LEARNING CONTINUOUS GRASPING FUNCTION WITH A DEXTEROUS HAND FROM HUMAN DEMONSTRATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose to learn to generate grasping motion for manipulation with a dexterous hand using implicit functions. With continuous time inputs, the model can generate a continuous and smooth grasping plan. We name the proposed model Continuous Grasping Function (CGF). CGF is learned via generative modeling with a Conditional Variational Autoencoder using 3D human demonstrations. We will first convert the large-scale human-object interaction trajectories to robot demonstrations via motion retargeting, and then use these demonstrations to train CGF. During inference, we perform sampling with CGF to generate different grasping plans in the simulator and select the successful ones to transfer to the real robot. By training on diverse human data, our CGF allows generalization to manipulate multiple objects. Compared to previous planning algorithms, CGF is more efficient and achieves significant improvement on success rate when transferred to grasping with the real Allegro Hand. Our anonymous project page is available at https://continuous-grasping.github.io/.

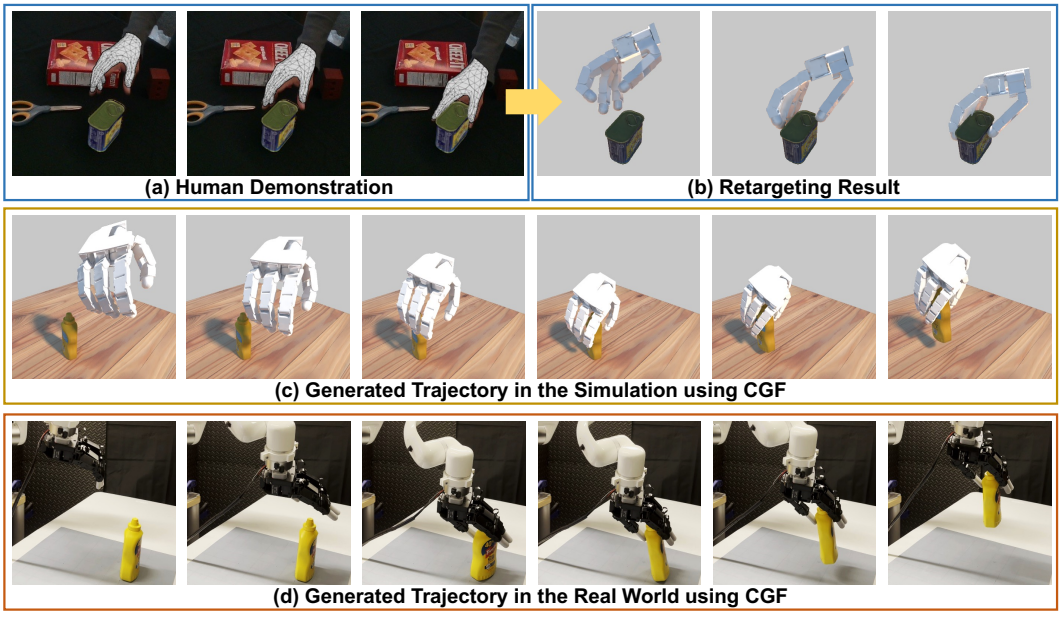

Figure 1: Examples of our generated trajectories learned from human demonstrations. Given hand-object trajectories from human video (a), we first translate them into robot manipulation demonstrations (b). We then train Continuous Grasping Function (CGF) to generate human-like trajectories and deploy them in simulation (c) and real robot (d).

1 INTRODUCTION

Learning to perform grasping with a multi-finger hand has been a long-standing problem in robotics (Salisbury & Craig, 1982; Rus, 1999; Okamura et al., 2000; Dogar & Srinivasa, 2010). Using a dexterous hand instead of a parallel gripper offers the robot the flexibility on operating with daily life objects like humans do, but also largely increases the difficulty given the large Degree-of-Freedom of the dexterous hand. A typical method for this task is a 2-step paradigm including grasp pose estimations following by motion planning (Varley et al., 2015; Brahmbhatt et al., 2019; Lu et al., 2020). Recent works have also studied on using Reinforcement Learning with human demonstration guidance for grasping (Mandikal & Grauman, 2021; Qin et al., 2021).

While these approaches have shown encouraging results, they plan the grasping with finite discrete time steps. On the other hand, human grasping motion is continuous, can we learn a continuous grasping process for robot hands? Making robot grasping continuous can lead to a more natural and human-like trajectory, and each step we sample will be differentiable which we can use a more robust augmented PD controller. Recent progress on neural implicit functions have shown successful applications in learning continuous image representation (Chen et al., 2021; Dupont et al., 2021) and continuous 3D shape representation (Park et al., 2019; Mescheder et al., 2019; Mildenhall et al., 2020). Can this success be migrated from representing 2D/3D space to time?

In this paper, we propose to learn Continuous Grasping Function (CGF) with a dexterous robotic hand. To mimic the continuous human motion, we utilize human grasp trajectories from videos to provide demonstrations and supervision in training. By training CGF with generative modeling on a large-scale of human demonstrations, it allows generalization to grasp multiple objects with a real Allegro robot hand as shown in Figure 1 (d).

Specifically, given the 3D hand-object trajectories from human videos (Figure 1 (a)), we first perform motion retargeting to convert the human hand motion to the robotic hand motion to obtain the robotic manipulation demonstrations (Figure 1 (b)). We then learn a CGF in the framework of a Conditional Variational AutoEncoder (CVAE) (Sohn et al., 2015a) by reconstructing the robotic hand motion with these demonstrations. Specifically, the conditional encoder of our CGF model will take the object point clouds as inputs and provides the object embedding. Taking the concatenation of the object embedding, a latent code z and a time parameter t, the decoder of CGF is an implicit function which outputs the dexterous hand parameters in the corresponding time t. By sampling a continuous-time sequence of t, we can recover a continuous grasping trajectory in any temporal resolution. By sampling the latent code z, we can achieve diverse trajectories for the same object. Figure 1 (c) shows an example of the inferred grasping trajectory in the simulator.

In our experiments on testing our model, we will perform sampling on the latent code z multiple times given a test object and generate diverse grasping trajectories. We then execute these trajectories in the simulator and select the one which can successfully grasp the object up. Different from the previous paradigm on grasping followed by planning, we empirically find our method is much more efficient since we avoid performing planning for each trajectory but directly generate the trajectory from CGF. Given the selected trajectories from the simulator, we can deploy them in the real world with an Allegro hand attached on an X-Arm 6 robot. Compared with planning, our method achieves better Sim2Real generalization with more natural and human-like motion, which leads to a better success rate.

We highlight our main contributions here: (i) A novel Continuous Grasping Function (CGF) model which allows smooth and dense sampling in time for generating grasping trajectory; (ii) CGF allows efficient generation of grasping plan and more robust control in simulation; (iii) We achieve much significant improvement on Sim2Real generation on Allegro hand by learning CGF from human demonstrations.

2 RELATED WORK

Generalization in Dexterous Manipulation. Dexterous manipulation is one of the most challenging problems in robotics (Salisbury & Craig, 1982; Rus, 1999; Okamura et al., 2000; Dogar & Srinivasa, 2010; Dafle et al., 2014; Bai & Liu, 2014; Calli et al., 2018). Recent studies on model-free (OpenAI et al., 2018; 2019; Huang et al., 2021; Chen et al., 2022) and model-based (Kumar et al., 2016b; Nagabandi et al., 2020) Reinforcement Learning (RL) have achieved encouraging re-

sults on multiple complex dexterous manipulation tasks. However, there is still a large challenge on generalization for RL. For example, when the RL policy in OpenAI et al. (2018) can be transferred to the real robot, it is learned specifically for one object. On the other hand, an RL policy trained with multiple objects in simulator (Chen et al., 2022) has not yet been transferred to real. Instead of using RL, one line of works on dexterous grasping is first performing a grasp estimation and then planning for execution (Andrews & Kry, 2013; Kappler et al., 2015; Varley et al., 2015; Lu et al., 2020), which have shown great success on generalization to multiple objects and in real robots at the same time. Our work is more close to this line of research, where we achieve generalization by learning a continuous grasping function from human videos. Different from previous works, our method allows more smooth and natural grasping generation and it is also more efficient and accurate at the same time. We will provide comparisons to previous planning approaches in our experiments.

Grasping Motion Synthesis. Synthesizing and predicting human grasp has been an active research field for both computer vision and robotics (Morrison et al., 2018; Brahmbhatt et al., 2019; Taheri et al., 2020; Karunratanakul et al., 2020; Jiang et al., 2021a; Grady et al., 2021; Yang et al., 2021; Wei et al., 2022). For example, Grasping Field (Karunratanakul et al., 2020) is proposed as an implicit function that generates plausible human grasps given a 3D object. However, to apply on the robot hand, we will need to synthesize the full motion instead of a static pose. This motivates the research on synthesizing the hand-object interaction motions (Hsiao & Lozano-Perez, 2006; Ye & Liu, 2012; Wu et al., 2021; Zhang et al., 2021; Taheri et al., 2022; Christen et al., 2022). For example, a full body and hand motion are synthesized together to grasp an object in Wu et al. (2021). While related to our work, most approaches are still focusing on modeling the human hand. In this paper, we provide a framework where we first retarget the human hand trajectories to the robot hand trajectories and learn the robot grasping function with them.

Learning from Human Demonstrations. Our work is related to imitation learning or RL with human demonstrations for not only parallel grippers (Schmeckpeper et al., 2020; Shao et al., 2020; Young et al., 2020) but also dexterous hands (Gupta et al., 2016; Kumar et al., 2016a; Rajeswaran et al., 2017; Christen et al., 2019; Garcia-Hernando et al., 2020; Qin et al., 2021; Sivakumar et al., 2022; Qin et al., 2022; Wu et al., 2022). For example, DexMV is a platform proposed in Qin et al. (2021) for extracting 3D human demonstrations from videos, generating robot demonstrations by retargeting, and augmenting Reinforcement Learning with these demonstrations for multiple manipulation tasks. While we both share similar methods on obtaining robot demonstrations for training, the RL approach taken GT states as inputs in DexMV limits its generalizability to multiple objects and real robot transfer. On the other hand, our implicit function is able to generate continuous grasping given a point cloud input and it can be deployed in the real robot. As most RL approaches are still with full access to GT states, they are not directly comparable to our method.

Implicit Functions for Robotic Tasks. Beyond its successful applications in computer vision, implicit functions have recently been explored in robotic manipulation tasks (Li et al., 2021; Simeonov et al., 2021; Li et al., 2022; Jiang et al., 2021b; Adamkiewicz et al., 2022). For example, NeRF (Mildenhall et al., 2020) is used as a manner for learning 3D representations for control in Li et al. (2021). Instead of using implicit functions to learn static 3D representations, our work focuses on the continuity in time. We build a grasping function to directly generate the trajectory instead of a static scene.

3 METHOD

3.1 OVERVIEW

We aim to learn human-like robot grasping given object point clouds as input. We emphasize that learning from human demonstration could lead to more natural trajectories and the continuity helps the following control. To this end, we train Continuous Grasping Function with a CVAE framework to generate continuous trajectories and deploy them in the simulator and real robot consecutively. The pipeline is shown in Fig. 2. During training, we first perform hand motion retargeting on a large-scale hand-object interaction dataset (Chao et al., 2021) to collect demonstrations. Then the retargeting results served as the supervision for the grasping function learning. During inference, numerous continuous trajectories are sampled for a specific object and tested in the simulator. The successful trajectories will be deployed to the real robot. Besides, we can take more advantage

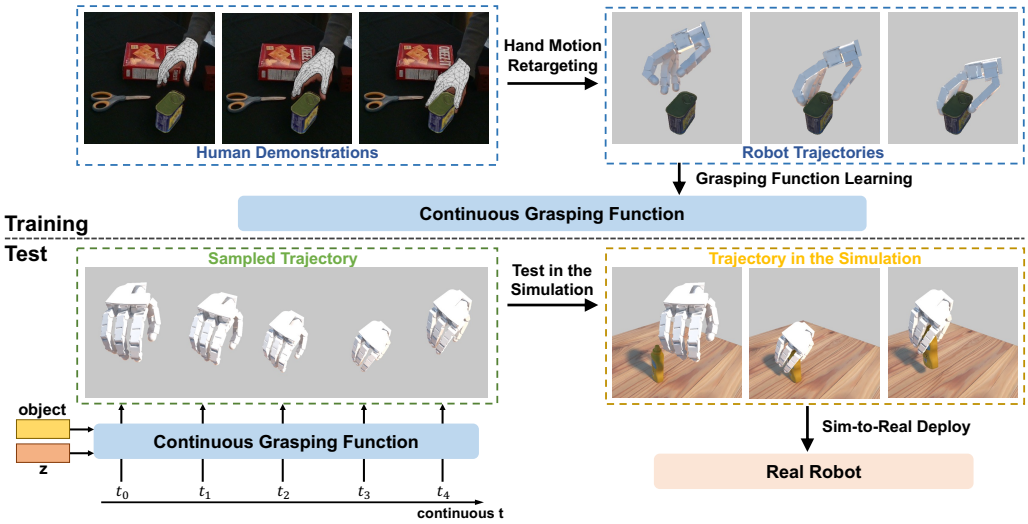

Figure 2: Pipeline overview. During training, human demonstrations are first translated to robot joint positions which serve as the supervision for grasping function learning. During inference, our trained CGF takes a sampled latent code z, object feature and query time sequence as inputs to generate the trajectory. We then execute these trajectories in the simulator and deploy successful ones to the real robot.

of continuous implicit function by utilizing augmented PD control with the derivative of the joint positions.

3.2 Human Demonstration Translation

Data collection on human hand-object interactions is relatively well established and accessible. Using large-scale human hand-object interaction data, we can learn patterns of how dexterous hands manipulate objects, and our goal is to generalize it to the robot hand. In this paper, we use ground truth trajectories from DexYCB (Chao et al., 2021). Translating human hand motion to robotics motion is the first step. Following Qin et al. (2022), we formulate our hand motion retargeting problem as a position-based optimization problem. We encourage the robot's joint position to be as close as possible to its corresponding human hand joint position,

$$\min_{q_t} \sum_{i=0}^{N} ||\mathcal{J}_i(q_t) - j_i||^2 + \lambda||\mathcal{J}_i(q_t) - \mathcal{J}_i(q_{t-1})||^2 \qquad \text{s.t. } q_{lower} \leq q_t \leq q_{upper}, \tag{1}$$

where q_t is joint position at time t, $\mathcal{J}_i$ is the forward kinematics function of the i-th joint and j_i is the Cartesian coordinates of the hand joint which matches the i-th joint of the robot. q_{lower} and q_{upper} are the joint limits. We also encourage smoothness by incorporating a normalization term to penalize a large distance between q_t and q_{t-1}. We set the initial $q_0 = \frac{1}{2}(q_{lower} + q_{upper})$.

3.3 Continuous Grasping Function Learning

Our generative model is based on Conditional Variational Auto-Encoder (CVAE) (Sohn et al., 2015b), where we propose Continuous Grasping Function (CGF) to replace the traditional decoder. During training, both encoder and CGF are used to learn the grasping generation in a robot hand reconstruction task with object point clouds and robot joint positions as inputs; During inference, only CGF is used to generate continuous grasping with only object point clouds as input. The architecture is shown in Figure 3.

During training, given the object point cloud $\mathcal{P}^o \in \mathbb{R}^{N\times 3}$ (N is the number of points) and a sequence of joints positions $\{q_t\}, t \in \{0, 1, \cdots, T\}$ (T is the number of frames) as inputs, we employ PointNet (Qi et al., 2017) and MLP to extract object feature $\mathcal{F}^o \in \mathbb{R}^{1024}$ and hand features $\{\mathcal{F}_t^h\} \in \mathbb{R}^{256}$ respectively. All these features are then concatenated as $\mathcal{F}^{oh}$ for the encoder input. The outputs of the encoder are the mean $\mu \in \mathbb{R}^{256}$ and variance $\sigma^2 \in \mathbb{R}^{256}$ of the Gaussian distribution $\mathcal{Q}\left(z \mid \mu, \sigma^2\right)$. The latent code z is sampled from the distribution for the hand reconstruction.

After the encoding and sampling, we use our CGF to decode continuous grasping. Inspired by implicit functions (Park et al., 2019; Mescheder et al., 2019; Chen & Zhang, 2019) for shape repre-

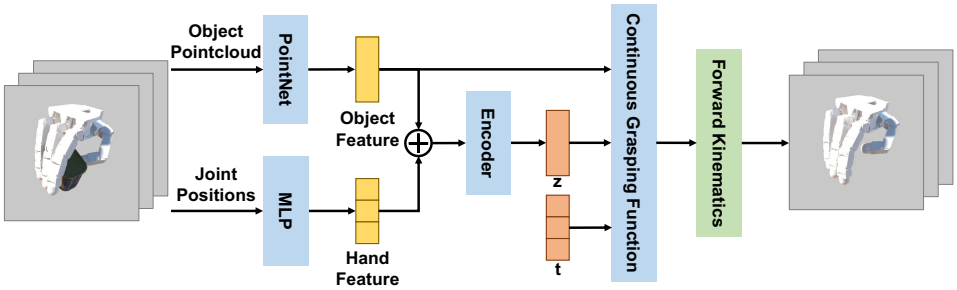

Figure 3: Network architecture. Our generative model takes object point cloud and a sequence of joint positions as input and recovers corresponding robot hands. The proposed CGF takes the latent code z, object feature, and the query time t as inputs to predict the corresponding joint position $\hat{q}_t$. $\oplus$ denotes concatenation.

sentation, our proposed CGF maps the query time t to the joint position q_t. We concatenate latent code z and the object feature $\mathcal{F}^o$ with time t as the input for CGF. Specifically, CGF is a MLP f parameterized by θ which predicts the corresponding joint position $\hat{q}_t$:

$$\hat{q}_t = f(t, z, \mathcal{F}^o; \theta). \tag{2}$$

We reverse the time and define $t = 0$ as the end of the grasping, i.e., when the robot's hand touches the object. And as t grows, the hand moves further away from the object. Given the predicted joint position $\hat{q}_t$, a differentiable forward kinematics layer $\mathcal{J}_i$ is utilized to get the Cartesian coordinate of the i-th joint.

The first objective function is the reconstruction error, which is defined as the $\mathcal{L}2$ distance between the joint positions as well as their Cartesian coordinates. We denote them as $\mathcal{L}_q = \sum_{t=0}^{T-1} \|\hat{q}_t - q_t\|_2^2$ and $\mathcal{L}_j = \sum_{t=0}^{T-1} \sum_{i=0} \|\mathcal{J}_i(\hat{q}_t) - \mathcal{J}_i(q_t)\|_2^2$ respectively. Following the training of VAE (Andrews & Kry, 2013), we define a KL loss to encourage the latent code distribution $\mathcal{Q}\left(z \mid \mu, \sigma^2\right)$ to be close to the standard Gaussian distribution, which is achieved by maximizing the KL-Divergence as $\mathcal{L}_{KL} = -D_{KL}\left(Q\left(z \mid \mu, \sigma^2\right) \|\mathcal{N}(0, I)\right)$. We also introduce a contact loss at the end of the grasping to push the tips of robot hand to the object surface, which is achieved by minimizing distances to their closest object points $\mathcal{L}_{contact} = \sum_i \min_{p \in \mathcal{P}^o} \|\mathcal{J}_i(q_0) - p\|_2^2$ where i belongs to the indices of all tips. The full training loss is:

$$\mathcal{L} = \lambda_q \mathcal{L}_q + \lambda_j \mathcal{L}_j + \lambda_{KL} \mathcal{L}_{KL} + \lambda_c \mathcal{L}_{contact}, \tag{3}$$

where $\lambda_q, \lambda_j, \lambda_{KL}$ and λ_c are weights for various losses.

During inference, our CGF can easily sample a large number of diverse trajectories. We use PointNet to get the object feature $\mathcal{F}^o$ and sample a random latent code z from the standard Gaussian distribution, then with a time query sequence, our CGF can produce a continuous and natural trajectory. By sampling a large amount of z, we can find trajectories with a smaller gap between the human hand and robot hand, which guarantees both natural and successful trajectories.

3.4 Augmented PD Control

Different from previous works, we utilize an implicit function to output the target joint position q_d, and given the continuity property of the implicit function, we can easily get the derivative (target joint velocity) $\dot{q}_d$ and the second-order derivative (target joint acceleration) $\ddot{q}_d$. Thus we can use a more robust controller, augmented PD control with the form of

$$\tau = \text{ID}(\ddot{q}_d, q, \dot{q}) - K_p e - K_v \dot{e}, \tag{4}$$

where $e = q - q_d$, $\dot{e} = \dot{q} - \dot{q}_d$, τ is the joint torque and $\text{ID}(\ddot{q}_d, q, \dot{q})$ is the inverse dynamics. K_p and K_v are hyperparameters. q, $\dot{q}$, and $\ddot{q}$ are the joint position, velocity, and acceleration of the robot. As far as we know, our method is the first end-to-end manner that can directly get the derivative to use the augmented PD control.

4 Experiments

We perform quantitative and qualitative evaluations of the grasping generated by our method. We show that by learning from human demonstrations, our method is more efficient in finding successful

grasping in the simulator. We do not plan for each trajectory, instead, we sample a large number of trajectories to get successful trajectories. For real-world experiments, we employ an Allegro hand attached on X-Arm 6. We perform evaluations to show that our generation results have the ability to be transferred to the real robot with a higher success rate.

4.1 Experimental Setting

Datasets. We utilize the DexYCB dataset (Chao et al., 2021) to serve as human demonstrations. DexYCB contains 1,000 sequences of human grasping motions with 20 YCB objects (Calli et al., 2015). We use 15 of them as training objects and 5 of them as test objects. We filter out all left-handed sequences and perform hand motion retargeting (Sec. 3.2) on other sequences to translate the hand motion into Allegro robot motion.

Baselines. We mainly compare CGF to two-step methods, i.e., grasping synthesis followed by motion planning. We provide two grasping synthesis baselines: GraspTTA (Jiang et al., 2021a) and GraspIt (Miller & Allen, 2004), where the former is also a generative model and the latter is based on searching. For GraspTTA, since its direct output is the Mano hand pose, we apply the same hand motion retargeting to obtain the Allegro hand joint positions. After that, we utilize rapidly exploring random tree (RRT) (LaValle et al., 1998) and cross-entropy method (CEM) (Rubinstein, 1999) with model predictive control (MPC) (Maciejowski, 2002) for planning the trajectory to reach the grasp pose. We name them GraspTTA+RRT and GraspTTA+CEM-MPC respectively. Note that GraspTTA+CEM-MPC is used for generating the demonstrations in Wu et al. (2022), and we use its reward for CEM-MPC. For GraspIt, due to the high Degree-of-Freedom of the dexterous hand, we firstly leverage the large set of grasping poses from ContactGrasp (Brahmbhatt et al., 2019) to construct a low-dimensional subspace via EigenGrasp (Ciocarlie et al., 2007). We then use GraspIt (Miller & Allen, 2004) and RRT for the grasping searching (including post-optimization) and motion planning respectively. We name it GraspIt+RRT. For the smoothness evaluation, we additionally perform linear interpolation between the beginning and ending joint positions generated by CGF and take it as a trivial baseline.

Implementation Details. For training CGF, we sample $2,000$ points on the object mesh as the input object point clouds. During training, we employ the Adam optimizer with the learning rate $5e-4$, where the learning rate is reduced by half per 500 epochs. The batch size is 32. The training takes 1000 epochs in total. The loss weights are $\lambda_q = 1$, $\lambda_j = 10$, $\lambda_{KL} = 1e-3$ and $\lambda_c = 50$. We sample $10,000$ latent codes for CGF and output the trajectories. We then pick the successful trajectories in the simulator to evaluate in the real world and count the success rate. For simulation experiments, environments are built upon the SAPIEN (Xiang et al., 2020) simulator.

For GraspTTA, we generate 200 grasp poses for each object. For CEM-MPC, we set popsize $M = 100$, time horizon $T = 5$, number of elites $e = 10$, and iterations $I = 2$. For GraspIt, we search for valid grasp poses with 100 different random seeds and use the contact energy as the objective function. For motion planning with RRT, we set 10,000 nodes in the tree and set step size $\epsilon = 0.01$ and probability of sampling $\beta = 0.5$.

4.2 Evaluation Metrics

Smoothness. To measure the continuity of the generated grasping, we propose to compare the smoothness. We first normalize joint positions by making all joints start at 0 and end at 1 in all trajectories. Then we calculate discrete first-order and second-order gradients, i.e. velocity and acceleration. Smoothness is defined as the sum of the $\mathcal{L}1$ distances of position, velocity, and acceleration between neighboring frames. Note that the linear interpolation is the upper bound with a joint position smoothness of 1.0.

Cost per successful trajectory in simulator. While sampling more grasps with a generative model or more random configurations with a planning algorithm could increase the probability of finding a successful trajectory, the issue of cost should not be neglected. At this point, it is important to evaluate the average cost of one successful trajectory. Therefore, we evaluate this average cost among baselines and our method in the simulator. Our method and GraspTTA+CEM-MPC require environment steps during the deployment and the cost is the number of steps per successful trajectory. GraspTTA+RRT requires collision checks during the planning and the cost is the number of collision checks per successful trajectory.

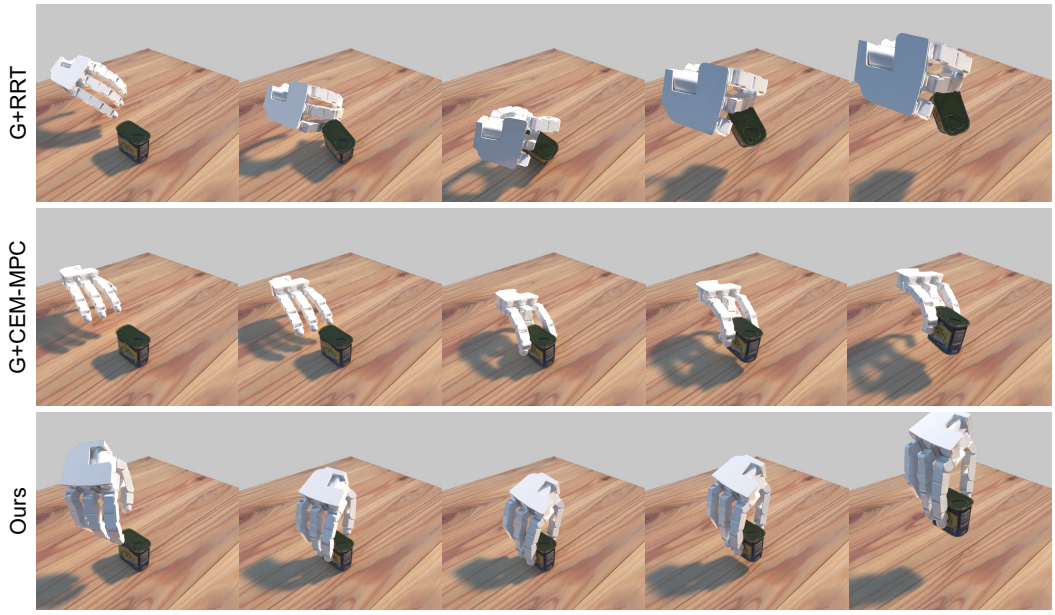

Figure 4: Qualitative evaluation in the simulation. Because of human demonstrations, our CGF generates a more natural and reasonable trajectory, which is helpful for the sim-to-real transfer. G is short for GraspTTA.

Method	Smoothness - Joints			Smoothness - Cartesian		
	Pos ↓	Vel (log) ↓	Acc (log) ↓	Pos ↓	Vel (log) ↓	Acc (log) ↓
GraspTTA (Jiang et al., 2021a) + RRT	**3.19**	3.40	6.43	7.82	3.80	6.83
GraspTTA (Jiang et al., 2021a) + CEM-MPC	6.96	3.09	5.50	13.18	3.33	5.74
Ours	9.77	**2.03**	**3.47**	**5.39**	**1.84**	**3.31**
Linear Interpolation	1.00	-3.58	-1.99	1.96	0.70	1.38

Table 1: For smoothness of joint positions and Cartesian coordinates, our CGF outperforms baselines by a large margin, which helps produce more natural trajectories and better control.

Success rate in the real world. We also evaluate the success rate in the real world. Note the successful trajectories in the simulator do not guarantee success in the real world because of the sim-to-real gap. For our method and all the baselines, we collect 20 successful trajectories, deploy them in the real world and count the success rate. This metric reflects the sim-to-real transfer ability.

4.3 Simulated Experimental Results

Smoothness evaluation. We compare the smoothness with three baselines and summarize the results in Tab. 1. For better comparison, the smoothness for velocity and acceleration is in the log form. For both joint positions and Cartesian coordinates, our CGF outperforms baselines by a large margin. Note that for the joint position smoothness, our method does not show an advantage over the baseline. This is due to that humans tend to follow a natural curve rather than the shortest path. Nevertheless, the improvement in the smoothness of velocity and acceleration plays an important role in producing natural trajectories, suppressing vibration, and achieving high accuracy control (Flash & Hogan, 1985; Piazzi & Visioli, 2000).

Success cost evaluation. The average cost per successful trajectory is shown in Tab. 2. For better comparison, the number of simulation steps and collision detection are in the log form. Since we directly execute the target joint positions generated from our CGF, through sampling a large number of trajectories, we can get a certain number of successful trajectories. Quantitatively, the average cost of getting a successful trajectory is significantly lower than the GraspTTA+RRT and GraspTTA+CEM-MPC. The main reasons are: (i) Our method does not require excessive exploration like MPC-CEM or sampling numerous configurations like RRT; (ii) Our method produces more natural trajectories than two-step methods, yielding a higher probability for finding a successful trajectory. We also evaluate the cost of unseen objects, although the cost increased a little, our method still surpasses the baselines. Although linear interpolation produces the smoothest trajectory, it never grasps an object successfully in the simulator.

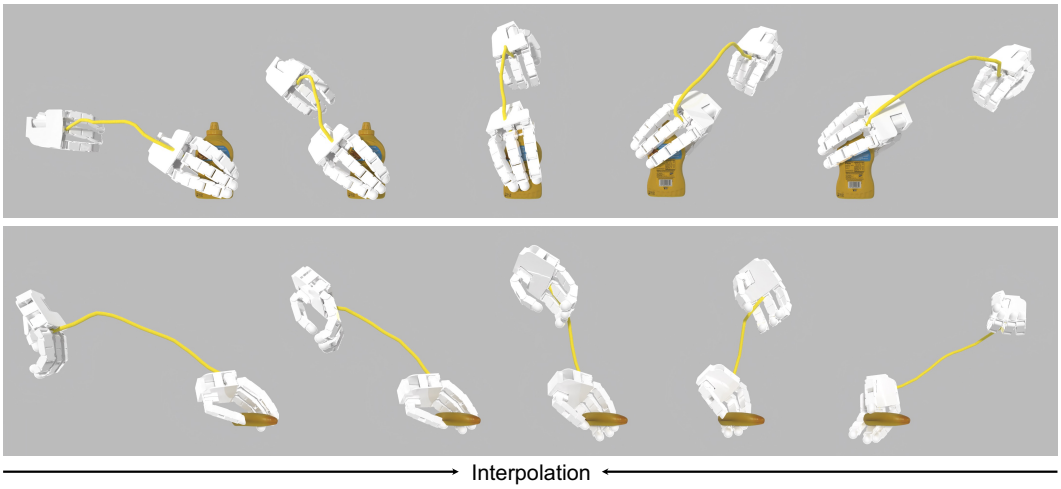

Figure 5: Grasping interpolation. We show the first frame and the last frame of the grasping trajectory. Yellow lines indicate the trajectory of the palm joint. Our method produces diverse grasping and the interpolation between them is also plausible. To the best of our knowledge, this result on interpolating both robot hand grasping pose and trajectory has not been shown before.

Method	Cost / Successful Trajectory (log) ↓ Seen Object	Unseen Object
GraspTTA (Jiang et al., 2021a) + RRT	5.41*	5.55*
GraspTTA (Jiang et al., 2021a) + CEM-MPC	5.96	6.16
Ours	**4.29**	**4.67**
Linear Interpolation	-	-

Table 2: Success cost evaluation. Since our method only requires fewer simulation steps to test a trajectory, the average cost for a successful trajectory is much lower than baselines. Note that we calculate the amount of collision detection for RRT.

Qualitative evaluation. We visualize the typical trajectories of the baselines and our method in Fig. 4. Since RRT is only planning the reachable target joint positions, the trajectory may not be natural. And a small perturbation in the execution process may cause it to collide with objects. CEM-MPC is not a long-time horizon method, which will make it fall into a local optimum quickly. As shown in Fig. 4, the middle finger is not in the ideal position. However, our CGF, learning from the human demonstration, could generate a much more natural and smooth trajectory. This is helpful for the sim-to-real transfer, which we will discuss in the next section.

Grasping Diversity. The ability to generate diverse outputs is one of the motivations for using CVAE. We perform interpolation in the latent space and show the results in Fig. 5. Our method produces diverse grasping and the interpolation between them is also plausible. We believe this is an interesting and potentially very useful property of our model. To the best of our knowledge, this result on interpolating both robot hand grasping pose and trajectory has not been shown before.

4.4 Real-World Robot Experiments

Setup. For the real-world robot experiments, we attached an Allegro hand on an X-Arm 6 robot. We select 6 real objects from YCB (Calli et al., 2015) which are *Banana*, *Bleach Cleanser*, *Ball Potted Meat Can*, *Tomato Soup Can* and *Mustard Bottle*, where the first 3 objects are unseen. The initial pose of the object is given and we use open-loop control for grasping.

Results with a Real Robot Hand. We evaluate the sim-to-real transfer ability of the trajectories generated by our method and baselines. For each object and method, we collect 20 successful trajectories in the simulator and deploy them in the real world. We report the success rate in Tab. 3. Although all the trajectories succeeded in the simulator, our method has a much higher success rate in the real world than the baselines. We believe there are two main reasons leading to better sim-to-real transfer ability: (i) Learning from human demonstration can lead to more natural behavior trajectory; (ii) The use of implicit function also helps provide a continuous and more smooth trajectory. In contrast, the motion planning used in the 2-step procedure baselines often generates unnatural trajectories which reduces the success rate when deploying in the real world. Specifically,

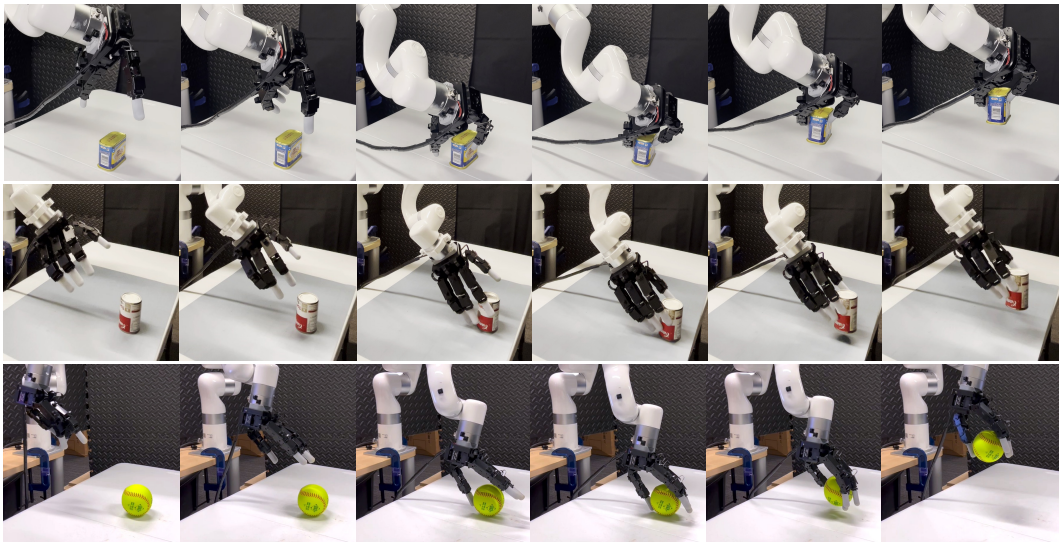

Figure 6: Real-world results on *Potted Meat Can*, *Tomato Soup Can* and *Ball*. Our CGF successfully transfers the simulation trajectory to the real robot.

Method	Success Rate (%) ↑					
	Banana	Cleanser	Meat Can	Soup Can	Bottle	Ball
GraspTTA (Jiang et al., 2021a) + RRT	10.0	15.0	10.0	5.0	5.0	10.0
GraspTTA (Jiang et al., 2021a) + CEM-MPC	10.0	10.0	15.0	5.0	0.0	15.0
GraspIt (Miller & Allen, 2004) + RRT	60.0	65.0	50.0	50.0	65.0	40.0
Ours	**70.0**	**85.0**	**70.0**	**80.0**	**85.0**	**65.0**

Table 3: Real-world experiments. For the successful trajectories in the simulation, our CGF has a higher success rate in the real world.

while GraspTTA (Jiang et al., 2021a) is able to generate grasp poses in the simulation, it does not ensure stable robotic grasping in the real world, and an unnatural trajectory approaching the grasp accumulates additional errors. On the other hand, GraspIt (Miller & Allen, 2004) is able to generate stable robot grasp pose in the real world, however, the 2-step procedure with it still performs worse than our method. We further provide a visualization of our successful trajectories in the real world in Fig. 6, which shows that our method could generate natural and human-like grasping.

4.5 Ablation Study

To demonstrate the advantages of continuous grasping, we ablate the augmented PD control (Sec. 3.4) which relies on the derivatives of CGF. We compare the performance of the augmented PD control to the default PD control in the simulator with different velocities and masses. Specifically, we set the speed to 1x, 2x and 3x and the mass of the robot to 1x, 2x and 3x. We combine them in pairs and report the augmented PD control's improvements on the success rate in Tab. 4. We show that the augmented PD control has improvements in all various cases, which proves its robustness.

	Mass 1x	Mass 2x	Mass 3x
Velocity 1x	5.66%	4.15%	7.10%
Velocity 2x	14.95%	7.73%	2.10%
Velocity 3x	4.29%	1.65%	16.77%

Table 4: Ablation study on the augmented PD control. We report improvements on the success rate of the augmented PD control over the default PD control with different velocities and masses.

5 Conclusion

We propose a novel implicit function named Continuous Grasping Function to generate smooth and dense grasping trajectories. CGF is learned in the framework of a Conditional Variational AutoEncoder using 3D human demonstrations. During inference, we sample various grasping plans in the simulator and deploy the successful ones to the real robot. By training on diverse human-object data, our method allows generalization to manipulate multiple objects. Compared to previous planning algorithms, CGF is more efficient and has a better sim-to-real generalization ability. We are committed to **releasing our code and environment**.

6 REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our work, we provide the following information in the main paper and appendix:

- Implementation details. We provide all implementation details and relevant hyper parameters of the proposed method and baselines in Sec. 4.1 and Appendix A, which are sufficient to reproduce our experiment results.
- Simulation environment. We describe our simulation environment settings, action space and success criteria in Appendix A.
- Real robot setup. We provide details on setting up real robot and conducting real-world experiments in Appendix B.

We are also committed to **releasing our code and environment**, which we believe is important to maintain an open research community.

REFERENCES

Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *IEEE Robotics and Automation Letters*, 7(2):4606–4613, 2022. 3

Sheldon Andrews and Paul G Kry. Goal directed multi-finger manipulation: Control policies and analysis. *Computers & Graphics*, 2013. 3, 5

Yunfei Bai and C Karen Liu. Dexterous manipulation using both palm and fingers. In *ICRA*, 2014. 2

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. 2, 3, 6

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. 6, 8

Berk Calli, Andrew Kimmel, Kaiyu Hang, Kostas Bekris, and Aaron Dollar. Path planning for within-hand manipulation over learned representations of safe states. In *International Symposium on Experimental Robotics*, pp. 437–447. Springer, 2018. 2

Yu-Wei Chao, Wei Yang, Yu Xiang, Pavlo Molchanov, Ankur Handa, Jonathan Tremblay, Yashraj S Narang, Karl Van Wyk, Umar Iqbal, Stan Birchfield, et al. Dexycb: A benchmark for capturing hand grasping of objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9044–9053, 2021. 3, 4, 6

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. 2, 3

Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8628–8638, 2021. 2

Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5939–5948, 2019. 4

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. 3

Sammy Christen, Muhammed Kocabas, Emre Aksan, Jemin Hwangbo, Jie Song, and Otmar Hilliges. D-grasp: Physically plausible dynamic grasp synthesis for hand-object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20577–20586, 2022. 3

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. 6

Nikhil Chavan Dafle, Alberto Rodriguez, Robert Paolini, Bowei Tang, Siddhartha S Srinivasa, Michael Erdmann, Matthew T Mason, Ivan Lundberg, Harald Staab, and Thomas Fuhlbrigge. Extrinsic dexterity: In-hand manipulation with external forces. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1578–1585. IEEE, 2014. 2

Mehmet R Dogar and Siddhartha S Srinivasa. Push-grasping with dexterous hands: Mechanics and a method. In *IROS*, 2010. 2

Emilien Dupont, Yee Whye Teh, and Arnaud Doucet. Generative models as distributions of functions. *arXiv preprint arXiv:2102.04776*, 2021. 2

Tamar Flash and Neville Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of neuroscience*, 5(7):1688–1703, 1985. 7

Guillermo Garcia-Hernando, Edward Johns, and Tae-Kyun Kim. Physics-based dexterous manipulations with estimated hand poses and residual reinforcement learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9561–9568. IEEE, 2020. 3

Patrick Grady, Chengcheng Tang, Christopher D Twigg, Minh Vo, Samarth Brahmbhatt, and Charles C Kemp. Contactopt: Optimizing contact to improve grasps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1471–1481, 2021. 3

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)*, pp. 3786–3793. IEEE, 2016. 3

Kaijen Hsiao and Tomas Lozano-Perez. Imitation learning of whole-body grasps. In *2006 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5657–5662. IEEE, 2006. 3

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. 2

Hanwen Jiang, Shaowei Liu, Jiashun Wang, and Xiaolong Wang. Hand-object contact consistency reasoning for human grasps generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11107–11116, 2021a. 3, 6, 7, 8, 9

Zhenyu Jiang, Yifeng Zhu, Maxwell Svetlik, Kuan Fang, and Yuke Zhu. Synergies between affordance and geometry: 6-dof grasp detection via implicit representations. *arXiv preprint arXiv:2104.01542*, 2021b. 3

Daniel Kappler, Jeannette Bohg, and Stefan Schaal. Leveraging big data for grasp planning. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4304–4311. IEEE, 2015. 3

Korrawe Karunratanakul, Jinlong Yang, Yan Zhang, Michael J Black, Krikamol Muandet, and Siyu Tang. Grasping field: Learning implicit representations for human grasps. In *2020 International Conference on 3D Vision (3DV)*, pp. 333–344. IEEE, 2020. 3

Vikash Kumar, Abhishek Gupta, Emanuel Todorov, and Sergey Levine. Learning dexterous manipulation policies from experience and imitation. *arXiv preprint arXiv:1611.05095*, 2016a. 3

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, 2016b. 2

Steven M LaValle et al. Rapidly-exploring random trees: A new tool for path planning. 1998. 6

Xueting Li, Shalini De Mello, Xiaolong Wang, Ming-Hsuan Yang, Jan Kautz, and Sifei Liu. Learning continuous environment fields via implicit functions. *arXiv preprint arXiv:2111.13997*, 2021. 3

Yunzhu Li, Shuang Li, Vincent Sitzmann, Pulkit Agrawal, and Antonio Torralba. 3d neural scene representations for visuomotor control. In *Conference on Robot Learning*, pp. 112–123. PMLR, 2022. 3

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. 2, 3

Jan Marian Maciejowski. *Predictive control: with constraints*. Pearson education, 2002. 6

Priyanka Mandikal and Kristen Grauman. Learning dexterous grasping with object-centric visual affordances. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6169–6176. IEEE, 2021. 2

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4460–4470, 2019. 2, 4

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pp. 405–421. Springer, 2020. 2, 3

Andrew T Miller and Peter K Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004. 6, 9

Douglas Morrison, Juxi Leitner, and Peter Corke. Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. In *Robotics: Science and Systems*, 2018. doi: 10.15607/RSS.2018.XIV.021. 3

Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pp. 1101–1112. PMLR, 2020. 2

Allison M Okamura, Niels Smaby, and Mark R Cutkosky. An overview of dexterous manipulation. In *ICRA*, 2000. 2

OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafał Józefowicz, 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. *arXiv*, 2018. 2, 3

OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving rubik's cube with a robot hand. *arXiv*, 2019. 2

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 165–174, 2019. 2, 4

Aurelio Piazzi and Antonio Visioli. Global minimum-jerk trajectory planning of robot manipulators. *IEEE transactions on industrial electronics*, 47(1):140–149, 2000. 7

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017. 4

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. 2, 3

Yuzhe Qin, Hao Su, and Xiaolong Wang. From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation. *arXiv preprint arXiv:2204.12490*, 2022. 3, 4

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. 3

Reuven Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and computing in applied probability*, 1(2):127–190, 1999. 6

Daniela Rus. In-hand dexterous manipulation of piecewise-smooth 3-d objects. *The International Journal of Robotics Research*, 1999. 2

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. 2

Karl Schmeckpeper, Oleh Rybkin, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Reinforcement learning with videos: Combining offline observations with interaction. *arXiv*, 2020. 3

Lin Shao, Toki Migimatsu, Qiang Zhang, Karen Yang, and Jeannette Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. In *RSS*, 2020. 3

Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, Joshua B Tenenbaum, Alberto Rodriguez, Pulkit Agrawal, and Vincent Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. *arXiv preprint arXiv:2112.05124*, 2021. 3

Aravind Sivakumar, Kenneth Shaw, and Deepak Pathak. Robotic telekinesis: learning a robotic hand imitator by watching humans on youtube. *arXiv preprint arXiv:2202.10448*, 2022. 3

Kihyuk Sohn, H. Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *NIPS*, 2015a. 2

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015b. 4

Omid Taheri, Nima Ghorbani, Michael J Black, and Dimitrios Tzionas. Grab: A dataset of whole-body human grasping of objects. In *European conference on computer vision*, pp. 581–600. Springer, 2020. 3

Omid Taheri, Vasileios Choutas, Michael J Black, and Dimitrios Tzionas. Goal: Generating 4d whole-body motion for hand-object grasping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13263–13273, 2022. 3

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. 2, 3

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*, 7 (2):1659–1666, 2022. 3

Yan Wu, Jiahao Wang, Yan Zhang, Siwei Zhang, Otmar Hilliges, Fisher Yu, and Siyu Tang. Saga: Stochastic whole-body grasping with contact. *arXiv preprint arXiv:2112.10103*, 2021. 3

Yueh-Hua Wu, Jiashun Wang, and Xiaolong Wang. Learning generalizable dexterous manipulation from human grasp affordance. *arXiv preprint arXiv:2204.02320*, 2022. 3, 6

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11097–11107, 2020. 6, 15

Lixin Yang, Xinyu Zhan, Kailin Li, Wenqiang Xu, Jiefeng Li, and Cewu Lu. Cpf: Learning a contact potential field to model the hand-object interaction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11097–11106, 2021. 3

Yuting Ye and C Karen Liu. Synthesis of detailed hand manipulations using contact sampling. *ACM Transactions on Graphics (TOG)*, 31(4):1–10, 2012. 3

Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. Visual imitation made easy. *arXiv*, 2020. 3

He Zhang, Yuting Ye, Takaaki Shiratori, and Taku Komura. Manipnet: neural manipulation synthesis with a hand-object spatial representation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 3

Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5745–5753, 2019. 15

A Implementation Details

Environment. Environments are built upon the SAPIEN (Xiang et al., 2020) simulator with timestep set to 0.004 and frame skip set to 5. The action space is the motor commands of the 22 actuators on the robot's hand. The first 6 motors control the global position and orientation of the robot while the last 16 motors control the fingers of the hand. The success criteria for our grasping task are (i) the object and the hand are in contact and (ii) the object is lifted by 4 cm.

Network Architecture. We show our network architectures during training and testing in Tab. 5 and Tab. 6 respectively. During training, the object point cloud and 20 frames of hand joint positions are utilized as inputs. Then a PointNet encoder and an MLP encoder are employed to extract object feature $\mathcal{F}^o$ and a set of hand features $\{\mathcal{F}_t^h\}$ respectively. Then all features are concatenated and passed to the CVAE encoder to predict the posterior distribution $\mathcal{Q}(z|\mu, \sigma^2)$. Then, a latent code z is sampled from the distribution, and concatenated with the object feature $\mathcal{F}^o$ and query time t as the input of CGF for regressing the hand joint positions. Note that the global rotation (in the form of axis angle) is converted to the continuous 6d representation (Zhou et al., 2019). In the end, a forward kinematics layer is utilized to convert the joint positions to the Cartesian coordinates.

Stage	Configuration	Output
0	Input object point cloud $\mathcal{P}^o$	2000×3
	Input joint positions $\{q_t\}$	20×22
	Feature extraction	
1	PointNet encoder	1024 ($\mathcal{F}^o$)
	MLP encoder $(22, 256, 256, 64)$	20×64 ($\{\mathcal{F}_t^h\}$)
	Calculating distribution (Input *concat*($\{\mathcal{F}_t^h\}$, $\mathcal{F}^o$))	
2	CVAE encoder $(2304, 256, 256)$	256 (μ)
		256 (σ^2)
	Latent code sampling	
3	Sampling from $\mathcal{Q}(z\|\mu, \sigma^2)$	256 (z)
	Hand reconstruction (Input *concat*($\mathcal{F}^o$, z, t))	
4	Continuous Grasping Function $(1281, 512, 256, 25)$	25 ($\hat{q}_t$)
4	Forward Kinematics Layer	15×3 ($\mathcal{J}_i(\hat{q}_t)$)

Table 5: Training time architecture.

Stage	Configuration	Output
0	Input object point cloud $\mathcal{P}^o$	2000×3
	Feature extraction	
1	PointNet Encoder	1024 ($\mathcal{F}^o$)
	Latent code sampling	
3	Sampling from $\mathcal{N}(z\|0, 1)$	256 (z)
	Hand reconstruction (Input *concat*($\mathcal{F}^o$, z, t))	
4	Continuous Grasping Function $(1281, 512, 256, 25)$	25 ($\hat{q}_t$)
4	Forward Kinematics Layer	15×3 ($\mathcal{J}_i(\hat{q}_t)$)

Table 6: Test-time architecture.

During inference, the latent code z is randomly sampled from the standard Gaussian distribution $\mathcal{N}(z|0, 1)$. Hence the hand joint positions and the CVAE encoder are not required.

B System Setup for Real-World Experiments

We attach an Allegro Hand to an X-Arm6 robot for deployment. Allegro Hand is a 16-DoF anthropomorphic hand with four fingers and XArm6 is a 6-DoF robot arm. The system setup is shown in Fig. 7. 6 YCB objects are chosen for real-world experiments (see Fig. 8).

Figure 7: System setup.

Figure 8: Selected YCB objects.

C QUALITATIVE EVALUATION ON UNSEEN OBJECTS

Given the point cloud of unseen objects, we also sample grasping trajectories and test them in the simulator and then in the real world. We visualize the successful grasping trajectories in Fig. 9, which demonstrates the generalization ability of our CGF.

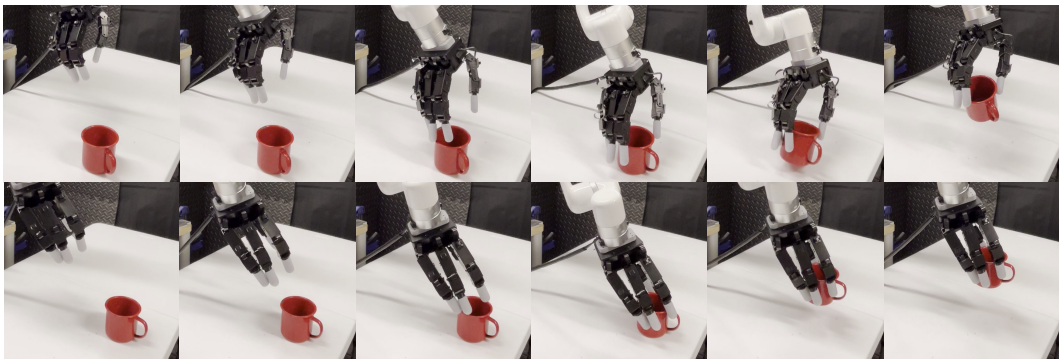

Figure 9: Real-world results on *Mug*. Our CGF allows generalization to manipulate unseen objects.