Observational and self-learning of multi-step robotic manipulation with unknown physical properties

Claudia Pérez-D'Arpino
Massachusetts Institute of Technology
cdarpino@csail.mit.edu

Julie A. Shah
Massachusetts Institute of Technology
julie_a_shah@csail.mit.edu

Abstract

Learning from both observations and self-exploration is a key capability of intelligent agents. It is present to various degrees in a number of animals that can combine information distilled from observing a task being performed with further knowledge acquired through self-exploration that contains variations over the demonstrations. We explore this mechanism in robotic arms in the context of learning to execute multi-step manipulation tasks. Specifically, the system accumulates geometric information on how humans typically manipulate objects with simple geometric shapes through learning from demonstrations. This geometric knowledge base is leveraged by a planner to find manipulation strategies given a novel scene. In addition, the new environment presents uncertainty in the values of the physical properties of the objects, such as their mass. We present an agent that has access to a simulator of the environment, instantiated by a physics engine, and uses it to explore multiple manipulation strategies and find successful sequences using Monte Carlo Tree Search. The search runs multiple forward simulations of the novel scene by hypothesizing plausible values for the unknown parameters of the mass of the objects. This abstract presents a preliminary implementation and results of this system, in which a dual-arm 16-DOF manipulator finds novel manipulation strategies by combining a geometric knowledge base learned from demonstrations with self-exploration in hypothesized simulated worlds in a simple task.

1 Introduction

Learning from both observations and self-exploration seems to be a key enabler of intelligent behaviors that can be generalized to variations of the original observed task. In the context of the manipulation of objects, it is possible to gather some information, especially of the geometric type, by observing another person performing it. Without further communication or an extensive number of demonstrations, it is challenging to observe all possible cases in which this task can occur, or even all the important features that were relevant to the teacher's execution. For example, we might observe the dimensions of an object being moved, but not its mass. This additional information could be relevant at the moment of performing the task as it might demand adjustments in the steps, e.g., a heavier object might need a different grasp from the one demonstrated. Demonstrations are useful as an initial guidance as to the steps and goals of the task, but generalizing this information to make plans for a similar task in a different context or arrangement might require further knowledge. One method to generate this body of knowledge is through self-exploration, in which the learning agent can experiment with different options to find new strategies for the task. Naturally, the initial set of demonstrations reduces the search space at both the task and motion levels.

Self-exploration is the process of generating and trying out new possible sequences of steps to perform a manipulation task. In robotics applications, the field of reinforcement learning has pioneered self-exploration on both real platforms and simulations [4] for the learning of manipulation tasks in the continuous control space of torques applied to joint motors. The combination of demonstrations and self-exploration through reinforcement learning in the continuous control setting has also been a topic of research [11], with findings that indicate that an initial set of demonstrations can speed up

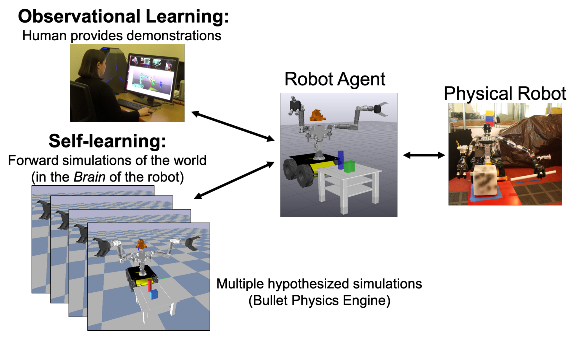

Observational Learning:
Human provides demonstrations
Self-learning:
Forward simulations of the world (in the Brain of the robot)
Robot Agent
Physical Robot
Multiple hypothesized simulations (Bullet Physics Engine)

Figure 1: A *robot agent* has access to human demonstrations as well as unlimited simulations of the world in which it can act and observe the outcomes of its actions. Simulations are run in the *brain* of the robot and are spawned using a physics engine (Bullet) with a world that represent the current scene observed by the real physical robot in its environment. The mass of the objects is unknown to the robot agent, and it uses the simulator to experiment with multiple hypothesized values of mass of the objects.

the policy-learning process as well as achieve results, in terms of accumulated rewards, that exceed what either learning from demonstrations or reinforcement learning can achieve in isolation with comparable effort.

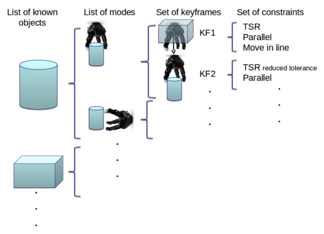
In our work, learning through observations and self-exploration is instantiated at the level of sequences of steps as logical reasoning and motion plans for each step as geometric reasoning. In the field of robotics, this approach is tightly related to methods for task and motion planning (TAMP) [3], in which sophisticated heuristics for logical and geometric reasoning are combined to solve mobile manipulation problems.

We propose an architecture in which a *robot agent* combines *observational learning* with *self-learning* through simulation, to find manipulation strategies that succeed in different scenarios that involve different physical properties of the objects (e.g., mass). This system is illustrated in Fig. 1.

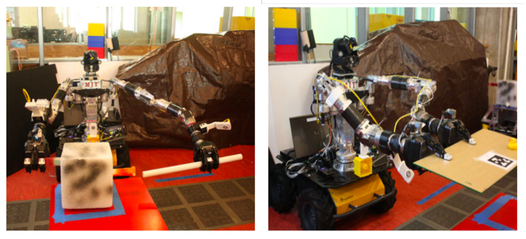
2 Observational Learning: Learning Multi-step Tasks from Few Demonstrations

In the observational learning stage, we leverage our previous work on learning multi-step manipulation tasks from a few human demonstrations [7]. These tasks are similar to assembly or disassembly of parts and objects, in which typically there is one or more logical sequence of steps (e.g., grasp object, move object, etc.), and geometric constraints are required to perform the task (e.g., move the object, keeping it perpendicular to the floor). The goal is to learn a multi-step task as a sequence of steps (keyframes) and a set of geometric constraints for each step. These geometric constraints are expressed in the form of SE(3) volumes in object coordinates. This algorithm, C-LEARN, works in two stages. First, it builds a geometric knowledge base (GKB) of how humans typically manipulate simple geometric shapes. This is done by collecting a small number of human demonstrations, on the order of 5 or 6. The structure of the GKB is depicted in Fig. 2a. The second stage consist of collecting a single demonstration of a multi-step task, such as the examples in Fig. 2b. C-LEARN combines the information in this single demonstration and in the GKB to compute a sequence of keyframes with parameterized constraints so that when they are given to an optimization-based motion planner, it can generate plans to execute this task in new environments where the objects might have different positions and orientations compared to the initial demonstrations. For a detailed explanation, we refer the reader to [7]. Additionally, video examples can be watched at `http://people.csail.mit.edu/cdarpino/CLEARN/index.html`.

The next challenge is to be able to generalize further, not only to new positions and orientations of the objects but also to different manipulation strategies. In a setting where we have available a full model of the world, this formulation would be a task and motion planning problem. However, additional challenges arise when we consider uncertainty in the outcome of actions when performed in the real world. One common source of uncertainty in real manipulation tasks without a full model of the world is the variations in physical properties of the objects that the agent does not know in advance. The human demonstrations are in some sense incomplete, as the robot is able to observe the geometric aspects but not the full set of features that the human took into account such as physical properties of the objects. Without lack of generality and for the sake of simplicity of implementation, we start by considering variations of only the mass of the objects. To learn multiple strategies that can



(a) Geometric knowledge base



(b) Example multi-step tasks

Figure 2

handle variations in the outcome of manipulation strategies due to this latent factor, we complement the observational learning stage with a self-exploration stage, which spawns multiple simulations with hypothesized values of the mass of objects. This stage is covered in the following section.

3 Self-learning: Finding Novel Manipulation Strategies

The self-learning stage incorporates elements from task and motion planning, optimization and sampling-based motion planning, and Monte Carlo Tree Search (MCTS). We give the robot agent access to create and run simulations using a physics engine. Analogous to the hypothesized existence of an intuitive physics engine in the human brain [2] [8] and computational approaches to emulate this physics modelling, [6] [10], we allow our virtual agent to envision multiple possible futures as it takes actions in the world. We start by using a full physics engine based on the open source Bullet [1].

Equipped with the geometric knowledge base from human demonstrations and the simulator, the robot agent runs traces of simulations following MCTS. MCTS was most recently successful at the game of Go [9], in which the outcome of simulation steps is determined according to the rules of the game of Go, and the uncertainty comes from the plays of the opponent in the game. Analogously, in our case the uncertainty comes from the physical outcome of an action which is not known a priori by the robot agent but instead revealed by the simulator as the simulation unfolds.

The rollout policy of the MCTS is guided by interleaved task and motion planning. For task planning, we use a PDDL planner [5] that, given a domain description, a problem description obtained from the current scene of the physical robot, can produce sequences of high-level actions that accomplish the task if all the outcomes follow the expected results. For example, in a task with a cylinder and a box object where the goal is to move the cylinder to the left and the box to the right, the actions of grasping, moving, and releasing these objects can be executed in a number of different ways. PDDL planners are suited to efficiently solve this type of problem. Each of the task-level actions has a potentially infinite number of geometric instances, but we reduce the search space by using the information in the geometric knowledge base.

Using this procedure, a PDDL planner produces a set of plans indexed by n_{plan}, each with a number of actions indexed by n_{action}. Each action can be executed with a number of geometric approaches indexed by n_{mode}, and each approach can be divided in an ordered sequence of motion plans indexed by $n_{keyframe}$. Rollouts use a binary reward that reflects success or failure in accomplishing the goal state of the task. Examples of simulation traces of the MCTS are shown in Fig. 3, together with a description of the tree branching. Each node in the tree represents a keyframe, which is the final state after a manipulation action has taken place (e.g., grasp object). Manipulation actions are taken in sequence in each trace according to the rollout policy. The result of the search is a candidate sequence of actions that tends to be successful in the Monte Carlo sense. Additionally, we have access to a number of other possible traces that were run in the simulator. For example, five traces are shown in Fig. 3. In World 1, the cylinder and box are moved by the left arm with a top grasp. In World 2, the box has a higher mass than in World 1, and the same top grasp fails because the box slips. A different rollout in World 2 finds a different sequence with a lateral top grasp that succeeds with the heavy box. Videos of these examples are available at `http://people.csail.mit.edu/cdarpino/neurips18_learnSimPhysics/index.html`

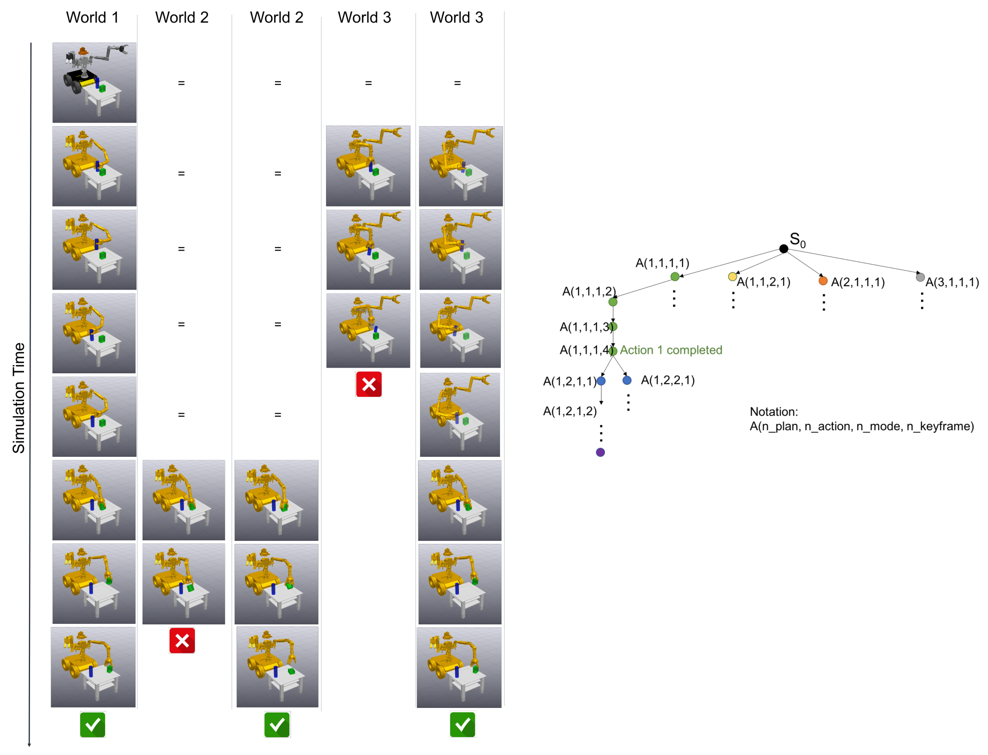


Figure 3: Traces of the Monte Carlo tree with simulations for worlds with different values of mass for the object cube.

Taking advantage of recent progress in the development of simulation tools, robot agents can be used to design and experiment with learning models of the physical world by hypothesizing possible scenarios in simulations before acting in the real world. These scenarios can consider both known and unknown physical properties by experimenting with many feasible values of the latent parameter(s). This framework enables the robot to computationally exercise a predictive engine of the physical world around it that goes beyond rules explicitly encoded in a system, removing the barriers for further generalization and adaptation to new situations beyond pre-designed sequences.

References

[1] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org`, 2016–2018.

[2] Jason Fischer, John G Mikhael, Joshua B Tenenbaum, and Nancy Kanwisher. Functional neuroanatomy of intuitive physical inference. *Proceedings of the national academy of sciences*, 113(34):E5072–E5081, 2016.

[3] Caelan Reed Garrett, Tomas Lozano-Perez, and Leslie Pack Kaelbling. Ffrob: Leveraging symbolic planning for efficient task and motion planning. *The International Journal of Robotics Research*, 37(1):104–136, 2018.

[4] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *ICRA*, pages 3389–3396, 2017.

[5] Michael Katz, Shirin Sohrabi, Octavian Udrea, and Dominik Winterer. A novel iterative approach to top-k planning. In *ICAPS*, pages 132–140, 2018.

[6] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2017.

[7] Claudia Pérez-D'Arpino and Julie A. Shah. C-LEARN: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy. In *IEEE ICRA 2017*, 2017.

[8] Sarah Schwettmann, Jason Fischer, Joshua Tenenbaum, and Nancy Kanwisher. Neural representation of the intuitive physical dimension of mass. *Journal of Vision*, 18(10):731–731, 2018.

[9] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484, 2016.

[10] Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *Advances in neural information processing systems*, pages 127–135, 2015.

[11] Josh Merel Andrei Rusu Tom Erez Serkan Cabi Saran Tunyasuvunakool János Kramár Raia Hadsell Nando de Freitas Nicolas Heess Yuke Zhu, Ziyu Wang. Reinforcement and imitation learning for diverse visuomotor skills. *Robotics: Science and Systems (RSS)*, 2018.