Case study of model-based reinforcement learning with skill library in peeling banana task

Fucai Zhu, ∘Akihiko Yamaguchi, Koichi Hashimoto

Abstract : We report a case study of model-based reinforcement learning with a library of skills in a peeling banana task. We summarize our methods and the experiments, and describe a comprehensive understanding of the features of our methods.

1. Introduction

Recent research progress makes artificial intelligence tools stronger, but it is still an open challenge to achieve complicated manipulation tasks such as cooking [1], peeling vegetables and fruits [2], and pouring liquids [3]. Based on the insight that we should represent and reason about alternative skills [3, 4], we are hypothesizing that combining symbolic representations (skills), and low-level reasoning and learning is a promising approach.

In this research, we chose a banana peeling task as an example task. It is complicated manipulation since bananas consist of deformable, elastic, and breakable components, each banana has different properties, and the property (e.g. softness of skin) changes even during manipulation. So far, we have:

- Developed a reinforcement learning method based on a library of skills [5].
- Developed a framework to use *human actors* to quickly implement and test perception and control algorithms [6, 5].
- Implemented a banana peeling system and conducted experiments [6, 5].

In this paper, we describe comprehensive understanding of these studies. We begin with summarizing our reinforcement learning method, the framework to use human actors, and the implementation and experiments.

2. Model-based Reinforcement Learning with Skill Library

This section summarizes our reinforcement learning method with a library of skills [5]. It is a method to learn and generate robot behaviors based on a given library of skills. Each skill is defined as a control policy with parameters; the parameters adjust the skill for different situations. We consider alternative skills, for example a tipping and a shaking skills for a pouring task. There is work focusing on acquiring skills [7], while our method focuses on how to combine learned primitive skills in solving complicated tasks. The problem solved here can be formulated as follows: for a given task represented by reward functions, find a combination of skills and their parameters such that the sum of rewards is maximized.

We consider situations where a part of the dynamics are not known in advance, i.e. this problem is reinforcement learning. We take a model-based approach, i.e. we train regression models of dynamics and generate a policy by solving dynamic programming with the learned models.

The proposed method unifies symbolic reasoning [8] and dynamic programming for graph-structured dynamical systems [9]. The problem to find a combination of skills is a form of symbolic reasoning, but it is difficult to completely remove the ambiguity of skill selections (alternative skill selections) with symbolic reasoning. Instead of a linear sequence of skills, we allow symbolic reasoning to find a graph structure of skills which includes alternative skill selections (behavior graph). On the other hand, unlike standard dynamic programming, graph dynamic programming [9] searches for selections of skills and skill parameters where the skills have a directed graph structure. Thus, by combining the ambiguous symbolic reasoning and graph dynamic programming, we can find a complete plan consisting of a sequence of skills and their parameters. From the result of executing the plan, we can update the dynamics models of the skills. Fig. 1 illustrates the overview of the proposed method.

The proposed method automates behavior generation of complicated tasks with a given set of skills. Compared to a hand designed policy, our approach may make behaviors more robust. There are a number of similar approaches such as skill-based approaches [10], hierarchical reinforcement learning [11, 12], policy decomposition and composition [13, 14, 15], a unified framework of symbolic task planning and geometric motion planning [16, 17]. The advantages of the proposed method are: (1) The skill library may include non-homogeneous skills, i.e. each parameterized skill may have a different time scale, state, and control spaces. A skill also may consist of the other skills. (2) The planning of the behavior considers the dynamics of the objects and the robot, which makes the proposed method applicable to manipulation tasks with non-rigid objects, for example peeling bananas and pouring liquids.

3. Human Actor in the Loop

In recent years, robot programming is becoming easier thanks to middleware such as the robot operating system (ROS). However it is still complicated and takes long time to implement a complicated manipulation task

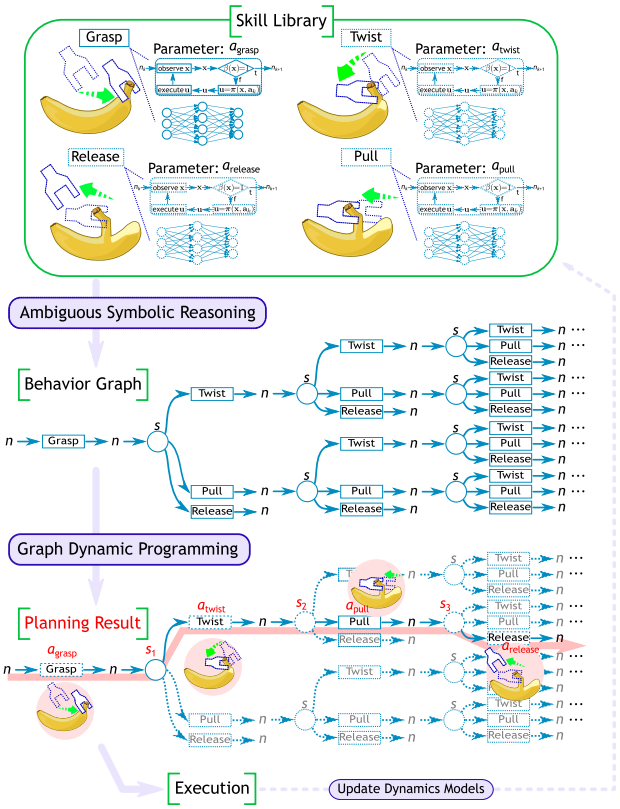

Fig. 1: Overview of our model-based reinforcement learning with skill library.

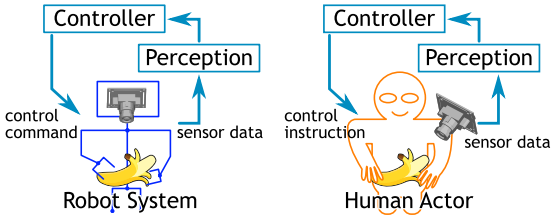

Fig. 2: Left: Typical robotic system for manipulation. Right: Conceptual illustration of the proposed framework.

like banana peeling. Such an implementation includes robotic arms, hands, perception systems (vision, tactile), and motion control, and implementing each of them has different difficulties. In order to reduce such complexity, we developed a framework to use human actors who follow computer-generated instructions rather than using robots [6].

Figure 2 shows the conceptual illustration of the proposed framework compared with a typical robotic system for manipulation. Both of them have the same perception system and controller. However, the robot is operated by commands, while a human actor obeys control instructions generated by the controller.

The design of the control instructions is important. The instructions should be human understandable, and ambiguity should be reduced as much as possible so that different human actors perform the instructions similarly. We chose to combine a text instruction and a

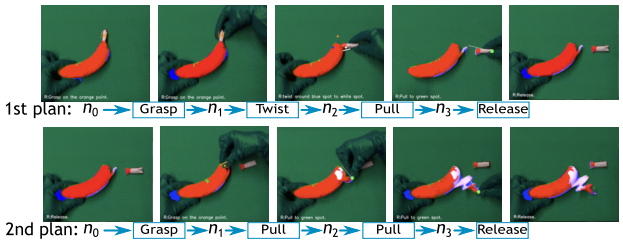

Fig. 3: Result of manipulation with the proposed method.

graphical user interface (GUI) to represent the control. For example, the grasping position and orientation are indicated through the GUI.

4. Experiments

In order to investigate how the proposed methods work, we applied them to a banana peeling task. We used human actors to implement the task. For simplicity, we considered peeling bananas on a table where the motions are performed in 2D. An RGB camera was used to perceive the banana state including the main body pose, end point positions, and flesh area. A monitor was used to display the control instructions with text and GUI.

From observations of humans peeling bananas, we designed the skill library with four skills: *Grasp, Twist, Pull, and Release*. For each skill, we designed symbolic dynamics models (precondition $\{0, 1\} = B(X_k, A_k)$ and effect $X_{k+1} = E(X_k, A_k)$) and a numerical dynamics model $\mathbf{x}_{k+1} = \mathbf{F}(\mathbf{x}_k, \mathbf{a}_k)$. Some of them were designed manually, and the others were modeled with neural networks. We considered the goal of the banana peeling task to be exposing the flesh area to the maximum. The reward function was designed as the flesh area.

During the experiment, each subject was asked to act as a human actor and execute the control instructions generated by the system. Before the experiment, each human actor was trained enough to become familiar with the GUI and text instructions. We used two control methods: one is the proposed reinforcement learning method, and the other is a manually-designed policy modeled with a state machine.

Fig. 3 shows an example of peeling behavior. The results of the experiments demonstrated that: (1) The framework to use human actors instead of robots worked to implement a complicated manipulation task. (2) With both control methods, our system peeled bananas successfully. (3) Our reinforcement learning controller was more efficient than a manually designed policy. The reason is that our method optimizes the skill parameters for the current state, while the manually designed policy uses pre-optimized parameters. (4) Our reinforcement learning controller was robust which was able to handle unexpected situations, such as flicking back of a banana skin due to its elasticity.

5. Discussions

5.1 Model-based Reinforcement Learning with Skill Library

The comparison of the model-based and model-free approaches discussed in [18] can be summarized as follows: (1) Model-based learns models of forward dynamics, while model-free directly learns policy. (2) Learning-complexity of model-based is supervised learning, while that of model-free is reinforcement learning. (3) Planning-complexity at execution of model-based is dynamic programming, while model-free does not require planning at execution. (4) Model-based is suffered from intractability of dynamics (simulation bias [19]). Model-free tends to acquire higher performance, and is more robust to POMDP. (5) Model-based is considered to generalize the learned behavior over wider situations than model-free. (6) Model-based is easier to share (reuse) learned components among different tasks than model-free, since the forward models are independent from tasks. A technique like importance sampling is necessary to reuse a policy in other tasks. Similarly, model-based is easier to modify the design of reward functions in the same task.

Since our reinforcement learning library takes a model-based approach, it inherits the features of model-based. For example, we were able to train each dynamics model independently without running complete peeling. It was also possible to redesign the reward functions after learning dynamics. Introducing skill library gives following advantages: (7) Representing a behavior with skills is considered as a temporal and spatial segmentation of behavior. It reduces the number of iterations of temporal integration, which leads the reduction of simulation-bias. (8) Efficiency to generate complicated behaviors is increased. For example, it was able to achieve a peeling banana task. (9) Robustness to adapt to various situations is increased. For example, it was able to handle failure recoveries in peeling banana.

On the other hand, the drawbacks are: (10) Many of dynamics (symbolic and numerical) are modeled manually. Due to that, the controller encountered unexpected situations, such as breaking body and skin, and flicking back of skin caused by the elasticity of skin. (11) Behavior graph will become complicated when we consider more skills, symbolic states, and actions, which will increase the planning complexity.

The first drawback may be solved by learning more models [20], although it will increase the learning cost. Ideas to tackle to the second one include introducing a hierarchy into symbols in order to shrink the graph size, and combining model-based and model-free approaches to reduce the planning cost.

6. Human Actor in the Loop

The followings are comparisons of a human-actor system and a robot system: (1) Implementation of skills for human actors takes a few weeks, while that for robots takes months to years. (2) Setup of experiments takes hours for both systems. The human actor system requires training time of actors, typically 0.5 to 1 hour per actor. (3) Human actors become tired and impatient after long run, while robots are stable in long run. (4) Maximum working time of human actors will be a few hours per day. Robots can work 24 hours. (5) Human actors are adaptive to situations, while robots are strict to commands. (6) Human actors behave with small feedback gain, while robots are typically operated with high feedback gain. (7) We typically use some human actors per experiment each of who has different physical and geometrical properties, while we use a single robot. (8) Commands for human actors can be ambiguous, while commands for robots should be accurate and detailed.

The advantages of the framework to implement manipulation tasks using human actors are summarized as follows: (9) It can handling complicated manipulation tasks such as banana peeling. (10) We can quickly implement and test the system, perception (computer vision), and control strategies. (11) Control commands can be ambiguous: it decreases the implementation cost. An example is a grasp skill.

On the other hand, there would be following disadvantages: (12) At each experiment, we need to train each subject (human actor) who may forget the training after some weeks. (13) Hundreds of trials are difficult to perform with human actors. Note that it does not mean that we can easily conduct such number of trials with robots. Cost of non-invertible materials (such as bananas) is the same for both systems. (14) Ambiguity of commands will cause disturbances of results.

7. Summary

This paper summarized our study of reinforcement learning with skill library, using human actors in implementing complicated manipulation tasks, and the application to a banana peeling task. Based on them, we described a comprehensive understanding of the features of our methods.

Acknowledgement

This work was supported in part by JSPS KAKENHI Grant Number JP16H06536.

References

[1] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mösenlechner, D. Pangercic, T. Rühr, and M. Tenorth, "Robotic roommates making pancakes," 10 2011, pp. 529–536.

[2] J. Hughes, L. Scimeca, I. Ifrim, P. Maiolino, and F. Iida, "Achieving robotically peeled lettuce," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4337–4342, 2018.

[3] A. Yamaguchi, C. G. Atkeson, and T. Ogasawara, "Pouring skills with planning and learning modeled from human demonstrations," *International Journal of Humanoid Robotics*, vol. 12, no. 3, p. 1550030, 2015.

[4] A. Yamaguchi, "Baxter peels banana," https://youtu.be/rEeixPBd3hc, 2016, [Online; accessed Oct-28-2016].

[5] F. Zhu, A. Yamaguchi, D. Beßler, M. Beetz, C. G. Atkeson, and K. Hashimoto, "Model-based reinforcement learning with a skill library and its verification with a banana peeling task," in *the IEEE International Conference on Robotics and Automation (ICRA'20)*, 2020, (Under review).

[6] F. Zhu, A. Yamaguchi, and K. Hashimoto, "Using human actors in robotic manipulation study: An initial attempt with peeling banana tasks," in *the 19th SICE System Integration Division Annual Conference (SI2018)*, 12 2018.

[7] A. Rajeswaran, V. Kumar, A. Gupta, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *ArXiv e-prints*, no. arXiv:1709.10087, 2017.

[8] M. Tenorth and M. Beetz, "A unified representation for reasoning about robot actions, processes, and their effects on objects," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 1351–1358.

[9] A. Yamaguchi and C. G. Atkeson, "Differential dynamic programming for graph-structured dynamical systems: Generalization of pouring behavior with different skills," in *the 16th IEEE-RAS International Conference on Humanoid Robots (Humanoids'16)*, 2016.

[10] T. Hasegawa, T. Suehiro, and K. Takase, "A model-based manipulation system with skill-based execution in unstructured environment," in *Fifth International Conference on Advanced Robotics*, 1991, pp. 970–975.

[11] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial Intelligence*, vol. 112, pp. 181–211, 1999.

[12] A. G. Barto and S. Mahadevan, "Recent advances in hierarchical reinforcement learning," *Discrete Event Dynamic Systems*, vol. 13, no. 4, pp. 341–379, 2003.

[13] F. Fernández, J. García, and M. Veloso, "Probabilistic Policy Reuse for inter-task transfer learning," *Robotics and Autonomous Systems*, vol. 58, no. 7, pp. 866–871, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889010000655

[14] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine, "Learning modular neural network policies for multi-task and multi-robot transfer," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2169–2176.

[15] X. B. Peng, M. Chang, G. Zhang, P. Abbeel, and S. Levine, "Mcp: Learning composable hierarchical control with multiplicative compositional policies," 2019.

[16] L. P. Kaelbling and T. Lozano-Pérez, "Learning composable models of parameterized skills," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 886–893.

[17] M. J. Aein, E. E. Aksoy, and F. Wörgötter, "Library of actions: Implementing a generic robot execution framework by using manipulation action semantics," *The International Journal of Robotics Research*, 2019.

[18] A. Yamaguchi and C. G. Atkeson, "A representation for general pouring behavior," in *in the Workshop on SPAR in the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'15)*, 2015.

[19] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[20] A. Yamaguchi and C. G. Atkeson, "Neural networks and differential dynamic programming for reinforcement learning problems," in *the IEEE International Conference on Robotics and Automation (ICRA'16)*, 2016.