On the Role of Structure in Manipulation Skill Learning

Eric Rosen* **Ben Abbatematteo*** **Skye Thompson** **Tuluhan Akbulut**

George Konidaris

Department of Computer Science
Brown University
United States

Abstract: We propose a new policy class, Composable Interaction Primitives (CIPs), specialized for learning sustained-contact manipulation skills like opening a drawer, pulling a lever, turning a wheel, or shifting gears. CIPs have two primary design goals: to minimize what must be learned by exploiting structure present in the world and the robot, and to support sequential composition by construction, so that learned skills can be used by a task-level planner. Using an ablation experiment in four simulated manipulation tasks, we show that the structure included in CIPs substantially improves the efficiency of motor skill learning. We then show that CIPs can be used for plan execution in a zero-shot fashion by sequencing learned skills.

Keywords: reinforcement learning; motor skills; manipulation

1 Introduction

The unique potential of robots lies in their ability to do physical work in the world—every process that currently requires a human to meaningfully interact with a physical object can only be automated by a robot. Despite this immense potential value, only a tiny fraction of the physical manipulation tasks that can be automated currently are [1]. There are multiple causes of this failure, but one of the most acute is that robots are currently not as flexible as humans in their ability to learn to interact with objects around them. A human factory worker can be trained to basic proficiency in an unfamiliar new task in a day; skillful and reliable execution of rote manual labor tasks rarely requires longer than a few weeks. Achieving the same level of flexibility, reliability, and skill in robots requires major advances in their learning capabilities, so that a robot can be trained to solve a new task, and subsequently improve its own performance, in a reasonable amount of time without the support of expert programmers.

There are two families of approaches to learning manipulation skills. The first combines end-to-end deep neural networks with reinforcement learning [2, 3] to learn "pixel to torque" controllers [4] that directly map sensor input to motor output. Such approaches couple reinforcement learning's promise of flexibility, generality, and autonomy, with the opportunity to exploit the power of deep networks. However, they face several challenges: dauntingly high sample complexity and difficulty incorporating other techniques from robotics such as forward and inverse kinematics, motion planning, wrench closure, and feedback control.

The second family aims to develop carefully designed and highly structured policy classes [5, 6, 7] to achieve sample-efficient learning, thereby trading design effort, flexibility, and generality for sample efficiency. Such approaches have been used to learn an impressive range of dynamic behaviors [3, 8]

*These authors contributed equally. Correspondence: {eric_rosen,abba}@brown.edu.

in a feasibly low number of interactions, but are best suited for targeting a restricted class of motor skills where there is structure to be exploited and sample efficiency is paramount.

We focus on one such class, *sustained contact manipulation skills*—where a robot must establish stable contact (in the form of a grasp) with an object in order to change its state, and sustain that contact throughout execution. Examples of such tasks include opening a drawer, pulling a lever, turning a doorknob, opening a door, turning a wheel, or shifting gears. We introduce a new policy class, *Composable Interaction Primitives* (or CIPs), that draws from the best of both motor skill learning approaches: it exploits the structure present in sustained contact tasks, resulting in a policy class that is structured, safe, and highly parameter- (and therefore data-) efficient; and then applies deep networks to the components where learning from high-dimensional input is unavoidable. Additionally, CIPs are sequentially composable by construction, so that learned skills can be sequenced to solve new tasks in an order determined at runtime by a task-level planner. Using an ablation experiment in four simulated manipulation tasks, we experimentally explore the role of structure in manipulation skill learning, and show that the components of CIPs substantially improve learning efficiency and safety. We then demonstrate the use of CIPs to efficiently learn, and subsequently sequence on-demand, two simulated sustained-contact manipulation skills.

2 Background

Motor skills are typically learned using reinforcement learning (RL) [2], where tasks are typically formalized as a Markov Decision Process $M = (S, A, R, T, \gamma)$, where S is a set of states which describe the current configuration of the task; A is a set of actions available to the robot; $R(s, a, s')$ is a reward function describing the reward obtained for executing action a in state s and transitioning to state s'; $T(s'|s, a)$ is the transition function, describing the probability that executing a in s leaving the robot in state s', and $\gamma \in (0, 1]$ is a discount factor expressing a preference for immediate over delayed rewards. The robot's goal is to learn a policy π mapping a state to the action it should execute in that state, such that it maximizes the discounted sum of expected future rewards (or *return*).

In many cases, the target motor skill is not the entirety of the robot's task, but should instead be used as an executable subroutine used as part of the solution. Such skills are typically modeled using the *options framework* [9], where an option o is defined by a tuple (I_o, π_o, β_o), where $I_o \subseteq S$ is the *initiation set*, the set of states from which the robot may choose to execute the option; $\beta_o : S \to [0, 1]$ is the *termination condition*, giving the probability that option execution ceases in state s; and π_o is the *option policy*. The robot can choose to execute o if the current state is inside I_o, whereupon execution proceeds according to π_o and halts at each encountered state according to β_o. When a motor skill is modeled this way, the skill's objective is typically specified by a reward function R_o, and the robot's task is to learn option policy π_o that maximizes return in the usual way. Modeling motor skills as options naturally supports reasoning about sequential compositionality—option o_2 can be executed after option o_1 if the state that o_1 leaves the robot in lies within o_2's initiation set; composable options therefore have small termination conditions that are highly likely to lie within many other options' (ideally large) initiation sets [10, 11, 12, 13].

Robotic state and action spaces are typically real-valued and high-dimensional. Consequently, the most popular family of approaches in this setting are policy search methods [14, 3], where the mapping from states to actions is directly represented as $\pi(a|s, \psi)$, controlled by parameter vector ψ. This form offers the opportunity to *structure the policy*: π need not be a simple learned mapping, but can instead include features such as safety constraints, stabilizing feedback, highly specified policy programs with just a few parameters, actuation limits, structured phases, and even motion planning. Additionally, in robotics there is the question of which state and action spaces the policy should use—the policy designer must choose where to place the policy on a spectrum ranging from directly mapping raw sensor input to motor torques, to wrapping the policy in highly processed input and output spaces (e.g., mapping object-level features extracted via computer vision to operation-space control). All of these choices are critical to effective learning; in section 3.1 we discuss implementation choices regarding the observation and action spaces in our experiments.

3 Composable Interaction Primitives

We identify four important properties present in sustained-contact motor skills. First, *skill execution can be decomposed into phases*: the robot first moves through free-space to reach a pre-grasp pose, then achieves a stable grasp, then manipulates the object, then releases its grasp, and finally controls its gripper back into free-space. Second, *most phases involve little or no per-task learning*: motion through free-space and to achieve or release a grasp can be computed using motion planning and feedback control, respectively; the choice of where to grasp the object is a supervised learning task that can be resolved (or at least bootstrapped) using a generic grasp detector. Only the sustained-contact controller itself need be largely learned on a per-task basis, though it could be bootstrapped using learning from demonstration [15] or kinematic motion planning [16]. Third, *the sustained-contact controller itself requires structure*: the controller must be a function of force- and tactile-feedback, learned using reinforcement learning; the goal of learning should be to reach a task-specific goal (e.g., opening a door, or switching a light on) while avoiding task-general failure modes (like losing contact with the object or becoming stuck); and during learning the policy should be able to explore while being position and torque constrained so as to never damage the robot or the object. Here, task-specific structures are components that are either learned or hand-specified for a specific object manipulation skill, whereas task-general structures are components that can be reused across different classes of object manipulation problems. Finally, *a natural means of composition is through free-space motion planning*: motor skills can be sequenced by simply motion planning from one skill's release point to another skill's grasp point.

We therefore propose *Composable Interaction Primitives* (CIPs), a new policy class structured by these insights and aimed at learning composable sustained-contact manipulation skills in tens, rather thousands, of real-world interactions. CIPs are structured as a tuple, where components subscripted by c are specific to the task, and the remainder are specific to the robot but generic across tasks:

$$C = (\pi_c, \sigma, \beta_c, I_c, h, t, B)\,,$$

where:

- $\pi_c : \phi \rightarrow \tau$ is a motor control policy that maps tactile sensor signals, proprioceptive data, and object state information ϕ to joint torques τ, parameterized by vector ψ.
- Policy π_c is constrained by σ, a safety envelope specific to the robot joint space but not to the task. Learning and execution are constrained to obey σ so that the agent does not damage the object it is interacting with or itself.
- $\beta_c : \phi \rightarrow \{0, 1\}$ is a task-specific success indicator that maps the robot's observations ϕ to a boolean indicating whether the interaction primitive has achieved its goal.
- B is a task-general classifier indicating interaction failure (e.g., that contact has been lost, the interaction has timed out, or execution cannot continue without a safety constraint being violated). Once initiated, π_c continues execution until either β_c indicates success or B indicates failure. The resulting signal informs a policy search algorithm to optimize π_c.
- $I_c : v,\ g \rightarrow [0, 1]$ is the grasp initiation set, a probabilistic classifier conditioned on visual data v that maps end-effector poses g to the probability with which executing π_c from grasp g terminates in β_c (success) as opposed to B (failure).
- h and t are the head and the tail, motion planners that control the robot through free space to achieve a grasp generated by I_c, and extract the robot from contact back into free space—or into the head of another skill—after the skill terminates, respectively. These serve to establish and break contact, and to sequence skills: the tail of one skill simply becomes the head of another.

For most tasks, we envision that all the skill components are given or designed except π_c and I_c, which leads to a problem of jointly learning a policy and affordance model for functional grasping. The CIP model structures the motor skill learning problem so that: only motor control involving contact with the object is learned, and free-space motion is generated using a planner; interaction

with an object is always safe; and motion planning is used for the remainder of motor control, especially to stitch motor skills together. At the same time, the components that must be learned offer natural opportunities for incorporating powerful deep network methods to learn rich sensorimotor policies. The result will be small, isolated pockets of motor skill learning connected by much longer trajectories generated by a motion planner. Note that the CIP model does not assume access to a simulator or dynamics model of the environment.

3.1 Instantiating CIPs

One benefit of the CIP framework is that its different components may be chosen to match the robot hardware it is being instantiated on. We now detail our specific choices of component instantiations used in the experiments (described in Section 4) as an illustrative example.

Motor control policy π_c. Our examples use sensor input from the touch sensors on the robot's grippers, the joint and Cartesian state of the robot, and object joint state, which are fed into a two-layered multi-layered perceptron (MLP) with 64 hidden nodes. For the action space, we chose to have the robot command the end-effector in Cartesian space while maintaining compliance with external forces, to promote ease-of-learning and safety during sustained contact. We therefore selected the Variable-Impedance Control in End-Effector Space [17] scheme as our action space. Motor policy π_c maps sensor readings ϕ to a desired delta end-effector position p_d and desired rotation R_d, as well as commanded stiffness terms k_p^p and k_p^R for position and rotation respectively. These terms are then use to directly map to joint torques τ via:

$$\tau = J_p[\Lambda_p[k_p^p(p_d - p) - k_d^p v]] + J_R[\Lambda_R[k_p^R(R_d \times R) - k_d^R \omega]], \tag{1}$$

where Λ_p and Λ_R are the position and orientation components of the inertia matrix $\Lambda \in \mathbb{R}^{6\times 6}$ in the end-effector frame, J_p and J_R are the position and orientation components of the end-effector Jacobian J, and $R_d \times R$ corresponds to subtraction in $SO(3)$. k_d^p and k_d^R are the damping values for position and rotation respectively, and are set with a damping ratio of 1 (critically damped). We also map directly to the gripper state $g \in \mathbb{R}$, with -1 being fully opened at 1 being fully closed. The resulting action space is therefore 13 dimensional.

Safety envelope σ. We limit the maximum value of stiffness parameters k_p^p and k_p^R, so that the robot remains compliant and does not generate high torque values when it contacts the object. In addition, the torques are clipped if they exceed the allowed range. In order to prevent joint limit violations, we use a two-fold strategy with two threshold parameters, σ_1 and σ_2 ($\sigma_1 > \sigma_2$), that check how close the robot joints are to its limits. If a joint position θ_i exceeds its threshold σ_1, we switch to a null-space controller [18] that attempts to move θ_i away from its limit without changing the end-effector pose. If the robot nonetheless exceeds σ_2 at joint index i (e.g. due to a high enough initial velocity to overcome the null-space controller), the controller generates a torque in the opposite direction for θ_i to attempt to return to a safe configuration.

Task-specific success indicator β_c. These were designed by hand for each task, and return true when the object's joints are above a threshold.

Task-general failure classifier B. In our case, B simply served as a joint limit safety check: if the robot is within 5 degrees of its joint limits, the classifier returns true and the learning episode ends early. Episodes are also terminated early if the agent loses contact with the object for sufficiently many timesteps.

Grasp initiation set I_c. In each case, the visual data v is represented as a point cloud of the scene, which is segmented to only include the part of the object that the robot should manipulate. An existing task-general grasp generator by ten Pas et al. [19] is used to sample a set of grasp poses G based on the normals calculated from the point cloud. Each grasp $g \in G$ is then checked for reachability and collision, and the stability of the grasp for sustained-contact manipulation is

evaluated by using a random noise policy to jiggle the gripper at the grasp pose g, and then the gripper is checked to still be in contact with the object. Grasps g which pass all these checks are added to a list of acceptable grasp poses that define the domain of I_c.

For sampling a grasp pose $g \in I_c$ for the head h during learning, we propose treating grasp pose sampling as a bandit problem that is solved with Upper Confidence Bounds (UCB) [20] where Q-values are task success rates. We therefore treat learning I_c and sampling grasp poses as an active-learning problem, and purposely choose UCB since it is particularly well-suited to balance exploration and exploitation.

Once a grasp pose g is sampled, we repeatedly solve inverse kinematics to obtain a joint configuration θ with high manipulability. A manipulability score is computed for a joint configuration θ as the product of two values: 1) the manipulability index introduced by Yoshikawa [21] that analyses the size of the manipulability ellipsoid: $w = \sqrt{det(JJ^T)}$ where J is the Jacobian for a particular joint configuration θ, and 2) a penalization term introduced by Tsai [22] based on the distance to the upper and lower joint limits for a particular joint configuration θ:

$$P(\theta) = 1 - exp(-k \prod_{j=1}^{n} \frac{(\theta_j - l_j^-)(l_j^+ - \theta_j)}{(l_j^+ - l_j^-)^2}), \tag{2}$$

where l_j^- and l_j^+ are the lower and upper joint limits for joint j. When these two metrics are multiplied together, they capture for a joint configuration θ how close the robot's end-effector is to a singularity and how close the robot's joints are to joint limits, respectively, which is termed the manipulability value.

Motion planners h and t. These were instantiated for each domain using the TRAC-IK inverse kinematics solver [23] and a basic grasping controller for establishing contact at the grasp pose sampled from the grasp initiation set I_c.

Algorithm 1 below describes the process of learning a CIP for one object manipulation task.

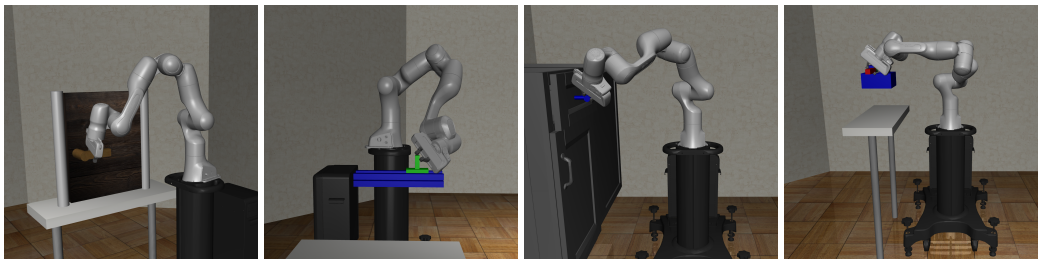
Algorithm 1 Learning a CIP

Require: CIP C = (policy π_c with parameters ψ, safety envelope σ, termination classifier β_c, failure classifier B, initiation set classifier I_c, head h, tail t), visual observation v, initial joint configuration θ_0

$v \leftarrow$ observe_scene()
$G \leftarrow$ get_grasp_candidates(v)
$I_c(g) \leftarrow 0 \quad \forall g \in G$ ▷ initialize initiation classifier
$D \leftarrow \{\}$ ▷ initialize policy replay buffer D
while not converged **do**:
  $g \leftarrow$ select_grasp(I_c, G) ▷ UCB for grasp selection
  $\Theta \leftarrow$ get_ik_solns(g)
  $\theta \leftarrow$ optimize_manipulability(Θ) ▷ choose suitable IK solution
  execute_motion_plan(θ, θ_0) ▷ head of CIP h
  execute_grasp(g)
  obs, actions, rewards, success $\leftarrow$ rollout(π_c, ψ, β_c, B)
  $D \leftarrow D \cup$ (obs, actions, rewards)
  $\psi \leftarrow$ improve_policy(D, π_c, ψ)
  $I_c(g) \leftarrow$ update_success_rate(g, success)
end while

4 Experiments

We evaluate the CIP framework in simulation using Robosuite [24]. We conducted experiments on four different articulated object tasks: opening a door, opening a drawer, sliding a knob, and lifting a lever.



(a) Open door task (b) Slide knob task (c) Open drawer task (d) Flip lever task

Figure 1: Simulation Task Environments

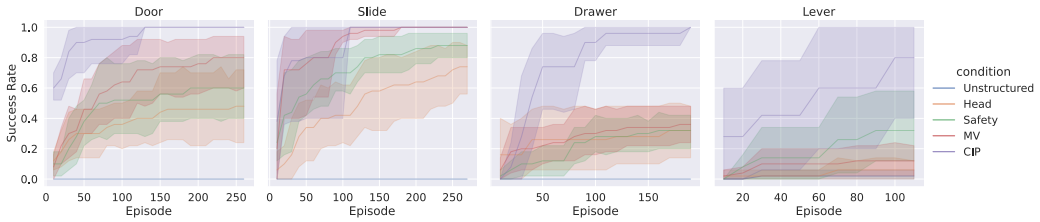
The state space for our policy is the state of the object, the position and velocities of the robot's joints, and tactile readings from the force sensors at the robot's grippers. We use TD3 [25] as our actor-critic method. To incorporate exploration during learning, we add Gaussian noise parameterized with 0 mean and 0.05 standard deviation to the policy. We use the Adam optimizer with a learning rate set to 0.0001. Our reward function is a dense reward based on the state of the object and how much its joint has progressed towards the goal, which leverages potential-based reward shaping [26] to ensure the optimal policy is not changed compared to the sparse reward setting based on success. Training episodes are ran for a maximum of 250 steps each, and trained for a total of $25,000$ training steps, which result in $100-1000$ training episodes depending on the task and experiment conditions. We evaluate the policies performance every 10 training episodes with 10 policy roll-outs.

We consider two evaluation metrics: 1) **Task Success Rate**, which measures how successful the policy is at manipulating the object, and 2) **Joint Limit Violation Rate**, which is a proxy measure for how safe the policy is. To analyze how each of the structures of CIP impact these metrics, we run 5 ablations that incrementally include structure described in Section 3.1:

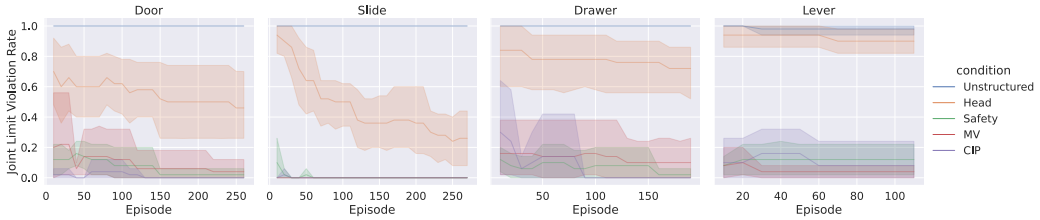
1. **Unstructured**: This setting is a baseline that incorporates none of the CIP structure. The robot begins in a home pose with no contact to the object, and must learn a complete policy for moving to the object and manipulating it.
2. **Head**: This setting is a baseline that incorporates the head h_c structure of the CIP. The agent has access to the domain of the grasp initiation set I_c, but samples grasp poses randomly and chooses random valid joint configurations.
3. **Safety**: This baseline extends the **Head** setting to additionally incorporate the safety envelope.
4. **Manipulability Value (MV)**: This baseline extends the **Safety** setting, and additionally incorporates the manipulability value into the sampling approach for I_c, which adds additional structure on top of the **Head** approach. After sampling a random grasp, we sample a set of inverse kinematics solutions and select the one with the highest manipulability value.
5. **CIP**: This setting incorporates all the structure of the CIP. Specifically, it extends the **MV** approach to additionally perform active learning with UCB over grasp poses.

5 Results

The results for all our experiments can be found in Figure 2, where we show the best-to-date performance for both metrics across all the tasks. Across all the tasks, the **Unstructured** baseline performs the worst and is unable to learn any meaningful policy, and also has many joint violation rates throughout learning. This is expected as exploration is extremely challenging in the absence of a strong reward signal for reaching the object and making contact. We also see that once the head of the CIP is incorporated (the baseline with minimal additional structure being **Head**), the agent is able to start achieving some amount of task success, but still encounters many joint state



(a) Success rates for simulated tasks (Door, Slide, Drawer, Lever).



(b) Joint limit violation rates for simulated tasks (Door, Slide, Drawer, Lever).

Figure 2: Task success rates and joint limit violation rates vs. the number of training episodes. The shaded region around the average is the 95% confidence interval over 5 seeds.

violations throughout the learning process. The **Safety** baseline is able to achieve a task success rate on par with **Head**, but is able to significantly reduce the number of joint state violation rates during learning. The **MV** baseline has improved task success over the **Head** baseline, which demonstrates the usefulness of incorporating the manipulability value when selecting joint configurations for sustained-contact manipulation tasks, but still has trouble learning an effective policy for the Lever and Drawer task in a small number of training episodes. Once the full structure of the CIP is incorporated (**CIP**), the agent is able to rapidly learn a policy with a high success rate (at least an average of 80%) within 100 training episodes. These results demonstrate that each structural component of the CIP is useful for ensuring that the agent is able to safely and efficiently learn across a diverse set of sustained-contact manipulation tasks.

5.1 CIPs for multi-step plan execution

One of the advantages of the CIP structure is it enables zero-shot composition by construction. The motion planning performed via the head h and tail t enable a robot to learn sustained-contact manipulation skills in isolation using a model-free learning algorithm, and then sequentially execute the skills when multiple objects are in the scene with no additional learning necessary. This is particularly useful when an agent performs high-level planning with the skill repertoire, since a plan involves composing actions together in sequence. We show a demonstration of this behavior in Figure 3, where the agent has been trained to separately slide a knob and open a door, and is now placed in a scene that has both. When tasked with a plan that involves first sliding the knob and then opening the door, the robot is able to use the head h to first motion plan to grasp the slide knob, execute the slide knob policy, and then motion plan to the grasp the door handle to finally execute the open door policy before returning to free space using the tail t.

6 Related Work

A great deal of recent work has examined the setting where a robot learns to map its sensor input directly to motor torques via deep reinforcement learning [4, 27]. These methods offer flexibility, generality, and autonomy by exploiting recent advances in learning deep networks, but face challenges due to high sample complexity. An alternative approach is to carefully design and structure a policy class to guarantee desirable properties (e.g., stability, joint and torque limits, and safety

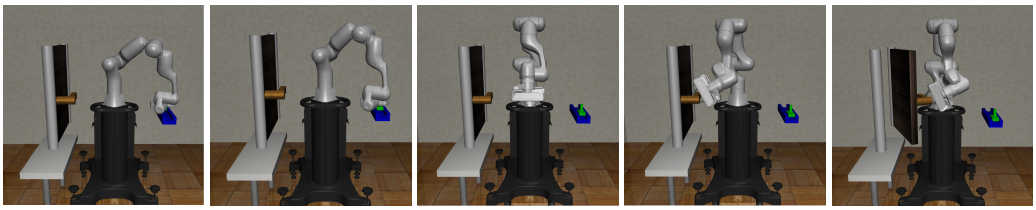


Figure 3: Two CIPs executed in succession.

constraints) while exploiting properties of a broad class of target tasks to support sample-efficient learning. The most historically important such policy class is *Dynamic Movement Primitives*, or DMPs [28, 5], which have been used to learn an impressive range of dynamic behaviors [3, 8] in tens or low hundreds of interactions, though they must typically be bootstrapped by an expert demonstration trajectory [15]. The key assumption underlying DMPs is that dynamic motions can be represented largely as a trajectory shape—represented separately for each joint, as a linear combination of learned weights with basis functions over time—coupled with a second-order dynamical system that safely and stably controls the robot towards the shape trajectory. Although DMPs have seen wide usage [1], they have largely not been integrated with high-dimensional, multi-modal data [29].

Other important policy classes overcome the standard shortcomings of DMPs. For example, Probabilistic Movement Primitives [30] learn a distribution over basis functions, so that variability across demonstrations and different DoFs are captured. Conditional Movement Primitives [31] encode demonstrations as a whole with high dimensional task parameters by a deep network, and in RL setting, safety and stability are achieved by encoding exploration trajectories into the same latent space [32] or coupling it with external controllers [33]. However, these approaches rely on demonstrations. Riemannian Motion Policies (RMPs) [6, 7] support combining multiple (second-order dynamical system) controllers defined in potentially different task spaces in a natural way to obtain a single controller that combines their effects in the joint space. RMPs provide a principled approach for integrating multiple concurrent controllers but have yet to see wide success in contact-rich manipulation tasks [34, 35].

A related problem formulation in the robotics literature is task and motion planning (TAMP) [36, 37] which relies on symbolic and geometric descriptions of the underlying environment to find abstract plans and corresponding feasible trajectories. Our goal is to explicitly structure a policy class so that it naturally integrates with this formalism.

7 Conclusion

We propose a new policy class for sustained-contact manipulation skills: Composable Interaction Primitives (CIPs). CIPs are designed to exploit readily-accessible structure in the world and robot to enable sample-efficient and safe policy learning, and be easily leveraged by high-level planners due to their sequential composability via motion planning. Future work will investigate efficient methods to learn effect models to autonomously construct a symbolic vocabulary to support integrating CIPs with a high-level task planner. In addition, because CIPs leverage motion planning for the head h and tail t, future work will investigate bootstrapping methods that leverage kinematic models of objects and the robot to provide demonstrations without human intervention [16]. Lastly, future work will validate CIPs on robot hardware, and include a ROS package to easily integrate CIPs into a robot manipulation pipelines.

Acknowledgments

The authors would like to thank Matthew Corsaro for sharing code and insightful discussions.

This research was supported by NSF CAREER Award 1844960 to Konidaris, an Amazon Faculty Research Award to Konidaris, NSF Fellowship Award to Thompson, AFOSR DURIP FA9550-21-1-0308, and the ONR under contracts N00014-22-1-2592, N00014-21-1-2584.

Disclosure: George Konidaris is the Chief Roboticist of Realtime Robotics, a robotics company that produces a specialized motion planning processor.

References

[1] O. Kroemer, S. Niekum, and G. Konidaris. A review of robot learning for manipulation: Challenges, representations, and algorithms. *Journal of Machine Learning Research*, 22(30): 1–82, 2021.

[2] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

[3] J. Kober and J. Peters. Policy search for motor primitives in robotics. *Machine Learning*, 84 (1-2):171–203, 2010.

[4] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[5] A. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1547–1554, 2002.

[6] C.-A. Cheng, M. Mukadam, J. Issac, S. Birchfield, D. Fox, B. Boots, and N. Ratliff. Rmpflow: A geometric framework for generation of multitask motion policies. *IEEE Transactions on Automation Science and Engineering*, 18(3):968–987, 2021.

[7] N. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox. Riemannian motion policies. *arXiv preprint arXiv:1801.02854*, 2018.

[8] K. Mülling, J. Kober, O. Kroemer, and J. Peters. Learning to select and generalize striking movements in robot table tennis. *The International Journal of Robotics Research*, 32(3):263–279, 2013.

[9] R. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, 1999.

[10] T. Lozano-Perez, M. Mason, and R. Taylor. Automatic synthesis of fine-motion strategies for robots. *International Journal of Robotics Research*, 3(1):3–24, 1984.

[11] R. Burridge, A. Rizzi, and D. Koditschek. Sequential composition of dynamically dextrous robot behaviors. *International Journal of Robotics Research*, 18(6):534–555, 1999.

[12] R. Tedrake. LQR-Trees: Feedback motion planning on sparse randomized trees. In *Robotics: Science and Systems V*, pages 18–24, 2009.

[13] G. Konidaris and A. Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. In *Advances in Neural Information Processing Systems 22*, pages 1015–1023, 2009.

[14] M. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1–2):1–142, 2013.

[15] B. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57:469–483, 2009.

[16] B. Abbatematteo, E. Rosen, S. Tellex, and G. Konidaris. Bootstrapping motor skill learning with motion planning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4926–4933. IEEE, 2021.

[17] R. Martín-Martín, M. A. Lee, R. Gardner, S. Savarese, J. Bohg, and A. Garg. Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1010–1017. IEEE, 2019.

[18] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE J. Robotics Autom.*, 3:43–53, 1987.

[19] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt. Grasp pose detection in point clouds. *The International Journal of Robotics Research*, 36(13-14):1455–1473, 2017.

[20] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[21] T. Yoshikawa. Manipulability of robotic mechanisms. *The international journal of Robotics Research*, 4(2):3–9, 1985.

[22] M.-J. Tsai. *WORKSPACE GEOMETRIC CHARACTERIZATION AND MANIPULABILITY OF INDUSTRIAL ROBOTS (KINEMATICS)*. The Ohio State University, 1986.

[23] P. Beeson and B. Ames. Trac-ik: An open-source library for improved solving of generic inverse kinematics. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 928–935. IEEE, 2015.

[24] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

[25] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[26] A. Y. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

[27] A. Khazatsky, A. Nair, D. Jing, and S. Levine. What can i do here? learning new skills by imagining visual affordances. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14291–14297. IEEE, 2021.

[28] S. Schaal. Dynamic movement primitives-a framework for motor control in humans and humanoid robotics. In *Adaptive motion of animals and machines*, pages 261–280. Springer, 2006.

[29] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel. Dynamic movement primitives in robotics: A tutorial survey. *ArXiv*, abs/2102.03861, 2021.

[30] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann. Probabilistic movement primitives. In C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[31] M. Y. Seker, M. Imre, J. Piater, and E. Ugur. Conditional neural movement primitives. In *Proceedings of Robotics: Science and Systems*, FreiburgimBreisgau, Germany, June 2019.

[32] M. Akbulut, E. Oztop, M. Y. Seker, H. X, A. Tekden, and E. Ugur. Acnmp: Skill transfer and task extrapolation through learning from demonstration and reinforcement learning via representation sharing. In J. Kober, F. Ramos, and C. Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 1896–1907. PMLR, 16–18 Nov 2021.

[33] M. T. Akbulut, U. Bozdogan, A. Tekden, and E. Ugur. Reward conditioned neural movement primitives for population-based variational policy optimization. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10808–10814, 2021.

[34] S. Shaw, B. Abbatematteo, and G. Konidaris. Rmps for safe impedance control in contact-rich manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2707–2713, 2022.

[35] M. Xie, K. Van Wyk, A. Handa, S. Tyree, D. Fox, H. Ravichandar, and N. D. Ratliff. Neural geometric fabrics: Efficiently learning high-dimensional policies from demonstration. In *6th Annual Conference on Robot Learning*.

[36] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4:265–293, 2021.

[37] Z. Wang, C. R. Garrett, L. P. Kaelbling, and T. Lozano-Pérez. Learning compositional models of robot skills for task and motion planning. *The International Journal of Robotics Research*, 40(6-7):866–894, 2021.