

A Simple Method for Complex In-Hand Manipulation

Tao Chen, Jie Xu, Pulkit Agrawal
Computer Science and Artificial Intelligence Laboratory (CSAIL)
Massachusetts Institute of Technology
Cambridge, MA 02139
{taochen, jiex, pulkitag}@mit.edu

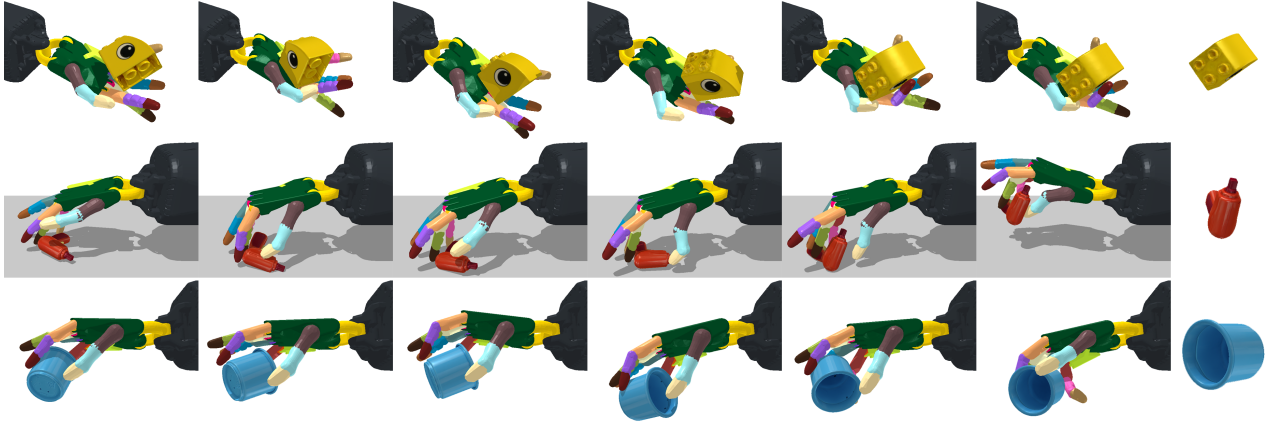


Fig. 1: We present a simple framework for learning policies for reorienting a large number of objects in scenarios where the (1) hand faces upward, (2) hand faces downward with a table below the hand and (3) without the support of the table. The object orientation in the rightmost image in each row shows the goal orientation.

Abstract—In-hand object reorientation has been a challenging problem in robotics due to high dimensional actuation space and the frequent change in contact state between the fingers and the objects. We present a simple model-free framework that can learn to reorient objects with both the hand facing upwards and downwards. We demonstrate the capability of reorienting over 2000 geometrically different objects in both cases. The learned policies show strong zero-shot transfer performance on new objects. We provide evidence that these policies are amenable to real-world operation by distilling them to use observations easily available in the real world. The videos of the learned policies are available at: <https://sites.google.com/view/in-hand-reorientation>.

I. INTRODUCTION

A common maneuver in many tasks of daily living is to pick an object, reorient it in hand and then either place it or use it as a tool. Consider three simplified variants of this maneuver in Figure 1. The first task shown in the top row requires an upward-facing multi-finger hand to manipulate an arbitrary object in a random orientation to a goal configuration. The next two rows show the second and third tasks where the hand is facing downward and is required to reorient the object either using the table as a support or without the aid of any support surface respectively. These two tasks are harder because the object is in an intrinsically unstable configuration owing to the downward gravitational force and lack of support from the

palm. Additional challenges in making such maneuvers with a multi-finger robotic hand stem from the control space being high-dimensional and reasoning about multiple transitions in the contact state between the finger and the object. Due to its practical utility and several unsolved issues, in-hand object reorientation remains an active area of research.

Past work has tackled the in-hand reorientation problem via several approaches: (i) The use of analytical models with powerful trajectory optimization methods [12, 2, 8]. While these methods demonstrated remarkable performance, the results were largely in simulation with simple object geometries and required detailed knowledge of the object model and physical parameters. As such, it remains unclear how to scale these methods to real-world and generalize to new objects. Another line of work has employed (ii) model-based reinforcement learning [10, 15]; or (iii) model-free reinforcement learning with [7, 9, 18, 17] and without expert demonstrations [14, 1, 16, 23]. While some of these works demonstrated learned skills on the real robots, they required substantial engineering and additional apparatus (e.g., motion capture system) to infer the object state, and the learned policies did not generalize to diverse objects. Furthermore, nearly all of these methods operate in a simplified setting where the hand is only allowed to face upward (with pick-and-place being an exception, but

pick-and-place doesn't incorporate any in-hand manipulation).

In this paper, our goal is to study the object reorientation problem with a multi-finger hand in its general form. We desire (a) manipulation with hand facing upward or downward; (b) the skill of using external surfaces to aid manipulation; (c) the ability to reorient objects of novel shapes to arbitrary orientations; (d) operation from sensory data that can be easily obtained in the real world such as RGBD images and joint positions of the hand. While some of these aspects have been individually demonstrated in prior work, we are unaware of any published method that realizes all four. Our main contribution is building a simple framework that can achieve these desiderata. We experimentally test this framework with a simulated Shadow hand. We also provide evidence indicating that the learned policies can be transferred to the real world in the future.

II. METHOD

We adopt the teacher-student training paradigm. First, we use PPO [20] to train a teacher policy with privileged full state information including joint positions/velocities, fingertip pose/velocities and object pose/velocities. Next, we distill the teacher policy into student policies that only use the sensory data that can be readily acquired in the real world via Dagger [19]. We designed two student policies depending on the type of sensory data available. Our first student policy takes as input the joint positions of the hand, object pose, goal orientation, and the action command at the previous time step a_{t-1} , and outputs control command. Our second student policy takes as input the point cloud of the manipulation scene, the joint positions and a_{t-1} . To process the point cloud data, we extend the IMPALA policy architecture [5] for processing RGB images to process colored point cloud data using 3D sparse convolution. We use Minkowski Engine [4], a PyTorch library for sparse tensors, to design a 3D version of IMPALA-Net with sparse convolutions (*Sparse3D-IMPALA-Net*) (see Fig. 2). The point cloud W_t is processed by a series of CNN residual modules and projected into an embedding vector. q_t and a_{t-1} are concatenated together and projected into an embedding vector via an MLP. Both embedding vectors from W_t and (q_t, a_{t-1}) are concatenated and passed through a recurrent network to output the action a_t . While the procedure described above works well for manipulation with the hand facing upwards and downwards with a table as a support, we find that in the case of the hand facing downward without a table, a good initialization of object pose such as the one obtained by an object lifting policy is key for successful reorientation, and the proposed gravity curriculum substantially improves performance.

Even though we do not test our policies on a real robot, we train and evaluate policies with domain randomization [22] to provide evidence that our work has the potential to be transferred to a real robotics system in the future. We randomize the object mass, friction coefficient, joint damping and add noise to the state observation s_t and the action a_t . More details about domain randomization are provided in Table I.

TABLE I: Dynamics Randomization and Noise

Parameter	Range	Parameter	Range	Parameter	Range
state observation	$+\mathcal{U}(-0.001, 0.001)$	action	$+\mathcal{N}(0, 0.01)$	joint stiffness	$\times\mathcal{E}(0.75, 1.5)$
object mass	$\times\mathcal{U}(0.5, 1.5)$	joint lower range	$+\mathcal{N}(0, 0.01)$	tendon damping	$\times\mathcal{E}(0.3, 3.0)$
object static friction	$\times\mathcal{U}(0.7, 1.3)$	joint upper range	$+\mathcal{N}(0, 0.01)$	tendon stiffness	$\times\mathcal{E}(0.75, 1.5)$
finger static friction	$\times\mathcal{U}(0.7, 1.3)$	joint damping	$\times\mathcal{E}(0.3, 3.0)$		

$\mathcal{N}(\mu, \sigma)$: Gaussian distribution with mean μ and standard deviation σ .
 $\mathcal{U}(a, b)$: uniform distribution between a and b . $\mathcal{E}(a, b) = \exp^{\mathcal{U}(\log(a), \log(b))}$.
 +: the sampled value is added to the original value of the variable. \times : the original value is scaled by the sampled value.

III. EXPERIMENTAL SETUP

We use the simulated Shadow Hand [21] in NVIDIA Isaac Gym [11]. In our experiments, we assume the base of the hand to be fixed. Twenty joints are actuated by agonist-antagonist tendons and the remaining four joints are under-actuated. We use the EGAD dataset [13] and YCB dataset [3] that contain objects with diverse shapes for in-hand manipulation experiments. We create 5 variants for each object mesh by randomly scaling the mesh. Mass is randomly sampled from $[0.05, 0.15]$ kg for every object. In total, we use 11410 EGAD object meshes and 390 YCB object meshes for training. The initial and goal orientations of the objects are both randomly sampled from the full $SO(3)$ space during both the training and testing. We consider a policy rollout to be a success if the angle difference $\Delta\theta$ between the object's current and the goal orientation is smaller than a threshold value $\bar{\theta}$, i.e., $\Delta\theta \leq \bar{\theta}$ ($\bar{\theta} = 0.1$). For vision experiments, $\bar{\theta} = 0.2$ and we also check the Chamfer distance [6] d_C between the object's point cloud sampled from its CAD mesh rotated to the current orientation and goal orientation respectively to handle the object symmetry issue. If d_C is less than a threshold value $\bar{d}_C = 0.01$, the episode is also considered a success. All the experiments in the paper were run on at most 2 GPUs with a 32GB memory.

IV. RESULTS

A. Reorient objects with the hand facing upward

We tried using a simple MLP as well as a simple RNN policy architecture for our teacher policy without using any object shape information. Both architectures work well. On the entire EGAD dataset, both policies are able to get a success rate greater than 90%. On the YCB dataset, the success rates for both policies are over 70% (see Table II). This result is surprising because intuitively, one would assume that information about the object shape is important for in-hand reorientation. The visualization of the policy rollout reveals that the agent throws the object in the air with a spinning motion and catches it at the precise time when the object's orientation matches the goal orientation. Throwing the object with a spin is a dexterous spin that automatically emerges! Even though we trained policies on only one dataset (either EGAD or YCB), we are able to get strong *zero-shot* transfer performance on the other untrained dataset. The RNN policy trained on EGAD gets a success rate of 68.82% on the YCB dataset, while the policy trained on the YCB dataset gets a success rate of 96.41% on the EGAD dataset. We are also able to distill our teacher policy to student policies that take as input the joint positions, object pose, goal orientation, and a_{t-1} without a performance drop in the success rates. We also

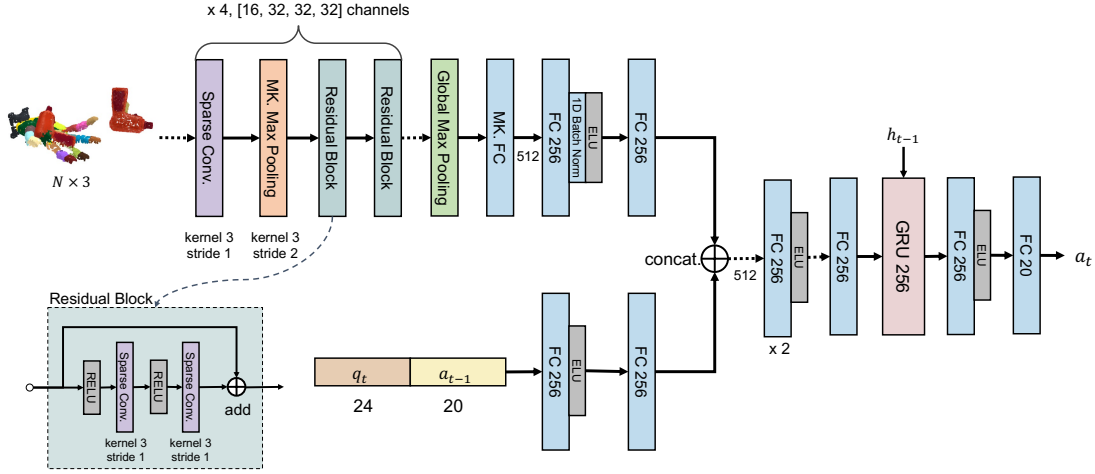


Fig. 2: Visual policy architecture. MK stands for Minkowski Engine. q_t is the joint positions and a_t is the action at time step t .

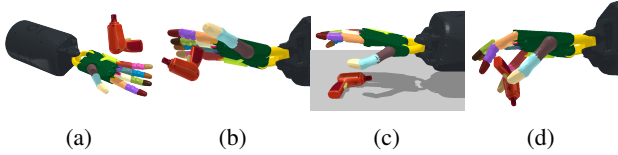


Fig. 3: Examples of initial poses of the hand and object. (a): hand faces upward. (b), (c), (d): hand faces downward. (b): both the hand and the object are initialized with random poses. (c): there is a table below the hand. (d): the hand and the object are initialized from the lifted poses.

trained our student policies with domain randomization that randomize the simulator dynamics. Our student policies are able to get a success rate of 80.29% on the EGAD dataset and 65.86% on the YCB dataset with randomized dynamics.

TABLE II: Success rates (%) of policies tested on different dynamics distribution. $\bar{\theta} = 0.1\text{rad}$. DR stands for domain randomization and observation/action noise. $X \rightarrow Y$: distill policy X into policy Y.

Dataset	State	Policy	1	2	3
			Train without DR		Train with DR
			Test without DR	Test with DR	Test with DR
EGAD	Full state	MLP	92.55 \pm 1.3	78.24 \pm 2.4	91.92 \pm 0.4
		RNN	95.95 \pm 0.8	84.27 \pm 1.0	88.04 \pm 0.6
	Reduced state	MLP \rightarrow MLP	55.55 \pm 0.2	25.09 \pm 3.0	23.77 \pm 1.8
		MLP \rightarrow RNN	85.32 \pm 1.2	68.31 \pm 2.6	81.05 \pm 1.2
		RNN \rightarrow RNN	91.96 \pm 1.5	78.30 \pm 1.2	80.29 \pm 0.9
YCB	Full state	MLP	73.40 \pm 0.2	54.57 \pm 0.6	66.00 \pm 2.7
		RNN	80.40 \pm 1.6	65.16 \pm 1.0	72.34 \pm 0.9
	Reduced state	MLP \rightarrow MLP	34.08 \pm 0.9	12.08 \pm 0.4	5.41 \pm 0.3
		MLP \rightarrow RNN	69.30 \pm 0.8	47.38 \pm 0.6	53.07 \pm 0.9
		RNN \rightarrow RNN	81.04 \pm 0.5	64.93 \pm 0.2	65.86 \pm 0.7

B. Reorient objects with the hand facing downward

1) *Reorient objects on a table (Fig. 3c)*: To tackle the problem of reorienting objects with the hand facing downward, we start with a simpler version of the task where there is a table below the hand. The success rate for the EGAD dataset with an MLP policy using full state information is 95.3%. The success rate for the YCB dataset is 81.59%.

2) *Reorient objects in air with hand facing downward (Fig. 3d)*: We experimentally verify that even in this case, the policies achieve a reasonably high success rate without any knowledge of object shape.

A good pose initialization is what you need: We first train an object-lifting policy to lift objects from the table, collect the ending state (joint positions, object position and orientation) in each successful lifting rollout episode, and reset the hand and objects to these states (velocities are all 0) for the pose initialization in training the reorientation policy. In every reset during the reorientation policy training, ending states are randomly sampled and used as the initial pose of the fingers and objects. With a good pose initialization, our teacher RNN policy trained on EGAD dataset gets a success rate of more than 80% while the teacher RNN policy trained on YCB dataset gets a success rate greater than 50%.

Improving performance using gravity curriculum: We also found that building a gravity curriculum boosts policy performance. Our gravity curriculum is constructed as follows: we start the training with $g = 1\text{m/s}^2$, then we gradually decrease g in a step-wise fashion if the evaluation success rate (w) is above a threshold value (\bar{w}) until $g = -9.81\text{m/s}^2$. By applying the gravity curriculum, our expert RNN policy gets 18% improvement in the success rate on the YCB training dataset.

C. Reorient objects with RGBD sensors

We also investigate whether we can train a student policy that directly uses vision as input to reorient objects with the hand facing upward. As vision-based experiments require much more compute resources, we train one vision policy for each object individually on six YCB objects. We use the expert MLP policy trained in Section IV-A to supervise the vision policy.

We also trained our vision policies with noise added to the point cloud input. We applied four types of transformations on the point cloud:

TABLE III: Success rates (%) of policies trained with hand facing downward and to reorient objects in the air. Due to the large number of environment steps required in this setup, we fine-tune the model trained without DR with randomized dynamics instead of training models with DR from scratch.

Dataset	State	Policy	1	2	3
			Train without DR		Finetune with DR
EGAD	Full state	MLP	84.29 \pm 0.9	38.42 \pm 1.5	71.44 \pm 1.3
		RNN	82.27 \pm 3.3	36.55 \pm 1.4	67.44 \pm 2.1
	Reduced state	MLP \rightarrow RNN	77.05 \pm 1.6	29.22 \pm 2.6	59.23 \pm 2.3
		RNN \rightarrow RNN	74.10 \pm 2.3	37.01 \pm 1.5	62.64 \pm 2.9
YCB	Full state	MLP	58.95 \pm 2.0	26.04 \pm 1.9	44.84 \pm 1.3
		RNN	52.81 \pm 1.7	26.22 \pm 1.0	40.44 \pm 1.5
		RNN + <i>g</i> -curr	74.74 \pm 1.2	25.56 \pm 2.9	54.24 \pm 1.4
	Reduced state	MLP \rightarrow RNN	46.76 \pm 2.5	25.49 \pm 1.4	34.14 \pm 1.3
		RNN \rightarrow RNN	45.22 \pm 2.1	24.45 \pm 1.2	31.63 \pm 1.6
		RNN + <i>g</i> -curr \rightarrow RNN	67.33 \pm 1.9	19.77 \pm 2.8	48.58 \pm 2.3







Object		Success rate (%)
	025_mug	89.67 \pm 1.2
	065-d_cups	68.32 \pm 1.9
	072-b_toy_airplane	84.52 \pm 1.4
	073-a_lego_duplo	58.16 \pm 3.1
	073-c_lego_duplo	50.21 \pm 3.7
	073-e_lego_duplo	66.57 \pm 3.1

TABLE IV: Vision policy success rate

- *RandomTrans*: Translate the point cloud by $[\Delta x, \Delta y, \Delta z]$ where $\Delta x, \Delta y, \Delta z$ are all uniformly sampled from $[-0.04, 0.04]$.
- *JitterPoints*: Randomly sample 10% of the points. For each sampled point i , jitter its coordinate by $[\Delta x_i, \Delta y_i, \Delta z_i]$ where $\Delta x_i, \Delta y_i, \Delta z_i$ are all sampled from a Normal distribution $\mathcal{N}(0, 0.01)$ (truncated at -0.015m and 0.015m).
- *RandomDropout*: Randomly dropout points with a dropout ratio uniformly sampled from $[0, 0.4]$.
- *JitterColor*: Jitter the color of points with the following 3 transformations: (1) jitter the brightness and rgb values, (2) convert the color of 30% of the points into gray, (3) jitter the color contrast. Each of this transformation can be applied independently with a probability of 30% if *JitterColor* is applied.

Each of these four transformations is applied independently with a probability of 40% for each point cloud at every time step. Table IV shows the success rates of the vision policies trained with the aforementioned data augmentations until policy convergence and tested with the same data augmentations. We found that adding the data augmentation in training helps improve the data efficiency of the vision policy learning even though the final performance might be a bit lower. Table IV shows that reorienting the non-symmetric objects including the toy and the mug has high success rates (greater than 80%). While training the policy for symmetric objects is much harder, using d_C as an episode termination criterion enables the policies to achieve a success rate greater than 50%.

V. DISCUSSION

Our results show that model-free RL with simple deep learning architectures can be used to train policies to reorient a

large set of geometrically diverse objects. Further for learning with hand facing downwards, we found that a good pose initialization obtained from a lifting policy was necessary, and the gravity curriculum substantially improved performance. The agent also learns to use an external surface (i.e., the table). The most surprising observation is that information about shape is not required despite the fact that we train a single policy to manipulate multiple objects. Perhaps in hindsight, it is not as surprising – after all, humans can close their eyes and easily manipulate novel objects into a specific orientation. However, even in this scenario, shape can be implicitly inferred using touch signals. As such, our work can serve a strong baseline for future in-hand object reorientation works that incorporate object shape information as part of the observation space to improve performance.

It should be noted that successful training of the teacher policy critically relied on shape information being expendable but required privileged information – a major obstacle in the real-world deployment. While we were not able to train reorientation policies from scratch using visual inputs, we show that we can distill this policy to operate from RGBD point clouds and joint positions that are readily available in the real world. While we only present results in simulation, we also provide evidence that our policies can be transferred to the real world. The experiments with domain randomization indicate that learned policies can work with noisy inputs.

REFERENCES

- [1] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.
- [2] Yunfei Bai and C Karen Liu. Dexterous manipulation using both palm and fingers. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1560–1565. IEEE, 2014.
- [3] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015.
- [4] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.
- [5] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416. PMLR, 2018.

- [6] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.
- [7] Abhishek Gupta, Clemens Eppner, Sergey Levine, and Pieter Abbeel. Learning dexterous manipulation for a soft robotic hand from human demonstrations. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3786–3793. IEEE, 2016.
- [8] Vikash Kumar, Yuval Tassa, Tom Erez, and Emanuel Todorov. Real-time behaviour synthesis for dynamic hand-manipulation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6808–6815. IEEE, 2014.
- [9] Vikash Kumar, Abhishek Gupta, Emanuel Todorov, and Sergey Levine. Learning dexterous manipulation policies from experience and imitation. *arXiv preprint arXiv:1611.05095*, 2016.
- [10] Vikash Kumar, Emanuel Todorov, and Sergey Levine. Optimal control with learned local models: Application to dexterous manipulation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 378–383. IEEE, 2016.
- [11] Jacky Liang, Viktor Makoviychuk, Ankur Handa, Nuttapong Chentanez, Miles Macklin, and Dieter Fox. Gpu-accelerated robotic simulation for distributed reinforcement learning. In *Conference on Robot Learning*, pages 270–282. PMLR, 2018.
- [12] Igor Mordatch, Zoran Popović, and Emanuel Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pages 137–144, 2012.
- [13] Douglas Morrison, Peter Corke, and Jürgen Leitner. Egad! an evolved grasping analysis dataset for diversity and reproducibility in robotic manipulation. *IEEE Robotics and Automation Letters*, 5(3):4368–4375, 2020.
- [14] Mayur Mudigonda, Pulkit Agrawal, Michael Deweese, and Jitendra Malik. Investigating deep reinforcement learning for grasping objects with an anthropomorphic hand. 2018.
- [15] Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pages 1101–1112. PMLR, 2020.
- [16] OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik’s cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.
- [17] Ilija Radosavovic, Xiaolong Wang, Lerrel Pinto, and Jitendra Malik. State-only imitation learning for dexterous manipulation. *arXiv preprint arXiv:2004.04650*, 2020.
- [18] 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. *arXiv preprint arXiv:1709.10087*, 2017.
- [19] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.
- [20] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [21] ShadowRobot. ShadowRobot Dexterous Hand, 2005. URL <https://www.shadowrobot.com/dexterous-hand-series/>.
- [22] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.
- [23] Henry Zhu, Abhishek Gupta, Aravind Rajeswaran, Sergey Levine, and Vikash Kumar. Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3651–3657. IEEE, 2019.