Supplementary Material for "Learning Diverse and Physically Feasible Dexterous Grasps with Generative Model and Bilevel Optimization"

Albert Wu
Computer Science Department
Stanford University, United States
amhwu@stanford.edu

Michelle Guo
Computer Science Department
Stanford University, United States
mguo95@cs.stanford.edu

C. Karen Liu
Computer Science Department
Stanford University, United States
karenliu@cs.stanford.edu

1 Method

1.1 Polyhedral Approximation of the Friction Cone

The friction cone can be modeled as in Equation (1).

$$0 \leq -\boldsymbol{f}_i \cdot \hat{\boldsymbol{n}}_i \quad \text{and} \quad \|\boldsymbol{f}_i - (\boldsymbol{f}_i \cdot \hat{\boldsymbol{n}}_i)\hat{\boldsymbol{n}}_i\|_2 \leq -\mu \boldsymbol{f}_i \cdot \hat{\boldsymbol{n}}_i. \tag{1}$$

The friction force direction may be approximated with a polyhedral cone spanned by a set of unit vectors perpendicular to $\hat{\boldsymbol{n}}_i$: $\left\{\hat{\boldsymbol{t}}_{i,j} \mid \hat{\boldsymbol{t}}_{i,j} \perp \hat{\boldsymbol{n}}_i, j \in \{1, 2, \dots\}\right\}$ ([1]). In this work, we chose an arbitrary orthogonal basis on the tangent plane of the contact point $(\hat{\boldsymbol{t}}_{i,1}, \hat{\boldsymbol{t}}_{i,2})$, which yields the following approximated friction cone constraints

$$0 \leq -\boldsymbol{f}_i \cdot \hat{\boldsymbol{n}}_i \quad \text{and} \quad |\boldsymbol{f}_i \cdot \hat{\boldsymbol{t}}_{i,j}| \leq -\mu \boldsymbol{f}_i \cdot \hat{\boldsymbol{n}}_i, \forall j \in \{1, 2\}. \tag{2}$$

1.2 Enforcing the Finger-Object Distance Constraint

The finger-object distance constraint $K_i(\boldsymbol{q}') \in \partial \boldsymbol{O}$ is implemented as a constraint on the signed distance $d_{min} \leq D(K_i(\boldsymbol{q}'), \boldsymbol{O}) \leq d_{max}$, where D is defined as

$$D(\boldsymbol{p}, \boldsymbol{O}) \triangleq \begin{cases} \min_{\boldsymbol{p}_o \in \partial \boldsymbol{O}} \|\boldsymbol{p}_o - \boldsymbol{p}\|_2, & \text{if } \boldsymbol{p} \notin \boldsymbol{O} \\ -\min_{\boldsymbol{p}_o \in \partial \boldsymbol{O}} \|\boldsymbol{p}_o - \boldsymbol{p}\|_2, & \text{otherwise.} \end{cases} \tag{3}$$

As the fingertips of the allegro hand are compliant, we chose $d_{min} = -0.68$cm and $d_{max} = -0.32$cm based on the thickness of the compliant material.

1.3 Creating a Physically Feasible Dexterous Grasping Dataset

We generate physically feasible grasps of six YCB objects [2] in simulation. The chosen objects are "cracker box," "sugar box," "tomato soup can," "mustard bottle," "gelatin box," and "potted meat can." We consider a realistic scenario where the object is placed on a flat surface instead of free floating. As such, we need to consider the variations due to different rest poses of on the surface in addition to finger placements. The following summarizes the steps for data generation:

1. Simulate dropping the object on a flat surface from different initial poses to find "rest poses".

6th Conference on Robot Learning (CoRL 2022), Auckland, New Zealand.

2. For each rest pose, produce 256 candidate contact points distributed across the object with Poisson disk sampling.
3. Remove any point in contact with the surface $\boldsymbol{E}$ and store the remaining as a point cloud $\hat{\boldsymbol{O}}$.
4. Evaluate Equation (4) for each of the C_3^{256} 3-point finger placements. Discard if it is not dynamically feasible.

$$\begin{aligned} \min_{\boldsymbol{f}_1,\boldsymbol{f}_2,\boldsymbol{f}_3} \quad & \|\sum_{i=1}^{3} \boldsymbol{f}_i\|_2^2 + \|\sum_{i=1}^{3} \boldsymbol{p}_i \times \boldsymbol{f}_i\|_2^2 \\ \text{subject to} \quad & 0 < f_{min} \leq -\boldsymbol{f}_i \cdot \hat{\boldsymbol{n}}_i, \;\; |\boldsymbol{f}_i \cdot \hat{\boldsymbol{t}}_{i,j}| \leq -\mu \boldsymbol{f}_i \cdot \hat{\boldsymbol{n}}_i, \;\; \forall i \in \{1,2,3\}, \forall j \in \{1,2\}. \end{aligned} \tag{4}$$

5. For each dynamically feasible finger placement, permute the $3! = 6$ possible assignments for the thumb, index finger, and middle finger. Check each assignment for kinematic feasibility. If a feasible grasping pose $\boldsymbol{q}$ can be found for a finger placement $\bar{\mathcal{P}}$, add $(\hat{\boldsymbol{O}}, \boldsymbol{q}, \bar{\mathcal{P}})$ to the dataset $\mathbb{P}$.
6. At training time, augment the dataset by randomly translating and varying the yaw angle for each grasp in the dataset.

2 Software

2.1 Implementation Details

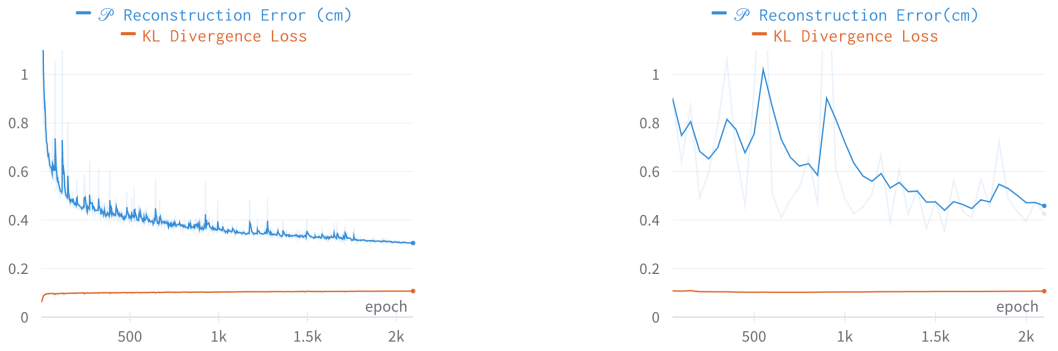
The CVAE is implemented using PyTorch [3] and trained on a Google Cloud virtual machine instance with 16 NVIDIA Tesla A100 GPUs. The selected model took approximately 76 hours to train.

The bilevel optimization pipeline is implemented using a mixture of automatic differentiation tools from Drake [4], OptNet [5], and Pytorch [3].

Our source code is available at `github.com/Stanford-TML/dex_grasp`.

2.2 CVAE Training Process

Figure 1 shows the training curve. Based on the test losses, we chose the model after 1550 epochs of training for all subsequent experiments.



(a) CVAE Training Loss.

(b) CVAE Testing Loss.

Figure 1: Training and testing loss of the CVAE. The model was able to achieve sub-centimeter reconstruction accuracy of the fingertip placement. Based on the testing results, the model weights at 1550 training epochs were selected for all subsequent experiments.

3 Hardware

3.1 Robot Control

The Franka arm was controlled using Polymetis [6]. The Allegro hand was commanded with an open source driver [7].

3.2 Scene observation

We collect depth images from the four calibrated cameras to obtain a point cloud of the tabletop scene in the robot base frame. To obtain a segmented object point cloud, we remove tabletop points (via plane fitting), crop points within a predefined axis-aligned bounding box (roughly corresponding to a small region near the center of the table), and filter the point cloud to remove outliers [8]. Finally, we extract a mesh from the object point cloud by computing its convex hull.

3.3 CVAE-based methods

For CVAE-based methods, which require sampling latent variables, we utilize unique random seeds across trials. Within each trial, we sample latent variables until we encounter the first successful grasp in simulation (kinematic and dynamic constraints satisfied). This is the grasp that we execute and evaluate on hardware. Each sampled latent variable attempts six IK initializations (orientations) before the next latent variable is sampled.

3.4 Challenges Stemming from Selected Object Shapes

The sandwich box is challenging to parallel jaw grippers as it is triangular. The (rigid) massage ball is challenging to suction-based grippers as the surface is highly irregular. The castle object requires a side grasp due to its cone-shaped top. Slender objects such as lego and hairspray bottle also benefit from a side grasp as there is more room for finger placement along the vertical axis compared to a horizontal cross section. The tetrahedron is difficult for any finger-based gripper to achieve force closure due to its shape, and a large squeezing force is necessary.

3.5 Hardcoded Baseline Design

The grasp finger joint angles are manually specified. The wrist translation is defined as $x = \frac{1}{N}\sum_{i=1}^{N} x_i + x_{\text{offset}}$, $y = \frac{1}{N}\sum_{i=1}^{N} y_i + y_{\text{offset}}$, and $z = \max_i z_i + z_{\text{offset}}$, where $(x_{\text{offset}}, y_{\text{offset}}, z_{\text{offset}})$ are hand-tuned offsets and (x_i, y_i, z_i) is the ith point in the point cloud containing N points. The orientation of the wrist is aligned with the short axis obtained from performing principal component analysis on the x-y projection of the observed point cloud.

4 Results

4.1 Grasp Trial Statistics

Tables 1, 2, 3 summarize the detailed success rates on each of the 20 test objects.

Table 1: Seen objects grasp statistics.

	Sugar box	Soup Can	Mustard Bottle	Overall	**Success %**
Ours	6/6	6/6	5/6	17/18	**94.4**
CVAE only	3/6	3/6	1/6	7/18	**38.9**
CVAE+kinematics	6/6	6/6	4/6	16/18	**88.9**
Hardcoded	3/3	2/3	3/3	8/9	**88.9**

Table 2: Familiar objects grasp statistics.

	Webcam	Mask	Chips	Soda	Overall	**Success %**
Ours	6/6	6/6	5/6	6/6	23/24	**95.8**
Hardcoded	3/3	3/3	2/3	1/3	9/12	**75.0**

Table 3: Novel objects grasp statistics.

	Glasses case	Ramen	Pill bottle	Sandwich (upright)	Massage ball
Ours	4/6	6/6	5/6	6/6	4/6
Hardcoded	3/3	3/3	1/3	3/3	3/3
	Castle	Tetrahedron	Hairspray bottle	Pear	Alcohol bottle
Ours	6/6	1/6	6/6	5/6	6/6
Hardcoded	0/3	0/3	0/3	0/3	1/3
	Condiment bottle	Sandwich (side)	Lego	**Overall**	**Success %**
Ours	4/6	6/6	5/6	**64/78**	**82.1**
Hardcoded	1/3	0/3	0/3	**15/39**	**38.5**

4.2 Quantitative Evaluation of Physical Constraints

We provide quantitative physical constraint evaluations on grasps planned by our method. The evaluations are reported for each of the object categories. Table 4 summarizes the signed distance (Equation (3)) between the fingertips to the fitted object meshes. The distances should satisfy $D(\boldsymbol{p}, \boldsymbol{O}) \in [d_{min}, d_{max}] = [-0.68, -0.32]$. Table 5 summarizes the force and torque ratios, which should both be zero under wrench closure.

Table 4: Evaluation of finger-object distance constraint on grasps planned by our method.

	Median	Min.	Max.	Max violation
Seen	-0.32	-0.68	-0.24	0.08
Familiar	-0.32	-0.68	-0.32	0.00
Novel	-0.32	-0.68	-0.24	0.08
Overall	-0.32	-0.68	-0.24	0.08

Table 5: Force and torque ratios of the grasps generated by our method. All values are reported as *median, (25th percentile, 75th percentile)*.

	Force ratio	Torque ratio
Seen	0.00 (0.00, 0.01)	0.01 (0.00, 0.27)
Familiar	0.00 (0.00, 0.00)	0.00 (0.00, 0.00)
Novel	0.00 (0.00, 0.03)	0.01 (0.00, 8.48)
Overall	0.00 (0.00, 0.02)	0.01 (0.00, 2.68)

4.3 Timing Statistics

In Table 6, we report the runtime and repeats of each stage in our pipeline during grasp planning.

Table 6: Timing statistics of our method, shown as *median, (25th percentile, 75th percentile)*.

	CVAE time (s)	CVAE repeats	IK time (s)	IK repeats	BO time (s)	BO repeats	**Total Time (s)**
Seen	0.95 (0.94, 0.97)	2 (1, 3.75)	13.02 (6.64, 18.88)	2 (1, 4)	1.99 (0.85, 2.90)	1.5 (1, 2.75)	**14.99 (9.04, 23.60)**
Familiar	0.95 (0.93, 0.97)	2 (1, 4)	10.91 (3.91, 21.52)	2 (1, 4)	0.78 (0.50, 2.25)	1.5 (1, 2.25)	**13.73 (5.49, 31.68)**
Novel	0.95 (0.93, 0.97)	2 (1, 4)	8.07 (1.47, 20.55)	2 (1,4)	2.61 (1.05, 5.35)	2 (1, 3)	**14.40 (5.59, 34.85)**
Overall	**0.95 (0.93, 0.97)**	**2 (1, 4)**	**9.81 (2.16, 20.07)**	**2 (1,4)**	**2.22 (0.82, 4.03)**	**2 (1,3)**	**14.40 (5.67, 34.71)**

References

[1] D. E. Stewart and J. C. Trinkle. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International Journal for Numerical Methods in Engineering*, 39(15):2673–2691, 1996.

[2] B. Calli, A. Singh, J. Bruce, A. Walsman, K. Konolige, S. Srinivasa, P. Abbeel, and A. M. Dollar. Yale-cmu-berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 36(3):261–268, 2017.

[3] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[4] R. Tedrake and the Drake Development Team. Drake: Model-based design and verification for robotics, 2019. URL `https://drake.mit.edu`.

[5] B. Amos and J. Z. Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pages 136–145. PMLR, 2017.

[6] Y. Lin, A. S. Wang, G. Sutanto, A. Rai, and F. Meier. Polymetis. https://facebookresearch.github.io/fairo/polymetis/, 2021.

[7] Allegro hand linux project. https://github.com/simlabrobotics/allegro_hand_linux_v4.

[8] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.