A Representation for General Pouring Behavior

Akihiko Yamaguchi[1] and Christopher G. Atkeson[1]

Abstract— **We introduce our work on pouring as an example of complicated tasks for robots. The architecture is a skill library with planning and learning methods. We briefly describe our representation for pouring, then discuss problems and future directions.**

I. Introduction

The research focus of the planning and learning community is shifting to more complicated tasks such as humans daily activity rather than planning trajectories. Examples are manipulation of non-rigid objects like folding towels [2], and baking cookies [3]. A challenge is to find a good way to model (represent) such behaviors. We think planning methods are necessary to adjust behavior models to current situations, and learning methods are necessary to adapt behavior models to unknown situations.

We are exploring these issues with a pouring task [1] in a learning from demonstration framework. The pouring task involves grasping as well as flow control, and considers a range of container shapes, types of materials, and initial positions. Thus designing a behavior covering these variations is very difficult. Our pouring behavior achieved a good generalization ability compared to those of other pouring approaches by carefully combining several technologies used in the robotics field, such as state machines, planning, and learning methods.

In this presentation, based on the pouring research [1] we discuss the essence of the representation, and future research directions. From human demonstrations, we found that there are many skills for pouring, especially flow control. Tipping is an example; shaking a container is useful to pour tomato sauce or ketchup, squeezing works well for shampoo bottles, and tapping a bottle of coffee powder is useful to pour it carefully. So our hypothesis is that humans have these skills in a library, and plan or learn how to use them. This hypothesis was empirically explored by making a pouring model with a skill library together with planning and learning methods, which achieved good generalization.

However, human developers did a lot of work in our pouring research. In order to automate the developers' work, we need to upgrade the planning and learning methods.

In the rest of this paper, we briefly introduce our pouring model, and then the issues are described. We discuss the future research directions and related research.

[1]A. Yamaguchi and C. G. Atkeson are with The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA 15213, United States info@akihikoy.net

II. Pouring Model

The video of a PR2 robot pouring is a good introduction of [1]: https://youtu.be/GjwfbOur3CQ

Phases of pouring include a preparation process (grasping a source container and moving the container to the receiving container's position), controlling the material flow, and a post pouring process (putting the source container at a final position). The most difficult thing to model is flow control, since it is a manipulation of complex materials, such as liquid and granular materials.

We consider a behavior model consisting of a library of skills, and planning and learning methods. Although our goal is automating the entire architecture, we model the skills manually. The skills are modeled with a well-known representation, finite state machines. Finite state machines represent both a behavioral structure (grasping, moving container, etc.) and feedback control (e.g. tipping until a target amount is achieved). A general flow controller is modeled as a hierarchical state machine using these skills. Each skill may have some parameters for selection and adjustment. Planning methods are introduced to obtain some situation specific parameters such as grasp parameters, pouring locations (where to put the mouth of the source container), and so on. Learning from practice methods are introduced for skill selection, for parameter adjustment, and for improving plan quality.

An important idea is the decomposition of pouring behavior. The entire pouring behavior is decomposed into several sub-skills and modeled by state machines, such as grasping and moving a container. Such a decomposition is possible because the sequence of sub skills is usually the same (moving an arm $\to$ grasping a container $\to$ moving the container $\to$ flow control $\to \dots$). This decomposition is also compatible with a memory-based approach to flow control, since the memorized skills are considered as decomposed skills.

Similarly, rather than planning many steps of the pouring behavior at once, we separate it into several small planning problems, such as grasping, pouring location, and path generation. The benefit is that we can reduce the computational cost of planning.

We consider two types of learning methods: (1) direct policy learning for selection (e.g. flow-control skill selection) and for adjustment (e.g. shaking angle), and (2) learning to improve planning where we update the evaluation function used in planning, rather than a policy.

Unifying these elements, we can achieve a pouring behavior model with wide generalization in terms of the variations mentioned before. Since some combinations of material types and container shapes require different flow control skills, state machines representing skills and skill selection learning are essential. Learning methods may also improve performance. Planning methods are necessary since container shape affects the grasp parameters, hand path, etc. The variation of initial poses is mainly handled by planning methods, but a state machine may increase the success rate in different initial poses; for example, the gripper position is improved by visual feedback control represented with a state machine. The difference of target amounts is handled by state machines for feedback control of the target amount observation. The learning method to improve planning improves plan quality.

We implemented the pouring behavior model on a PR2 robot and conducted experiments. We verified the generalization ability of the model in terms of materials, container shapes, contexts, container locations, and target amounts.

III. Discussion

Through the pouring research, we found that the essential components were a skill library, and planning and learning methods. Since our pouring behavior depends a lot on human engineering, our next challenge is to increase the autonomy. In this section we discuss the learning and planning methods in pouring by comparing with other research in order to consider the future directions.

We decomposed the entire pouring task into several planning problems, and introduced learning methods to improve performance. Since each planning problem is a maximization of an evaluation function with respect to action parameters, and a learning method updates the evaluation function from actual samples, we can consider that this method is a kind of temporal difference learning. Namely we are solving a reinforcement learning (RL) problem.

The issues of our method are that: (1) learning methods to update evaluation functions are slow, and (2) a lot of human engineering is necessary to design good evaluation functions. These issues would be also problematic when learning new skills. We consider that these issues are due to reinforcement learning with learning value functions.

There are two approaches to plan behaviors under totally or partially unknown dynamics: a model-free approach and a model-based approach. Another issue is whether value functions are learned. In a model-based approach, we learn a dynamics of the system, then we apply dynamic programming, such as differential dynamic programming (DDP; [4]). Examples are [5], [6]. In model-free approaches, we do not learn dynamics models, but learn policies. There are two types: learning value functions (typically $Q(x, a)$) then applying local planning ($\max_a Q(x, a)$), e.g. [7], [8]. The other is updating policy parameters directly, e.g. [9], [10].

Learning dynamics is typically easier than RL, since this is a supervised learning problem. However it needs many samples to cover the entire state-action space, and DDP often converges to poor local maxima. Model-free methods tend to acquire better policies. Direct policy learning methods converge faster than value function based methods. However their issue is less generalization ability compared to that of model-based methods. Recent research is trying to break the weak points. Kober *et al.* [11] introduced a meta-parameter learning architecture to their direct policy learning method [9] in order to increase the generalization ability. Levine *et al.* [12] proposed a practical approach where a trajectory optimization method for unknown dynamics is combined with a local linear model learned from samples. They trained neural network polices to increase generalization.

Although those approaches are based on direct policy learning, we think approaching from learning dynamics is also valuable due to its generalization ability. Actually in our pouring behavior, planning methods are playing a very important roll in supporting generalization. For a smooth connection of those planning methods and learning methods, a learning-dynamics approach makes a lot of sense. Through our pouring research we noticed that although the entire dynamics is complicated, it can be decomposed into a sequence of dynamical models, for example the relationship between the mouth position of the source container and the flow trajectory. We are currently taking this approach and our work will be published soon.

References

[1] 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.

[2] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei, and P. Abbeel, "Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding," in *the IEEE International Conference on Robotics and Automation (ICRA'10)*, 2010, pp. 2308–2315.

[3] M. Bollini, S. Tellex, T. Thompson, N. Roy, and D. Rus, "Interpreting and executing recipes with a cooking robot," in *the 13th International Symposium on Experimental Robotics*, 2013, pp. 481–495.

[4] D. Mayne, "A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems," *International Journal of Control*, vol. 3, no. 1, pp. 85–95, 1966.

[5] S. Schaal and C. Atkeson, "Robot juggling: implementation of memory-based learning," in *the IEEE International Conference on Robotics and Automation (ICRA'94)*, vol. 14, no. 1, 1994, pp. 57–71.

[6] J. Morimoto, G. Zeglin, and C. Atkeson, "Minimax differential dynamic programming: Application to a biped walking robot," in *the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'03)*, vol. 2, 2003, pp. 1927–1932.

[7] D. Ernst, P. Geurts, and L. Wehenkel, "Tree-based batch mode reinforcement learning," *Journal of Machine Learning Research*, vol. 6, pp. 503–556, 2005.

[8] A. Yamaguchi, J. Takamatsu, and T. Ogasawara, "DCOB: Action space for reinforcement learning of high dof robots," *Autonomous Robots*, vol. 34, no. 4, pp. 327–346, 2013.

[9] J. Kober and J. Peters, "Policy search for motor primitives in robotics," *Machine Learning*, vol. 84, no. 1-2, pp. 171–203, 2011.

[10] E. Theodorou, J. Buchli, and S. Schaal, "Reinforcement learning of motor skills in high dimensions: A path integral approach," in *the IEEE International Conference on Robotics and Automation (ICRA'10)*, may 2010, pp. 2397–2403.

[11] J. Kober, A. Wilhelm, E. Oztop, and J. Peters, "Reinforcement learning to adjust parametrized motor primitives to new situations," *Autonomous Robots*, vol. 33, pp. 361–379, 2012, 10.1007/s10514-012-9290-3.

[12] S. Levine, N. Wagener, and P. Abbeel, "Learning contact-rich manipulation skills with guided policy search," in *the IEEE International Conference on Robotics and Automation (ICRA'15)*, 2015.