Value Guided Exploration with Sub-optimal Controllers for Learning Dexterous Manipulation

Gagan Khandate*[†], Cameron Paul Mehlman*[‡], Xingsheng Wei*[‡], Matei Ciocarlie[‡]

***Abstract*— Recently, reinforcement learning has allowed dexterous manipulation skills with increasing complexity. Nonetheless, learning these skills in simulation still exhibits poor sample-efficiency which stems from the fact these skills are learned from scratch without the benefit of any domain expertise. In this work, we aim to improve the sample-efficiency of learning dexterous in-hand manipulation skills using sub-optimal controllers available via domain knowledge. Our framework optimally queries the sub-optimal controllers and guides exploration toward state-space relevant to the task thereby demonstrating improved sample complexity. We show that our framework allows learning from highly sub-optimal controllers and we are the first to demonstrate learning hard-to-explore finger-gaiting in-hand manipulation skills without the use of an exploratory reset distribution.**

***Index Terms*— Dexterous manipulation, Reinforcement learning, In-hand manipulation, Sub-optimal controllers**

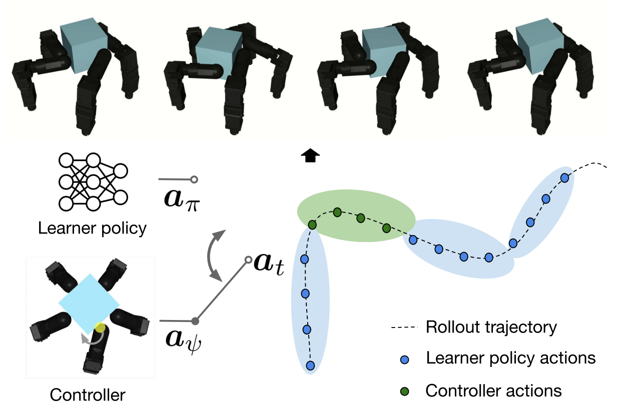


Fig. 1: Our method of interweaving sub-optimal controller (ex. contact switching) and policy enables learning effective finger-gaiting skills as shown.

I. Introduction

Reinforcement learning has led to significant improvements in our ability to achieve a range of dexterous manipulation skills. Prior works have demonstrated in-hand manipulation skills such as object reorientation to a desired pose [1, 2] or continuous re-orientation about a given axis[3, 4]. However, learning these dexterous skills can still require up to billions of simulation steps, particularly for in-hand manipulation.

Methods using domain expertise to improve the sample efficiency for learning in-hand manipulation skills achieve this mainly through engineered reset distributions and model-based controllers. The methods using reset distributions use a set of stable grasps to enable exploration which has been particularly effective for finger-gaiting in-hand manipulation tasks[3]. However, the reset distributions are object and task specific and may require heavy engineering.

Alternatively, other methods use model-based controllers that leverage the model to ensure the stability of the grasp and combine them with reinforcement learning. Predominantly, this involves decomposing the in-hand manipulation task into appropriate sub-skills (i.e in-grasp manipulation, finger-pivoting, finger-gaiting, etc,.) and designing model-based controllers for each to be used as low-level controllers in hierarchical reinforcement learning [5].

However, this approach of using model-based controllers as low-level controllers has a number of limitations. First, from a training perspective, computing actions from these controllers can be expensive, particularly if it involves optimization. Hence, querying these controllers at every step can outweigh the benefits in terms of the net computational cost. Next, restricting the policy to only executing one of the predefined sub-skills can lead to learning a sub-optimal policy. Finally, a hierarchical learning framework implicitly requires that low-level model-based controllers must be deployable on hardware. Thus, these model-based controllers must be designed such that only sensor feedback available on the hardware is used as input, and controllers must be robust to uncertainties present on the hardware.

In this work, we seek to leverage model-based controllers to improve sample efficiency while retaining the benefits of learning an end-to-end policy. The key idea that we propose is to interleave learner policies and exploratory controllers during episode rollout during the initial phase and use only learner policy during the later phase of training. Consequently, we use these controllers only during training and not at run-time. We show that our method guides exploration towards relevant regions of the state-space. We also use the current action-value estimates provided by the critic to guide action selection which further improves exploration.

Critically, we use controllers that are simple to design and inexpensive to query actions from. Our controllers do not involve multi-step horizon motion planning or other sources of complexity and computational cost. This makes them inexpensive to compute - favorably reducing the cost of querying the expert. However, inexpensive controllers will often be sub-optimal in their behavior. Still, our method enables learning despite using such significantly sub-optimal

* denotes joint first authorship
[†]Dept. of Computer Science, [‡]Dept. of Mechanical Engineering
Columbia University, New York, NY 10027, USA
Corresponding email: `gagank@cs.columbia.edu`

controllers. Overall, the main contributions of our work include:

- We demonstrate that dexterous in-hand manipulation tasks such as finger-gaiting can be learned by leveraging a set of with simple and significantly sub-optimal controllers.
- In contrast to previous work combining model-based controllers and learning for manipulation, we use the proposed controllers for training an end-to-end policy that does not require the experts for deployment. This approach alleviates many constraints on the controllers we use, in terms of both performance and sensory input.
- We also show that sub-optimal controllers can enable effective exploration of the complex state space of our problem, without exploratory reset distributions used in previous studies with similar goals.

II. Related Work

Model-free RL has led to significant progress in dexterous manipulation skills. As mentioned, this typically involves demonstrating reorientation to a desired object pose [1]. However, this required hundreds of hours of training. Recently, it was shown that GPU physics could be used to accelerate learning these skills [6, 7]. Chen et al. [2, 8] demonstrated in-hand re-orientation for a wide range of objects under palm-up and palm-down orientations of the hand with extrinsic sensing providing dense object feedback. While Khandate et al. [3] showed object-agnostic dexterous finger-gaiting and finger-pivoting skills using precision fingertip grasps and sensing intrinsic to the hand only. Qi et al. [4] used rapid motor adaptation to achieve effective sim-to-real transfer of in-hand manipulation skills for small cylindrical and cube-like objects. Nonetheless, learning these skills still remains extremely sample inefficient.

Dexterous in-hand manipulation has long been of interest and numerous methods have been proposed ranging from simple low-level feedback primitive controllers to trajectory optimization for the complete task. Early model-based work on finger-gaiting [9–11] and finger-pivoting [12] propose controllers using simplified models. More recently, Fan et al. [13] and Sundaralingam et al. [14] use model-based online trajectory optimization and demonstrate finger-gaiting in simulation. Chen et al. [15] combine trajectory optimization and tree search to follow a desired object pose trajectory.

Such model-based controllers can be used to improve sample complexity by using them as low-level policies in hierarchical reinforcement learning. Li et al. [5] derive model-based controllers for reposing, sliding, flipping, and learn a hierarchical policy for 2D in-hand manipulation. Veiga et al. [16, 17] used a low-level controller to maintain grasp stability based on tactile feedback and demonstrated in-grasp manipulation.

Imitation learning (IL) methods provide an alternative to using model-based controllers. DAgger [18] and AggreVaTe [19] are popular frameworks for imitation with a single expert. Typically, these methods require the expert to be near-optimal. MAMBA [20] provides an approach for IL with multiple sub-optimal experts. However, MAMBA has only been demonstrated with partially trained policies, which does not capture the complexities of a realistic sub-optimal expert.

Several methods also propose learning from a dataset of demonstrations through augmentation [21, 22] or offline RL [23]. These demonstrations are collected either by a human expert or a sufficiently exploratory model-based controller. However, human demonstrations are hard to collect for tasks with complex dynamics such as ours, and model-based controllers are often sub-optimal and cannot be used for any extended period of time.

Several methods have used off-policy RL with sub-optimal experts as part of the behavioral policy. Kurenkov et al. [24] proposed AC-Teach framework to sample from multiple experts based on the critic's estimate of the action-value. Jeong et al. [25] derived REQ to better utilize the off-policy transitions sampled from a sub-optimal expert. However, these methods considered relatively simple robot control tasks for evaluation. Our method is most similar to AC-Teach with the following differences. We use softmax procedure instead of greedy procedure to select between action proposals. In addition, we use a behavior cloning loss which we found to improve performance.

III. Method

We demonstrate our method as the task of finger-gaiting in-hand manipulation with only fingertip grasps and no support surfaces - a necessary skill for continuous object re-orientation in arbitrary orientations of the hand. Learning this skill purely with reinforcement learning is challenging as random exploration from a fixed state does not sufficiently explore the desired state-space. Due to the complexity and contact-rich nature of the task, it also challenging to use model-based methods to design an expert for finger-gaiting.

Finger-gaiting is a complex skill consisting of a range of behaviors. We can split this skill into two sub-skills: in-grasp manipulation, and contact switching. In-grasp manipulation entails maintaining the object in a stable grasp and reorienting it without breaking or making contact. Therefore, due to limits on the range of motion of the hand, the maximum object rotation is also limited. Contact switching involves breaking and making contact to re-grasp the object in order to further object reorientation via in-grasp manipulation.

Thus, instead of designing a single controller for finger-gaiting we can decompose it into two controllers, an in-grasp manipulation controller and a contact switching controller. Still, ensuring optimality and robustness is challenging for these controllers. Nevertheless, sub-optimal model-based controllers obtained from simplified models or heuristics can still generate reasonable actions and are able to generate exploratory sequences of state transitions.

Thus, in this work, we will (i) design simple controllers for in-grasp manipulation and contact switching, and (ii) learn an effective end-to-end policy for finger-gaiting using these sub-optimal controllers by following their actions.

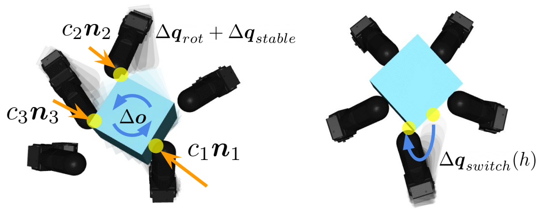

$c_2\boldsymbol{n}_2$
$\Delta\boldsymbol{q}_{rot} + \Delta\boldsymbol{q}_{stable}$
$c_3\boldsymbol{n}_3$
$\Delta\boldsymbol{o}$
$c_1\boldsymbol{n}_1$
$\Delta\boldsymbol{q}_{switch}(h)$

Fig. 2: Sub-skill controllers used for learning finger-gaiting: (left) In-grasp manipulation controller (right) Contact switching controller.

A. Problem definition and sub-optimal controllers

We consider the task of finger-gaiting in-hand manipulation by learning continuous re-orientation about z-axis in the hand-centric frame [3]. We aim to learn such a policy by rewarding angular velocity of the object along the z-axis. Let us consider a fully dexterous hand with m fingers and d dofs. Assuming position control, our policy outputs joint target updates from observations $\boldsymbol{s}$:

$$\boldsymbol{s} = [\boldsymbol{q}, \boldsymbol{q}', \boldsymbol{p}_1, .., \boldsymbol{p}_m, \boldsymbol{n}_1, .., \boldsymbol{n}_m, c_1, .., c_m] \tag{1}$$

where $\boldsymbol{q} \in \mathcal{R}^d$, $\boldsymbol{q}' \in \mathcal{R}^d$ are current and target joint positions, $\boldsymbol{p}_i \in \mathcal{R}^3$, $\boldsymbol{n}_i \in \mathcal{R}^3$ are contact positions, contact normal in the global co-ordinate frame for contacts on all fingertips and $c_i \in \mathcal{R}$ is the magnitude of contact forces. We aim to train this policy by rewarding the angular velocity of the object about the z-axis when making at least 3 contacts as in [3].

As previously discussed, finger-gaiting skill can be broken down into two sub-skills - in-grasp manipulation and contact switching. We will now design controllers for each.

1) In-grasp manipulation controller (IGM): The in-grasp manipulation sub-skill maintains a stable grasp on the object and, simultaneously, reorients the object about the desired axis. We construct this controller by superposition of two controllers - one responsible for stability and another responsible for object re-orientation.

To maintain grasp stability, we assume at least three contacts and desire the net object wrench be close to zero. We use the kinematic model of the hand to derive our stability controller as follows. Let S be the set of contact points and $\boldsymbol{n}_1, \ldots, \boldsymbol{n}_k$ ($k \geq 3$) be the contact normals. $\boldsymbol{G}_S$ and $\boldsymbol{J}_S$ are the grasp map matrix and the Jacobian for the set of contacts. Our stability controller keeps the object in equilibrium by minimizing the net wrench applied to the object. This can be achieved by solving the following Quadratic Program (QP):

unknowns: normal force magnitudes $c_i,\ i = 1 \ldots k$

minimize $\|\boldsymbol{w}\|$ subject to:

$$\boldsymbol{w} = \boldsymbol{G}_S^T \left[c_1\boldsymbol{n_1} \ldots c_k\boldsymbol{n_k}\right]^T \tag{2}$$

$$c_i \geq 0 \ \forall i \tag{3}$$

$$\exists j \text{ such that } c_j = 1 \text{ (ensure non-zero solution)} \tag{4}$$

Let c_i^*'s be the desired contact forces obtained as solution of above QP. We can compute the actions Δq_{stable} needed for stability (expressed as joint position setpoint changes) by solving the following system:

$$\Delta\boldsymbol{p}_i = \alpha(c_i^* - c_i)\boldsymbol{n}_i \tag{5}$$

$$\boldsymbol{J}_S \Delta q_{stable} = [\Delta\boldsymbol{p}_1, \ldots, \Delta\boldsymbol{p}_k]^T \tag{6}$$

where α is the stability controller gain.

Separately, let $\Delta\boldsymbol{o} \in \mathcal{R}^6$ denote the desired change in object pose. We can compute the action $\Delta\boldsymbol{q}_{rot}$ required to achieve this object re-orientation by solving

$$\boldsymbol{J}_S \Delta\boldsymbol{q}_{rot} = \boldsymbol{G}_S^T \Delta\boldsymbol{o} \tag{7}$$

Finally, $\Delta\boldsymbol{q}_{stable}$ and $\Delta\boldsymbol{q}_{rot}$ are combined together simply via superposition. The complete set of steps involved in in-grasp manipulation controller is outlined in Alg 1. When used by itself in a rollout the in-grasp manipulation controller achieves a cumulative reward of only 6.9 before reaching joint limits. This value is far lower than the cumulative reward achievable by a successful finger-gaiting policy within the same duration.

Algorithm 1 In-grasp manipulation (IGM) controller

```
1: Get Jacobian, $\boldsymbol{J}_S$ and grasp map matrix $\boldsymbol{G}_S$
2: Compute optimal contact force $c_i^*$'s for stability from QP
3: Solve for $\Delta\boldsymbol{q}_{stable}$ using Eq 6
4: Solve for $\Delta\boldsymbol{q}_{rot}$ from sampled $\Delta\boldsymbol{o}$ using Eq 7
5: $\psi_{IGM}(\boldsymbol{s}) = \Delta\boldsymbol{q}_{stable} + \Delta\boldsymbol{q}_{rot}$
```

2) Contact switching controller (CS): Our insight here is to simply use a fixed trajectory as a controller,

$$\psi_{CS}(\boldsymbol{s}) = \Delta\boldsymbol{q}_{switch}(h) \tag{8}$$

where $h = 1, \ldots, H$ and H is the length of the controller trajectory.

This trajectory is a simple primitive for breaking and making one contact as described next. We design a trajectory that only breaks and makes contact with one selected finger at a given time even though contact switching as a skill can involve multiple contact switches. We select a finger in contact at random and follow a hand-designed trajectory as shown in Fig 2 for the selected finger to break contact and remake contact with the object at a different location. This trajectory is easily achieved as shown in Fig 3b. As one can expect, the switching controller by itself, virtually collects, no reward.

3) Finger-gaiting controller (FG): Finally, we also construct a third sub-skill controller. Combining the in-grasp manipulation and contact switching controller we construct a finger-gaiting controller. While a randomly selected fingertip breaks and makes contact, this controller also reorients the object with the fingertips in contact. Although this controller comes closer than the rest, it does not achieve the whole finger-gaiting skill we are looking to learn. Continuous object re-orientation is not possible with this controller as the fingers selected to switch contacts are still selected at random without any coordination.

We construct this controller again by superposition. We superpose in-grasp manipulation and contact switching controller.

$$\psi_{FG}(\boldsymbol{s}) = \psi_{IGM}(\boldsymbol{s}) + \psi_{CS}(\boldsymbol{s}) \tag{9}$$

Note that these controllers are orthogonal to each other by construction. The in-grasp manipulation controller commands zero actions for joints of the fingers without contact, whereas, the contact switching controller does the same for joints of the fingers in contact i.e $\psi_{IGM}(\boldsymbol{s})^T \psi_{CS}(\boldsymbol{s}) = \mathbf{0}$.

The average reward collected by finger-gaiting controller is 1.4 which is even smaller than that of the in-grasp manipulation controller.

B. *Learning from sub-optimal controllers*

We use off-policy actor-critic reinforcement learning, where the policy used for episode rollouts (i.e the behavior policy) can be significantly different from the end-to-end policy being learned (i.e the learner policy). We use this dichotomy to enable exploration by constructing a behavior policy that is a composition of the learner policy and the sub-skill controllers. Importantly, this approach also sidesteps the need for the controller during deployment - once trained, the learner policy by itself is sufficient for deployment.

We construct the behavior policy to achieve sufficient exploration as follows. Our behavior policy periodically selects between the controllers and learner policy. We bias this selection towards higher value controllers using the critic's action-value estimate of the actions sampled from these controllers. This heuristic allows the probability of selecting a controller action to vary across the state-space. We show that this heuristic of using action-value estimates to select between available controllers and learner policy improves exploration. Optionally, we use value-weighted behavior cloning in addition to the nominal RL objective of the chosen off-policy RL method to update the policy. The method is detailed next.

Let π_θ represent the learner policy and ψ represent the controller. At every step of the rollout, we query actions from both the learner policy and controllers. Let these actions be $\boldsymbol{a}_\pi$, $\boldsymbol{a}_\psi$. The probability p_ψ of selecting the action from a controller, as well as the probability p_π of selecting the action prescribed by the learner policy, are computed as:

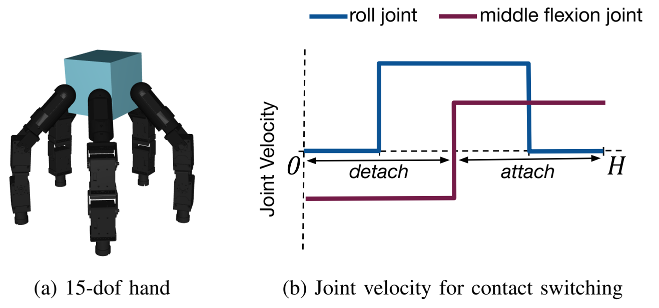
$$p_\psi = \frac{\exp(Q(\boldsymbol{s}, \boldsymbol{a}_\psi))}{\exp(Q(\boldsymbol{s}, \boldsymbol{a}_\pi)) + \exp(Q(\boldsymbol{s}, \boldsymbol{a}_\psi))} \tag{10}$$

$$p_\pi = 1 - p_\psi \tag{11}$$

As $Q(\boldsymbol{a}_\pi, \boldsymbol{s})$ increases, p_ψ decreases, i.e., as the learner policy improves, the probability of following the controller decays exponentially. In practice, we also impose a hard stop, i.e., we stop querying the controllers altogether after a few million steps of training and only use the learner policy for exploration. This point typically corresponds to the convergence of the critic.

Algorithm 2 Learning from sub-optimal controllers

```
Require: Controller $\psi$
 1: Initialize policy $\pi_\theta$, $Q_\phi$ and off-policy RL algorithm with
    replay buffer $\mathcal{D}$
 2: for each iteration do
 3:     for $t = 0, \ldots, T-1$ do
 4:         for every $H$ steps do
 5:             Compute $p_\pi$, $p_\psi$ from $\boldsymbol{a}_\pi, \boldsymbol{a}_\psi$ (Eq 10 & 11)
 6:             Select $\mu$: $\mu \sim \{\pi, \psi\}$ given $p_\pi$, $p_\psi$
 7:         Follow selected $\mu$ for next H steps: $\boldsymbol{a}_t \sim \mu(\boldsymbol{s}_t)$
 8:         $\mathcal{D} \leftarrow \mathcal{D} \cup \{\boldsymbol{s}_t, \boldsymbol{a}_t, r(\boldsymbol{s}_t, \boldsymbol{a}_t), \boldsymbol{s}_{t+1}\}$
 9:     for each gradient step do
10:         $\theta \leftarrow \theta - \nabla_\theta \mathcal{L}_{RL}(\theta)$
11:         $\theta \leftarrow \theta - \nabla_\theta \mathcal{L}_{BC}(\theta)$ (Optional)
```



(a) 15-dof hand (b) Joint velocity for contact switching

Fig. 3: (a) The 5-fingered, 15-dof dexterous hand grasping the cube we use in our experiments (b) Joint velocity profile of the joints of the finger making and breaking contact. It involves detach and attach phases. Note that the distal flexion joint is held fixed.

Our method is summarized in Alg 2. The first difference w.r.t the standard off-policy learning framework is steps 4-7 when the behavior policy is different from the learning policy and is composed as described previously. The second difference, in step 11, entails updating the learner policy parameters using a value weighted behavior cloning loss $\mathcal{L}_{BC}$ (see Eq 12), in addition to the nominal actor loss $\mathcal{L}_{RL}$ as determined by the off-policy algorithm of choice. We follow a linear decay schedule for β.

$$\mathcal{L}_{BC}(\theta) = \sum_{(\boldsymbol{s},\boldsymbol{a}),\ \boldsymbol{a} \neq \boldsymbol{a}_\pi} \beta \log \pi_\theta(\boldsymbol{a}|\boldsymbol{s}) Q(\boldsymbol{s}, \boldsymbol{a}) \tag{12}$$

IV. Experiments and Results

In this section, we discuss the experimental results of our method for learning finger-gaiting in-hand manipulation with sub-optimal controllers as described in the previous section.

A. *Experimental setup*

In all experiments, we use a fully actuated dexterous hand in a simulated environment as shown in Fig 3a. The 15-dof hand consists of five fingers with one roll joint and two flexion joints per finger. The object is a cube of dimension similar to the distance between adjacent fingers, unless otherwise specified.

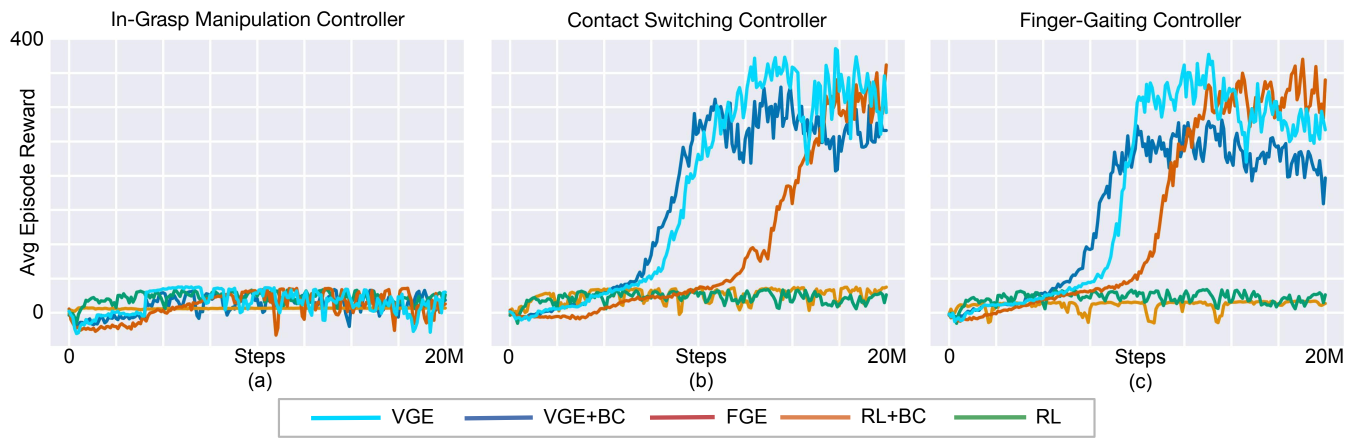


Fig. 4: Training curves for using (a) In-grasp manipulation controller (b) Contact switching controller (c) Finger-gaiting controller with conditions listed in Sec IV-A. Our method VGE, VGE + BC learn finger-gaiting with similar performance while the other RL, RL + BC baselines don't.

TABLE I: Training robustness (percentage of seeds that successfully learn), calculated using 5-10 seeds for each method.

	CS Controller	FG Controller
VGE	27%	**100%**
VGE + BC	36%	**100%**
FS	67%	**73%**

For the in-grasp manipulation controller, we use $\Delta \boldsymbol{o} = [0, 0, 0, 0, 0, 0.0225]^T$. For the contact switching controller, the velocity profile of the trajectory followed by the controller is shown in Fig 3b where the roll-joint and middle-flexion-joint velocities are sampled in the range of $[0.1, 0.4]$ rad/s.

For training, we use SAC [26] and TD3 [27] with similar results. Nonetheless, we show results only for SAC. Note that with SAC we set the exploration co-efficient α to 0 because using any non-zero value results in failure to learn including all the baselines discussed in the following section. We stop querying the controllers after 4M steps of training.

B. Evaluated conditions

We use our experimental setup to evaluate and compare the following approaches:

1) RL: The first baseline consists of standard off-policy RL, without the use of controllers and without any modification to the behavior policy or loss function.

2) RL + BC loss: We also test a version of off-policy RL augmented using the same behavior cloning loss w.r.t the behavior policy as used in our method.

3) Fixed Guided Exploration (FGE): Instead of informing the probability for following the controllers with action-value estimate, this baseline uses a preset linear decay schedule for the probability of sampling from the controllers. Unlike our method, here the probability of following the controller is independent of the state and purely a function of training progress. We consider this baseline to check the advantage of allowing the probability of following the controllers to vary across the state-space.

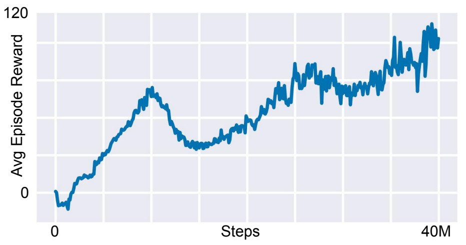


Fig. 5: Training curve for multiple objects using finger-gaiting controller.

4) Value Guided Exploration (VGE): This is the method proposed in this paper, combining sub-optimal controllers and value-guided controller sampling probability.

5) VGE + BC loss: This is a variant of our method that additionally uses behavior cloning with loss as in Eq 12.

C. Results and discussion

First, we evaluate our method for each of our sub-optimal controllers individually. The results for these experiments are shown Fig 4, (a) through (c). As we see, our method generally outperforms the baselines considered. Importantly, we also can see that the contact switching controller is critical for learning finger-gaiting skill. While the difference in contact switching controller and finger-gaiting controller is not significant the success rate of learning to gait w.r.t varying seeds is markedly improved. See Table I. We believe this can be attributed to using the stability controller as part of the in-grasp manipulation controller.

The inability of baselines RL + BC loss and RL to learn is important to note. It underscores the key idea of our method that it is critically essential to follow actions sampled by the sub-optimal controllers to enable exploration.

We can learn finger-gaiting policy for multiple objects. In this experiment, alongside the cube, we use sphere, cylinder, dodecahedron, and icosahedron. In Fig 5, we show that finger-gaiting controller in a single controller setting

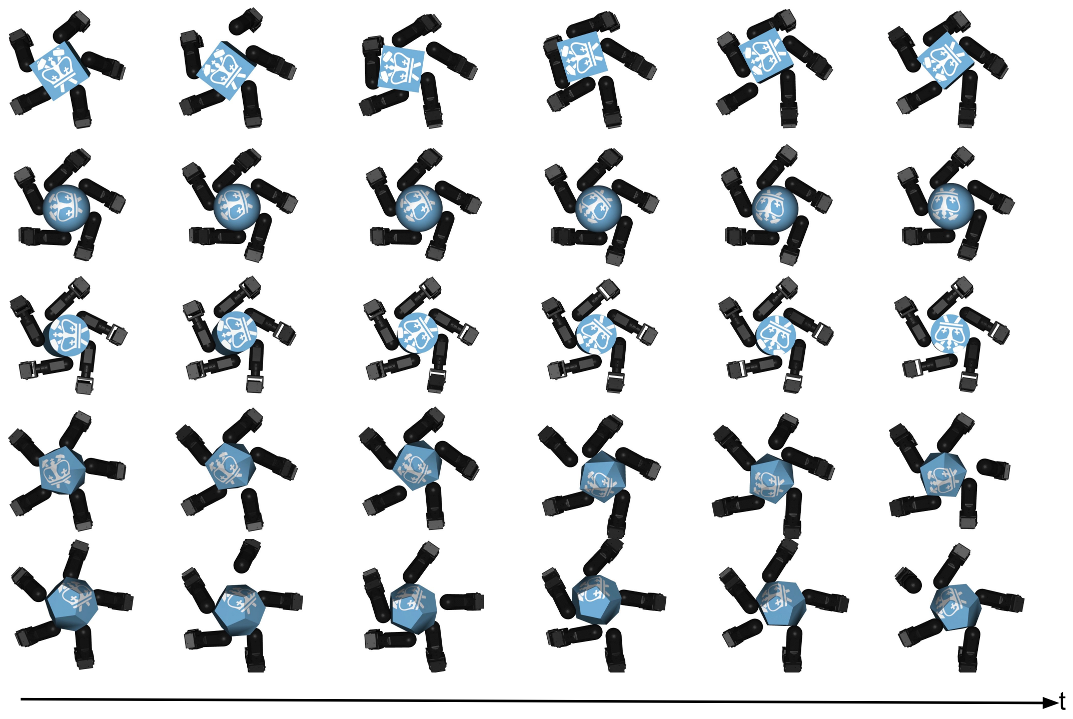


Fig. 6: Keyframes of the gaits achieved for multiple objects simultaneously with a single policy.

enables learning finger-gaiting policy simultaneously for the above objects. Also, Fig 6 shows the keyframes of the gaits achieved with this policy.

V. Conclusion

In this work, we proposed the use of model-based controllers to assist exploration in reinforcement learning of dexterous manipulation tasks. To this end, we use off-policy RL and take advantage of the freedom with construction of behavior policy to follow actions from the controller while collecting rollout trajectories and update the learning policy with such data to learn a successful policy.

We evaluated our method with the challenging dexterous manipulation task of learning finger-gaiting in-hand manipulation with only fingertip grasps. We designed simple controllers for key sub-skills in finger-gaiting - in-grasp manipulation and contact switching - and used these to enable learning effective finger-gaiting skills without any other additional means of improving exploration. Moreover, we show that our method benefits from model-based controllers even if they are significantly sub-optimal.

Overall, we demonstrate a method to use domain expertise for learning dexterous manipulation tasks with improved sample-efficiency. We believe our method of using sub-skill controllers for exploration is a promising approach for achieving dexterous skills with complex dynamics. While this work uses only model-based controllers, our method can also use learned sub-skill experts when available. It can also be extended to consider multiple experts. We hope to explore these directions in future work.

References

[1] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. "Learning Dexterous In-Hand Manipulation". In: (Aug. 2018). arXiv: 1808.00177 [cs.LG].

[2] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. "Visual Dexterity: In-hand Dexterous Manipulation from Depth". In: (Nov. 2022). arXiv: 2211.11744 [cs.RO].

[3] Gagan Khandate, Maximilian Haas-Heger, and Matei Ciocarlie. "On the Feasibility of Learning Finger-gaiting In-hand Manipulation with Intrinsic Sensing". In: *2022 International Conference on Robotics and Automation (ICRA)*. May 2022, pp. 2752–2758.

[4] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. "In-Hand Object Rotation via Rapid Motor Adaptation". In: (Oct. 2022). arXiv: 2210.04887 [cs.RO].

[5] Tingguang Li, Krishnan Srinivasan, Max Qing-Hu Meng, Wenzhen Yuan, and Jeannette Bohg. "Learning Hierarchical Control for Robust In-Hand Manipulation". In: (Oct. 2019). arXiv: 1910.10985 [cs.RO].

[6] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. "Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning". In: (Aug. 2021). arXiv: 2108.10470 [cs.RO].

[7] Arthur Allshire, Mayank Mittal, Varun Lodaya, Viktor Makoviychuk, Denys Makoviichuk, Felix Widmaier, Manuel Wüthrich, Stefan Bauer, Ankur Handa, and Animesh Garg. "Transferring Dexterous Manipulation from GPU Simulation to a Remote Real-World TriFinger". In: (Aug. 2021). arXiv: 2108.09779 [cs.RO].

[8] Tao Chen, Jie Xu, and Pulkit Agrawal. "A System for General In-Hand Object Re-Orientation". Nov. 2021.

[9] Susanna Leveroni and Kenneth Salisbury. "Reorienting Objects with a Robot Hand Using Grasp Gaits". In: *Robotics Research*. Springer London, 1996, pp. 39–51.

[10] L Han and J C Trinkle. "Dextrous manipulation by rolling and finger gaiting". In: *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*. Vol. 1. May 1998, 730–735 vol.1.

[11] Jean-Philippe Saut, Anis Sahbani, Sahar El-Khoury, and Veronique Perdereau. "Dexterous manipulation planning using probabilistic roadmaps in continuous grasp subspaces". In: *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Oct. 2007, pp. 2907–2912.

[12] T Omata and M A Farooqi. "Regrasps by a multifingered hand based on primitives". In: *Proceedings of IEEE International Conference on Robotics and Automation*. Vol. 3. Apr. 1996, 2774–2780 vol.3.

[13] Yongxiang Fan, Te Tang, Hsien-Chung Lin, Yu Zhao, and Masayoshi Tomizuka. "Real-Time Robust Finger Gaits Planning under Object Shape and Dynamics Uncertainties". In: (Oct. 2017). arXiv: 1710.10350 [cs.RO].

[14] Balakumar Sundaralingam and Tucker Hermans. "Geometric In-Hand Regrasp Planning: Alternating Optimization of Finger Gaits and In-Grasp Manipulation". In: (Apr. 2018). arXiv: 1804.04292 [cs.RO].

[15] Claire Chen, Preston Culbertson, Marion Lepert, Mac Schwager, and Jeannette Bohg. "TrajectoTree: Trajectory Optimization Meets Tree Search for Planning Multi-contact Dexterous Manipulation". In: (Sept. 2021). arXiv: 2109.14088 [cs.RO].

[16] Filipe Veiga, Benoni B Edin, and Jan Peters. "In-Hand Object Stabilization by Independent Finger Control". In: (June 2018). arXiv: 1806.05031 [cs.RO].

[17] Filipe Veiga, Riad Akrour, and Jan Peters. "Hierarchical Tactile-Based Control Decomposition of Dexterous In-Hand Manipulation Tasks". en. In: *Front Robot AI* 7 (Nov. 2020), p. 521448.

[18] Stephane Ross, Geoffrey J Gordon, and J Andrew Bagnell. "A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning". In: (Nov. 2010). arXiv: 1011.0686 [cs.LG].

[19] Stephane Ross and J Andrew Bagnell. "Reinforcement and Imitation Learning via Interactive No-Regret Learning". In: (June 2014). arXiv: 1406.5979 [cs.LG].

[20] Ching-An Cheng, Andrey Kolobov, and Alekh Agarwal. "Policy improvement via imitation of multiple oracles". In: (July 2020). arXiv: 2007.00795 [cs.LG].

[21] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations". In: (Sept. 2017). arXiv: 1709.10087 [cs.LG].

[22] Ilija Radosavovic, Xiaolong Wang, Lerrel Pinto, and Jitendra Malik. "State-Only Imitation Learning for Dexterous Manipulation". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Sept. 2021, pp. 7865–7871.

[23] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. "Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems". In: (May 2020). arXiv: 2005.01643 [cs.LG].

[24] Andrey Kurenkov, Ajay Mandlekar, Roberto Martin-Martin, Silvio Savarese, and Animesh Garg. "AC-Teach: A Bayesian Actor-Critic Method for Policy Learning with an Ensemble of Suboptimal Teachers". In: (Sept. 2019). arXiv: 1909.04121 [cs.LG].

[25] Rae Jeong, Jost Tobias Springenberg, Jackie Kay, Daniel Zheng, Yuxiang Zhou, Alexandre Galashov, Nicolas Heess, and Francesco Nori. "Learning Dexterous Manipulation from Suboptimal Experts". In: (Oct. 2020). arXiv: 2010.08587 [cs.RO].

[26] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor". In: (Jan. 2018). arXiv: 1801.01290 [cs.LG].

[27] Scott Fujimoto, Herke van Hoof, and David Meger. "Addressing Function Approximation Error in Actor-Critic Methods". In: (Feb. 2018). arXiv: 1802.09477 [cs.AI].