Generalizable Point Cloud Reinforcement Learning for Sim-to-Real Dexterous Manipulation

Anonymous authors
Paper under double-blind review

Abstract

We propose a sim-to-real framework for dexterous manipulation which can generalize to new objects of the same category in the real world. The key of our framework is to train the manipulation policy with point cloud inputs and dexterous hands. We propose two new techniques to enable joint learning on multiple objects and sim-to-real generalization: (i) using imagined hand point clouds as augmented inputs; and (ii) designing novel contact-based rewards. We empirically evaluate our method using an Allegro Hand to grasp novel objects in both simulation and real world. To the best of our knowledge, this is the first policy learning-based framework that achieves such generalization results with dexterous hands. Our project page is available at `http://dexpc.github.io/`.

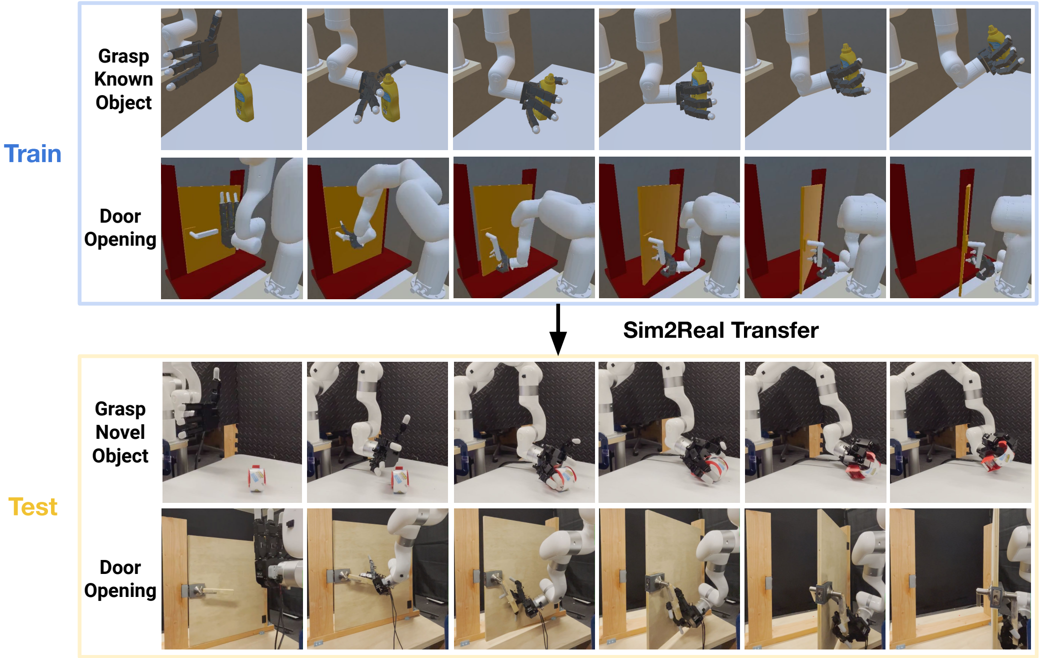


Figure 1: We introduce a reinforcement learning method which takes the point cloud as input for two manipulation tasks: grasping and door opening. By introducing several techniques in the policy learning process, our point cloud based policy trained purely in simulation can successfully generalize to novel objects and transfer to real-world without any real-world data.

1 Introduction

Dexterous manipulation has remained to be one of the most challenging problems in robotics Andrychowicz et al. (2020). While multi-finger hands create ample opportunities for our robots to flexibly manipulate objects in our daily life, the nature of the high degree of freedom and high-dimensional action space creates significant optimization challenges for both search-based planning algorithms and policy learning algorithms. Recent efforts using model-free Reinforcement Learning have achieved encouraging results on complex manipulation tasks Andrychowicz et al. (2020); Zhu et al. (2019); however, it still faces many challenges in generalizing to diverse objects and being deployed on real robots.

For example, the dexterous manipulation framework proposed by OpenAI et al. Andrychowicz et al. (2020) can solve in-hand manipulation of Rubik's Cube with RL and transfer to the real robot hand. However, the policy is only trained with one particular object and it is not able to *generalize to diverse objects*. To achieve cross-object generalization, recent efforts proposed to learn robust 3D point cloud representations Chen et al. (2022); Wu et al. (2022); Huang et al. (2021); Mu et al. (2021) with diverse objects using RL in simulation. While point cloud input has also been shown easier for Sim2Real transfer Zhang et al. (2022) given its focus on geometry instead of texture, the assumption on the access of complete object point clouds and ground truth states limit the transferability of above methods to the *real robot deployment*. Among these works, Chen et al. Chen et al. (2022) showed that cross-object generalization is achievable in simulation without knowing the shape of the object to grasp, but its requirement of real-time access to the object states is itself a very challenging robot perception problem, especially under large occlusions caused by real robot hands.

In this paper, we provide a sim-to-real reinforcement learning framework for generalizable dexterous manipulation, using two tasks with the Allegro HandRobotics: (i) object grasping where the test objects has not been seen during training; (ii) door opening where the robot needs to first rotate the lever to unlock the door latch and then pull the lever in a circular motion to open the door. The tasks are visualized in Figure 1. With both tasks, we list the three discoveries of our framework for learning generalizable point cloud policy below:

(i) We justified that it is possible to achieve *direct sim-to-real transfer* for a dexterous grasping policy with category-level generalizability when we use point cloud as the data representation;

(ii) Raw point clouds captured by sensors often come with heavy occlusions and noise. For example, a very small portion of the points from the observation are representing the robot fingers. We propose to *imagine* the complete robot hand point clouds according to the internal robot states and use them to augment the occluded real point cloud observations. We find that explicitly augmenting the input by *imagined* points can help achieve better robustness and sample efficiency for policy learning.

(iii) Different from existing works that add contact information to the input of RL, we design a novel reward using contact pair information without adding contact to the observation. This practice remarkably improves sample efficiency as well as learning stability and avoids the dependency on contact sensor that is often unavailable for real robot models.

2 Related Work

Dexterous Manipulation. Dexterous manipulation aims to enable robotic hands to achieve human-level dexterity for grasping and manipulating objects. Analytical methods have been proposed to adopt planning to solve this problem (Salisbury & Craig, 1982; Han & Trinkle, 1998; Rus, 1999; Prattichizzo & Trinkle, 2016; Bicchi & Kumar, 2000; Bohg et al., 2013; Dogar & Srinivasa, 2010). However, they rely on detailed object models, which are not accessible when testing on unseen objects. To mitigate this issue, researchers propose a learning-based method for dexterous manipulation (Kroemer et al., 2021). Some methods (Kumar et al., 2016; Levine et al., 2015; Andrychowicz et al., 2020; Zhu et al., 2019; Chen et al., 2022; Huang et al., 2021) take a reinforcement learning (RL) approach, while another line of work (Gupta et al., 2016; Zhang et al., 2018; Radosavovic et al., 2020; Johns, 2021) also proposes to learn from demonstration using imitation learning (IL) to acquire the control policy. Recently, the combination of RL and IL (Rajeswaran et al., 2018; Christen et al., 2019; Jeong et al., 2020; Qin et al., 2021; Wu et al., 2022) has also shown encouraging dexterous manipulation results in simulation. However, most methods are either learning policy for one single object or assume access to object states produced by perfect detector which increases challenges to Sim2Real transfer. In this paper, we surpass these limitations by introducing training on multiple objects and applying point cloud inputs for control.

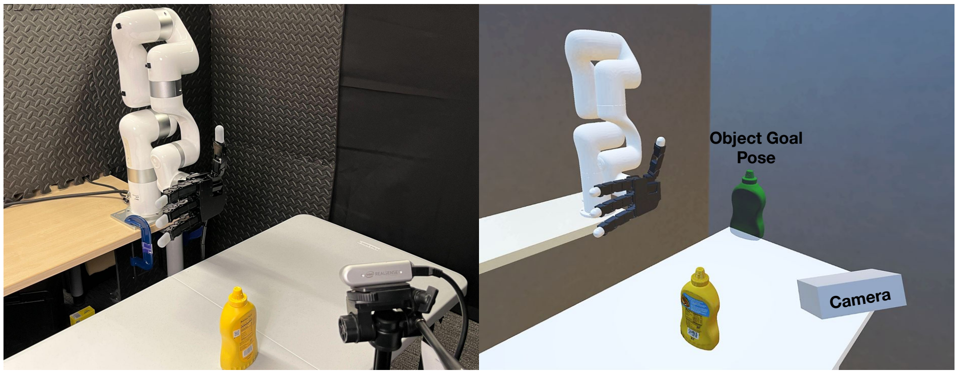


Figure 2: **Our Dexterous Manipulation System.** Left: Real-world setup with an Allegro Hand attached on an XArm6 and a RealSense D435. Right: Simulator setup with the transparent green object representing goal pose.

Point Cloud in Robotic Manipulation. Point cloud representation has been widely applied in the robotics community. Researchers have studied matching the observed point clouds to an object in a grasping dataset (Pomerleau et al., 2015) and executing the corresponding grasping action (Mahler et al., 2017; Bohg et al., 2013). For learning-based approaches, one line of research has focused on first estimating grasp proposals or affordance given the point cloud input and then planning accordingly for manipulation (ten Pas et al., 2017; Liang et al., 2019; Mousavian et al., 2019; Qin et al., 2020; Ni et al., 2020; Wu et al., 2020; Wei et al., 2021; 2022). While these methods are designed for parallel-jaw grippers, recent advancements have also been made for grasping with dexterous hands with similar approaches Andrews & Kry (2013); Varley et al. (2015); Brahmbhatt et al. (2019); Lu et al. (2020). However, this line of methods requires feedback from motion planning to estimate whether a grasp is plausible. Oftentimes, a stable grasp pose is proposed but it is not achievable by planning. To achieve efficient and flexible manipulation, a Reinforcement Learning policy with point cloud inputs is proposed Chen et al. (2022); Huang et al. (2021); Wu et al. (2022), which allows the robot hand to flexibly adjust its pose while interacting with the object. However, these approaches still face challenges in transferring to the real robot, given the noisy, occluded point clouds in the real world, and training RL policy with noisy point clouds increases the optimization difficulty. In this paper, we propose to use imagined point clouds, contact information, and multi-object training to combat these problems and achieve Sim2Real generalization.

Using Contact Information for Manipulation. Humans can manipulate objects purely from tactile sensing without seeing them. This biological fact inspires researchers to integrate contact and tactile information into the learning pipeline Chebotar et al. (2014); Yuan et al. (2017); Calandra et al. (2018); Murali et al. (2018); Lambeta et al. (2020); Lee et al. (2020); Bhirangi et al. (2021). For example, both tactile and visual information inputs are combined together in (Lee et al., 2020) for decision making. The tactile information is also utilized as inputs for RL-based manipulation policies Van Hoof et al. (2015); Vulin et al. (2021); Huang et al. (2019); Christen et al. (2019); Xu et al. (2021). For example, visual-tactile sensor is used with an Allegro hand in simulator for playing piano (Xu et al., 2021). Instead of using contacts as inputs, some methods also use contact and tactile information as reward to encourage exploration and boost policy learning Vulin et al. (2021); Christen et al. (2019). Motivated by these works, we provide a novel design of reward based on each contact link of the robot hand, which encourages more reasonable grasping behavior. Our design does not require a real tactile sensor when training in simulation or deploying in the real robot.

3 Approach

Our objective is to train a generalizable point cloud policy on a dexterous robot hand-arm system that is able to grasp a wide range of objects or open an closed door with RL. We aims at Sim-to-Real transfer without any real-world training or data. During testing, the robot can only access the single-viewed point cloud and the robot proprioception data. As is discussed before, training such a policy comes with numerous technical challenges, including reward design and imperfect point cloud information. In this work, we propose a novel reward design technique based on contact and imagined point cloud model to deal with these challenges. In the following sections, we first introduce the setup in section 3.1. We introduce reward design and imagined point cloud in section 3.2 and 3.3. Finally, we briefly discuss the implementation in section 3.4.

Preliminaries: We model the dexterous manipulation problem as a Partially Observable Markov Decision Process (POMDP) $\mathcal{M} = (\mathcal{O}, \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \mathcal{U})$. Here, $\mathcal{O}$ is the observation space, $\mathcal{S}$ is the underlying state space, $\mathcal{A}$ is the action space, $\mathcal{R}$ is the reward function, $\mathcal{T}$ is the transition dynamics, and $\mathcal{U}$ generates agent's observation. At timestep t, the environment is at the state $s_t \in \mathcal{S}$. The agent observes $o_t \sim \mathcal{U}(\cdot|s_t) \in \mathcal{O}$. The agent takes action a_t and receives reward $r_t = \mathcal{R}(s_t, a_t)$. The environment state at timestep $t+1$ then transit to $s_{t+1} \sim \mathcal{T}(s_t, a_t)$. The objective of the agent is to maximize the return $\sum_{t=0}^{T} \gamma^t r_t$, where γ is a discount factor.

3.1 Environment Setup

System Setup: Many previous works on learning-based dexterous manipulation attach the hand to a fixed platform to simplify the experiment environment in the real world. In this work, we create a more flexible and powerful dexterous manipulation system which includes both the robotic hand and the arm (Figure 2). Concretely, we attach the Allegro Hand to an XArm6 robot. Allegro Hand is a 16-DoF anthropomorphic hand with four fingers and XArm6 is a 6-DoF robot arm. We place a RealSense D435 camera at the right front of the robot to capture the point cloud. This setup brings additional challenges to RL exploration and Sim2Real deployment. We use SAPIEN Xiang et al. (2020) platform with uses a full-physics simulator to build the environment of the whole system. The simulation time step is $0.005s$ to ensure stable contact simulation. Each control step lasts for $0.05s$.

Tasks and Objects: In this paper, we expect our robot to perform the grasping task over a diverse set of objects and to open a locked door by rotating the lever. In the grasping task, we first select a random object from an object dataset and place it on the table. The robot is then required to move it to a target pose. Moreover, the robot should be able to generalize to different initial states, so we randomize both the initial pose and the goal pose for each trial. In simulation experiments, we use bottles and cans from both ShapeNet Chang et al. (2015) dataset and YCB Calli et al. (2015) dataset. In real world experiments, we only use bottles from YCB. More details about the object set can be found in appendix.

Observation Space: The observation contains both visual and proprioceptive information with four modalities: (1) Observed point cloud provided by the camera; (2) Proprioception signals of the robot including joint positions and end-effector position; (3) Imagined hand point cloud proposed in Sec. 3.3; (4) Object goal position provided in each trial. All the information is accessible on the real robot. The dimension of each observation modality is shown in Figure 3.

Action Space: The action is responsible for controlling both the 6-DOF robot arm and the 16-DOF hand. It has $6 + 16 = 22$ dimension in total. The robot arm is parametrized by the 6D translation and rotation of the end-effector relative to a reference pose. We use the damped least square inverse kinematics solver with a damping constant $\lambda = 0.05$ to compute the joint motion. Each finger joint of the Allegro hand is controlled by a position controller. Both robot arm and hand are controlled by PD controllers.

Network Architecture: The network architecture is visualized in Figure 3,

3.2 Reward Design with Oracle Contact

Since we aim to solve the dexterous manipulation problem with pure RL, the reward design is central to the method. We need a good reward function to ensure proper interaction between the robotic hand and the object. The whole interaction process consists of two phases. The first phase is to simply reach the object. The second phase is to grasp the object and move it to the target, which is more challenging. For the first phase, we encourage reaching with the following reaching reward:

$$r_{\text{reach}} = \sum_{\text{finger}} \frac{1}{\epsilon_r + d(\mathbf{x}_{\text{finger}}, \mathbf{x}_{\text{obj}})}. \tag{1}$$

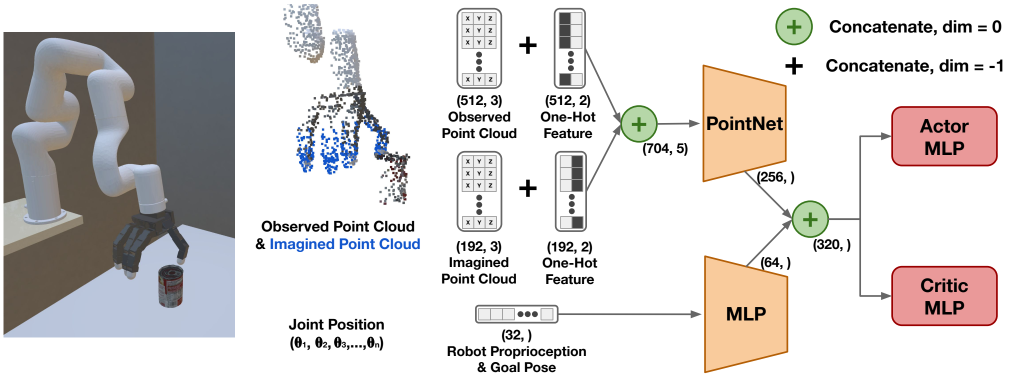


Figure 3: **Architecture:** our feature extractor takes the observed point cloud, imagined point cloud, robot proprioception, and goal pose as input to output a feature embedding. Both actor and critic take the same feature to predict action and value. The blue point represented the imaged point cloud. Note that our network does not require RGB information.

Here, $\mathbf{x}_{\text{finger}}$ and $\mathbf{x}_{\text{obj}}$ are the Cartesian position of each fingertip and the target object. Note that $\mathbf{x}_{\text{obj}}$ is available when we perform training in simulation. However, using this reward alone cannot ensure proper contact for grasping. For example, the robot can touch the object with the back of the hand rather than the palm and then get stuck in this local minimum. Therefore, we introduce a novel contact reward to guarantee meaningful contact behavior:

$$r_{\text{contact}} = \textbf{IsContact}(\text{thumb}, \text{object}) \textbf{ AND } \left(\sum_{\text{finger}} \textbf{IsContact}(\text{finger}, \text{object}) \geq 2 \right). \quad (2)$$

This contact reward function outputs a boolean value in $\{0, 1\}$. It outputs 1 only if the thumb is in contact with the object and there are more than one finger in contact with the object. Intuitively, it encourages the robot to cage the object within fingers. In this case, the robot can quickly find out stable grasping and lift the object to the target location. The lifting behavior is encouraged by

$$r_{\text{lift}} = r_{\text{contact}} \textbf{Lift}(\mathbf{x}_{\text{obj}}, \mathbf{x}_{\text{target}}). \quad (3)$$

The **Lift** function is basically in the form of Equation 1 and the main difference is that it will return a large reward value upon task completion. The overall reward function is a weighted combination of the terms above plus a control penalty:

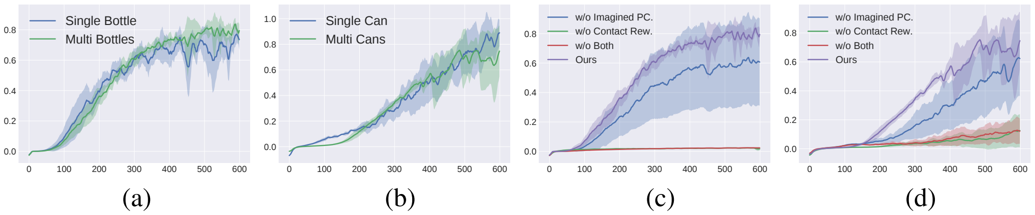
$$\mathcal{R} = w_{\text{reach}} r_{\text{reach}} + w_{\text{contact}} r_{\text{contact}} + w_{\text{lift}} r_{\text{lift}} + w_{\text{penalty}} r_{\text{penalty}}. \quad (4)$$

We refer readers to the appendix for the detailed calculation of each term and the hyper-parameters.

3.3 Imagined Hand Point Cloud

The usage of point cloud comes with two challenges. The first challenge is occlusion, which may occur to both the object under manipulation and the hand itself. When the robot hand is interacting with an object, the fingers may be occluded by the object. Since we do not assume tactile sensors in this work, this occlusion problem can be serious. The second challenge is the low point cloud resolution during RL training, where we can only use a limited number of points due to the memory limit. In this case, the number of points from hand finger may not be adequate to precisely capture the spatial relationship between the robot and the object. We propose a simple yet effective method to handle both issues in a unified manner. Our idea is to use an imagined hand point cloud in the observation to help the robot to "see" the interaction.

We provide one example in Figure 3. Black points indicate the point cloud captured by the camera, in which some important details of fingers are missing. These missing details provide crucial information of the interaction. Though such interaction information can also be inferred by combining the information from both the proprioception and visual input, we find that the best way is to synthesize these missing details. Concretely, we can compute the pose for each finger link via forward kinematics given the joint position from the robot joint encoder and the robot kinematics model. Then, we synthesize the imagined point cloud (blue points in Figure 3) by sampling the points from the mesh of each finger link. This process is possible in both simulation and the real world.



(a) (b) (c) (d)

Figure 4: **Training Curves.** The left two plots show the single-object and multi-object training curve of (a) bottle category and (b) can category. The right two plots show the ablation results on the (c) bottle category and (d) can category. The x-axis is the training episode and y-axis is the success rate of the corresponding task. The shaded area indicates standard error and the performance is evaluated on five random seeds.

Settings	**Bottle**		**Can**	
	Known Obj.	Novel Obj.	Known Obj.	Novel Obj.
Single Obj. Training	0.81 ± 0.09	0.60 ± 0.06	0.96 ± 0.04	0.63 ± 0.18
Multi Obj. Training	**0.83 ± 0.16**	**0.81 ± 0.15**	**0.93 ± 0.07**	**0.68 ± 0.09**

Table 1: **Comparison for training with single and multiple objects.** We evaluate the policy trained with single and multiple objects on bottle (upper two rows) and can (bottom two rows) categories. We train policy via two different approaches: (1) training the point cloud RL directly, and (2) distilling an oracle policy trained on oracle-states to point cloud RL policy. We test them on both known or novel objects. The results are reported on 5 seeds.

3.4 Training

We adopt Proximal Policy Optimization(PPO) Schulman et al. (2017) to train the agent in simulation, and then deployed to real without real-world fine-tuning. The network architecture is illustrated in Figure 3. Both value and policy networks share the same visual feature extraction backbone. We concatenate the observed point cloud with the imagined hand point cloud together as the input to the feature extractor. We also attach a one-hot encoding to each point which indicates whether it is observed or an imagined point. The detailed implementation can be found in appendix.

4 Experiments

4.1 Experimental Setup

Point Cloud Preprocessing: To enable smooth transfer from simulation to real-world, we apply the same data preprocessing procedure to the point cloud captured by the camera. It involves four steps: (i) Crop the point cloud to the work region with a manually-defined bounding-box; (ii) Downsample the point cloud uniformly to 512 points; (iii) Add a distance-dependent Gaussian noise to the simulated point cloud to improve the sim2real robustness; (iv) Transform point cloud from the camera frame to the robot base frame using camera pose. In simulation, we use the ground-truth camera pose with multiplicative noise for frame transformation. In the real-world, we perform hand-eye calibration to get the relative transformation from camera to robot base.

Evaluation Criterion: We evaluate the performance of a policy by its success rate. In simulation, a task is considered a success if $d_{ot} < 0.05m$, where d is the distance between object position and goal position. In real world, the task is considered as success if the XY position of the object is within 5cm from the target position and the height of the object is at least 15cm from table top. For each experiment, we train policies with **5 different random seeds** and report the mean and standard deviation.

. The new metric also considers the horizontal distance, which better captures the task performance. The new real-world experimental results are also performed based on this new metric.

EigenGrasp Baseline: We choose the widely-used EigenGrasp Ciocarlie et al. (2007) as the grasp representation. Given an object mesh model, we use the well-known GraspIt Miller & Allen (2004) to search valid grasp for Allegro Hand. Then, we use the RRTConnect Kuffner & LaValle (2000)

Settings	**Bottle**		**Can**	
	Known Obj.	Novel Obj.	Known Obj.	Novel Obj.
w/o Imagined PC.	0.60 ± 0.46	0.56 ± 0.51	0.91 ± 0.17	0.63 ± 0.07
w/o Contact Rew.	0.00 ± 0.00	0.00 ± 0.00	0.06 ± 0.08	0.03 ± 0.06
w/o Both	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Ours	$\mathbf{0.83 \pm 0.16}$	$\mathbf{0.81 \pm 0.15}$	$\mathbf{0.93 \pm 0.07}$	$\mathbf{0.68 \pm 0.09}$

Table 2: **Comparison for different settings on bottle and can categories:** we study the influence of contact-based reward design and imaged point cloud. We evaluate the trained policy on both known and novel objects. The left figure is the learning curve and the right table is the success rate. *PC.* means point cloud and *Rew.* means reward.

motion planner implemented in OMPL Sucan et al. (2012) to plan a joint trajectory to the pre-grasp pose and use a screw motion controller to move the robot from pre-grasp pose to the grasp pose. Finally, we close all fingers in a direction specially designed for Allegro Hand and lift the object to the target. Note that different from our approach, the baseline method requires complete object model to search for grasps and ground-truth object pose to align the grasp pose in the robot space. To evaluate the performance of baseline on novel objects, we first build a grasp database on ShapeNet bottle and can categories using GraspIt. Given the sensory data of a new object, we search for the most similar objects in the dataset and use the query grasps for the novel object. Here we compare the performance of our method with baselines in the real-world.

Training: We train RL in two settings: (i) training on a single object (ii) training on multiple objects jointly. For the single object setting, we perform experiments on both "tomato soup can" and "mustard bottle" from YCB. For the multi-object training, we choose 10 objects from the can or bottle categories of ShapetNet. We randomly choose one object to train for each episode in the multi-object training. Here, we use can and bottle as experimental subjects since they represent two different basic grasping patterns Napier (1956) for anthropomorphic hand: precision grasp and power grasp. More training details can be found in appendix.

4.2 Comparison of Single-object and Multi-object Training

We plot the training curve of RL in Figure 4. In general, our method can learn to grasp and move the object to the goal pose within 600 episodes, where each episode contains 20K environment steps. Then we evaluate the policy trained on both known and novel objects, and the results are shown in Table 1. We run 100 trials to compute the average success rate. The policies trained on multiple objects can outperform the policies trained on a single object by a large margin. In particular, policies trained with multiple objects can generalize much better to novel objects compared with those trained on a single object.

Besides the end-to-end RL training, we also experiment an alternative approach: training RL on oracle states, and distill the orcale-state-based policy into the point cloud policy. The oracle state is composed of robot proprioception, target pose, object pose, the distance vector between the object and each fingertips, and a finger contact vector indicating whether which finger is in contact with the object. Note that the contact vector can act like a haptic sensor, which enables the robot to manipulate the object even without vision.

We perform single-object and multi-object training for this alternative approach. The results are shown in Table 1. On grasping known objects, we find that agent trained on single-object outperforms the agent trained on multi-object by a small margin. However, agent trained on multi-objects does much better at grasping novel objects. Our results suggest that using multiple object during training is important, and is of great importance for novel object generalization.

4.3 Ablation Results in Simulation

We ablate two key innovations of the work: the reward design with oracle contact and the imaged hand point cloud. We perform experiments on four different variants: (i) without imagined point cloud; (ii) without contact-based reward design; (iii) without both imagined point cloud and contact based reward design; (iv) our standard approach with both techniques. Note that the variant (iii) is an approximation of (Chen et al., 2022) in our environments and tasks. We compare both the learning curve and the evaluation success rate of these four variants. Figure 4 (c) and (d) show the results on bottle and can categories and Table 2 shows the success rate. Our findings can be summarized as follows.

First, we find that contact reward information is of vital importance for training the point cloud RL policy on the multi-finger robot hand. Without using contact reward, the agent can hardly learn anything (red and green curve in the figure) and get nearly zero success rate during evaluation for both bottle and can categories. By encouraging the contact between fingers and object, the RL agent can avoid getting stuck in local minimums and learn meaningful manipulation behavior.

Second, the imagined point cloud can also improve the training and test performance for both categories, though it is not so important as the contact reward. As is shown in the learning curves of Figure 4 (c) and (d), the policy utilizing the imagined point cloud as input can learn faster in the early stage of the training and show much smaller variances. An interesting fact is that imagined point cloud is more beneficial for the bottle category than the can category. One possible reason is that grasping a bottle requires multi-finger coordination to perform a power grasp. Such coordination is highly dependent on the detailed finger information provided by the imagined point cloud. The imagined point cloud can help the agent to better see the fingers even if they are occluded by the object, e.g., fingers behind the object.

4.4 Real-World Evaluation

Method	Bottle	Can
EigenGrasp Oracle	0.66	0.45
EigenGrasp	0.50	0.41
Single Obj. Train	0.75 ± 0.06	0.75 ± 0.08
Multi Obj. Train	**0.87 ± 0.03**	**0.83 ± 0.13**

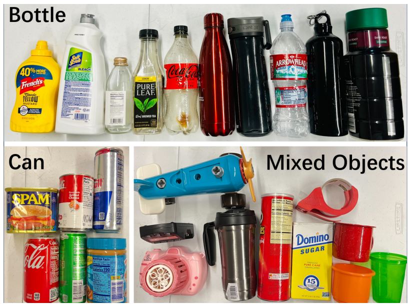


Table 3: **Real-World Experiments**: The real-world experiments consist of 3 categories, which incorporate 26 objects: 10 bottles, 6 cans, and 10 other objects mixed with multiple categories. The mixed categories are outside training categories such as mug, toys, camera, box, and tap. We train policies and both bottle and can categories simultaneously with different random seeds and evaluate these policies on the **Mixed Category**, which also show a reasonable success rate: **0.73** ± **0.12**. The left table shows the task performance and the right figure shows the objects we use for evaluation.

We perform sim2real experiments to evaluate the performance of our method in the real world. As is shown in Figure 2, we attached an Allegro hand onto an XArm-6 robot arm to grasp the object on the front table. We apply the same data-preprocessing steps for both simulated environment and real-world as mentioned in Sec.4.1. The task execution sequence is visualized on the bottom row of Figure 1. We run 10 independent trials seeds for each object-policy pair.

The real-world evaluation results are shown in Table 3. The policy trained in simulator can directly transfer to the real-world without fine-tuning. Our policy can even grasp objects that has never been seen during the training. We find that for grasping a mustard bottle, training on the same object can ensure better performance while for novel objects like bleach cleanser and water bottle II, training with more diverse objects can improve the performance.

5 Conclusion

To the best of our knowledge, our approach is the first work to train a dexterous manipulation policy with point cloud inputs that can transfer to the real world. We justified that direct sim-to-real transfer is possible for dexterous grasping with point cloud representation.

REFERENCES

Sheldon Andrews and Paul G Kry. Goal directed multi-finger manipulation: Control policies and analysis. *Computers & Graphics*, 2013.

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.

Raunaq Bhirangi, Tess Hellebrekers, Carmel Majidi, and Abhinav Gupta. Reskin: versatile, replaceable, lasting tactile skins. *arXiv preprint arXiv:2111.00071*, 2021.

Antonio Bicchi and Vijay Kumar. Robotic grasping and contact: A review. In *IEEE International Conference on Robotics and Automation*, volume 1, pp. 348–353. IEEE, 2000.

Jeannette Bohg, Antonio Morales, Tamim Asfour, and Danica Kragic. Data-driven grasp synthesis—a survey. *IEEE Transactions on robotics*, 30(2):289–309, 2013.

Samarth Brahmbhatt, Ankur Handa, James Hays, and Dieter Fox. Contactgrasp: Functional multi-finger grasp synthesis from contact. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2386–2393. IEEE, 2019.

Roberto Calandra, Andrew Owens, Dinesh Jayaraman, Justin Lin, Wenzhen Yuan, Jitendra Malik, Edward H Adelson, and Sergey Levine. More than a feeling: Learning to grasp and regrasp using vision and touch. *IEEE Robotics and Automation Letters*, 3(4):3300–3307, 2018.

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)*, pp. 510–517. IEEE, 2015.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Yevgen Chebotar, Oliver Kroemer, and Jan Peters. Learning robot tactile sensing for object manipulation. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3368–3375, 2014. doi: 10.1109/IROS.2014.6943031.

Tao Chen, Jie Xu, and Pulkit Agrawal. A system for general in-hand object re-orientation. In *Conference on Robot Learning*, pp. 297–307. PMLR, 2022.

Sammy Christen, Stefan Stevšić, and Otmar Hilliges. Guided deep reinforcement learning of control policies for dexterous human-robot interaction. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 2161–2167. IEEE, 2019.

Matei Ciocarlie, Corey Goldfeder, and Peter Allen. Dexterous grasping via eigengrasps: A low-dimensional approach to a high-complexity problem. In *Robotics: Science and systems manipulation workshop-sensing and adapting to the real world*, 2007.

Mehmet R Dogar and Siddhartha S Srinivasa. Push-grasping with dexterous hands: Mechanics and a method. 2010.

Abhishek Gupta, Clemens Eppner, Sergey Levine, and Pieter Abbeel. Learning dexterous manipulation for a soft robotic hand from human demonstrations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3786–3793. IEEE, 2016.

Li Han and Jeffrey C Trinkle. Dexterous manipulation by rolling and finger gaiting. In *IEEE International Conference on Robotics and Automation*, volume 1, pp. 730–735. IEEE, 1998.

Sandy H. Huang, Martina Zambelli, Jackie Kay, Murilo F. Martins, Yuval Tassa, Patrick M. Pilarski, and Raia Hadsell. Learning gentle object manipulation with curiosity-driven deep reinforcement learning. *arXiv preprint arXiv:1903.08542*, 2019.

Wenlong Huang, Igor Mordatch, Pieter Abbeel, and Deepak Pathak. Generalization in dexterous manipulation via geometry-aware multi-task learning. *arXiv preprint arXiv:2111.03062*, 2021.

Rae Jeong, Jost Tobias Springenberg, Jackie Kay, Daniel Zheng, Alexandre Galashov, Nicolas Heess, and Francesco Nori. Learning dexterous manipulation from suboptimal experts. In Jens Kober, Fabio Ramos, and Claire J. Tomlin (eds.), *Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 915–934. PMLR, 2020.

Edward Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4613–4619. IEEE, 2021.

Oliver Kroemer, Scott Niekum, and George Dimitri Konidaris. A review of robot learning for manipulation: Challenges, representations, and algorithms. *Journal of machine learning research*, 22(30), 2021.

James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2, pp. 995–1001. IEEE, 2000.

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)*, pp. 378–383. IEEE, 2016.

Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian Yang, Benjamin Maloon, Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, et al. Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation. *IEEE Robotics and Automation Letters*, 5(3):3838–3845, 2020.

Michelle A Lee, Yuke Zhu, Peter Zachares, Matthew Tan, Krishnan Srinivasan, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Jeannette Bohg. Making sense of vision and touch: Learning multimodal representations for contact-rich tasks. *IEEE Transactions on Robotics*, 36(3):582–596, 2020.

Sergey Levine, Nolan Wagener, and Pieter Abbeel. Learning contact-rich manipulation skills with guided policy search. In *IEEE International Conference on Robotics and Automation, ICRA*, pp. 156–163. IEEE, 2015.

Hongzhuo Liang, Xiaojian Ma, Shuang Li, Michael Görner, Song Tang, Bin Fang, Fuchun Sun, and Jianwei Zhang. Pointnetgpd: Detecting grasp configurations from point sets. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 3629–3635, 2019. doi: 10.1109/ICRA.2019.8794435.

Qingkai Lu, Kautilya Chenna, Balakumar Sundaralingam, and Tucker Hermans. Planning multi-fingered grasps as probabilistic inference in a learned deep network. In *Robotics Research*, pp. 455–472. Springer, 2020.

Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *Robotics: Science and System*, 2017.

Andrew T Miller and Peter K Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004.

Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2901–2910, 2019.

Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. *arXiv preprint arXiv:2107.14483*, 2021.

Adithyavairavan Murali, Yin Li, Dhiraj Gandhi, and Abhinav Gupta. Learning to grasp without seeing. In *International Symposium on Experimental Robotics*, pp. 375–386. Springer, 2018.

John R Napier. The prehensile movements of the human hand. *The Journal of bone and joint surgery. British volume*, 38(4):902–913, 1956.

Peiyuan Ni, Wenguang Zhang, Xiaoxiao Zhu, and Qixin Cao. Pointnet++ grasping: Learning an end-to-end spatial grasp generation algorithm from sparse point clouds. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3619–3625, 2020. doi: 10.1109/ICRA40945.2020.9196740.

François Pomerleau, Francis Colas, and Roland Siegwart. A review of point cloud registration algorithms for mobile robotics. *Foundations and Trends in Robotics*, 4(1):1–104, 2015.

Domenico Prattichizzo and Jeffrey C Trinkle. Grasping. In *Springer handbook of robotics*, pp. 955–988. Springer, 2016.

Yuzhe Qin, Rui Chen, Hao Zhu, Meng Song, Jing Xu, and Hao Su. S4g: Amodal single-view single-shot se(3) grasp detection in cluttered scenes. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 53–65. PMLR, 2020.

Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. *arXiv preprint arXiv:2108.05877*, 2021.

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)*, pp. 7865–7871. IEEE, 2020.

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 *Robotics: Science and Systems*, 2018.

WONIK Robotics. Allegro robot hand. `https://www.wonikrobotics.com/research-robot-hand`.

Daniela Rus. In-hand dexterous manipulation of piecewise-smooth 3-d objects. *The International Journal of Robotics Research*, 1999.

J Kenneth Salisbury and John J Craig. Articulated hands: Force control and kinematic issues. *The International journal of Robotics research*, 1(1):4–17, 1982.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Ioan A Sucan, Mark Moll, and Lydia E Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012.

Andreas ten Pas, Marcus Gualtieri, Kate Saenko, and Robert Platt. Grasp pose detection in point clouds. *The International Journal of Robotics Research*, 36(13-14):1455–1473, 2017.

Herke Van Hoof, Tucker Hermans, Gerhard Neumann, and Jan Peters. Learning robot in-hand manipulation with tactile features. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 121–127. IEEE, 2015.

Jacob Varley, Jonathan Weisz, Jared Weiss, and Peter Allen. Generating multi-fingered robotic grasps via deep learning. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 4415–4420. IEEE, 2015.

Nikola Vulin, Sammy Christen, Stefan Stevšić, and Otmar Hilliges. Improved learning of robot manipulation tasks via tactile intrinsic motivation. *IEEE Robotics and Automation Letters*, 6(2): 2194–2201, 2021.

Wei Wei, Yongkang Luo, Fuyu Li, Guangyun Xu, Jun Zhong, Wanyi Li, and Peng Wang. Gpr: Grasp pose refinement network for cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4295–4302. IEEE, 2021.

Wei Wei, Daheng Li, Peng Wang, Yiming Li, Wanyi Li, Yongkang Luo, and Jun Zhong. Dvgg: Deep variational grasp generation for dextrous manipulation. *IEEE Robotics and Automation Letters*, 2022.

Chaozheng Wu, Jian Chen, Qiaoyu Cao, Jianchi Zhang, Yunxin Tai, Lin Sun, and Kui Jia. Grasp proposal networks: An end-to-end solution for visual learning of robotic grasps. *Advances in Neural Information Processing Systems*, 33:13174–13184, 2020.

Yueh-Hua Wu, Jiashun Wang, and Xiaolong Wang. Learning generalizable dexterous manipulation from human grasp affordance. *arXiv preprint arXiv:2204.02320*, 2022.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Huazhe Xu, Yuping Luo, Shaoxiong Wang, Trevor Darrell, and Roberto Calandra. Towards learning to play piano with dexterous hands and touch. *arXiv preprint arXiv:2106.02040*, 2021.

Wenzhen Yuan, Siyuan Dong, and Edward H Adelson. Gelsight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, 2017.

Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5628–5635, 2018.

Xiaoshuai Zhang, Rui Chen, Fanbo Xiang, Yuzhe Qin, Jiayuan Gu, Zhan Ling, Minghua Liu, Peiyu Zeng, Songfang Han, Zhiao Huang, et al. Close the visual domain gap by physics-grounded active stereovision depth sensor simulation. *arXiv preprint arXiv:2201.11924*, 2022.

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)*, pp. 3651–3657. IEEE, 2019.