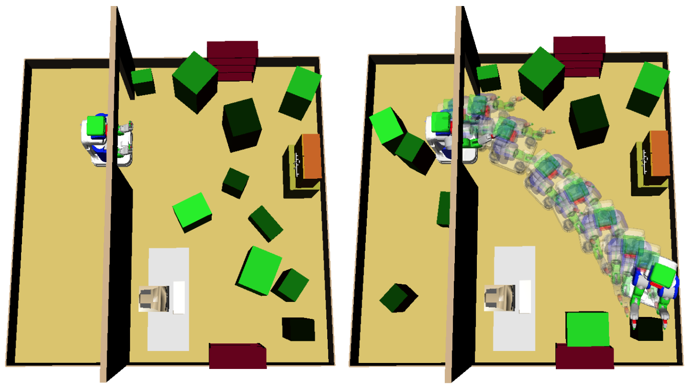

Figure 1: Object fetching problem. Initial state (left) and a goal state with a fetching path (right)

Topic 3. A new project

If you would like to propose a new topic that is relevant for using learning to make planning more efficient, then you should write a proposal that has the following components:

1. Problem formulation: what are the unknowns, givens, and constraints?
2. Justification: if we solve this problem, how would it improve planning efficiency?
3. One or two papers that are relevant to the proposed the problem.

You should also do the programming exercises in section 4.

Programming exercises

We are interested in a robot planning problem where the robot has to clear a path to fetch a target object. This requires planning a sequence of pick-and-place motions to clear obstacles. An example planning scene and its solution is shown in Figure 1. The goal of this exercise is to train a policy that maps a representation of a state to an object placement.

1. Download the ROBOT OBJECT PLACEMENT dataset (link). Use Python Pickle library to load the dataset. This dataset is a dictionary that contains *pose_states*, *collision_states*, and *action* fields, obtained from solving past object fetching problems. We have about approximately 1000 data points. An action corresponds to choosing a base pose for placing the object down. Using supervised learning, train a regressor that maps a pose state to an object placement. Report the mean-squared-error (MSE).

2. Do anything you can to further minimize the MSE. Report the improved MSE, the tricks you have used, and why you thought the tricks would be helpful.

3. *collision_states* field includes a set of states with a collision-based state representation, which has collision information at some important robot configurations. Each point represents collisions at a state with a one-hot-encoding, where the first bit is set to 1 if there is a collision at the corresponding configuration. Design an architecture that is appropriate for this representation, and train a regressor that maps this representation to an object placement.

4. What other information do you think are needed to predict the placements?

5. Submit your code and answers.