Embodiment Part	Finger Tip	Finger Link	Palm
Link No.	tip_1, tip_2, tip_3, tip_4	1,2,3,5,6,7,9,10,11,14,15	palm_link
Number of Contact Candidates / each	96	16	128

TABLE II: Number of contact candidates on different parts of the Allegro hand. We specify potential contacts all over the hand to encourage whole-hand (especially palm) nonprehensile manipulation on the object.

APPENDIX

A. Dataset Generation and Statistical Analysis

Parameter	Value
w_{fc}	0.5
w_{dis}	500
w_{pen}	300.0
w_{spen}	100.0
w_{joints}	1.0
w_{ff}	3.0
w_{fp}	0.0
w_{tpen}	100.0
$w_{\text{direction}}$	200.0
$w_{\text{kinematics}}$	100.0

TABLE III: Weight parameters.

Parameter	Value
Switch Possibility	0.5
μ	0.98
Step Size	0.005
Stepsize Period	50
Starting Temp.	18
Annealing Period	30
Temp. Decay	0.95

TABLE IV: Optimization hyperparameters.

During dataset generation, we specify the contact candidates according to Figure 11 and Table II, and we set the weight parameters (from Eq. 1) according to values listed in Table III. For the optimization we discussed in Sec. IV-A, the detailed hyperparameters are in Table IV.

In the original hand pose generation procedure, we mainly consider the object geometry and encourage contact between selected contact candidates all over the hand and the object surface. However, it is crucial to test pushing to validate the quality of the nonprehensile hand poses. Initially, we obtain a low success rate of all generated hand poses, so we augment each successful hand pose 10 times. These perturbations involve small changes in rotation (max 2.5 deg), translation (max 0.005 m) and joint pose (0.05 rad) using a Halton sequence. Figure 12 shows an example of a random original hand pose (lightblue color) and 4 different perturbed hand poses (lightyellow color). By doing so, we get a large dataset of only successful hand poses, which we use for training the diffusion model.

Figure 13 shows the distribution of joint angle values across our dataset. Most joints span the full range between their lower and upper bounds, and tend to have one or several modes. Those modes may lead to "general" stable hand poses for pushing motions. Other joint values may vary depending on particular object geometries. Figure 14 shows a breakdown of object categories and the frequency of the top 20 objects in our dataset.

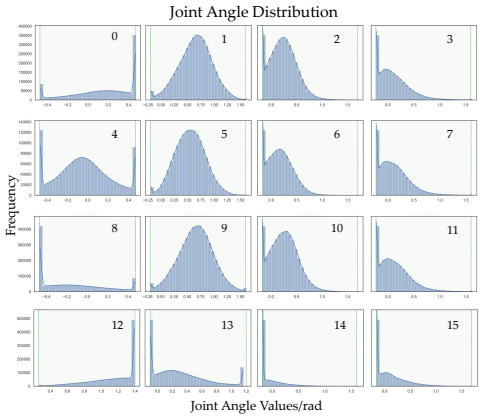

Fig. 13: Visualization of the distribution of joint angle values in our proposed dataset, demonstrating the diversity of our generated hand poses. The number on the top right corner of each subfigure indicates the joint index. The *green dashed lines* on the edge of x-axis indicate the lower/upper bounds of each joint angle values.

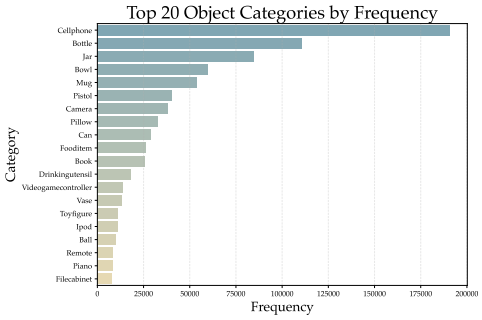

Fig. 14: Visualization of the top 20 objects in terms of pushing hand poses frequency in our proposed dataset.

B. Training Details

We train our model with one NVIDIA 4090 GPU on a desktop. Detailed training and model parameters are shown in Table V. We also show the training curves with training loss and validation loss on different scales of the dataset in Figure 15, which is relevant to our experiments in Sec. V-A.

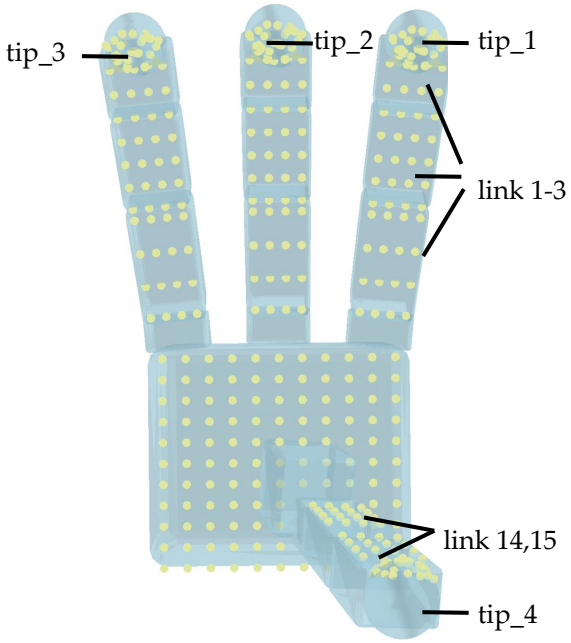

Fig. 11: Contact candidates on the Allegro hand. Refer to Table II for the number of contacts on each link.

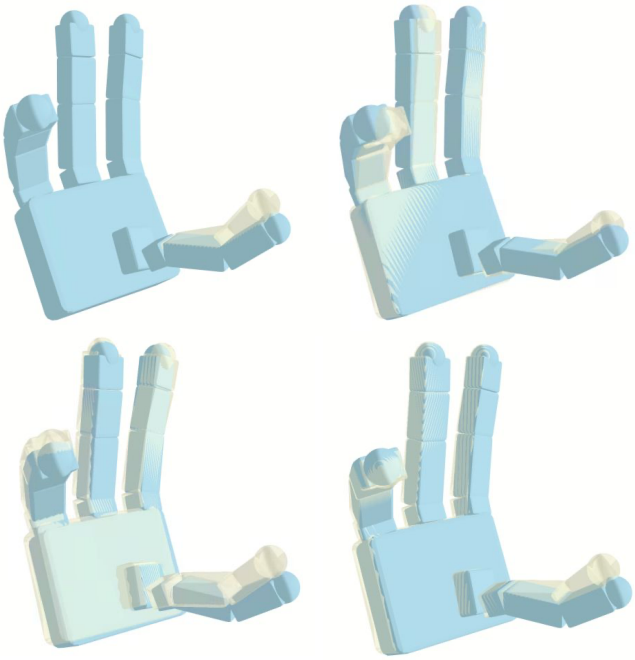

Fig. 12: A visualization of an example of augmentations. *Lightyellow* indicates the hand pose with the perturbation, and *lightblue* is the original one.

Component	Parameter	Default / value
`Data Config`	`observation_dim`	4096
	`pushingpose_dim`	25
`Model Config`	`name`	ConditionalUnet1D
	`input_dim`	25
	`global_cond_dim`	4096
`DDPM Scheduler`	`beta_schedule`	squaredcos_cap_v2
	`clip_sample`	`True`
	`num_diffusion_timesteps`	100
	`prediction_type`	epsilon
`Training Config`	`batch_size`	16
	`n_epochs`	200
	`print_freq`	10
	`snapshot_freq`	25
`Optim Config`	`optimizer`	Adam
	`lr`	1×10^{-4}
	`weight_decay`	1×10^{-6}
	`beta1`	0.9
	`amsgrad`	`False`
	`eps`	1×10^{-8}
	`grad_clip`	1.0
`lr Scheduler`	`name`	cosine
	`num_warmup_steps`	500
`EMAModel`	`power`	0.75

TABLE V: Configuration and training hyperparameters of the diffusion model.

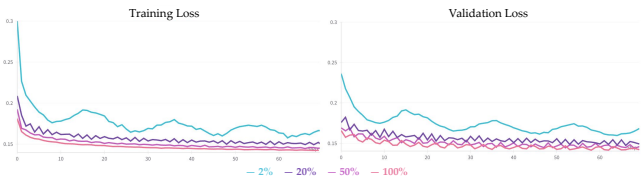

Fig. 15: Training curves on different scales of the dataset. See Sec. V-A for more discussion.

C. Experiment Details

Our physical experiment setup consists of a Franka Panda manipulator equipped with an Allegro Hand, as shown in Figure 16. We also place an L515 RealSense camera above the table, which is *only* used for path planning in multi-step planning experiments in Sec. V-C and Sec. F. The surface we use for all experiments is a commercially available product purchased from Amazon (product_link). Since our

focus is on nonprehensile hand pose generation, we assume that the surface's friction properties are sufficient to support pushing interactions. We leave a more detailed investigation of how physical properties influence dexterous nonprehensile manipulation as future work.

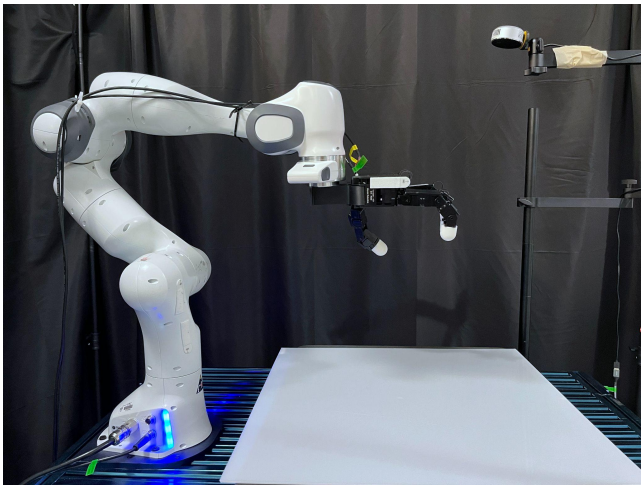

Fig. 16: Our physical experiment setup including a Frank Panda robot with an attached Allegro Hand. The camera is only used for high-level path planning.

We select 8 3D-printed objects and 6 real-world objects, covering flat, volumetric, and tall objects, as shown in Figure 17. Each object presents unique challenges for pushing. For example, when the robot hand approaches flat objects (e.g., Cake, Cookie Box) it may risk colliding with the table. In addition, tall objects (e.g., Lamp, Spray) frequently topple during pushing due to a high center of mass. While our method also suffers from these failure modes (particularly object toppling), it outperforms baselines, which topple objects more frequently. This motivates our case study on using a fixed hand pose to push objects taller than 20 cm. While fixed hand poses can reliably work for objects with simple geometries, they frequently fail on these taller objects. As discussed in Sec. V-C, our results highlight the need for hand poses that provide more stable object support for transporting.

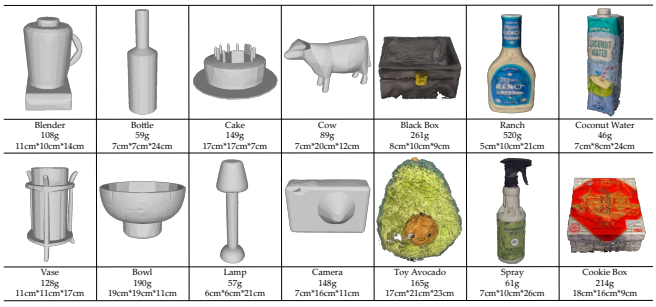

Fig. 17: 3D meshes, mass and physical dimensions of all objects tested in real-world experiments. Dimensions are listed as (x, y, z).

We list the number of successful trials out of 5 for each method and direction in Table VI. A blank entry (-) indicates that the robot could not execute the motion due to kinematic infeasibility. While GD2P has marginally more infeasible trials than the baselines, this is expected because GD2P generates diverse hand orientations beyond top-down poses. All methods execute pushes for 20 cm, which is relatively long within the robot's workspace, and this can be infeasible for many hand poses. In contrast, the Pre-Trained Grasp Pose baseline tends to result in consistently top-down hand poses, which are generally easier to execute due to reachability and kinematic constraints. Despite counting all kinematically infeasible trials as failures, GD2P outperforms the baseline methods, demonstrating its robustness on pushing or pulling tasks.

D. More Successful Rollouts

We provide additional example visualizations of successful rollouts of GD2P in Figure 18. For videos, please refer to our website: dexnoma.github.io.

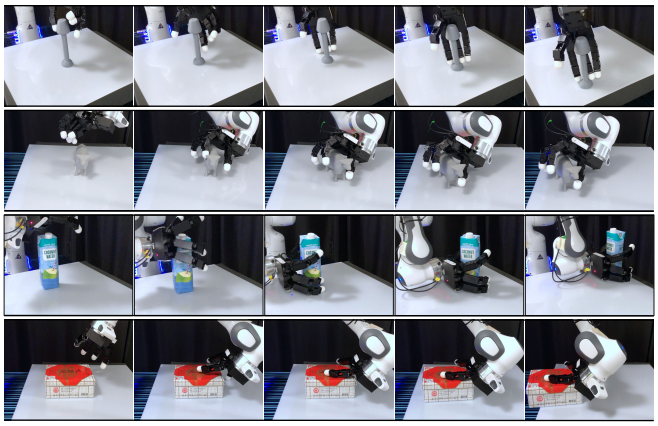

Fig. 18: Successful rollouts of GD2P, one per row.

E. Results and Analysis of Baseline Methods

We visualize 3 examples of the nearest neighbor (NN) retrieval results and the trained NeRF representation in Figure 19. The retrieved NN objects are similar in shape and scale of the query object (left 3 columns in Figure 19). However, their coarse geometry granularity is insufficient to generate robust hand poses. For example, with the *Toy Avocado*, our method selects a hand pose that pushes from the bottom to avoid sliding or toppling. In contrast, the NN method retrieves a vase-like object, where pushes from the middle make more sense. The irregular geometric shape at the bottom of the vase-like object could potentially cause more collisions and may increase the difficulty of solving the kinematics. The right 3 columns in Figure 19 visualize the NeRF input to the Pre-Trained Grasp Pose method, since we use their pre-trained model taking in NeRF representations. Though a common failure mode of the pre-trained grasp pose is that the object slips from the hand because the palm is oriented at an improper angle, we observe notable visual noise in the NeRF representation, which may also deteriorate performance of this baseline. For more discussions of baseline performance, see Sec. V-C.

	GD2P			GD2P w/o Ranking			Nearest Neighbor			Pre-Trained Grasp Pose		
	Dir.1	Dir.2	Dir.3	Dir.1	Dir.2	Dir.3	Dir.1	Dir.2	Dir.3	Dir.1	Dir.2	Dir.3
Blender	5/5	4/5	4/5	3/5	3/5	5/5	2/5	2/5	2/5	1/5	1/5	1/5
Vase	5/5	3/5	4/5	2/5	4/5	4/5	4/5	4/5	3/5	2/5	3/5	2/5
Bottle	4/5	4/5	5/5	3/5	3/5	3/5	0/5	4/5	3/5	3/5	2/5	2/5
Bowl	4/5	1/5	-	4/5	1/5	-	2/5	2/5	1/5	3/5	2/5	2/5
Cake	4/5	3/5	4/5	4/5	4/5	3/5	3/5	1/5	1/5	1/5	0/5	1/5
Lamp	1/5	1/5	1/5	2/5	2/5	2/5	1/5	0/5	0/5	0/5	1/5	1/5
Cow	5/5	3/5	3/5	3/5	2/5	3/5	1/5	1/5	1/5	0/5	3/5	2/5
Camera	2/5	2/5	4/5	2/5	3/5	3/5	1/5	1/5	3/5	1/5	4/5	2/5
3D Avg./ %	67.5	52.5	62.5	57.5	55.0	57.5	35.0	37.5	35.0	27.5	40.0	32.5
Black Box	4/5	4/5	3/5	3/5	1/5	2/5	1/5	1/5	2/5	3/5	3/5	2/5
Toy Avocado	4/5	-	1/5	3/5	-	2/5	-	-	1/5	3/5	0/5	4/5
Ranch	3/5	2/5	3/5	4/5	1/5	2/5	3/5	1/5	4/5	1/5	-	2/5
Spray	3/5	-	1/5	0/5	-	1/5	2/5	-	2/5	0/5	0/5	2/5
Coconut Water	2/5	3/5	4/5	2/5	2/5	1/5	2/5	1/5	2/5	0/5	0/5	0/5
Cookie Box	-	5/5	3/5	-	2/5	5/5	-	3/5	2/5	2/5	2/5	1/5
DO Avg./ %	53.3	40.0	50.0	40.0	20.0	43.3	30.0	16.7	30.0	26.7	20.0	43.3
All Avg./ %	61.4	47.1	57.1	50.0	40.0	51.4	32.9	28.6	32.9	27.1	31.4	37.1

TABLE VI: Detailed experiment results for each object and direction combination. "3D Avg." refers to the average success rate over all 3D-printed objects, "DO Avg." is that of daily objects and "All Avg." is that of all 14 test objects. These results correspond to the bar charts in Figure 6.

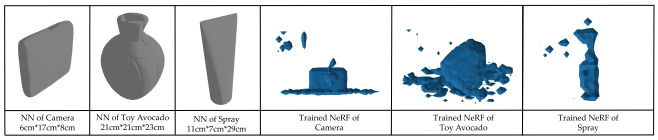

Fig. 19: Nearest Neighbor retrieval results of three test objects (left three columns) and visualization of trained NeRF (right three columns).

F. Multi-step Planning

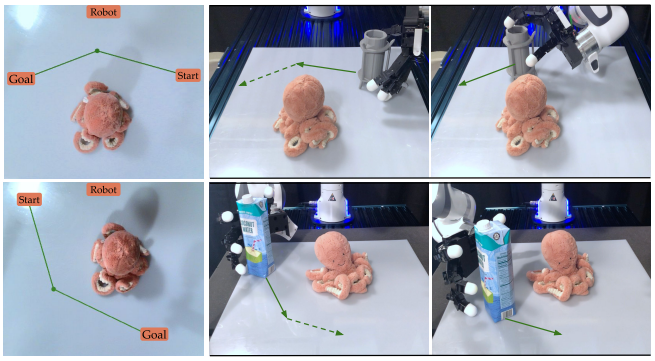

Fig. 20: Path planning using RRT* for multi-step planning. The first column shows the visualization of path planning results. The second and third columns show two consecutive hand poses for pushing the object along the path. The first example is the same as the one shown in Fig. 9.

Here, we provide more information and context on top of the *Multi-step Planning* section in Sec. V-C. These experiments explore the potential for GD2P's hand poses to support long-horizon planning. As shown in Figure 16, an Intel RealSense L515 camera captures a top-down view of the scene (see Figure 20). A toy placed in the scene serves as an obstacle. We extract its segmentation mask using Grounded SAM 2 [66]–[70], define the toy's position at its (estimated) center, and set a fixed 20 cm radius for path planning. The start and goal positions are manually assigned. We use RRT* as a high-level planner to compute a collision-free path in the 2D image space. Through camera calibration, we convert the 2D waypoints into 3D coordinates in the robot frame. For each edge along the planned path, GD2P generates a corresponding hand pose, and the robot pushes the object towards the next waypoint.

We test with two episodes that cover more pushing directions. The key insight in these experiments is that hand poses should be considered and evaluated while considering the kinematics of the arm as the motion becomes more complex. In the second row of Figure 20, a similar hand pose is able to finish the two-step pushing tasks while avoiding the obstacle. However, the first row of Figure 20 shows the need to change hand poses to better fit the object pose and the intended pushing direction. This motivates our use of motion planning and pose ranking to facilitate stable and smooth multi-step pushing motions.