Sirui Chen University of Hong Kong HKSAR, China ericcsr@connect.hku.hk Albert Wu Stanford University California, USA amhwu@stanford.edu C. Karen Liu Stanford University California, USA karenliu@cs.stanford.edu



Figure 1: We propose a method to synthesize physically realistic nonprehensile pregrasp motions. Our method automatically discovers various strategies to leverage contacts with the surrounding environment and the hand

ABSTRACT

Daily objects embedded in a contextual environment are often ungraspable initially. Whether it is a book sandwiched by other books on a fully packed bookshelf or a piece of paper lying flat on the desk, a series of nonprehensile pregrasp maneuvers is required to manipulate the object into a graspable state. Humans are proficient at utilizing environmental contacts to achieve manipulation tasks that are otherwise impossible, but synthesizing such nonprehensile pregrasp behaviors is challenging to existing methods. We present a novel method that combines graph search, optimal control, and a learning-based objective function to synthesize physically realistic and diverse nonprehensile pre-grasp motions that leverage the external contacts. Since the "graspability" of an object in context with its surrounding is difficult to define, we utilize a dataset of dexterous grasps to learn a metric which implicitly takes into account the exposed surface of the object and the finger tip locations. Our method can efficiently discover hand and object trajectories that are certified to be physically feasible by the simulation and kinematically achievable by the dexterous hand. We evaluate our method on eight challenging scenarios where nonprehensile pregrasps are required to succeed. We also show that our method can be applied to unseen objects different from those in the training dataset. Finally, we report quantitative analyses on generalization and robustness of our method, as well as an ablation study.

 SIGGRAPH '23 Conference Proceedings, August 6–10, 2023, Los Angeles, CA, USA
 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM. ACM ISBN 979-8-4007-0159-7/23/08...\$15.00
 https://doi.org/10.1145/3588432.3591528

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

Character animation, Dexterous manipulation, Robotics

ACM Reference Format:

Sirui Chen, Albert Wu, and C. Karen Liu. 2023. Synthesizing Dexterous Nonprehensile Pregrasp for Ungraspable Objects. In Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings (SIGGRAPH '23 Conference Proceedings), August 6–10, 2023, Los Angeles, CA, USA. ACM, New York, NY, USA, 11 pages. https://doi.org/10. 1145/3588432.3591528

1 INTRODUCTION

Manipulating objects with a dexterous multi-fingered hand is a key human ability. In particular, humans are proficient at leveraging environmental contacts to perform tasks that are otherwise impossible, utilizing *nonprehensile manipulation*, a strategy to move objects without establishing a firm grasp first. For instance, when removing a book from a densely packed bookshelf, one would pivot the book outwards with one finger while keeping the book stabilized with lateral environment contacts. Once sufficient area of the book is exposed, the book can be picked up with a simple grasp (Figure 1). This type of non-prehensile "pregrasps" is an essential skill for operating in an ecological human living space in which objects are often initially ungraspable due to occlusions by other surrounding objects.

Replicating human-like nonprehensile manipulation is challenging to existing motion planning methods. The inclusion of environmental contacts leads to combinatorial complexity in the number of possible contact configurations, which is intractable to even state-ofthe-art motion planners. Moreover, it is difficult to define heuristics for the purpose of trajectory optimization. Consequently, many

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

papers in both the dexterous grasp generation literature (e.g., [Jiang et al. 2021; Karunratanakul et al. 2020; Taheri et al. 2020]) and the motion generation literature (e.g., [Zhang et al. 2021]) involve motion generation through learning-based methods. Nevertheless, the rich contact constraints in manipulation makes it difficult for learning methods to produce physically realistic motions with no interpenetrating or telekinetic interactions. The tradeoff between computational efficiency and physical realism remains a major hurdle in synthesizing manipulation motion.

In this paper, we propose a method to synthesize physically realistic nonprehensile pregrasp motions that make ungraspable objects in cluttered environments graspable. In particular, this is achieved through leveraging contacts with the surrounding environment and the hand. Our key observation is that, for any object placed in the environment, *at most two* finger contacts are necessary for nonprehensile pregrasp since there exist at least one external contact point which collectively achieves wrench closure with the two finger contacts. This significantly reduces the search space of our motion synthesis problem, and allows us to formulate pregrasp planning as a reasonably sized mixed-integer optimization problem. Under this observation, our method solves for sequences of at most two contact points on the object by formulating a combination of discrete graph search and trajectory optimization.

Defining an exact metric for "graspability" during the pregrasp phase is not trivial. The obvious goal in this phase is to expose the surface of the object, but it has to be done in such a way that the finger contacts during the pregrasp phase can be fluidly transitioned to a firm grasp later. As such, the graspability we would like to maximize depends on the current state of the object, the environment, the fingers that have established contact points and the fingers that are still free of contact. To define a generalizable metric without relying on heuristics, we learn a general metric of "graspability" through a dataset of dexterous grasps in various scenarios. Using this metric, we perform particle-based trajectory optimization in a physics-based simulator. This allows our method to efficiently discover trajectories that are certified to be physically feasible by the simulation.

We demonstrate our method on eight challenging scenarios where nonprehensile manipulation is required to successfully grasp the object. Our method is able to discover diverse strategies that successfully completes the grasping tasks while satisfying physical constraints. We compare our method to kinematic-based motion generation([Zhang et al. 2021]) and show the motions generated by our method is visually superior with no hand-object interpenetration or telekinetic interaction.

2 RELATED WORK

Manipulating objects with a dexterous hand is a long-standing research challenge that interests both the graphics and robotics communities. In this section, we review literature on physically plausible manipulation planning.

2.1 Dexterous manipulation without environmental interaction

Due to the increased complexity when considering physical laws, many existing contributions assume either that manipulation tasks are done in free space, or that the hand-object interaction is the only relevant interaction. We review two of the most discussed manipulation tasks in this domain.

2.1.1 Dexterous Grasping. Grasping describes the task of generating hand and finger configurations to firmly holds an object of interest. To synthesize grasping motion analytically, some works leverage motion data to design control laws [Pollard and Zordan 2005], compute contact interactions [Kry and Pai 2006], and formulate optimization problems [Zhao et al. 2013]. Other works assume the object trajectory is known and use it as a basis to synthesize hand motion through trajectory optimization [Gleicher 1998; Liu 2009; Ye and Liu 2012]. Additionally, physics-inspired grasp metrics, such as matching geometry [Li et al. 2007], wrench closure and no collision [Ciocarlie and Allen 2009; Ferrari and Canny 1992; Pokorny and Kragic 2013], are commonly applied in these works. More recently, advances in deep generative models(e.g., [Goodfellow et al. 2014; Sohn et al. 2015]) has given rise to learning-based grasp-generation methods for different object geometries [Jiang et al. 2021; Lu et al. 2020; Lundell et al. 2021; Mandikal and Grauman 2021; Romero et al. 2017; Shao et al. 2020; Taheri et al. 2020], some of which also ensures physics feasibility and stability of the grasp [Christen et al. 2022; Wu et al. 2022]. The common limitation of these methods is that they only produce grasps for non-occluded object in free space. We note that there is another branch in the grasping literature, often dubbed as "grasping in clutter" or "bin picking," which studies cluttered-scene object picking with simple grippers. As these works seldom use a dexterous hand, we excluded them from our review.

2.1.2 In-hand manipulation. In-hand manipulation seeks to move an object in a dexterous hand to a desired pose relative to the hand, using only the hand itself. In recent years, in-hand manipulation has become a popular task in the robotics community as a benchmark for challenging physical interaction. Mordtach and colleagues [Mordatch et al. 2012] formulate in-hand manipulation as a trajectory optimization problem with hand-object contact constraints. More recently, reinforcement learning using physics-based simulation has been applied to reorienting objects [Andrychowicz et al. 2020; Chen et al. 2022; Qi et al. 2022] and solving a Rubik's cube [Akkaya et al. 2019]. The complexity of in-hand manipulation originates from the hand-object interaction. Environment contacts do not need to be considered in this task.

2.2 Extrinsic dexterity: dexterous manipulation without environmental interaction

Nonprehensile manipulation with *extrinsic dexterity* [Lynch and Mason 1999] aims to utilize external contact forces from environment. This is especially useful for manipulating objects in cluttered space and greatly expands the scope of possible actions. For instance, extrinsic dexterity may facilitate downstream tasks by exposing previously occluded parts of an object in clutter [Serhan et al. 2022]. Common strategies that rely on external contact interactions include pushing, pivoting and tilting [Aiyama et al. 1993; Eppner et al. 2015]. Among these strategies, pushing has been most extensively investigated. Researchers have explored how to manipulate objects

SIGGRAPH '23 Conference Proceedings, August 6-10, 2023, Los Angeles, CA, USA

on a two-dimensional plane with up to two active contact points applied from a robot [Arruda et al. 2017; Lee et al. 2015; Lowrey et al. 2018; Serhan et al. 2022; Woodruff and Lynch 2017]. Pivoting has also been used in robotics for robot to grasp objects that are initially not graspable [Sun et al. 2020; Zhou and Held 2022]. Combining those primitives, [Eppner and Brock 2015] proposed a graph-based planner for a single DoF dexterous hand to grasp objects. However, general non-prehensile manipulation requires more complex motion planning and often resorts to sampling-based planner [Pollayil et al. 2021]. Nevertheless, all these works use either a fixed-geometry manipulator, simple parallel jaw grippers or underactuated multi-finger hand. Motion synthesis for fully actuated dexterous hand is more challenging because both finger movement planning and grasp generation become much more complex. To the best of our knowledge, no existing work is capable of generating physically plausible motion sequences for a fully actuated multi-fingered dexterous hand using extrinsic dexterity.

3 PRELIMINARY

Before describing our method, we first formally define the "feasibility" of a grasp. A grasp is represented by at most five contact points on the object surface, corresponding to five finger tips: $[p^1, p^2, p^3, p^4, p^5]$. A feasible grasp needs to be both *dynamically* and *kinematically* feasible [Wu et al. 2022].

Dynamic feasibility requires each finger tip to apply a minimal normal force f_{\min} on the object while maintaining zero net wrench. This is referred to as wrench closure [Miller and Allen 2004]. Meanwhile, the contact force applied on each contact point must be bounded within the friction cone specified by the friction coefficient μ . To summarize, the two requirements for a dynamically feasible grasp are:

$$\min_{f} \|\sum_{i=1}^{5} f^{i}\|_{2}^{2} + \|\sum_{i=1}^{5} \boldsymbol{p}^{i} \times f^{i}\|_{2}^{2} = 0,$$
subject to $0 < f_{\min} < -f^{i} \cdot \hat{\boldsymbol{n}}^{i}, \forall i \in \{1, \cdots, 5\}$
 $|f^{i} \cdot \hat{\boldsymbol{t}}^{i,j}| \le \mu f^{i} \cdot \hat{\boldsymbol{n}}^{i}, \forall i \in \{1, \cdots, 5\}, \forall j \in \{1, 2\}$

where f^i is the contact force at the contact point p^i and \hat{n}^i is the surface normal at the contact point. $\hat{t}^j, \forall j \in \{1, 2\}$ are orthogonal basis vectors used to approximate the projection of the friction cone. If the above constrained quadratic optimization problem can be solved with zero optimal value, the grasp is dynamically feasible.

Kinematic feasibility requires the contact points to be reachable by the dexterous hand. Such reachability can be confirmed by solving the inverse kinematics problem for the given contact points.

4 METHOD

We introduce a method to synthesize nonprehensile manipulation given the point cloud of the object of interest O, the environment state relative to the initial object pose S_0 , the learned *score function* f_{θ} : $(p^1, p^2, S, O) \mapsto \mathbb{R}$, and the learned *grasp generator* g_{ϕ} : $(p^1, p^2, O) \mapsto (p^1, p^2, p^3, p^4, p^5)$. The initial environment state S_0 is represented as a signed distance function (SDF) from the point cloud of the object to its surrounding. $p^i, i = 1 \cdots 5$ indicates the position of the five fingertips of an anthropomorphic hand, ordered from the thumb to the little finger and represented in the object coordinate frame. Given the object state, the thumb's and the index finger's locations on the object, the score function f_{θ} evaluates how likely such a two finger contacts may lead to a feasible grasp. With the same finger contact locations, the grasp generator g_{ϕ} , which is the decoder of a conditional variational autoencoder (CVAE), predicts the best locations for the other three fingers. f_{θ} and g_{ϕ} are produced by an offline training process described in Section 4.3.

Our method consists of three steps: 1) Construct the contact state graph; 2) Optimize dynamically feasible contact trajectories and the object trajectory for nonprehensile manipulation; and 3) Synthesize animation. We first construct a contact state graph \mathcal{G} based on the input object O, the initial environment SDF S_0 , and the learned score function f_{θ} . Since at most two finger contacts are required to achieve nonprehensile pregrasp, we can efficiently optimize contact trajectories of the thumb and the index finger on the contact state graph via a sampling-based optimization process. We take a learning approach to define optimality that rewards the hand to manipulate the object into a graspable configuration in a cluttered environment, subject to dynamic constraints. Finally, we solve a sequence of inverse kinematic (IK) problems to produce the animation of a dexterous hand, conditioned on the fingertip contacts and the object motion produced by the previous step. Figure 2 gives an overview of our method.

4.1 Contact state graph

Given an object point cloud O and an initial environment SDF S_0 , we construct a contact state graph G to encode the relationships between different different regions of the object surface. G allows our method to efficiently explore the rich hand-object contact behaviors with optimization (Section 4.2).

Algorithm 1 summarizes the process of constructing the contact graph \mathcal{G} . We first approximate the object surface by fitting a mesh containing 30 to 50 triangles to O via an off-the-shelf mesh simplification algorithm [Garland and Heckbert 1997]. Since we only need to consider the thumb and the index finger thanks to external contacts, we define each node in \mathcal{G} as $v = (\mathcal{T}^1, \mathcal{T}^2)$. \mathcal{T}^1 and \mathcal{T}^2 are the triangles in the mesh that the thumb and the index

ALGORITHM 1: Building contact state graph
Input: Object point cloud O ; SDF of environment obstacles S_0
$\mathcal{M} \leftarrow create_mesh(\mathcal{O})$
$\mathcal{V}_{all} \leftarrow ordered_triangle_pairs(\mathcal{M})$
$S_c \leftarrow calc_nodal_score_in_context(\mathcal{V}_{all}, f_{\theta})$
$S_n \leftarrow calc_nodal_score_no_context(\mathcal{V}_{all}, f_{\theta})$
Remove low scoring nodes
$\mathcal{V}_{c} \leftarrow \texttt{select_top_M}(\mathcal{S}_{c}, \mathcal{V}_{all})$
$\mathcal{V} \longleftarrow \mathcal{V}_c \cup \texttt{select_top_M}(\mathcal{S}_n, \mathcal{V}_{\texttt{all}})$
Connect nodes with edges
$\emptyset \longrightarrow 3$
for u, v in \mathcal{V} do
$\begin{vmatrix} \mathbf{if} \ \mathcal{T}_u^1 = \mathcal{T}_v^1 \ or \ \mathcal{T}_u^2 = \mathcal{T}_v^2 \ \mathbf{then} \\ \ \mathcal{E}. \operatorname{add}(u, v) \end{vmatrix}$
end
end
$\mathcal{G} \leftarrow \{\mathcal{V}, \mathcal{E}\}$
Output: Nodes for initial state \mathcal{V}_c ; Contact state graph \mathcal{G}

Sirui Chen, Albert Wu, and C. Karen Liu



Figure 2: An overview of our pipeline. Using plate grasping as an example, (a)-(e) illustrate the steps. (a) Input point cloud. (b) Contact state graph. (c) Thumb and index fingertip motion, as well as object motion obtained from the trajectory optimizer. (d) A full grasp generated conditioned on the last frame of the trajectory. (e) Resulting hand motion after the IK process.

finger are in contact with respectively. Because many of the triangle pairs are not ideal contact locations for nonprehensile pregrasp, we use the learned score function f_{θ} to prune \mathcal{G} and only keep the high-scoring nodes. For each node v, we compute a score s_c that approximates the success likelihood of the node within the context of the environment, and a score s_n that does not consider the environment:

$$s_{\boldsymbol{c}} = f_{\boldsymbol{\theta}}(\boldsymbol{p}^1, \boldsymbol{p}^2, \mathcal{S}_0, \boldsymbol{O}'); \quad s_{\boldsymbol{n}} = f_{\boldsymbol{\theta}}(\boldsymbol{p}^1, \boldsymbol{p}^2, \boldsymbol{\emptyset}, \boldsymbol{O}), \quad (1)$$

where O' is a subset of O with occluded points removed.

We sample five fingertip locations within \mathcal{T}^1 and \mathcal{T}^2 associated with \boldsymbol{v} for each of the three scenarios: thumb-only contact, indexfinger-only contact, and two-finger contact. We query f_{θ} for these 15 fingertip placements and compute the average s_c and s_n for \boldsymbol{v} . If either s_c or s_n is within the top M percentile among all nodes, we include \boldsymbol{v} in \mathcal{G} . If \boldsymbol{v} is selected due to a high s_c , we put it in a subset of selected nodes called \mathcal{V}_c . This subset of nodes will be used for the initial contact state in the optimization process (Section 4.2).

Each edge of \mathcal{G} represent a feasible contact state transition. To prevent both fingers from simultaneously changing contact locations, we only connect two nodes by an edge if they share either \mathcal{T}^1 or \mathcal{T}^2 . Each node also connects to itself to allow for consecutive unchanged contact states. This is illustrated in Figure 3.

4.2 Trajectory optimization

Once \mathcal{G} is constructed, the next step is to search for optimal trajectories for the thumb contact, the index-finger contact, and the 6D pose of the object such that a wrench-closure grasp using all five fingers can be achieved successfully at the end. The optimization is a double-loop iterative process. The outer loop solves a discrete path planning problem while the inner loop solves a continuous optimal control problem. Algorithm 2 shows the detailed procedure of this optimization.

A *path* is a sequence of *N* nodes connected by edges on *G*. It determines the *N contact stages* of the animation. Since the object is positioned in the context of the environment initially, we restrict the first node of each path to be a node in the subset V_c . The remaining N - 1 nodes can be any node on *G*. The score of a path



Figure 3: Different types of connectivity between nodes in the contact graph.

is computed by accumulating the score at each contact stage:

$$s_p = s_c(\boldsymbol{v}_1) + \sum_{i=2}^N s_n(\boldsymbol{v}_i), \qquad (2)$$

where s_c and s_n are overloaded to indicate functions that return the respective score of the input node. Using Equation 2, we compute the scores for all possible paths and sort the paths by their scores from high to low.

The outer loop of the optimization iterates over the sorted list of paths until a successful trajectory is found by the inner loop. For each given path (v_1, \dots, v_N) , the inner loop solves for contact locations of the fingertips subject to staying within the triangles associated with the nodes in the path. Formally, we optimize the state variable $x_i = (p_i^1, p_i^2, b_i^1, b_i^2)$ where $i = 1 \dots N$, and the control variables $u_t = (\bar{p}_t^1, \bar{p}_t^2)$ where $t = K \cdot N$. p_i^1 and p_i^2 are the thumb position and the index-finger position in the object coordinate frame at contact stage *i*. To distinguish between thumb-only, index-finger-only, or two-finger contact scenarios, we define binary variables b_i^1 and b_i^2 to indicate whether the respective finger is in contact with the object at contact stage *i*. We use a proportional derivative-like control scheme to control the contact forces exerted

by the fingers on the object: $f = -k(p^j - \bar{p}^j)$, where k = 50 is a predefined coefficient. p^j and \bar{p}^j are the current fingertip location and its target location. We define the control variables $(\bar{p}_t^1, \bar{p}_t^2)$ as the target locations of the thumb and the index fingertip in the world coordinate frame at frame *t*. Because the contact forces tend to change at a higher rate than the contact locations, we allow the control variables to change *K* times within each contact stage.

The objective function of the inner optimization is as follows:

$$\mathcal{L}(\boldsymbol{x}_{1:N}, \boldsymbol{u}_{1:KN}) = \sum_{i=i}^{N} f_{\theta}(\boldsymbol{p}_{i}^{1} \wedge b_{i}^{1}, \boldsymbol{p}_{i}^{2} \wedge b_{i}^{2}, \mathcal{S}_{i}, O_{i}'), \quad (3)$$

where we use $p \wedge b$ to denote "p if b is true, \emptyset otherwise." Note that the environment SDF S_i depends on the object pose and is essentially a function of x and u via physics simulation. We compute S_i as the environment SDF corresponding to the object pose at the beginning of contact stage i. Similarly, we remove occluded points from O to form O'_i based on the object pose at the beginning of contact stage i.

To ensure the contact locations staying within the boundary of the given triangles, we use barycentric coordinates to parameterize p^1 and p^2 . To further ensure the physical plausibility of the finger motion, a finger can only change its location after it is detached from the object in the previous contact stage. For example, b_{i-1}^2 must be false for the index finger to change the location at contact stage *i*. This additional constraint prevents fingers from jumping instantaneously from one location to another.

We use a sampling-based optimizer, MPPI [Williams et al. 2017] since gradient-based optimization algorithms tend to be stuck in local minima when solving contact-rich control tasks [Suh et al. 2022]. After solving each trajectory optimization problem, we query the grasp generator g_{ϕ} to complete a five-finger grasp based on the final locations of thumb and/or the index fingertips, as well as the object point cloud:

$$(\boldsymbol{p}^{1}, \boldsymbol{p}^{2}, \boldsymbol{p}^{3}, \boldsymbol{p}^{4}, \boldsymbol{p}^{5}) = g_{\phi}(\boldsymbol{p}^{1}_{N}, \boldsymbol{p}^{2}_{N}, O)$$
(4)

There are three possible outcomes during the nonprehensile manipulation phase: 1) only thumb is in contact, 2) only index finger is in contact 3) both thumb and index finger are in contact. [100,100,100] is used as a invalid token if a finger is not in contact. Lastly, we test whether a force-closure grasp can be formed by any combination of the three finger contacts from the set $(\mathbf{p}_1^1, \mathbf{p}_2^2, \mathbf{p}^3, \mathbf{p}^4, \mathbf{p}^5)$. Since g_{ϕ} is the decoder of a CVAE, we sample the latent space 20 times to generate 20 different grasps. If any one of them is feasible, we exit the outer loop and the optimization is complete.

4.3 Grasp generator and score function

Grasp generator: We train the grasp generator, g_{ϕ} , as three conditional variational autoencoders (CVAE). Once trained, the decoder is used to predict the contact locations for the middle, ring, and little fingers conditioned on the initial contact locations of the thumb and/or the index finger, as well as the object point cloud expressed in the object coordinate frame (Figure 4). The training data for the grasp generator is a synthetic dataset of feasible grasps of various objects in different environmental contexts. We create the dataset using the YCB object set [Calli et al. 2015]. For each object, we first manually define 3 feasible grasps as seed grasps. For each seed SIGGRAPH '23 Conference Proceedings, August 6-10, 2023, Los Angeles, CA, USA



Figure 4: Overview of the grasp CVAE and the score function.

grasp, we perturb the finger contact locations and check whether the perturbed grasp is still both kinematically and dynamically feasible. If so, the perturbed grasp becomes a new seed grasp and added to the dataset. The process repeats until sufficient feasible grasps are generated for this object.

Score function: Given the thumb and the index finger contact locations, the environment SDF, and the object point cloud expressed in the object coordinate frame, we train a Multilayer Perceptron (MLP), f_{θ} , to evaluate how likely the two finger contacts will lead to a successful feasible grasp (Figure 4). We use PointNet++ [Qi et al. 2017] to encode the input point cloud and the SDF. The output encoding is concatenated with the contact locations of the thumb and the index finger to form the input of the MLP. We leverage the grasp generator to create the labels for our training data. For each data point, we first randomly select one or two contact points on the object which are allocated as thumb and/or index finger contact location. Conditioned on these contact points, we sample the latent space of the corresponding grasp generator *P* times to complete *P* grasps. We then check the feasibility of the *P* grasps and assign the label as "the ratio of the number of feasible grasps to *P*."

ALGORITHM 2: Pregrasp nonprehensile manipulation optimization

4.4 Hand motion synthesis

Given the finger contact trajectories and the object trajectory from the previous step, we solve an inverse kinematics (IK) problem to obtain the detailed hand pose for each frame of the final animation. We frame the IK problem as an optimization with an objective function that (1) ensures the fingertips of the hand satisfy the contact trajectories, (2) avoids collisions with the other objects in the environment and self-collision with the hand, and (3) encourages smoothness in the kinematic trajectory.

The solution to such a nonconvex optimization is very sensitive to the initial guess. A typical strategy is to solve the IK problem frame-by-frame in chronological order, so the decision variables of the current frame can be initialized with the solution from the previous frame for increaseed motion continuity. However, we found that using a reverse chronological order, multi-layered strategy to solve the animation frames results in higher quality animation. We first solve IK for the N+1 "keyframes", which include the first frame of each contact stage and the last frame of the animation. Once all keyframes are solved, we can interpolate the hand poses to create initial guesses for the in-between frames. The key frames are solved in a reverse chronological order such that the current keyframe's optimization is initialized with the solution of the optimizating for the next keyframe in the optimization. This is because the final keyframe involves all five fingers, resulting in a more constrained IK problem than keyframes earlier in the animation sequence. In practice, the IK almost always solved. If the IK solver fails after multiple attempts, we make a slight compromise on quality and manually loosen the optimization's constraint tolerances, such as those for avoiding collision and matching desired contact locations.

5 EVALUATION

We evaluate our method qualitatively by demonstrating a rich and diverse set of synthesized hand animations. We also provide quantitative analysis on our method's robustness, generalizability, and ablations. Finally, we compare with the most relevant method in literature, ManipNet[Zhang et al. 2021]. We use a five-finger Shadow Hand model with 28 degrees of freedom (22 joints and 6*D* wrist pose) in our implementation. Our approach can be applied to other hand models as long as all relevant training is done with the new hand model. We use PyBullet [Coumans and Bai 2016] for physics simulation and MPPI [Williams et al. 2017] for trajectory optimization. Inverse kinematics problems are solved using the collision-free IK Module in Drake [Tedrake et al. 2019], which uses [Gill et al. 2005] as the backend optimizer.

5.1 Motion quality and diversity

We design eight diverse scenarios to test our method, including (1) adjacent books on a bookshelf, (2) a ruler lying flat on the edge of a table, (3) a computer keyboard and (4) a flat cardboard placed on a shelf against a wall, (5) a dinner plate and (6) a food container placed on a the table, (7) a densely packed spice rack, and (8) a marker stored in a pencil box. We show that in all scenarios, our method is able to synthesize natural and physically plausible nonprehensile pregrasps to manipulate the objects into a graspable configuration. We observe four distinct strategies used by the dexterous hand: (1) repositioning the object to expose the bottom surface (ruler,

cardboard), (2) pivoting the object against its surroundings (marker, keyboard, book, spice bottle), (3) utilizing finger friction (food container), and (4) utilizing geometric features (plate). Figure 5 and the supplemental video show animations of our results.

To further demonstrate the diversity of motions generated by our method, we showcase some scenarios where multiple successful pregrasp strategies exist. By searching over more paths on the contact graph instead of exiting upon the first successful path as in 2, our method is able to generate visually and functionally different manipulation strategies. Figure 6(a) and 6(b) show two different ways to pick up the cardboard: one uses the index finger to pull the cardboard outward, while the other one uses the thumb. Changing the surrounding of the object can also result in different strategies. If the keyboard is placed near the backboard of the shelf, the hand will push the keyboard is placed near the edge of the shelf, the hand will lift the keyboard from the corner that is fully exposed.

5.2 Quantitative evaluation of algorithms

We define two metrics to evaluate our method. Based on these two metrics, we analyze the generalizability of our method to unseen objects and conduct ablation studies on the design of score function.

Grasp success rate Feasibility of the grasp generated by the grasp generator is the most critical indicator of the success of our method. A feasible grasp must satisfy both dynamic and kinematic constraints (Section 3). Since we use a sampling-based grasp generator (i.e. a CVAE), we compute the grasping success rate r_{suc} based on the first feasible path. r_{suc} is defined as the ratio between the number of feasible grasps and the total number of samples (20 in our implementation) drawn from the latent space.

Number of paths attempted: The sampling-based optimizer may need to sample multiple candidate contact paths before a feasible one is found. The number of paths attempted is another metric to measure the efficiency of our algorithm. For each candidate path, we test a set of 20 grasps generated by different latent space samples of the grasp generator. If there exists one feasible grasp, the path is considered successful. The total number of paths attempted to obtain a feasible grasp is denoted as N_{paths} .

We evaluate r_{suc} and N_{paths} on all eight scenarios in Figure 5 and report the results in Table 1. The results demonstrate that our grasp generator can reasonably predict full grasp condition on final object pose and thumb and index finger contact location with high success rate. All tasks are completed within 20 minutes of runtime on a desktop computer with Intel i9-9900K and NVIDIA RTX2080.

5.3 Generalization to unseen objects

Since our method contains learning-based components trained on a dataset of objects, it is crucial to demonstrate generalizability to unseen objects. All eight aforementioned scenarios were tested with unseen objects not included in the YCB training dataset. In addition, we evaluate the method on two nonconvex objects. For those objects that can be reasonably approximated by their convex hulls, such as a paper roll and a cookie jar, our method can successfully synthesize physically plausible motions. However, for highly non-convex objects, our current implementation is limited by two

$r_{\rm suc}$ and $N_{\rm path}$ of all tasks					
	Bookshelf	Plate	Marker	Ruler	
r _{suc}	0.917(0.058)	0.800(0.050)	0.933(0.058)	0.817(0.144)	
N _{path}	1.667(1.155)	1.667(1.155)	3.667(2.082)	3.333(1.528)	
	Waterbottle	Food container	Keyboard	Cardboard	
r _{suc}	0.850(0.180)	0.733(0.340)	0.817(0.076)	0.800(0.229)	
N _{path}	3.000(1.732)	2.000(1.000)	1.667(0.577)	1.333(0.577)	

Table 1: Quantitative result of all environments. Means and standard deviations are obtained from 3 experiments. Computational budget is 10 paths.

components that requires convexity: the mesh reconstruction algorithm, and the collision avoidance objective in Drake's IK solver (Figure 9). More failures and limitations are in our video.

We also evaluate generalizability across object size scaling using r_{suc} and N_{paths} as metrics. Table 2 shows that our method could perform reasonably well with different scale of object in Keyboard environment. In more challenging Bookshelf environment, changing object scale may affect object's physics property such as center of mass and accessibility, causing performance fluctuation.

$r_{\rm suc}$ and $N_{\rm path}$ of different scales						
Bookshelf			keyboard			
scale(x,y,z)	r _{suc}	N _{path}	r _{suc}	N _{path}		
[1,1,1]	0.917(0.058)	1.667(1.155)	0.816(0.076)	1.667(0.577)		
[0.5,1,1]	0.967(0.029)	1.000(0.000)	0.683(0.126)	1.333(0.577)		
[1,0.5,1]	NA*	NA*	0.816(0.104)	2.333(1.155)		
[1,1,0.5]	0.833(0.104)	1.667(0.577)	0.900(0.050)	2.333(1.528)		
[1,1,0.02]	$0.500(0.071)^*$	$2.000(1.414)^*$	0.800(0.218)	3.000(2.000)		

Table 2: Results of various object sizes. Means and standard deviations are obtained from 3 optimizations. Computation budget is 10 paths. * indicates the algorithm occasionally failed within budget. NA* means all 3 executions failed.

5.4 Ablation on score function design

We perform an ablation study on the design choice of the score function. We evaluate two variants of the score function: 1) No SDF: only use positions of the points on the object surface as input to the point cloud encoder; 2) No Shape: only use signed distance values as input to the point cloud encoder. We compute r_{suc} and N_{paths} on these two variants using the plate and the marker examples. 3 shows that having both SDF and object shape information is crucial for accurately assessing the score of a grasp.

5.5 Ablation on different orders of solving IK

We compare solving sequential IK in a forward and reverse order. It shows that the reverse order achieves better consistency (Figure 7). Solving IK in a forward order results in palm flipping abruptly because the IK solver cannot foresee the next grasping pose. SIGGRAPH '23 Conference Proceedings, August 6-10, 2023, Los Angeles, CA, USA

$r_{\rm suc}$ and $N_{\rm nath}$ of score function variants						
	Plate		Food container			
	r _{suc}	N _{path}	r _{suc}	N _{path}		
Ours	0.800(0.050)	1.667(1.155)	0.733(0.340)	2.000(1.000)		
No SDF	0.583(0.058)	5.333(2.887)	0.450(0.436)	2.667(1.528)		
No pretrain	0.400(0.132)	6.333(1.155)	NA	NA		

Table 3: Result of the ablation study. Means and standard deviations are obtained from 3 experiments. Computational budget is 10 paths. NA denotes infeasible with budget.

5.6 Comparison with ManipNet

Due to the lack of literature on non-prehensile manipulation for dexterous hand, we compare our method with the state-of-theart, kinematics-based method, ManipNet [Zhang et al. 2021], in a scenario similar to the spice rack example. We pick this example because ManipNet is trained on grasping objects with similar cylinder shapes. In experiment, the object and wrist trajectory is obtained from running our method as ManipNet assumes this information as input. Figure 8 shows that, despite the ability to sense the manipulated object, ManipNet attempts to grasp the bottle without considering the surrounding objects. This results in significant finger-object interpenetration. In contrast, our method pulls out then grasps the bottle while avoiding collision with other objects. While the comparison may arguably be more fair if ManipNet was trained with similar context, recording training data that covers all possible surroundings for all different objects is impractical. This makes ManipNet difficult to extend to contextual environments.

6 CONCLUSIONS

We propose a physics-based method for synthesizing nonprehensile pregrasp animations that grasp an initially ungraspable object. Our method leverages extrinsic dexterity and uses a learned function to evaluate the "graspability" of an object in the context of the environment. We show that our method is capable of discovering a diverse set of pregrasp strategies and producing realistic and physically plausible hand and object motions.

7 LIMITATIONS

Our method has a number of limitations. First, the duration of contact stages is predefined and may lead to unnatural behaviors in some scenarios. Nevertheless, optimizing the timing of pregrasp behaviors is possible once motion data is available. Our approach is also limited to grasping rigid object as the contact graph construction assumes the distances between triangles on the mesh are fixed. Furthermore, the point contact assumption in this work makes two-finger grasps unstable and challenging to achieve. A potential extension is to incorporate other contact points, such as the palm and the knuckles, for exploring different grasp strategies. Lastly, our pipeline is limited by a number of implementation choices. Currently, it cannot manipulate nonconvex objects that are poorly approximated by their convex hulls due to the selected mesh reconstruction method and IK solver. Improving the implementation can remove the convexity requirement and speed up the pipeline.

Sirui Chen, Albert Wu, and C. Karen Liu

ACKNOWLEDGMENTS

We'd like to thank Yifeng Jiang for designing action space. The work is supported by the NSF:CCRI:2120095, Toyota Research Institute(TRI) and Stanford Institute for Human-Centered AI(HAI).

REFERENCES

- Yasumichi Aiyama, Masayuki Inaba, and Hirochika Inoue. 1993. Pivoting: A new method of graspless manipulation of object by robot fingers. In Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93), Vol. 1. IEEE, 136–143.
- Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. 2019. Solving rubik's cube with a robot hand. arXiv preprint arXiv:1910.07113 (2019).
- Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. 2020. Learning dexterous in-hand manipulation. *Int. J. Robotics Res.* 39, 1 (2020). https://doi.org/10.1177/0278364919887447
- Ermano Arruda, Michael J Mathew, Marek Kopicki, Michael Mistry, Morteza Azad, and Jeremy L Wyatt. 2017. Uncertainty averse pushing with model predictive path integral control. In 2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids). IEEE, 497–502.
- Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. 2015. The ycb object and model set: Towards common benchmarks for manipulation research. In 2015 international conference on advanced robotics (ICAR). IEEE, 510–517.
- Tao Chen, Jie Xu, and Pulkit Agrawal. 2022. A system for general in-hand object re-orientation. In Conference on Robot Learning. PMLR, 297–307.
- Sammy Christen, Muhammed Kocabas, Emre Aksan, Jemin Hwangbo, Jie Song, and Otmar Hilliges. 2022. D-grasp: Physically plausible dynamic grasp synthesis for hand-object interactions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 20577–20586.
- Matei T Ciocarlie and Peter K Allen. 2009. Hand posture subspaces for dexterous robotic grasping. The International Journal of Robotics Research 28, 7 (2009), 851–867.

Erwin Coumans and Yunfei Bai. 2016. Pybullet, a python module for physics simulation for games, robotics and machine learning. (2016).

Clemens Eppner and Oliver Brock. 2015. Planning grasp strategies that exploit environmental constraints. In 2015 IEEE international conference on robotics and automation (ICRA). IEEE, 4947–4952.

Clemens Eppner, Raphael Deimel, José Alvarez-Ruiz, Marianne Maertens, and Oliver Brock. 2015. Exploitation of environmental constraints in human and robotic grasping. *The International Journal of Robotics Research* 34, 7 (2015), 1021–1038.

Carlo Ferrari and John F Canny. 1992. Planning optimal grasps.. In *ICRA*, Vol. 3. 6.

- Michael Garland and Paul S Heckbert. 1997. Surface simplification using quadric error metrics. In Proceedings of the 24th annual conference on Computer graphics and interactive techniques. 209–216.
- Philip E. Gill, Walter Murray, and Michael A. Saunders. 2005. SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization. *SIAM Rev.* 47, 1 (2005), 99–131. https://doi.org/10.1137/S003614450446096
- Michael Gleicher. 1998. Retargeting Motion to New Characters. In Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1998, Orlando, FL, USA, July 19-24, 1998, Steve Cunningham, Walt Bransford, and Michael F. Cohen (Eds.). ACM, 33–42. https://doi.org/10.1145/280814.280820
- Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In NIPS.
- Hanwen Jiang, Shaowei Liu, Jiashun Wang, and Xiaolong Wang. 2021. Hand-Object Contact Consistency Reasoning for Human Grasps Generation. In 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021. IEEE, 11087–11096. https://doi.org/10.1109/ICCV48922.2021. 01092
- Korrawe Karunratanakul, Jinlong Yang, Yan Zhang, Michael J Black, Krikamol Muandet, and Siyu Tang. 2020. Grasping field: Learning implicit representations for human grasps. In 2020 International Conference on 3D Vision (3DV). IEEE, 333–344.
- Paul G Kry and Dinesh K Pai. 2006. Interaction capture and synthesis. ACM Transactions on Graphics (TOG) 25, 3 (2006), 872–880.
- Gilwoo Lee, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. 2015. Hierarchical planning for multi-contact non-prehensile manipulation. In 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2015, Hamburg, Germany, September 28 - October 2, 2015. 264–271. https://doi.org/10.1109/IROS.2015.7353384
- Ying Li, Jiaxin L Fu, and Nancy S Pollard. 2007. Data-driven grasp synthesis using shape matching and task-based pruning. *IEEE Transactions on visualization and* computer graphics 13, 4 (2007), 732–747.
- C. Karen Liu. 2009. Dextrous manipulation from a grasping pose. ACM Trans. Graph. 28, 3 (2009), 59. https://doi.org/10.1145/1531326.1531365
- Kendall Lowrey, Svetoslav Kolev, Jeremy Dao, Aravind Rajeswaran, and Emanuel Todorov. 2018. Reinforcement learning for non-prehensile manipulation: Transfer from simulation to physical system. In 2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR). IEEE, 35–42.



Figure 5: Results of our method in different scenarios

Qingkai Lu, Kautilya Chenna, Balakumar Sundaralingam, and Tucker Hermans. 2020. Planning multi-fingered grasps as probabilistic inference in a learned deep network.

In Robotics Research. Springer, 455-472.

Sirui Chen, Albert Wu, and C. Karen Liu



Figure 6: (a), (b) are of same configuration. (c), (d) are different. Keyboard in (c) is closer to the wall, in (d) it is closer to the edge.



(a). Reversed chronological order



(b). Normal chronological order

Figure 7: Comparison of normal and reverse order of solving IK.

- Jens Lundell, Francesco Verdoja, and Ville Kyrki. 2021. Ddgc: Generative deep dexterous grasping in clutter. *IEEE Robotics and Automation Letters* 6, 4 (2021), 6899–6906.
- Kevin M. Lynch and Matthew T. Mason. 1999. Dynamic Nonprehensile Manipulation: Controllability, Planning, and Experiments. *Int. J. Robotics Res.* 18, 1 (1999), 64–92. https://doi.org/10.1177/027836499901800105
- Priyanka Mandikal and Kristen Grauman. 2021. Learning dexterous grasping with object-centric visual affordances. In 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 6169–6176.
- Andrew T. Miller and Peter K. Allen. 2004. Graspit! A versatile simulator for robotic grasping. *IEEE Robotics Autom. Mag.* 11, 4 (2004), 110–122. https://doi.org/10.1109/ MRA.2004.1371616
- Igor Mordatch, Zoran Popović, and Emanuel Todorov. 2012. Contact-invariant optimization for hand manipulation. In Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation. 137–144.
- Florian T Pokorny and Danica Kragic. 2013. Classical grasp quality evaluation: New algorithms and theory. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 3493–3500.
- Nancy S Pollard and Victor Brian Zordan. 2005. Physically based grasping control from example. In Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation. 311–318.

- George Jose Pollayil, Giorgio Grioli, Manuel Bonilla, and Antonio Bicchi. 2021. Planning robotic manipulation with tight environment constraints. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 9385–9392.
- Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems 30 (2017).
- Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. 2022. In-Hand Object Rotation via Rapid Motor Adaptation. arXiv preprint arXiv:2210.04887 (2022).
- Javier Romero, Dimitris Tzionas, and Michael J Black. 2017. Embodied Hands: Modeling and Capturing Hands and Bodies Together. ACM Transactions on Graphics 36, 6 (2017).
- Baris Serhan, Harit Pandya, Ayse Kucukyilmaz, and Gerhard Neumann. 2022. Push-to-See: Learning Non-Prehensile Manipulation to Enhance Instance Segmentation via Deep Q-Learning. Institute of Electrical and Electronics Engineers.
- Lin Shao, Fabio Ferreira, Mikael Jorda, Varun Nambiar, Jianlan Luo, Eugen Solowjow, Juan Aparicio Ojea, Oussama Khatib, and Jeannette Bohg. 2020. Unigrasp: Learning a unified model to grasp with multifingered robotic hands. *IEEE Robotics and Automation Letters* 5, 2 (2020), 2286–2293.
- Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. Advances in neural

Figure 8: Comparison with ManipNet on spice rack.



Figure 9: Performance of our method on nonconvex objects.

SIGGRAPH '23 Conference Proceedings, August 6-10, 2023, Los Angeles, CA, USA

information processing systems 28 (2015).

- Hyung Ju Terry Suh, Tao Pang, and Russ Tedrake. 2022. Bundled gradients through contact via randomized smoothing. *IEEE Robotics and Automation Letters* 7, 2 (2022), 4000-4007.
- Zhaole Sun, Kai Yuan, Wenbin Hu, Chuanyu Yang, and Zhibin Li. 2020. Learning pregrasp manipulation of objects from ungraspable poses. In 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 9917–9923.
- Omid Taheri, Nima Ghorbani, Michael J Black, and Dimitrios Tzionas. 2020. GRAB: A dataset of whole-body human grasping of objects. In European conference on computer vision. Springer, 581–600.
- R Tedrake et al. 2019. Drake: Model-based design and verification for robotics. URL https://drake. mit. edu (2019).
- Grady Williams, Andrew Aldrich, and Evangelos A Theodorou. 2017. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance*, *Control, and Dynamics* 40, 2 (2017), 344–357.
- J Zachary Woodruff and Kevin M Lynch. 2017. Planning and control for dynamic, nonprehensile, and hybrid manipulation tasks. In 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 4066–4073.
- Albert Wu, Michelle Guo, and C. Karen Liu. 2022. Learning diverse and physically feasible dexterous grasps with generative model and bilevel optimization. arXiv preprint arXiv:2207.00195 (2022).
- Yuting Ye and C. Karen Liu. 2012. Synthesis of detailed hand manipulations using contact sampling. ACM Trans. Graph. 31, 4 (2012), 41:1–41:10. https://doi.org/10. 1145/2185520.2185537
- He Zhang, Yuting Ye, Takaaki Shiratori, and Taku Komura. 2021. ManipNet: neural manipulation synthesis with a hand-object spatial representation. *ACM Trans. Graph.* 40, 4 (2021), 121:1–121:14. https://doi.org/10.1145/3450626.3459830
- Wenping Zhao, Jianjie Zhang, Jianyuan Min, and Jinxiang Chai. 2013. Robust realtime physics-based motion control for human grasping. ACM Transactions on Graphics (TOG) 32, 6 (2013), 1–12.
- Wenxuan Zhou and David Held. 2022. Learning to Grasp the Ungraspable with Emergent Extrinsic Dexterity. In ICRA 2022 Workshop: Reinforcement Learning for Contact-Rich Manipulation.