2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Learning Geometry-Aware Nonprehensile Pushing and Pulling with Dexterous Hands

Yunshuang Li, Yiyang Ling, Gaurav S. Sukhatme, Daniel Seita

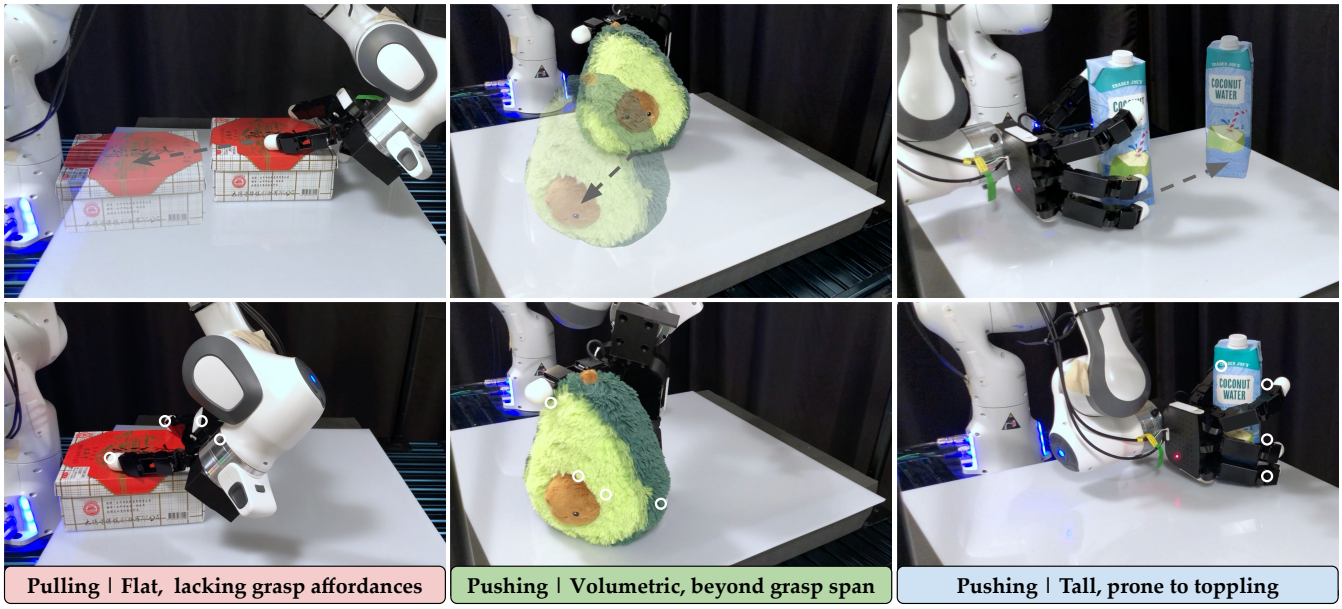

Fig. 1: Three examples of nonprehensile manipulation using GD2P with a 4-finger, 16-DOF Allegro Hand. The top row shows the starting object configuration with its goal rendered as a transparent overlay, while the bottom row shows the result after the robot's motion. GD2P synthesizes diverse hand poses conditioned on object geometry, handling flat (left), volumetric (middle), and tall (right) objects. Grey arrows represent the transporting direction, whereas white volumetric dots mark the estimated fingertip contact with the object.

***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. In contrast, 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. Therefore, we propose Geometry-aware Dexterous Pushing and Pulling (GD2P) for nonprehensile manipulation with dexterous robotic hands. We study pushing and pulling by framing the problem as synthesizing and learning pre-contact dexterous hand poses that lead to effective manipulation. 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 planners to select and execute pushing and pulling actions. We perform extensive real-world experiments with an Allegro Hand and a LEAP Hand, demonstrating that GD2P offers a scalable route for generating dexterous nonprehensile manipulation motions with its applicability to different hand morphologies. Our project website is available at: *geodex2p.github.io*.**

All authors are with the Thomas Lord Department of Computer Science at the University of Southern California, USA.
Correspondence: yunshuan@usc.edu

I. Introduction

Nonprehensile actions are fundamental to how humans and robots interact with the physical world [4]–[7]. 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 [8]–[12], most work uses simple end-effectors such as parallel-jaw grippers, rods [13], [14], or spatulas [15]. In contrast, multi-fingered hands with high degrees-of-freedom (DOF) such as the Allegro Hand or LEAP Hand [16] enable contact patterns that can be especially useful for stabilizing complex, awkward, or top-heavy objects, or for coordinating contact across multiple objects, compensating for the challenges of modeling nonprehensile manipulation dynamics. However, despite their promise and recent progress [17] , 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 and LEAP Hands. We select pushing and pulling as representative tasks of nonprehensile manipulation because they are more commonly used for manipulating general daily objects and are more amenable

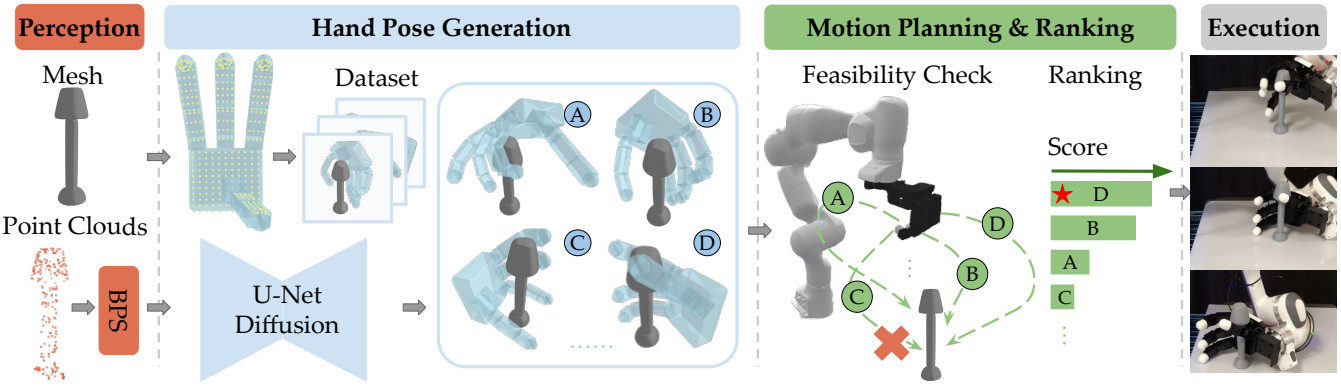

Fig. 2: Overview of GD2P. We present a large-scale dataset of hand poses specifically for pushing or pulling, and leverage it to train a diffusion model. During execution time, given an object, we obtain its basis point set representation [1] and pass that to our trained diffusion model, which uses the architecture from [2]. This model synthesizes diverse floating pre-contact hand poses formed from our large-scale data generation pipeline (Sec. IV-A). Given these hand poses, we then check their feasibility in a physics simulator by adding the arm back in and performing motion planning [3]. We rank the feasible hand poses (e.g., "C" is infeasible in the example here since the motion planner detects an unavoidable collision between the arm and the table.) and select the best performing one (e.g., "D" in our example with no collision detected, while optimally facilitating the pushing direction.) and execute it in the real world.

to scaling. 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 manipulation [18]–[23]. 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 [24]. These filtered hand poses are then used to train a generative diffusion model conditioned on object geometry, represented using basis point sets [1].

At test time, we use visual data to reconstruct an object mesh in physics simulation. The trained diffusion model 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 **G**eometry-aware **D**exterous **P**ushing and **P**ulling (GD2P) with multi-fingered hands. Fig. 1 shows several real-world examples where the hand pose differs depending on object geometry. Overall, our experimental results across diverse daily objects demonstrate that GD2P 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 nonprehensile pushing and pulling.
- A diffusion model for geometry-conditioned hand pose prediction for nonprehensile pushing and pulling.
- A motion planning framework to execute these poses in the real world, with results across 840 trials showing that GD2P outperforms alternative methods.
- A dataset of 1.3 million hand poses for nonprehensile pushing and pulling across 2.3k objects with corresponding canonical point cloud observations.

II. Related Work

Nonprehensile Robot Manipulation. Classical nonprehensile manipulation includes pushing, sliding, rolling, and tilting, and has a long history in robotics [4]–[7]. Planning methods for nonprehensile manipulation often assume access to object models or priors [25], [26]. Another recent planning-based method explores nonprehensile interaction with high-DOF hands in simulation by analyzing contact reasoning and wrench closure [27]. In contrast, our work targets real-world pushing and pulling using a high-DOF hand applied to diverse and geometrically complex objects. Applying traditional planning methods in this setting is highly challenging due to the complexity of modeling the diverse contact interactions between the hand and objects. Recent learning-based methods have extended nonprehensile manipulation beyond classical planning, including extrinsic dexterity systems [8], [28] and those based on predicting object dynamics such as HACMan [9], [10], CORN [11], and DyWA [12]. Other works approach pushing as a precursor to grasping, often in planar settings with parallel-jaw grippers for multi-object manipulation [29], or use bimanual systems for nonprehensile tasks using multi-link tools [30]. 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 [14], or use spatulas to move small cubes and granular media [13], [15]. 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 [31] create a dataset of 6.9K grasps using the GraspIt! [32] software tool, while [33] synthesize human hand poses by using a conditional Variational Autoencoder [34]. More recent efforts significantly scale grasp generation with tools such as differentiable contact simulation [35], [36] or optimization

over an energy function based on Differentiable Force Closure (DFC) [37]. Our work falls in the latter category, which has facilitated the generation of diverse grasping datasets such as DexGraspNet [19] with 1.32M grasps followed by DexGraspNet 2.0 [20] with 427M grasps, with recent work [38] further incorporating language into the dataset. 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 [2], [18]. Most recently, Dex1B [39] scales to one billion demonstrations by integrating optimization-based control with generative modeling. While our pipeline also uses energy-based pose optimization and filtering, we focus on synthesizing hand poses for nonprehensile manipulation, specifically pushing and pulling, demonstrating the effectiveness of a tailored pipeline beyond grasping.

Learning-Based Dexterous Manipulation. Learning-based approaches for robotic grasping and manipulation have rapidly expanded in recent years [40]. While some recent work emphasizes fine-grained bimanual manipulation using parallel-jaw grippers [41], [42], our focus is on learning single-arm manipulation with high-DOF dexterous hands such as the LEAP [16], Allegro, and Shadow Hands. These hands have been applied to a variety of tasks, such as in-hand object rotation [43]–[45], object singulation [46], [47], multi-object manipulation [23], [29], [48], and bimanual systems [49], [50]. 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 [51] or human videos [52]. Recently, [53] 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.

III. Problem Statement and Assumptions

We study nonprehensile object pushing and pulling 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 emphasize the distinction from "prehensile pushing [54]." 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 pushing and pulling 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 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 $u_{\text{targ}} \in \mathbb{R}^3$. 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.

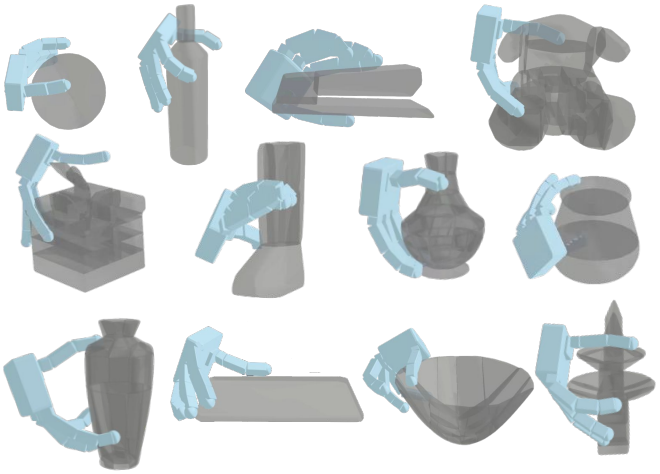

Fig. 3: Examples of pushing and pulling hand poses from optimizing our energy function (Eq. 1). These have all been validated in IsaacGym simulation. In all examples, the intended object pushing direction is to the right. These data points are used to train our diffusion model (see Sec. IV-B).

IV. Method

GD2P consists of the following steps. First, we generate a large dataset of hand poses for pushing and pulling (Sec. IV-A). Second, we use this data to train a diffusion model to synthesize hand poses conditioned on object geometry (Sec. IV-B). Third, during deployment, we generate hand poses and perform motion planning (Sec. IV-C).

A. Dataset Generation for Dexterous Pushing and Pulling

We first generate diverse 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* [18]–[21], [23], [48] by casting the hand synthesis problem as minimizing an energy function via optimization [37]. Unlike those works, our focus is on pushing and pulling instead of grasping. To enable hand pose optimization, we 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), 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 website for candidate contact points details.

With the sampled contact point candidates, we run an optimization algorithm following the sampling strategy from [18], [19] that iteratively minimizes an energy function E to generate hand poses. We adapt the energy function from [18] to suit our manipulation tasks, resulting in:

$$E = E_{\text{fc}} + w_{\text{dis}} E_{\text{dis}} + w_{\text{j}} E_{\text{j}} + w_{\text{pen}} E_{\text{pen}} + w_{\text{dir}} E_{\text{dir}} + w_{\text{arm}} E_{\text{arm}} \quad (1)$$

where E_{fc} is a force closure estimator [37], E_{dis} penalizes hand-to-object distance (thus encouraging proximity), E_{j}

penalizes joint violations, and E_{pen} penalizes penetration between hand-object, hand-table and hand self-collision contacts. See [18], [19] for further details. The w terms are all scalar coefficients; we adopt the values from prior work and tune the weights (see the Appendix on our website) 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:

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

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 a subset of the contact point indices from the set of valid candidates when generating a new hand pose. We use RMSProp [55] to update translation, rotation and joint angles with step size decay, then minimize the energy function with Simulated Annealing [56] 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 [24], a GPU-accelerated physics simulator that has been used in prior work for filtering grasp poses [18], [19]. 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. From extensive parallel experiments, we generate a dataset containing 2.3k objects with 1.3 million successful hand poses. See the Appendix on our website for more details.

B. Training a Diffusion Model to Predict Hand Poses

To generate hand poses, we adapt a conditional U-Net [57] from the diffusion policy architecture [14], and train it with the Denoising Diffusion Probabilistic Models (DDPM) objective [58]. 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) [1] representation $B \in \mathbb{R}^{4096}$ based on the object's point cloud P. 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 [59] to obtain real-world object point clouds (and thus, the BPS). Specifically, we reconstruct object meshes by using Nerfstudio [60] and we use Stable Normal [61] to generate normal maps. Then, we employ 2D Gaussian Splatting [62] to obtain the point clouds. This representation, which is also used in [2], [18], 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 Fig. 2 and Appendix on our website for more information.

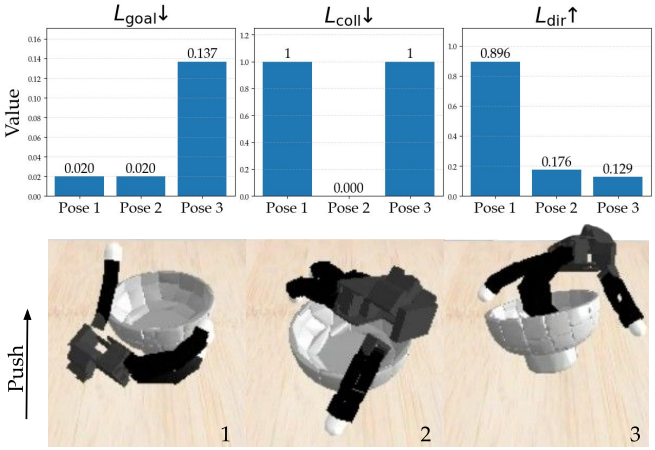

Fig. 4: Visualization of L_{goal}, L_{coll}, and L_{dir} values in $V(H)$ from Eq. 3 on three simulated hand poses. See Sec. IV-C and Sec. V-C for more details.

C. Arm-Hand Motion Planning and Evaluation

During deployment, the diffusion model generates candidate hand poses. We then integrate the robotic 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. IV-A). See Fig. 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 [3] to generate a complete motion plan for the robotic 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

Fig. 5: The objects we use in our real-world experiments, including 3D printed and daily objects. See Sec. V-B for more details.

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 tuned empirically.

To provide additional context, Fig. 4 presents three candidate hand poses along with their corresponding scores on L_{goal}, L_{coll} (where lower values are preferable), and L_{dir} (where higher values are preferable). Hand pose 1 achieves the highest score on L_{dir}, indicating effective alignment with the pushing direction, yet it results in an avoidable hand–table collision. Hand pose 3 exhibits a low score on L_{goal}, suggesting limited effectiveness in pushing the object toward the target. Balancing all three objectives, the motion planner would select hand pose 2 for execution in the real world. We tune these weight terms according to their influence on pushing feasibility and overall task success.

Multi-step Planning. While we mainly study GD2P 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* [63]. Then, we sequentially plan hand poses to reach 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, highlighting the importance of generating diverse hand poses for varying object positions and directions.

V. Experiments

Through simulation and real-world experiments, we aim to investigate the following questions: (1) Can we learn feasible and effective hand poses from our large-scale dataset? (2) Is GD2P more effective and robust for dexterous pushing and pulling compared to baselines and ablations? (3) Can GD2P serve as a reliable module for downstream manipulation tasks such as multi-step pushing around obstacles? (4) Does GD2P perform consistently across different hand morphologies?

A. Simulation Experiments and Results

We investigate the first question in simulation. We evaluate the quality of the hand pose generation pipeline using IsaacGym [24]. 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. The objects we use for training and evaluation come from [18], consisting of diverse daily objects. 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 on our website for more discussion). A common failure mode in the preliminary experiments is that some generated poses still collide with the object, which motivates the later inclusion of the collision term in Eq. 3 for motion planning to further avoid collisions before executing pushing and pulling motions in the real world.

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

TABLE I: Number of objects with ≥ 1 feasible pushing hand pose out of 300.

B. Real-World Experiments Setup

We evaluate GD2P on a real robot to check if our pushing and pulling hand poses successfully transfer to reality. Our hardware setup consists of a Franka Panda arm equipped with a four-finger, 16-DOF Allegro Hand (see the Appendix on our website for detailed setup visualization). It operates over a tabletop cutting board with dimensions 60 cm×60 cm. We use a mix of objects, including 3D-printed and common daily items (shown in Fig. 5). All evaluation objects are unseen during training. We obtain the object BPS representation as stated in IV-B. While this pipeline introduces some noise, it is sufficient for GD2P 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 GD2P with:

- **Pre-Trained Grasp Pose**: We use a pre-trained grasp synthesis model from [18] using NeRF [64]. 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 GD2P. This tests out-of-distribution generalization with a retrieval-only approach compared to our proposed generative model.

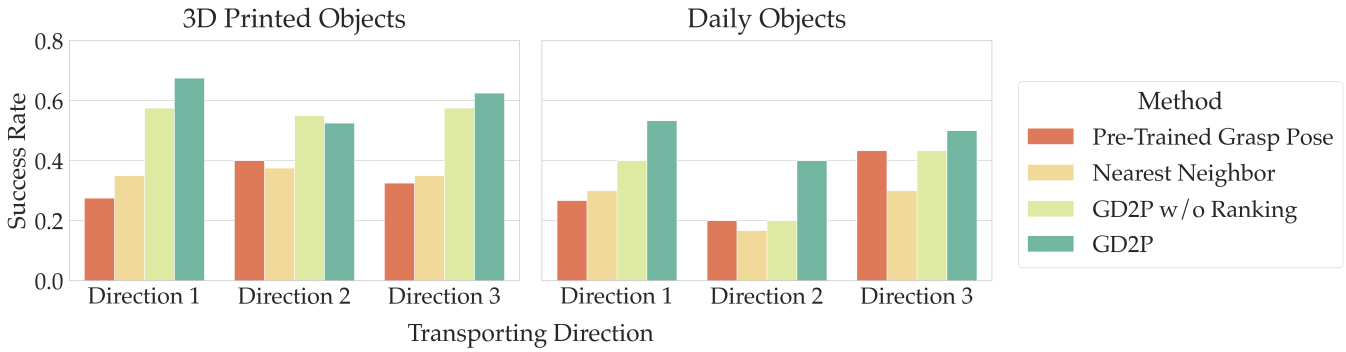

Fig. 6: Pushing and pulling success rates from GD2P and baselines, across different 3D printed (left) and daily objects (right), and with three directions evaluated. Each bar aggregates success rates from 40 trials (left bar plot) and 30 trials (right bar plot). See Sec. V-B and V-C for more details.

- **GD2P w/o Ranking**: An ablation that excludes analytical ranking of hand poses (ignores Eq. 3) and executes a random feasible pose. This tests the usefulness of Eq. 3 in selecting poses.

Experiment Protocol and Evaluation. For each object, we test three pushing and pulling directions uniformly distributed around a circle. We consider "pulling" to involve applying force towards the body to move an object. This aligns with "Direction 2" in our real-world experiments. 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 GD2P 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 score from Eq. 3.

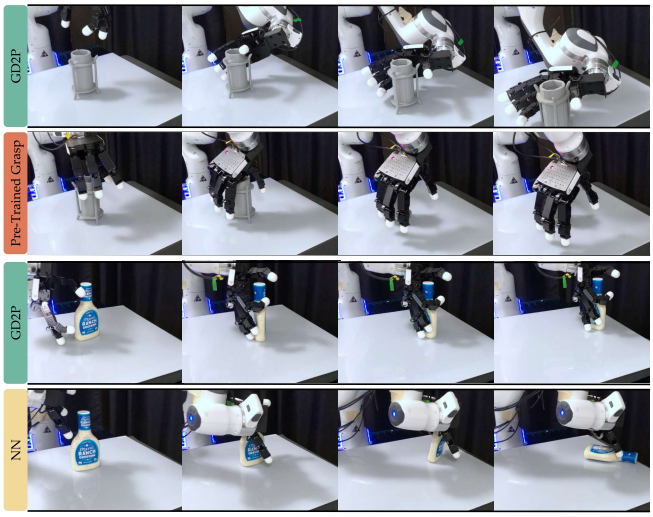

Fig. 7: Comparison between GD2P and baselines. The first two rows show GD2P (success) and Pre-Trained Grasp (failure) while pushing a 3D-printed vase forward (i.e., away from the robot). The last two rows show GD2P (success) and NN (failure) while pushing a ranch bottle to the right.

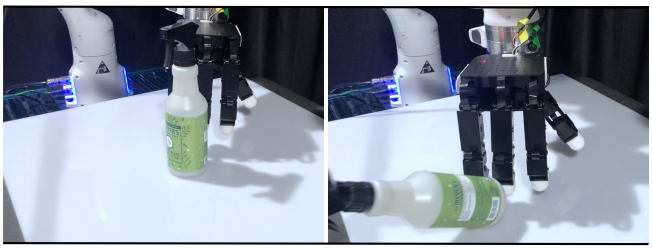

Fig. 8: Example of a typical failure case using the Fixed Hand Pose strategy (see Sec. V-C for details), which topples the spray.

C. *Is GD2P Effective for Diverse Objects and Directions?*

We summarize quantitative results in Fig. 6, which shows that GD2P outperforms or matches alternative methods for both object categories. As shown in Fig. 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 (Fig. 7, second row). Second, some objects are unsuitable for grasping with a single hand due to their geometry or awkward aspect ratios (e.g., a flat box with limited area for enclosure). 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 **GD2P w/o ranking**, we observe that its hand poses are more likely to collide with the table or objects. To further investigate this ablation, Fig. 4 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. **GD2P** outperforms baselines in all directions tested in Fig. 6, demonstrating the robustness of its generated hand poses for pushing and pulling objects. Fig. 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.

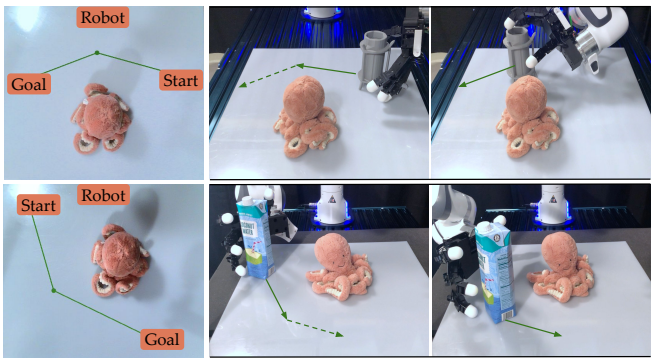

Fig. 9: Path planning using RRT* for multi-step planning. The first column shows the visualization of path planning results. The second and third columns show two consecutive hand poses for pushing the object along the path.

For more rollouts, see the Appendix and the website.

Fixed Hand Pose: Inspired by prior pushing work [15], we manually define a "spatula" hand pose with the fingers spread flat (see Fig. 8) to assess whether simple flat-hand strategies suffice for diverse objects. We perform a case study on the 6 objects in Fig. 5 that are taller than 20 cm. We push each object 10 times and get a relatively low 18/60 success rate, suggesting insufficient object support.

D. Can GD2P Support Multi-step Manipulation Tasks?

GD2P can serve as a reliable module for multi-step pushing. 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. IV-C). By doing so, GD2P can be used to perform multiple pushes. Fig. 9 shows a multi-step pushing sequence using GD2P. 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.

E. Does GD2P Apply to Different Hand Morphologies?

GD2P achieves consistent performance across various robotic hand designs. To evaluate the robustness of GD2P across different robotic hands, we applied the same pipeline on a four-finger, 16-DOF LEAP Hand [16] with an xArm7 arm and conducted experiments following the procedure described in Section V-B (see Fig. 10). Specifically, we tested on the same set of 14 objects over three directions in 210 new trials (not part of the 840 trials in Sec. V-C) and achieved a comparable success rate of 68.1%. These results highlight the potential of GD2P for generating effective dexterous pushing and pulling policies across different robotic hands.

VI. Limitations and Conclusion

While promising, the GD2P approach has some limitations that motivate exciting directions for future work. First, our study considers only pushing and pulling as examples of nonprehensile manipulation and thus does not exhaustively characterize all possible nonprehensile manipulation procedures; we aim to expand to a broader repertoire (e.g., tilting, rolling) that leverages environmental interactions with dexterous hands to enhance nonprehensile dexterity. Second, we aim to develop a closed-loop nonprehensile manipulation policy that adapts in real time to different object physical properties and environmental changes.

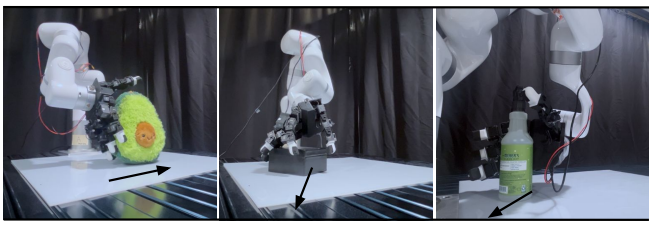

Fig. 10: Pushing and pulling experiments with a LEAP Hand, following the same protocol and evaluations from Section V-B. We evaluates using the same 14 objects across three directions. The first two images illustrate pushing motions and the third illustrates a pulling motion of the spray bottle toward the robot base.

To conclude, we propose GD2P, a dataset and method for nonprehensile object pushing and pulling using high-DOF robotic hands, such as the Allegro and LEAP Hands. Our extensive real-world results show that GD2P enables diverse and effective pre-contact hand poses for different combinations of objects and pushing directions. We also demonstrate its usage for multi-step planning. We hope this inspires future work on dexterous nonprehensile manipulation.

References

[1] S. Prokudin, C. Lassner, and J. Romero, "Efficient learning on point clouds with basis point sets," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[2] Z. Weng, H. Lu, D. Kragic, and J. Lundell, "Dexdiffuser: Generating dexterous grasps with diffusion models," in *IEEE Robotics and Automation Letters (RA-L)*, 2024.

[3] B. Sundaralingam *et al.*, "Curobo: Parallelized collision-free minimum-jerk robot motion generation," *arXiv preprint arXiv:2310.17274*, 2023.

[4] M. T. Mason, "Mechanics and Planning of Manipulator Pushing Operations," in *International Journal of Robotics Research*, 1986.

[5] K. Lynch, "Nonprehensile robotic manipulation: Controllability and planning," Ph.D. dissertation, Carnegie Mellon University, 1996.

[6] M. T. Mason, "Progress in Nonprehensile Manipulation," in *International Journal of Robotics Research (IJRR)*, 1999.

[7] K. M. Lynch and M. T. Mason, "Dynamic nonprehensile manipulation: Controllability, planning, and experiments," in *International Journal of Robotics Research (IJRR)*, 1999.

[8] W. Zhou and D. Held, "Learning to Grasp the Ungraspable with Emergent Extrinsic Dexterity," in *Conference on Robot Learning (CoRL)*, 2022.

[9] W. Zhou, B. Jiang, F. Yang, C. Paxton, and D. Held, "HACMan: Learning Hybrid Actor-Critic Maps for 6D Non-Prehensile Manipulation," in *Conference on Robot Learning (CoRL)*, 2023.

[10] B. Jiang, Y. Wu, W. Zhou, C. Paxton, and D. Held, "Hacman++: Spatially-grounded motion primitives for manipulation," in *Robotics: Science and Systems (RSS)*, 2024.

[11] Y. Cho, J. Han, Y. Cho, and B. Kim, "Corn: Contact-based object representation for nonprehensile manipulation of general unseen objects," in *International Conference on Learning Representations (ICLR)*, 2024.

[12] J. Lyu, Z. Li, X. Shi, C. Xu, Y. Wang, and H. Wang, "Dywa: Dynamics-adaptive world action model for generalizable non-prehensile manipulation," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

[13] K. Zhang, B. Li, K. Hauser, and Y. Li, "Adaptigraph: Material-adaptive graph-based neural dynamics for robotic manipulation," in *Robotics: Science and Systems (RSS)*, 2024.

[14] C. Chi *et al.*, "Diffusion Policy: Visuomotor Policy Learning via Action Diffusion," in *Robotics: Science and Systems (RSS)*, 2023.

[15] Y. Wang, Y. Li, K. Driggs-Campbell, L. Fei-Fei, and J. Wu, "Dynamic-resolution model learning for object pile manipulation," in *Robotics: Science and Systems (RSS)*, 2023.

[16] K. Shaw, A. Agarwal, and D. Pathak, "LEAP Hand: Low-Cost, Efficient, and Anthropomorphic Hand for Robot Learning," in *Robotics: Science and Systems (RSS)*, 2023.

[17] Y. Wang, Y. Li, Y. Yang, and Y. Chen, "Dexterous non-prehensile manipulation for ungraspable object via extrinsic dexterity," *arXiv preprint arXiv:2503.23120*, 2025.

[18] T. G. W. Lum *et al.*, "Get a Grip: Multi-Finger Grasp Evaluation at Scale Enables Robust Sim-to-Real Transfer," in *Conference on Robot Learning (CoRL)*, 2024.

[19] R. Wang *et al.*, "DexGraspNet: A large-scale robotic dexterous grasp dataset for general objects based on simulation," in *International Conference on Robotics and Automation (ICRA)*, 2023.

[20] J. Zhang *et al.*, "DexGraspNet 2.0: Learning Generative Dexterous Grasping in Large-scale Synthetic Cluttered Scenes," in *Conference on Robot Learning (CoRL)*, 2024.

[21] Y. Xu *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.

[22] W. Wan *et al.*, "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.

[23] Y. Li *et al.*, "Grasp multiple objects with one hand," in *IEEE Robotics and Automation Letters (RA-L)*, 2024.

[24] V. Makoviychuk *et al.*, "Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning," *arXiv preprint arXiv:2108.10470*, 2021.

[25] N. Chavan-Dafle *et al.*, "Extrinsic dexterity: In-hand manipulation with external forces," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[26] W. Yang and M. Posa, "Dynamic on-palm manipulation via controlled sliding," in *Robotics: Science and Systems (RSS)*, 2024.

[27] S. Chen, A. Wu, and C. K. Liu, "Synthesizing dexterous nonprehensile pregrasp for ungraspable objects," in *ACM SIGGRAPH*, 2023.

[28] A. Wu, R. Wang, S. Chen, C. Eppner, and C. K. Liu, "One-Shot Transfer of Long-Horizon Extrinsic Manipulation Through Contact Retargeting," *arXiv preprint arXiv:2404.07468*, 2024.

[29] T. Yonemaru, W. Wan, T. Nishimura, and K. Harada, "Learning to Group and Grasp Multiple Objects," *arXiv preprint arXiv:2502.08452*, 2025.

[30] J. J. Liu, Y. Li, K. Shaw, T. Tao, R. Salakhutdinov, and D. Pathak, "Factr: Force-attending curriculum training for contact-rich policy learning," in *Robotics: Science and Systems (RSS)*, 2025.

[31] M. Liu, Z. Pan, K. Xu, K. Ganguly, and D. Manocha, "Deep differentiable grasp planner for high-dof grippers," in *Robotics: Science and Systems (RSS)*, 2020.

[32] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," *IEEE Robotics & Automation Magazine*, 2004.

[33] H. Jiang, S. Liu, J. Wang, and X. Wang, "Hand-object contact consistency reasoning for human grasps generation," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[34] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *International Conference on Learning Representations*, 2014.

[35] D. Turpin *et al.*, "Grasp'd: Differentiable contact-rich grasp synthesis for multi-fingered hands," in *European Conference on Computer Vision (ECCV)*, 2022.

[36] D. Turpin *et al.*, "Fast-grasp'd: Dexterous multi-finger grasp generation through differentiable simulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[37] T. Liu, Z. Liu, Z. Jiao, Y. Zhu, and S.-C. 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.

[38] J. He *et al.*, "Dexvlg: Dexterous vision-language-grasp model at scale," *arXiv preprint arXiv:2507.02747*, 2025.

[39] J. Ye *et al.*, "Dex1b: Learning with 1b demonstrations for dexterous manipulation," in *Robotics: Science and Systems (RSS)*, 2025.

[40] O. Kroemer, S. Niekum, and G. Konidaris, "A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms," in *Journal of Machine Learning Research (JMLR)*, 2021.

[41] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware," in *Robotics: Science and Systems (RSS)*, 2023.

[42] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile ALOHA: Learning Bimanual Mobile Manipulation with Low-Cost Whole-Body Teleoperation," in *Conference on Robot Learning (CoRL)*, 2024.

[43] H. Qi, A. Kumar, R. Calandra, Y. Ma, and J. Malik, "In-Hand Object Rotation via Rapid Motor Adaptation," in *Conference on Robot Learning (CoRL)*, 2022.

[44] J. Wang *et al.*, "Lessons from Learning to Spin "Pens"," in *Conference on Robot Learning (CoRL)*, 2024.

[45] OpenAI *et al.*, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.

[46] H. Jiang, Y. Wang, H. Zhou, and D. Seita, "Learning to Singulate Objects in Packed Environments using a Dexterous Hand," in *International Symposium on Robotics Research (ISRR)*, 2024.

[47] L. Xu *et al.*, "Dexsingrasp: Learning a unified policy for dexterous object singulation and grasping in cluttered environments," *arXiv preprint arXiv:2504.04516*, 2025.

[48] S. He *et al.*, "Sequential multi-object grasping with one dexterous hand," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

[49] T. Chen, E. Cousineau, N. Kuppuswamy, and P. Agrawal, "Vegetable Peeling: A Case Study in Constrained Dexterous Manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[50] T. Lin, Z.-H. Yin, H. Qi, P. Abbeel, and J. Malik, "Twisting Lids Off with Two Hands," in *Conference on Robot Learning (CoRL)*, 2024.

[51] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu, "3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations," in *Robotics: Science and Systems (RSS)*, 2024.

[52] T. G. W. Lum, O. Y. Lee, C. K. Liu, and J. Bohg, "Crossing the Human-Robot Embodiment Gap with Sim-to-Real RL using One Human Demonstration," in *Conference on Robot Learning (CoRL)*, 2025.

[53] J. Chen, Y. Chen, J. Zhang, and H. 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.

[54] P. Perugini, J. Lundell, K. Friedl, and D. Kragic, "Pushing everything everywhere all at once: Probabilistic prehensile pushing," *IEEE Robotics and Automation Letters*, 2025.

[55] T. Tieleman and G. Hinton, *Lecture 6.5—rmsprop: Divide the gradient by a running average of its recent magnitude*, COURSERA: Neural Networks for Machine Learning, 2012.

[56] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, 1983.

[57] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *MICCAI*, 2015.

[58] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Neural Information Processing Systems*, 2020.

[59] H. Lou *et al.*, "Robo-gs: A physics consistent spatial-temporal model for robotic arm with hybrid representation," *arXiv preprint arXiv:2408.14873*, 2024.

[60] M. Tancik *et al.*, "Nerfstudio: A modular framework for neural radiance field development," in *SIGGRAPH*, 2023.

[61] C. Ye *et al.*, "Stablenormal: Reducing diffusion variance for stable and sharp normal," *arXiv preprint arXiv:2406.16864*, 2024.

[62] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, "2D Gaussian Splatting for Geometrically Accurate Radiance Fields," in *SIGGRAPH*, ACM, 2024.

[63] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research (IJRR)*, 2011.

[64] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," in *European Conference on Computer Vision (ECCV)*, 2020.