DexNoMa: Learning Geometry-Aware Nonprehensile Dexterous Manipulation

Author Names Omitted for Anonymous Review.

***Abstract*—Nonprehensile manipulation, such as pushing and pulling, enables robots to move, align, or reposition objects that may be difficult to grasp due to their geometry, size, or relationship to the robot or the environment. Much of the existing work in nonprehensile manipulation relies on parallel-jaw grippers or tools such as rods and spatulas. Multi-fingered dexterous hands offer richer contact modes and versatility for handling diverse objects to provide stable support over the objects, which compensates for the difficulty of modeling the dynamics of nonprehensile manipulation. We propose Dexterous Nonprehensile Manipulation (DexNoMa), a method for nonprehensile manipulation which frames the problem as synthesizing and learning pre-contact dexterous hand poses that lead to effective pushing and pulling. We generate diverse hand poses via contact-guided sampling, filter them using physics simulation, and train a diffusion model conditioned on object geometry to predict viable poses. At test time, we sample hand poses and use standard motion planning tools to select and execute pushing and pulling actions. We perform 840 real-world experiments with an Allegro Hand, comparing our method to baselines. The results indicate that DexNoMa offers a scalable route for training dexterous nonprehensile manipulation policies. Our pre-trained models and dataset, including 1.3 million hand poses across 2.3k objects, will be open-source to facilitate further research. Supplementary material is available here: dexnoma.github.io.**

I. Introduction

Nonprehensile actions are fundamental to how humans and robots interact with the physical world [35, 31, 36, 32]. These actions permit the manipulation of objects that may be too large, heavy, or geometrically complex to grasp directly. While there has been tremendous progress in nonprehensile robot manipulation [72, 73, 13, 8, 33], most work uses simple end-effectors such as parallel-jaw grippers, rods [70, 7], or spatulas [57]. In contrast, multi-fingered hands with high degrees-of-freedom (DOF) such as the Allegro Hand or LEAP Hand [48] enable contact patterns that can be especially useful for stabilizing complex, awkward, or top-heavy objects, or for coordinating contact across multiple objects. However, despite their promise and recent progress [58], leveraging high-DOF hands for nonprehensile manipulation remains relatively underexplored due to the challenges of modeling hand-object relationships and planning feasible contact-rich motions.

In this paper, we study pushing and pulling objects using the 4-finger, 16-DOF Allegro Hand. Our insight is to recast this problem into one of synthesizing effective pre-contact hand poses, an approach inspired by recent success in generating large-scale datasets for dexterous *grasping* [29, 56, 69, 62, 54, 22]. We propose a scalable pipeline for generating hand poses for pushing and pulling objects. This involves contact-guided optimization and validation via GPU-accelerated physics simulation with IsaacGym [34]. These filtered hand poses are then used to train a generative diffusion policy conditioned on object geometry, represented using basis point sets [42].

At test time, we use visual data to reconstruct an object mesh in physics simulation. The trained diffusion policy uses this mesh to generate diverse hand poses for pushing or pulling. We then validate the resulting hand poses in simulation, and execute the best-performing action in the real world. We call this pipeline **Dex**terous **No**nprehensile **Ma**nipulation (DexNoMa). Figure 1 shows several real-world examples where the hand pose differs depending on object geometry. Overall, our experimental results across diverse common and 3D-printed objects demonstrate that DexNoMa is a promising approach for generalizable object pushing and pulling. It outperforms alternative methods such as querying the nearest hand pose in our data or using a fixed spatula-like hand pose, highlighting the need for a diffusion model to generate diverse hand poses.

To summarize, the contributions of this paper include:

- A scalable pipeline for generating and filtering dexterous hand poses for pushing and pulling.
- A diffusion model for geometry-conditioned hand pose prediction for nonprehensile manipulation.
- A motion planning framework to execute these poses for nonprehensile manipulation in the real world, with results across 840 trials showing that DexNoMa outperforms alternative methods.
- A dataset of 1.3 million hand poses for pushing and pulling across 2.3k objects with corresponding canonical point cloud observations.

II. Problem Statement and Assumptions

We study nonprehensile object manipulation on a flat surface using a single-arm robot with a high-DOF multi-finger dexterous hand (e.g., the Allegro Hand). By "nonprehensile," we specifically refer to *pushing* or *pulling* in this paper. We assume that there exists one object O on the surface with configuration $S_{\text{obj}} \in SE(3)$, and that the surface's friction properties facilitate object pushing. We use P to indicate the object's point cloud sampled from its surface. Let $\mathcal{H}$ be the space of possible nonprehensile hand poses, where $H \in \mathcal{H}$ is defined as $H = (\theta, T)$. Here, $\theta \in \mathbb{R}^d$ is the joint configuration of the d-DOF robot hand, and $T \in SE(3)$ is the end-effector pose of the robot's wrist consisting of translation and orientation. A *trial* is an instance of nonprehensile pushing or pulling, defined by a given direction $u_{\text{dir}} \in \mathbb{R}^3$ (with z-component of 0) resulting in the target object position as

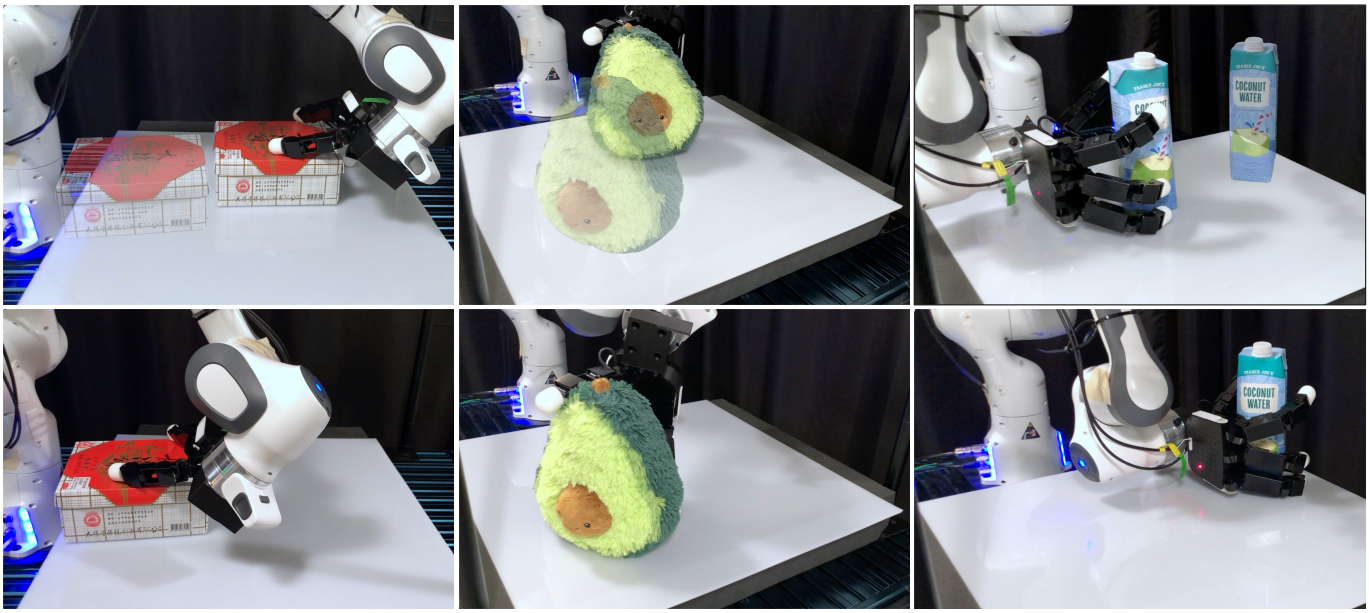

Fig. 1: Three examples (one per column) of nonprehensile manipulation using DexNoMa with a 4-finger, 16-DOF Allegro Hand. The top row shows the starting object configuration with its goal rendered as a transparent overlay, while the bottom row shows the result after the robot's motion. DexNoMa synthesizes diverse hand poses conditioned on object geometry, handling flat (left), volumetric (middle), and tall (right) objects.

$u_{\text{targ}} \in \mathbb{R}^3$. The objective is to generate a hand pose H such that, if a motion planner moves the hand to H and then translates it along u_{dir}, the object moves closer to the target u_{targ}. The object's distance to u_{targ} must be below a threshold for a trial to be considered a success.

III. Method

DexNoMa consists of the following steps. First, we generate a large dataset of hand poses for nonprehensile pushing and pulling (Sec. III-A). Second, we use this data to train a diffusion model to synthesize hand poses conditioned on object geometry (Sec. III-B). Third, during deployment, we generate hand poses and perform motion planning to do the pushing or pulling (Sec. III-C).

A. Dataset Generation for Nonprehensile Pushing and Pulling

We first generate hand poses for pushing and pulling various objects in simulation. To do this, we take inspiration from prior work on generating diverse hand poses for *grasping* [29, 56, 69, 62, 22, 10] by casting the hand synthesis problem as minimizing an energy function via optimization [27]. Unlike those works, our focus is on pushing and pulling actions instead of grasping. To enable optimization, we first define a set of candidate contact points sampled across the hand surface. Different regions of the hand have different candidate points to encourage broad contact across the palm and fingers. For the palm and finger (excluding fingertips) regions, we sample points uniformly over the rigid body surface. For the fingertips, we sample from a denser set of points uniformly on the unit hemisphere for each tip. See the Appendix for details of the distribution of candidate contact points (Figure 10 and Table II).

With the sampled contact point candidates, we run an optimization algorithm following the sampling strategy from [29, 56] that iteratively minimizes an energy function E to generate hand poses. We adapt the energy function from [29] to better suit our nonprehensile manipulation tasks, resulting in:

$$E = E_{\text{fc}} + w_{\text{dis}} E_{\text{dis}} + w_{\text{joints}} E_{\text{joints}} + w_{\text{pen}} E_{\text{pen}} + w_{\text{dir}} E_{\text{dir}} + w_{\text{arm}} E_{\text{arm}}, \tag{1}$$

where E_{fc} is a force closure estimator [27], E_{dis} penalizes hand-to-object distance (thus encouraging proximity), E_{joints} penalizes joint violations, and E_{pen} penalizes penetration between hand-object, hand-table and hand self-collision contacts. See [29, 56] for further details. The w terms are all scalar coefficients; we adopt the values from prior work and tune the weights for the following two new terms. To adapt the energy from Eq. 1 to pushing or pulling in a particular direction $u_{\text{dir}} \in \mathbb{R}^3$, we introduce E_{dir} and E_{arm}, which use the normal vector of the palm $v_{\text{palm}} \in \mathbb{R}^3$. The E_{dir} term encourages v_{palm} to align with u_{dir}, and E_{arm} encourages hand poses that are kinematically feasible when attached to the robot arm. Formally, we define E_{dir} and E_{arm} as:

$$E_{\text{dir}} = -\frac{u_{\text{dir}}^{\mathsf{T}} v_{\text{palm}}}{\|u_{\text{dir}}\|_2 \|v_{\text{palm}}\|_2} \quad \text{and} \quad E_{\text{arm}} = \max\big(0, (v_{\text{palm}})_z\big) \tag{2}$$

where $(v_{\text{palm}})_z$ is the z-component of the palm's normal vector (in the world frame). Intuitively, aligning u_{dir} and v_{palm} promotes more stable object-palm directional contact. Furthermore, if the palm faces upwards, then the rest of the arm must be below it. Thus, it is likely to lead to an infeasible robot configuration due to robot-table intersections, so E_{arm} is nonzero (i.e., worse). To inject randomness (and thus diversity) in the sampling process, we randomly resample

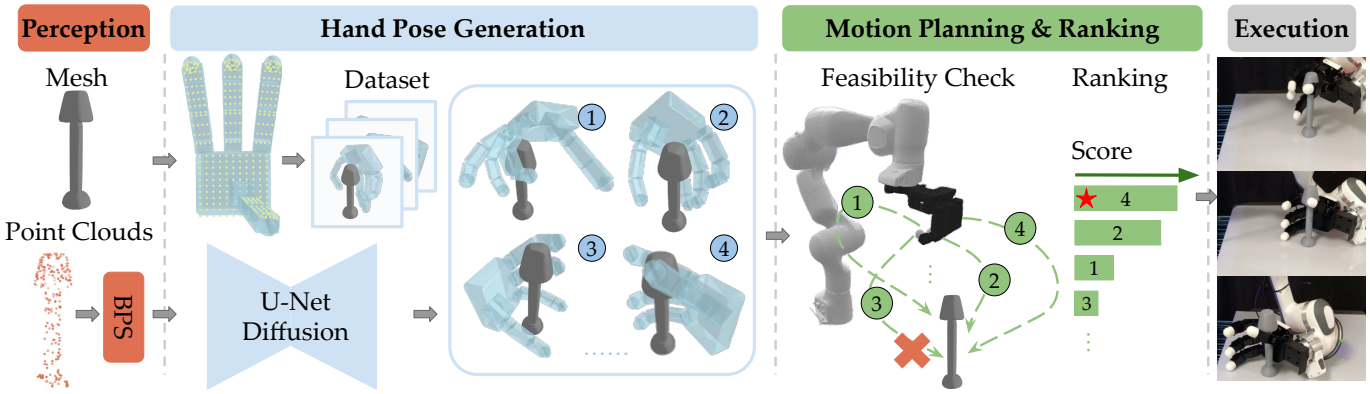

Fig. 2: Overview of DexNoMa. We present a large-scale dataset of hand poses specifically for pushing or pulling, and leverage it to train a diffusion model. During execution time, given an object, we obtain its basis point set representation [42] and pass that to our trained diffusion model, which uses the architecture from [59]. This model synthesizes diverse floating pre-contact hand poses formed from our large-scale data generation pipeline (Sec. III-A). Given these hand poses, we then check their feasibility in a physics simulator by adding the arm back in and performing motion planning [49]. We rank the feasible hand poses (e.g., "3" is infeasible in the example here) and select the best performing one (e.g., "4" in our example) and execute it in the real world.

a subset of the contact point indices from the set of valid candidates (Figure 10) when generating a new hand pose. We use RMSProp [51] to update translation, rotation and joint angles with step size decay, then minimize the energy function with Simulated Annealing [20] to adjust parameters.

Hand Pose Validation in Simulation. After optimizing contact points to generate candidate hand poses, we must *validate* whether they can lead to successful pushing or pulling. To do this, we use IsaacGym [34], a GPU-accelerated physics simulator that has been used in prior work for filtering grasp poses [29, 56]. We define a push or a pull as successful if, after executing a 20 cm translation, the object's center is within 3 cm of the target position *and* the object's orientation changes by no more than 45 degrees relative to its original configuration. The optimization process has a low success rate because it does not account for the full dynamics of pushing and pulling. Thus, we augment successful hand poses by adding slight noise to the pose parameters. We get 10X more augmented hand poses. From extensive parallel experiments, we generate a dataset containing 2,391 objects with 1,387,632 successful hand poses.

B. Training a Diffusion Model to Predict Hand Poses

To generate hand poses, we adapt a conditional U-Net [46] from the diffusion policy architecture [7], and train it with the Denoising Diffusion Probabilistic Models (DDPM) objective [11]. Diffusion models are well-suited for this task as they can learn complex, high-dimensional distributions. The forward process gradually adds Gaussian noise to the hand configuration H, while the reverse process reconstructs the original pose H by iteratively denoising conditioned on the object's geometry. The model is trained to minimize denoising error. To represent the observation, we use a 4096-dimensional Basis Point Set (BPS) [42] representation $B \in \mathbb{R}^{4096}$ based on the object's point cloud P. This representation, which is also used in [29, 59], encodes each object as a fixed-length vector of shortest distances between canonical basis points and the points in P. BPS captures geometric properties in a compact manner and simplifies the design of the diffusion model. Given this trained diffusion model, at test time it can be used to generate diverse hand poses which we can select for motion planning. See Figure 2 and Appendix VII-B for more information.

C. Arm-Hand Motion Planning and Evaluation

During deployment, the diffusion model generates candidate hand poses. We then integrate the Franka arm into full arm-hand motion planning to select hand poses which are kinematically feasible and avoid environment collisions, such as arm-table intersections (which are not considered in Sec. III-A). See Figure 2 (right half) for an overview. Each hand pose $H = (\theta, T)$ is initially expressed in the object frame. We use the object's initial configuration S_{obj} and intended direction u_{dir} to transform H to the world frame, and supply that to the cuRobo planner [49] to generate a complete motion plan for the Franka arm. In this process, we discard infeasible trajectories (and thus, the associated hand poses) to only keep the feasible arm-hand trajectories. To select which of the feasible trajectories to execute, we associate each with a custom analytical score V, defined as:

$$V(H = (\theta, T)) = \alpha L_{\text{goal}} + \beta L_{\text{coll}} + \gamma L_{\text{dir}}, \tag{3}$$

where L_{goal} measures the Euclidean distance between the object's final position and the target position, L_{coll} indicates whether a collision occurred during execution (1 if a collision occurs, 0 otherwise), and L_{dir} encourages the palm's orientation to align with the pushing direction. For L_{dir}, we set it equal to the E_{dir} term from the energy function (Eq. 1). The α, β, and γ are hyperparameters.

Multi-step Planning. While we mainly study DexNoMa for single open-loop pushes (or pulls) to targets, our framework naturally extends to multi-step planning. In scenarios with obstacles, we first compute a collision-free global path using RRT* [17]. Then, we sequentially plan hand poses to reach

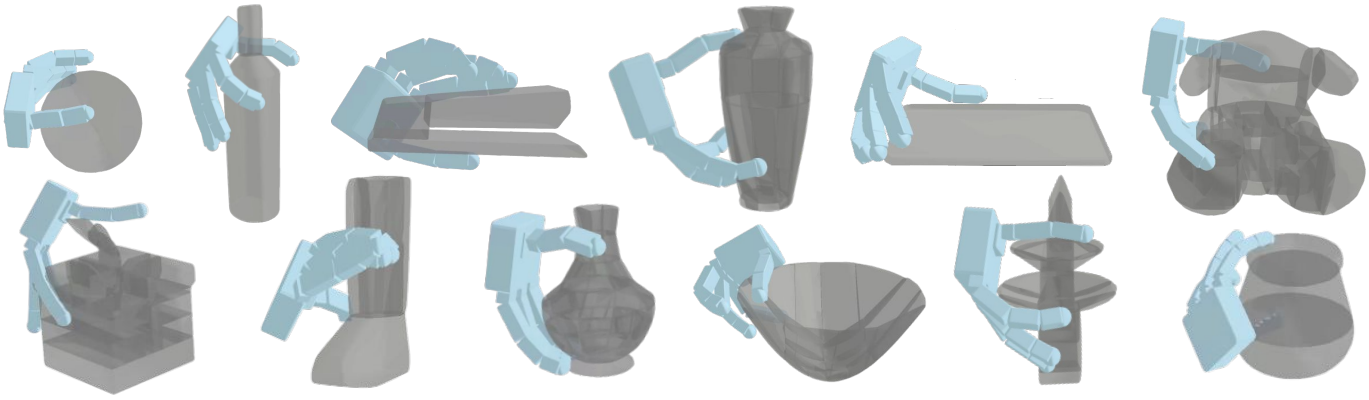

Fig. 3: Examples of nonprehensile hand pushing poses from optimizing our energy function (Eq. 1). These have all been validated in IsaacGym simulation. In all examples, the intended object pushing direction is to the right. These data points are used to train our diffusion model (see Sec. III-B).

each intermediate waypoint. Given an object, the same hand pose may be feasible only in certain pushing or pulling directions due to robot and hand kinematics. The waypoints from RRT* may require planning pushes across challenging directions, which highlights the importance of generating diverse hand poses for varying object positions and directions.

IV. Experiments

A. Real-World Experiments

We evaluate DexNoMa on a real robot to check if our nonprehensile hand poses successfully transfer to reality. Our hardware setup consists of a Franka Panda arm equipped with a four-finger, 16-DOF Allegro Hand. It operates over a tabletop cutting board with dimensions 60 cm×60 cm. We use a mix of objects, including 3D-printed and common items (shown in Figure 4). All evaluation objects are unseen during training. For 3D-printed objects, we use their known meshes to directly compute their BPS representation. For the other objects, we follow the pipeline proposed in [28] to obtain real-world object point clouds (and thus, the BPS). We reconstruct object meshes by using Nerfstudio [50] to compute COLMAP reconstructions [47]. We also use Stable Normal [65] to generate normal maps. Then, we employ 2D Gaussian Splatting [12] to obtain the point clouds. While this reconstruction pipeline introduces some noise, it is sufficient for DexNoMa to predict effective hand poses. In contrast, we empirically observed that optimization-based methods are more sensitive to mesh quality and often fail under these conditions.

Baselines and Ablations. We compare DexNoMa with the following methods: Pre-trained Grasp Pose, Nearest Neighbor, DexNoMa w/o Ranking. Please refer to the Appendix for detailed explanations on baseline methods and the experiment protocol.

B. Real-World Results

We summarize quantitative results in Figure 6, which shows that DexNoMa outperforms or matches alternative methods for both object categories. As shown in Figure 7, the **Pre-Trained Grasp Pose** baseline suffers from two major issues. First, the hand pose is not conditioned on the pushing direction, which means during the push, the object is likely to slide off the hand due to limited support (Figure 7, second row). Second, some objects are unsuitable for grasping due to their geometry or awkward aspect ratios. Additionally, the similarity-based **Nearest Neighbor** baseline struggles due to limited granularity in object geometry matching, motivating the need for our geometry-conditioned generative model. For **DexNoMa w/o ranking**, we observe that its hand poses are more likely to collide with the table or objects. To further investigate this ablation, Figure 5 shows three different hand poses. The first one has a low collision score because it is easy to collide with the table, while the third collides with the objects and scores low on the palm direction. The second hand pose leads to a successful push in real-world experiments. This suggests the importance of our ranking system via Eq. 3. **DexNoMa** outperforms baselines in all directions tested in Figure 6, demonstrating the robustness of its generated hand poses for nonprehensile manipulation. Figure 7 (first row) demonstrates using the palm and thumb to provide strong support moving the object forward, and the third row shows using the thumb and index finger to form a circular shape support for the thinner upper parts of the object while providing force at the bottom, aiding stable movement. For more rollouts, see the Appendix and the website.

Fixed Hand Pose: Inspired by prior pushing work [57], we manually define a "spatula" hand pose with the fingers spread flat (see Figure 8) to assess whether simple flat-hand strategies suffice for diverse objects. We perform a case study on the 6 objects in Figure 4 that are taller than 20 cm. We push each object 10 times, with 5 pushes for each of 2 directions, (the third direction results in kinematic errors). We get a relatively low 18/60 success rate, suggesting insufficient object support.

V. Conclusion

In this work, we propose DexNoMa, a dataset and method for nonprehensile object pushing and pulling using a high-DOF Allegro Hand. We hope that this inspires future work on dexterous nonprehensile robotic manipulation.

References

[1] Wisdom C Agboh, Jeffrey Ichnowski, Ken Goldberg, and Mehmet R Dogar. Multi-object grasping in the plane. In *International Symposium on Robotics Research (ISRR)*, 2022.

[2] Jeannette Bohg, Antonio Morales, Tamim Asfour, and Danica Kragic. Data-driven grasp synthesis—a survey. In *IEEE Transactions on Robotics (T-RO)*, 2014.

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

[4] Jiayi Chen, Yuxing Chen, Jialiang Zhang, and He Wang. Task-oriented dexterous hand pose synthesis using differentiable grasp wrench boundary estimator. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.

[5] Sirui Chen, Albert Wu, and C. Karen Liu. Synthesizing dexterous nonprehensile pregrasp for ungraspable objects. In *ACM SIGGRAPH*, 2023.

[6] Tao Chen, Eric Cousineau, Naveen Kuppuswamy, and Pulkit Agrawal. Vegetable Peeling: A Case Study in Constrained Dexterous Manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[7] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

[8] Yoonyoung Cho, Junhyek Han, Yoontae Cho, and Beomjoon Kim. Corn: Contact-based object representation for nonprehensile manipulation of general unseen objects. In *International Conference on Learning Representations (ICLR)*, 2024.

[9] Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile ALOHA: Learning Bimanual Mobile Manipulation with Low-Cost Whole-Body Teleoperation. In *Conference on Robot Learning (CoRL)*, 2024.

[10] Sicheng He, Zeyu Shangguan, Kuanning Wang, Yongchong Gu, Yuqian Fu, Yanwei Fu, and Daniel Seita. Sequential multi-object grasping with one dexterous hand. *arXiv preprint arXiv:2503.09078*, 2025.

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[12] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers '24*, SIGGRAPH '24. ACM, 2024.

[13] Bowen Jiang, Yilin Wu, Wenxuan Zhou, Chris Paxton, and David Held. Hacman++: Spatially-grounded motion primitives for manipulation. In *Robotics: Science and Systems (RSS)*, 2024.

[14] Hanwen Jiang, Shaowei Liu, Jiashun Wang, and Xiaolong Wang. Hand-object contact consistency reasoning for human grasps generation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[15] Hao Jiang, Yuhai Wang, Hanyang Zhou, and Daniel Seita. Learning to Singulate Objects in Packed Environments using a Dexterous Hand. In *International Symposium on Robotics Research (ISRR)*, 2024.

[16] Qing Jiang, Feng Li, Zhaoyang Zeng, Tianhe Ren, Shilong Liu, and Lei Zhang. T-rex2: Towards generic object detection via text-visual prompt synergy, 2024.

[17] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research (IJRR)*, 2011.

[18] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[19] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment Anything. *arXiv preprint arXiv:2304.02643*, 2023.

[20] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598): 671–680, 1983.

[21] Oliver Kroemer, Scott Niekum, and George Konidaris. A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms. In *Journal of Machine Learning Research (JMLR)*, 2021.

[22] Yuyang Li, Bo Liu, Yiran Geng, Puhao Li, Yaodong Yang, Yixin Zhu, Tengyu Liu, and Siyuan Huang. Grasp multiple objects with one hand. In *IEEE Robotics and Automation Letters (RA-L)*, 2024.

[23] Toru Lin, Zhao-Heng Yin, Haozhi Qi, Pieter Abbeel, and Jitendra Malik. Twisting Lids Off with Two Hands. In *Conference on Robot Learning (CoRL)*, 2024.

[24] Jason Jingzhou Liu, Yulong Li, Kenneth Shaw, Tony Tao, Ruslan Salakhutdinov, and Deepak Pathak. Factr: Force-attending curriculum training for contact-rich policy learning. In *Robotics: Science and Systems (RSS)*, 2025.

[25] Min Liu, Zherong Pan, Kai Xu, Kanishka Ganguly, and Dinesh Manocha. Deep differentiable grasp planner for high-dof grippers. In *Robotics: Science and Systems (RSS)*, 2020.

[26] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[27] Tengyu Liu, Zeyu Liu, Ziyuan Jiao, Yixin Zhu, and Song-Chun Zhu. Synthesizing diverse and physically stable grasps with arbitrary hand structures using differentiable force closure estimator. In *IEEE Robotics and Automation Letters (RA-L)*, 2022.

[28] Haozhe Lou, Yurong Liu, Yike Pan, Yiran Geng, Jianteng Chen, Wenlong Ma, Chenglong Li, Lin Wang, Hengzhen Feng, Lu Shi, et al. Robo-gs: A physics consistent spatial-temporal model for robotic arm with hybrid representation. *arXiv preprint arXiv:2408.14873*, 2024.

[29] Tyler Ga Wei Lum, Albert H. Li, Preston Culbertson, Krishnan Srinivasan, Aaron D. Ames, Mac Schwager, and Jeannette Bohg. Get a Grip: Multi-Finger Grasp Evaluation at Scale Enables Robust Sim-to-Real Transfer. In *Conference on Robot Learning (CoRL)*, 2024.

[30] Tyler Ga Wei Lum, Olivia Y. Lee, C. Karen Liu, and Jeannette Bohg. Crossing the Human-Robot Embodiment Gap with Sim-to-Real RL using One Human Demonstration. *arXiv preprint arXiv:2504.12609*, 2025.

[31] Kevin M. Lynch. *Nonprehensile Robotic Manipulation: Controllability and Planning*. PhD thesis, Carnegie Mellon University, The Robotics Institute, 1996.

[32] Kevin M. Lynch and Matthew T. Mason. Dynamic nonprehensile manipulation: Controllability, planning, and experiments. In *International Journal of Robotics Research (IJRR)*, 1999.

[33] Jiangran Lyu, Ziming Li, Xuesong Shi, Chaoyi Xu, Yizhou Wang, and He Wang. Dywa: Dynamics-adaptive world action model for generalizable non-prehensile manipulation. *arXiv preprint arXiv:2503.16806*, 2025.

[34] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning. *arXiv preprint arXiv:2108.10470*, 2021.

[35] Matthew T. Mason. Mechanics and Planning of Manipulator Pushing Operations. In *International Journal of Robotics Research (IJRR)*, 1986.

[36] Matthew T. Mason. Progress in Nonprehensile Manipulation. In *International Journal of Robotics Research (IJRR)*, 1999.

[37] 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 (ECCV)*, 2020.

[38] Andrew T Miller and Peter K Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 2004.

[39] João Moura, Theodoros Stouraitis, and Sethu Vijayakumar. Non-prehensile planar manipulation via trajectory optimization with complementarity constraints. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[40] 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 preprint arXiv:1910.07113*, 2019.

[41] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, 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. In *International Journal of Robotics Research (IJRR)*, 2019.

[42] Sergey Prokudin, Christoph Lassner, and Javier Romero. Efficient learning on point clouds with basis point sets. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[43] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-Hand Object Rotation via Rapid Motor Adaptation. In *Conference on Robot Learning (CoRL)*, 2022.

[44] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[45] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024.

[46] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

[47] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[48] Kenneth Shaw, Ananye Agarwal, and Deepak Pathak. LEAP Hand: Low-Cost, Efficient, and Anthropomorphic Hand for Robot Learning. In *Robotics: Science and Systems (RSS)*, 2023.

[49] Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, Nathan Ratliff, and Dieter Fox. curobo: Parallelized collision-free minimum-jerk robot motion generation. *arXiv preprint arXiv:2310.17274*, 2023.

[50] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23, 2023.

[51] T. Tieleman and G. Hinton. Lecture 6.5—rmsprop: Divide the gradient by a running average of its recent

magnitude, 2012. COURSERA: Neural Networks for Machine Learning.

[52] Dylan Turpin, Liquan Wang, Eric Heiden, Yun-Chun Chen, Miles Macklin, Stavros Tsogkas, Sven Dickinson, and Animesh Garg. Grasp'd: Differentiable contact-rich grasp synthesis for multi-fingered hands. In *European Conference on Computer Vision (ECCV)*, 2022.

[53] Dylan Turpin, Tao Zhong, Shutong Zhang, Guanglei Zhu, Eric Heiden, Miles Macklin, Stavros Tsogkas, Sven Dickinson, and Animesh Garg. Fast-grasp'd: Dexterous multi-finger grasp generation through differentiable simulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[54] Weikang Wan, Haoran Geng, Yun Liu, Zikang Shan, Yaodong Yang, Li Yi, and He Wang. UniDexGrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[55] Jun Wang, Ying Yuan, Haichuan Che, Haozhi Qi, Yi Ma, Jitendra Malik, and Xiaolong Wang. Lessons from Learning to Spin "Pens"'. In *Conference on Robot Learning (CoRL)*, 2024.

[56] Ruicheng Wang, Jialiang Zhang, Jiayi Chen, Yinzhen Xu, Puhao Li, Tengyu Liu, and He Wang. DexGraspNet: A large-scale robotic dexterous grasp dataset for general objects based on simulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[57] Yixuan Wang, Yunzhu Li, Katherine Driggs-Campbell, Li Fei-Fei, and Jiajun Wu. Dynamic-resolution model learning for object pile manipulation. In *Robotics: Science and Systems (RSS)*, 2023.

[58] Yuhan Wang, Yu Li, Yaodong Yang, and Yuanpei Chen. Dexterous non-prehensile manipulation for ungraspable object via extrinsic dexterity. *arXiv preprint arXiv:2503.23120*, 2025.

[59] Zehang Weng, Haofei Lu, Danica Kragic, and Jens Lundell. Dexdiffuser: Generating dexterous grasps with diffusion models. In *IEEE Robotics and Automation Letters (RA-L)*, 2024.

[60] Albert Wu, Ruocheng Wang, Sirui Chen, Clemens Eppner, and C. Karen Liu. One-Shot Transfer of Long-Horizon Extrinsic Manipulation Through Contact Retargeting. *arXiv preprint arXiv:2404.07468*, 2024.

[61] Lixin Xu, Zixuan Liu, Zhewei Gui, Jingxiang Guo, Zeyu Jiang, Zhixuan Xu, Chongkai Gao, and Lin Shao. Dexsingrasp: Learning a unified policy for dexterous object singulation and grasping in cluttered environments. *arXiv preprint arXiv:2504.04516*, 2025.

[62] Yinzhen Xu, Weikang Wan, Jialiang Zhang, Haoran Liu, Zikang Shan, Hao Shen, Ruicheng Wang, Haoran Geng, Yijia Weng, Jiayi Chen, et al. UniDexGrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[63] William Yang and Michael Posa. Dynamic on-palm manipulation via controlled sliding. In *Robotics: Science and Systems (RSS)*, 2024.

[64] Kunpeng Yao and Aude Billard. Exploiting kinematic redundancy for robotic grasping of multiple objects. In *IEEE Transactions on Robotics (T-RO)*, 2023.

[65] Chongjie Ye, Lingteng Qiu, Xiaodong Gu, Qi Zuo, Yushuang Wu, Zilong Dong, Liefeng Bo, Yuliang Xiu, and Xiaoguang Han. Stablenormal: Reducing diffusion variance for stable and sharp normal. *arXiv preprint arXiv:2406.16864*, 2024.

[66] Takahiro Yonemaru, Weiwei Wan, Tatsuki Nishimura, and Kensuke Harada. Learning to Group and Grasp Multiple Objects. *arXiv preprint arXiv:2502.08452*, 2025.

[67] Ying Yuan, Haichuan Che, Yuzhe Qin, Binghao Huang, Zhao-Heng Yin, Kang-Won Lee, Yi Wu, Soo-Chul Lim, and Xiaolong Wang. Robot Synesthesia: In-Hand Manipulation with Visuotactile Sensing. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[68] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In *Robotics: Science and Systems (RSS)*, 2024.

[69] Jialiang Zhang, Haoran Liu, Danshi Li, XinQiang Yu, Haoran Geng, Yufei Ding, Jiayi Chen, and He Wang. DexGraspNet 2.0: Learning Generative Dexterous Grasping in Large-scale Synthetic Cluttered Scenes. In *Conference on Robot Learning (CoRL)*, 2024.

[70] Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Adaptigraph: Material-adaptive graph-based neural dynamics for robotic manipulation. In *Robotics: Science and Systems (RSS)*, 2024.

[71] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Robotics: Science and Systems (RSS)*, 2023.

[72] Wenxuan Zhou and David Held. Learning to Grasp the Ungraspable with Emergent Extrinsic Dexterity. In *Conference on Robot Learning (CoRL)*, 2022.

[73] Wenxuan Zhou, Bowen Jiang, Fan Yang, Chris Paxton, and David Held. HACMan: Learning Hybrid Actor-Critic Maps for 6D Non-Prehensile Manipulation. In *Conference on Robot Learning (CoRL)*, 2023.

VI. Appendix

A. Related Work

Nonprehensile Robot Manipulation. Classical nonprehensile manipulation includes pushing, sliding, rolling, and tilting, and has a long history in robotics [35, 31, 36, 32]. Planning methods for nonprehensile manipulation often assume access to object models or priors [39, 3, 63]. Another recent planning-based method explores nonprehensile interaction with high-DOF hands in simulation by analyzing contact reasoning and wrench closure [5]. In contrast, our work targets real-world pushing and pulling using a high-DOF hand applied to diverse and geometrically complex objects. Recent learning-based methods have extended nonprehensile manipulation beyond classical planning, including extrinsic dexterity systems [72, 60] and those based on predicting object dynamics such as HACMan [73, 13], CORN [8], and DyWA [33]. Other works approach pushing as a precursor to grasping, often in planar settings with simple parallel-jaw grippers for multi-object manipulation [1, 66], or use bimanual systems for nonprehensile tasks using multi-link tools [24]. None of these works study learning for single-hand pushing and pulling with dexterous hands. Furthermore, many prior benchmarks focus on pushing single flat objects on a surface, such as a T-shape object [7], or use spatulas to move small cubes and granular media [57, 70]. Our work directly targets larger and more complex objects, including those that might topple or require coordinated multi-surface contact.

Dexterous Grasping Synthesis and Datasets. A substantial body of research focuses on generating and evaluating grasp poses for multi-fingered hands. Pioneering efforts such as Liu et al. [25] create a dataset of 6.9K grasps using the GraspIt! [38] software tool, while Jiang et al. [14] synthesize human hand poses by using a conditional Variational Autoencoder [18]. More recent efforts significantly scale grasp generation with tools such as differentiable contact simulation [52, 53] or optimization over an energy function based on Differentiable Force Closure (DFC) [27]. Our work falls in the latter category, which has facilitated the generation of diverse grasping datasets such as DexGraspNet [56] with 1.32M grasps followed by DexGraspNet 2.0 [69] with 427M grasps. These pipelines generate hand poses by optimization over an energy function, filter them using physics simulators, train generative diffusion models for grasp synthesis, and typically include some fine-tuning or evaluation modules [59, 29]. While our pipeline also uses energy-based pose optimization and filtering, our focus is on generating hand poses for nonprehensile manipulation.

Learning-Based Dexterous Manipulation. Learning-based approaches for robotic grasping and manipulation have rapidly expanded in recent years [2, 21]. While some recent work emphasizes fine-grained bimanual manipulation using parallel-jaw grippers [71, 9], our focus is on learning single-arm manipulation with high-DOF dexterous hands such as the LEAP [48], Allegro, and Shadow hands. These hands have been applied to a variety of tasks, such as in-hand object rotation [43, 55, 40, 41, 67], object singulation [15, 61], multi-object manipulation [10, 22, 64, 66], and bimanual systems [6, 23]. While showing the versatility of dexterous hardware, these works focus on largely prehensile interactions. Prior learning-based systems with high-DOF hands for nonprehensile behaviors demonstrate tasks such as rolling objects or picking up plates as examples of learning from 3D data [68] or human videos [30]. Recently, Chen et al. [4] synthesize task-oriented dexterous hand poses for certain nonprehensile tasks such as pulling drawers. However, none of these methods directly study pushing or pulling as their primary manipulation mode.

B. Supplementary Experiments

1) Simulation Experiments and Results: We evaluate the quality of the hand pose generation pipeline using IsaacGym [34]. To quantify the effectiveness of our trained model and dataset, we report the number of successfully pushed objects as a function of training data size. We train our diffusion model on varying subsets of the full dataset (of 1.3M hand poses) and evaluate on 300 unseen objects from the test set. For each test object, we sample 200 candidate hand poses. An object is considered "successful" if at least one feasible hand pose results in success. Table I reports results over 3 different seeds, which shows that our model generates feasible pushing poses more reliably with larger training sets, which validates large-scale supervision. The growth is not strictly linear, suggesting room for improvement via better model tuning or data strategies. Qualitatively, our generated hand poses are diverse across object geometries and exhibit pushing intent (see the Appendix for more discussion). A common failure mode is that some poses still collide with the object, which motivates the inclusion of the collision term in Eq. 3.

Data Size	# of Objects
2%	41.67 $\pm$ 10.21
20%	102.67 $\pm$ 5.85
50%	110.33 $\pm$ 29.67
100%	169.33 $\pm$ 15.18

TABLE I: Number of objects with at least one feasible pushing hand pose out of 300.

2) Real-World Experiments: **Baselines and Ablations.** We compare DexNoMa with the following methods.

- **Pre-Trained Grasp Pose**: We use a pre-trained grasp synthesis model from Lum et al. [29] using NeRF [37]. For each object, we train a NeRF representation, then query their pre-trained model for a grasp. This evaluates how well a grasping-centric model generalizes to nonprehensile tasks.
- **Nearest Neighbor (NN)**: Given a test object, we find the training object with the most similar BPS representation (in terms of Euclidean distance) and retrieve its associated hand poses. We then do the same motion planning pipeline as in DexNoMa. This tests out-of-distribution generalization with a retrieval-only approach compared to our proposed generative model.
- **DexNoMa w/o Ranking**: An ablation that excludes analytical ranking of hand poses (ignores Eq. 3) and executes a

Fig. 4: The objects we use in our real-world nonprehensile manipulation experiments, including 3D printed and common ("Daily") objects. See Sec. IV-A for more details.

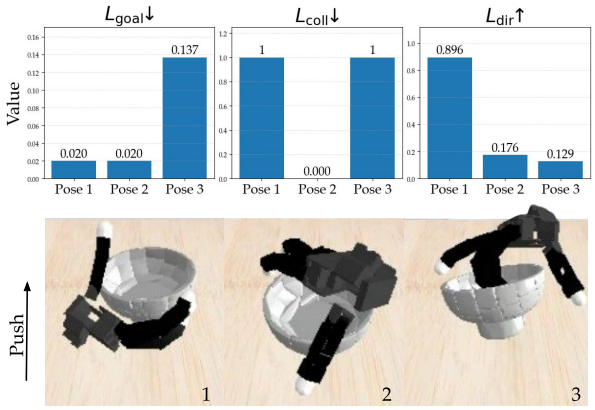

Fig. 5: Visualization of L_{goal}, L_{coll}, and L_{dir} values in $V(H)$ from Eq. 3 on three simulated hand poses. See Sec. IV-B for more details.

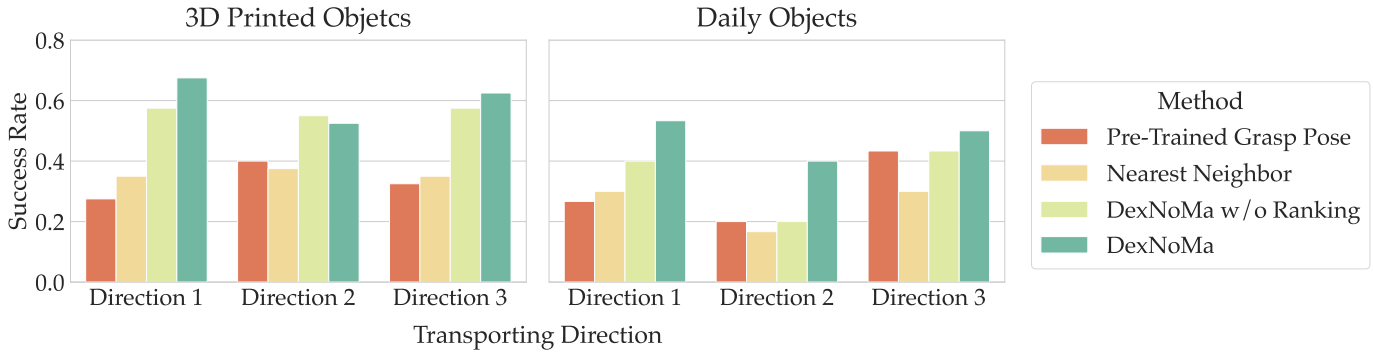

Fig. 6: Nonprehensile manipulation success rates from DexNoMa and baselines, across different 3D printed (left) and daily objects (right), and with three directions evaluated. Each bar aggregates success rates from 40 trials (left bar plot) and 30 trials (right bar plot). See Sec. IV-A and IV-B for more details.

random feasible pose. This tests the usefulness of Eq. 3 in selecting poses.

Experiment Protocol and Evaluation. For each object, we test three pushing directions uniformly distributed around a circle. Along each direction, the robot executes the hand pose and planned motion five times, all with a fixed push length of 20 cm. A human manually places the object in a relatively consistent pose between trials. A trial is successful if the object's center is within 3 cm of the target position, the hand maintains contact throughout, and it does not lead to task failure modes such as toppling or loss of control. For NN and DexNoMa w/o Ranking, we randomly sample hand poses among the feasible planned actions. For Pre-trained Grasp Pose, we execute the best actions from its output. For our method, we execute the one with the highest analytical score from Eq. 3.

Multi-step Planning. Selecting a kinematically feasible hand pose for a given object state S_{obj} and direction u_{dir} is challenging in multi-step planning, as different waypoints may require different hand poses. Our method resolves this by identifying valid poses across object configurations and coupling pose selection with kinematic feasibility (see Sec. III-C). By doing so, DexNoMa can be used to perform multiple pushes. Figure 9 shows a multi-step pushing sequence using DexNoMa. The robot uses two different hand poses to push the 3D-printed vase, as the first hand pose may not be ideal for the second hand pose, which shows the benefit of re-planning.

VII. Additional Details of DexNoMa

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.

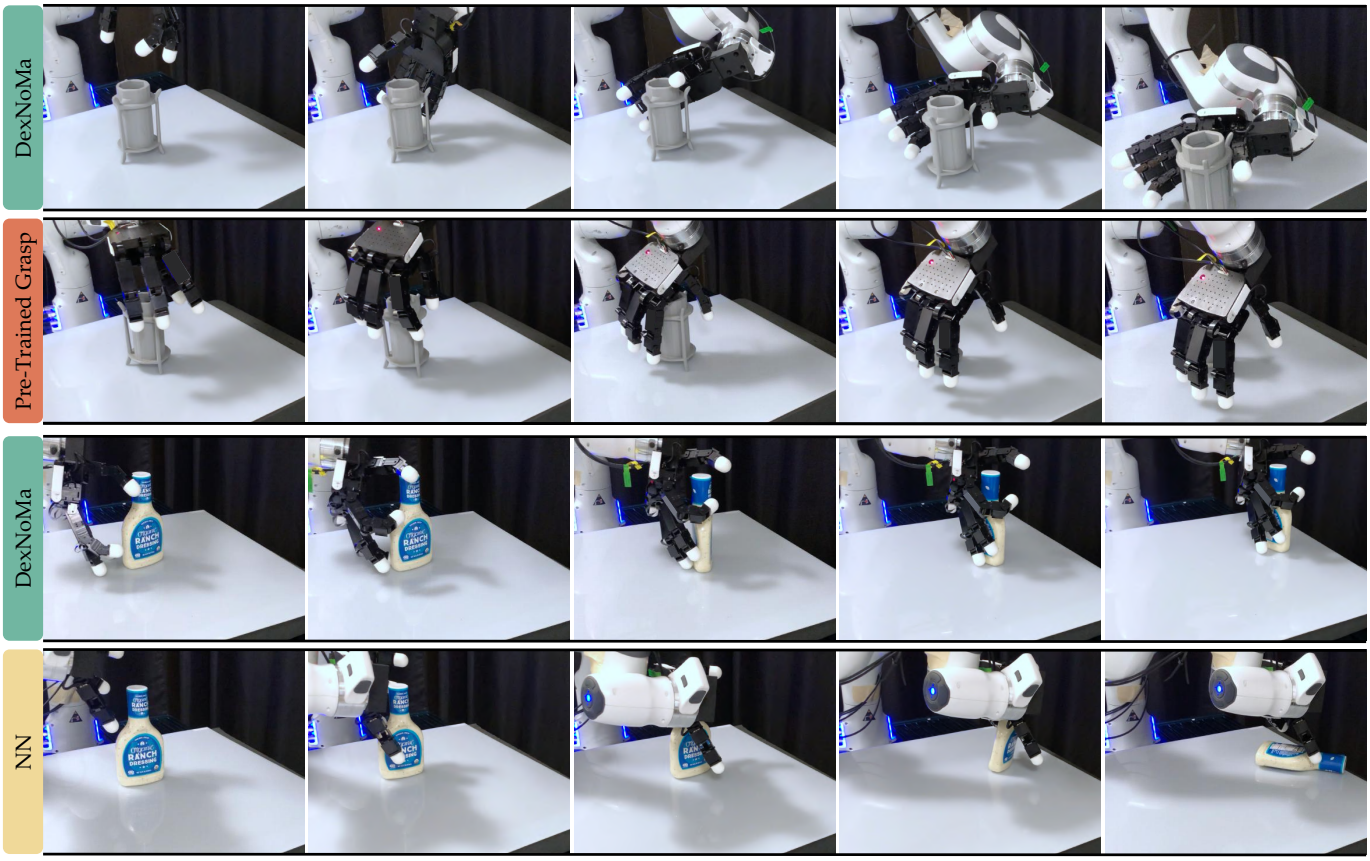

Fig. 7: Comparison between DexNoMa and baselines. The first two rows show DexNoMa (success) and Pre-Trained Grasp (failure) while pushing a 3D-printed vase forward (i.e., away from the robot). The last two rows show DexNoMa (success) and NN (failure) while pushing a ranch bottle to the right.

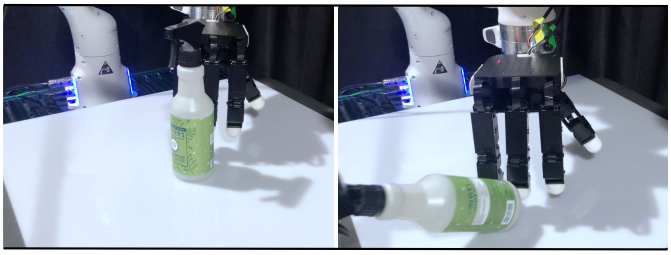

Fig. 8: Example of a typical failure case using the Fixed Hand Pose strategy, which topples the spray.

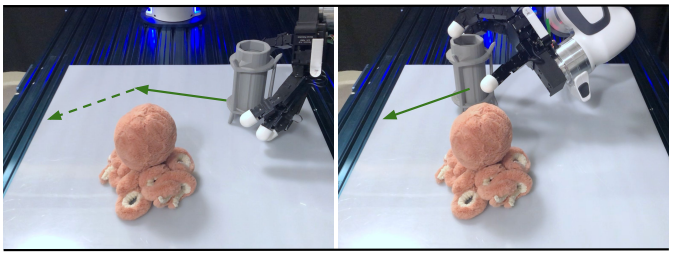

Fig. 9: Example of multi-step pushes using DexNoMa, which avoids the central obstacle.

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 10 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. III-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 11 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.

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.

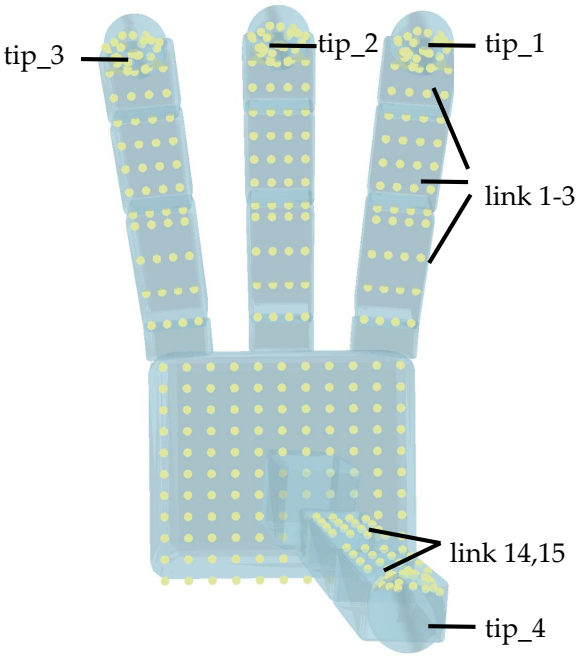

Fig. 10: Contact candidates on the Allegro hand. Refer to Table II for the number of contacts on each link.

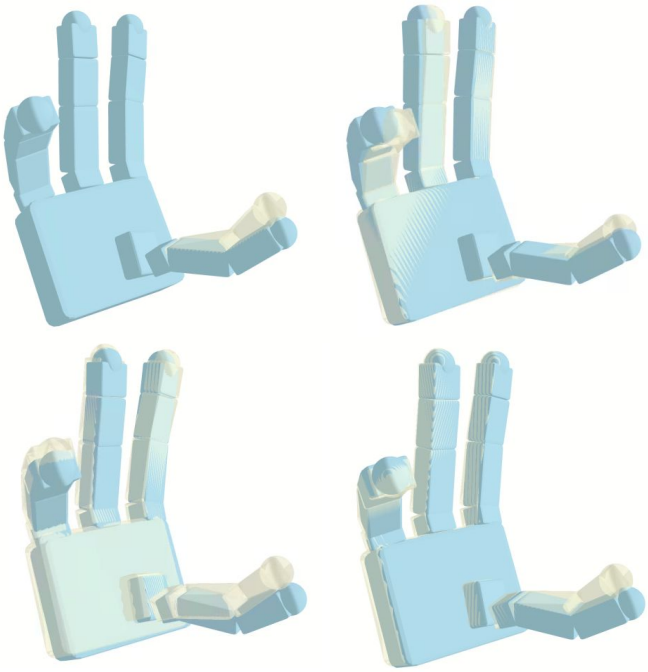

Fig. 11: A visualization of an example of augmentations. *Lightyellow* indicates the hand pose with the perturbation, and *lightblue* is the original one.

Figure 12 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 13 shows a breakdown of object categories and the frequency of the top 20 objects in our dataset.

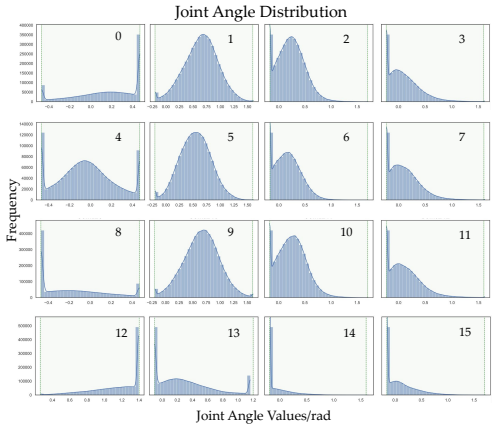

Fig. 12: Visualization of the distribution of joint angle values in our proposed dataset, demonstrating the diversity of our generated hand poses. The number on the top right corner of each subfigure indicates the joint index. The *green dashed lines* on the edge of x-axis indicate the lower/upper bounds of each joint angle values.

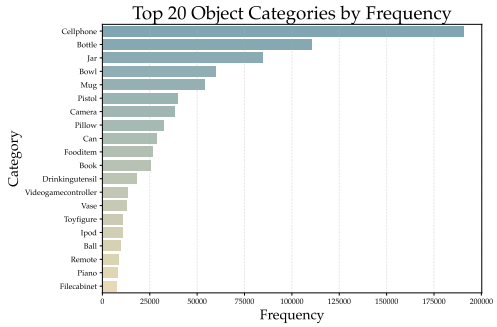

Fig. 13: Visualization of the top 20 objects in terms of pushing hand poses frequency in our proposed dataset.

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 14, which is relevant to our experiments in Sec. VI-B1.

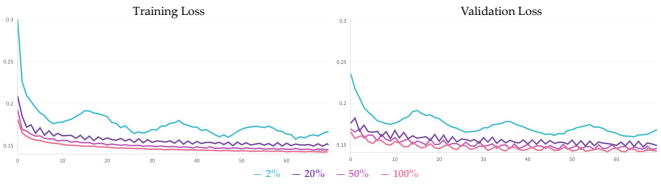

Fig. 14: Training curves on different scales of the dataset. See Sec.VI-B1 for more discussion.

VIII. Additional Details of Experiments

A. Experiment Details

Our physical experiment setup consists of a Franka Panda manipulator equipped with an Allegro Hand, as shown in Figure 15. We also place an L515 RealSense camera above the table, which is *only* used for path planning in multi-step planning experiments in Sec. IV-B and Sec. VIII-D. 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.

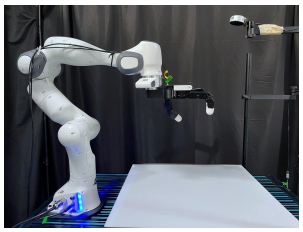

Fig. 15: Our physical experiment setup including a Frank Panda robot with an attached Allegro Hand. The camera is only used for high-level path planning.

We select 8 3D-printed objects and 6 real-world objects, covering flat, volumetric, and tall objects, as shown in Figure 16. 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. IV-B, our results highlight the need for hand poses that provide more stable object support for transporting.

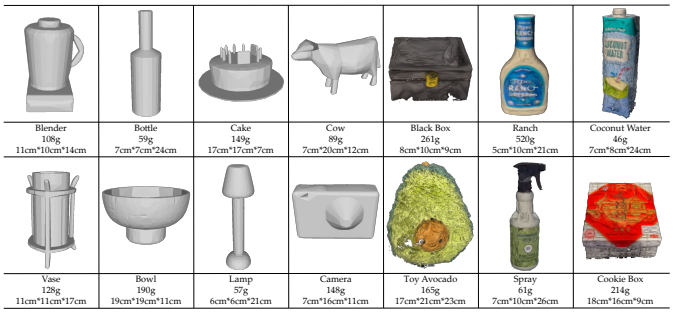

Fig. 16: 3D meshes, mass and physical dimensions of all objects tested in real-world experiments. Dimensions are listed as (x, y, z).

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 DexNoMa has marginally more infeasible trials than the baselines, this is expected because DexNoMa 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, DexNoMa outperforms the baseline methods, demonstrating its robustness on pushing or pulling tasks.

B. More Successful Rollouts

We provide additional example visualizations of successful rollouts of DexNoMa in Figure 17. For videos, please refer to our website: dexnoma.github.io.

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.

	DexNoMa			**DexNoMa 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.

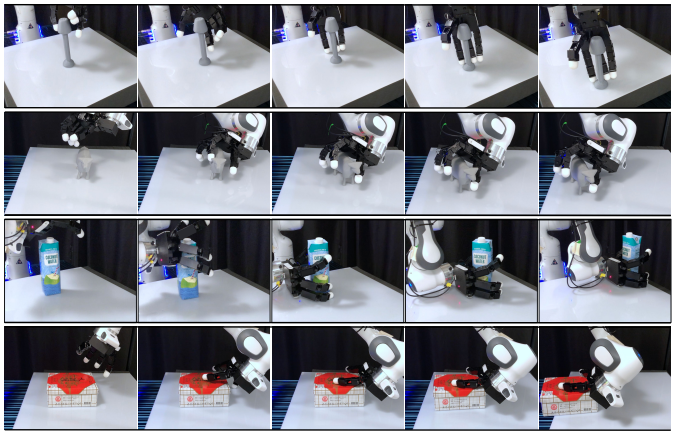

Fig. 17: Successful rollouts of DexNoMa, one per row.

C. Results and Analysis of Baseline Methods

We visualize 3 examples of the nearest neighbor (NN) retrieval results and the trained NeRF representation in Figure 18. The retrieved NN objects are similar in shape and scale of the query object (left 3 columns in Figure 18). 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 18 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. IV-B.

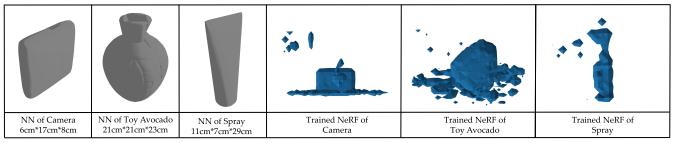

Fig. 18: Nearest Neighbor retrieval results of three test objects (left three columns) and visualization of trained NeRF (right three columns).

D. Multi-step Planning

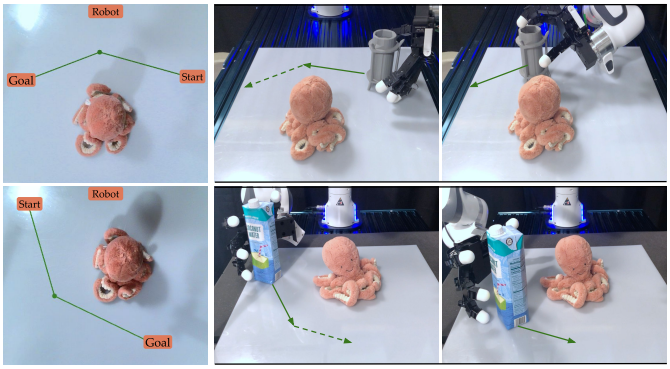

Fig. 19: Path planning using RRT* for multi-step planning. The first column shows the visualization of path planning results. The second and third columns show two consecutive hand poses for pushing the object along the path. The first example is the same as the one shown in Fig. 9.

Here, we provide more information and context on top of the *Multi-step Planning* section in Sec. IV-B. These experiments explore the potential for DexNoMa's hand poses to support long-horizon planning. As shown in Figure 15, an Intel RealSense L515 camera captures a top-down view of the scene (see Figure 19). A toy placed in the scene serves as an obstacle. We extract its segmentation mask using Grounded SAM 2 [26, 44, 45, 19, 16], 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, DexNoMa 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 19, a similar hand pose is able to finish the two-step pushing tasks while avoiding the obstacle. However, the first row of Figure 19 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.