Robotics: Science and Systems 2025
Los Angeles, CA, USA, June 21–25, 2025

Dexonomy: Synthesizing All Dexterous Grasp Types in a Grasp Taxonomy

Jiayi Chen[1,2*], Yubin Ke[1,2*], Lin Peng[2], He Wang[1,2,3†]
[1]Peking University, [2]Galbot, [3]Beijing Academy of Artificial Intelligence
[*]Equal contribution, [†]Corresponding author

Fig. 1: For **any grasp type** in the GRASP taxonomy [20], **any object**, and **any articulated hand**, our pipeline efficiently synthesizes contact-rich, penetration-free, and physically plausible dexterous grasps, starting from only one human-annotated grasp template to specify an initial hand pose and contact information per hand and grasp type.

***Abstract*—Generalizable dexterous grasping with suitable grasp types is a fundamental skill for intelligent robots. Developing such skills requires a large-scale and high-quality dataset that covers numerous grasp types (i.e., at least those categorized by the GRASP taxonomy), but collecting such data is extremely challenging. Existing automatic grasp synthesis methods are often limited to specific grasp types or object categories, hindering scalability. This work proposes an efficient pipeline capable of synthesizing contact-rich, penetration-free, and physically plausible grasps for any grasp type, object, and articulated hand. Starting from a single human-annotated template for each hand and grasp type, our pipeline tackles the complicated synthesis problem with two stages: optimize the object to fit the hand template first, and then locally refine the hand to fit the object in simulation. To validate the synthesized grasps, we introduce a contact-aware control strategy that allows the hand to apply the appropriate force at each contact point to the object. Those validated grasps can also be used as new grasp templates to facilitate future synthesis. Experiments show that our method significantly outperforms previous type-unaware grasp synthesis baselines in simulation. Using our algorithm, we construct a dataset containing 10.7k objects and 9.5M grasps, covering 31 grasp types in the GRASP taxonomy. Finally, we train a type-conditional generative model that successfully performs the desired grasp type from single-view object point clouds, achieving an 82.3% success rate in real-world experiments. Project page: https://pku-epic.github.io/Dexonomy.**

I. Introduction

Dexterous grasping is a fundamental skill for intelligent robots, enabling flexible interaction with the environment. However, this potential remains largely under-explored. Although dexterous grasping has received increasing attention, most prior work focuses on whether a dexterous hand can successfully grasp an object, rather than considering *how* to grasp it. As a result, the dexterous hand loses its dexterity and becomes functionally similar to a large parallel gripper. True dexterous grasping is not merely about "grasping with dexterous hands", but about "grasping dexterously with appropriate grasp types based on the task requirement". For example, when a robot needs to securely grasp an apple or hold a knife to cut, it should use a power grasp to envelop the object. Conversely, when grasping a lightweight or flat object from the table, a precision grasp using the fingertips would be more suitable.

To develop such intelligent skills, there are two key challenges: (1) selecting the appropriate grasp type based on the

task and (2) generating high-quality grasps for specified types and objects. The first challenge is a high-level decision-making problem and can take advantage of recent advances in large vision-language models, e.g., GPT-4o [22], as a temporary solution. However, the second challenge is less studied and represents a significant bottleneck, which is the main focus of this paper. To address the problem of type-aware grasp synthesis with data-driven methods, the first step is to build a large-scale grasp dataset that at least includes most of the grasp types in the GRASP taxonomy [20]. However, collecting and annotating grasp data, particularly for multi-fingered hands in contact-rich scenarios, remains a big challenge.

Several approaches have been explored for automatically synthesizing a large-scale dexterous grasp dataset, but most of them suffer from various limitations. Analytical grasp synthesis methods [47, 44, 25, 8, 7] are often applicable to any object, but most of them are type-unaware and the synthesized grasps only belong to limited types. This is because specifying flexible grasp types solely through analytical metrics is challenging. Moreover, these methods often produce unnatural hand poses, as they prioritize *force closure*, which does not always align with human habits. Another line of research [55, 48, 52] focuses on transferring functional dexterous grasps by mapping object contact regions. While these methods generate more human-like grasps and support a wider range of grasp types, they are limited to objects that are geometrically similar or axis-aligned with the initial demonstration, making them less scalable.

In this work, we propose a novel pipeline based on sampling and optimization to address these challenges. As shown in Figure 1, our algorithm can efficiently synthesize high-quality dexterous grasps for any grasp type, object, and articulated hand, requiring only one human-annotated grasp template per hand and grasp type. Our synthesized grasps achieve rich hand-object contact (e.g., > 10 hand links within 2 mm of the object for power grasps), guarantee penetration-free poses via collision mesh verification, and satisfy force closure under six-axis testing in MuJoCo [42] — all with shared hyperparameters across grasp types, objects, and hands.

Our key insight is that grasping can be framed as a geometric matching problem, where the hand and object should align through contact points. We begin by introducing a human-annotated grasp template that specifies the initial hand pose and contact information (i.e., points and normals). Unlike previous methods that directly adjust the hand pose to fit the object, we first use a lightweight stage that samples and optimizes the object pose to match the hand contacts defined in the grasp template. This stage supports hundreds of thousands of initial samples processed in parallel on a single GPU and leaves only a small number of promising results for the next stage. The combination of dense sampling and optimization helps avoid local optima without sacrificing efficiency.

After aligning the object pose, the hand only needs a slight refinement to get a good grasp. This dual-stage design not only eases the hand refinement stage, but also ensures that the final grasp remains similar to the initial grasp template and thus remains natural. In contrast to most prior work [24, 47, 8, 7] that develops custom objective functions and optimizers to refine the hand pose, we propose a novel method based on the transposed Jacobian control in MuJoCo. This approach is key to achieving rich contacts while ensuring no penetration, with minimal coding effort and parameter tuning.

Next, we evaluate the synthesized grasps in MuJoCo to assess their ability to withstand external forces applied to the object. Unlike previous work [47, 60] that designs heuristics to squeeze the hand for applying force on the object, which is not suitable for all grasp types, we design a contact-aware control strategy that computes the desired forces for each contact point and controls the hand to apply them approximately, also based on the transposed Jacobian control. Finally, high-quality grasps that pass the simulation tests can be used to construct new grasp templates, reducing the need for human annotations and broadening the range of objects that can be grasped.

Experiments show that our method greatly outperforms previous type-unaware grasp synthesis baselines in simulation, especially under more challenging testing conditions (i.e., smaller friction coefficients) and with a wider variety of objects (i.e., from Objaverse [18]). Using our proposed pipeline, we also build a large-scale dataset covering different grasp types. This dataset further enables training a type-conditional generative model that generates desired grasp types for novel objects from single-view point clouds, achieving a success rate of 82.3% on the Shadow hand in real-world experiments. Finally, we show that our algorithm can be used to develop an annotation UI for collecting semantic grasps on the specified object regions with only a few mouse clicks.

In summary, our main contributions are:

- An efficient pipeline to synthesize high-quality grasps for any grasp type, object, and articulated hand, starting from one human-annotated template per hand and grasp type.
- A large-scale dataset containing 9.5M grasps and 10.7k objects, covering 31 grasp types in the GRASP taxonomy.
- A type-conditional generative model that can use the specified grasp types to grasp novel objects in the real world, with only a single-view point cloud as input.
- An annotation UI for collecting semantic grasps with only a few mouse clicks.

II. Related Work

A. Analytical Grasp Synthesis

Analytical grasp synthesis aims to find good grasps, typically measured by wrench-based force closure metrics [21, 40]. These methods generally assume complete knowledge of the object's geometry for the metric calculation, making them more suitable for data preparation than for real-world applications, where objects are often only partially observed. For parallel grippers and suction cups, where hand-object contact patterns are simple and the dimension of the hand pose is low (around 7), many work [32, 33, 19, 4] randomly sample a large number of grasps and select those with better metrics as the final results. For dexterous hands, which have more

complex contact patterns and higher pose dimensions (often exceeding 20), random sampling is insufficient to find good grasps, necessitating optimization. Early work [35, 14] used sampling-based optimization, such as simulated annealing, since the commonly used metrics are non-differentiable. More recent approaches [27, 43, 44, 47, 6, 25, 8, 7] introduce differentiable energies to leverage gradient-based optimization. However, these methods typically start from a rest pose, requiring many iterations to adjust the hand pose, which can be inefficient, easily lead to local minima, or produce unnatural poses. Our approach benefits from both random sampling and gradient-based optimization, greatly reducing the complexity.

B. Grasp Taxonomy

The GRASP taxonomy [20] categorizes 33 distinct grasp types based on human daily activities. Many previous analytical methods [6, 7, 25, 8] can only synthesize limited grasp types, such as fingertip grasp. Although some work [43, 44, 27, 47] supports contact beyond the fingertip, they can only synthesize some simple grasp types, e.g. warping fingers around the object, and typically have high randomness and cannot specify a desired type. Some work on functional dexterous grasping [23, 48, 1, 52, 61] tackles more complex, human-like grasp types, but is often limited to axis-aligned objects with similar geometries, hindering scalability. In contrast, our method can synthesize all 33 grasp types without assumptions about the object, requiring only one template per grasp type.

C. Data Collection for Learning-based Grasp Synthesis

Learning-based dexterous grasp synthesis [53, 51, 50] is often built on generative models such as CVAE, diffusion models, and normalizing flows. While they can handle partial observation and be deployed in the real world, their performance is highly dependent on the quality of the training dataset. To build dexterous grasp datasets, several approaches are explored, including teleoperation [38, 11, 56], transferring grasps from human hands [3, 37, 30, 5, 55, 10], and reinforcement learning (RL) [59, 46, 13]. Our work also presents a possible approach for scaling up grasp data collection, because our method requires less human effort than teleoperation, avoids the morphological gap seen with human hands, and is often more efficient than RL.

D. Physics Simulation for Robotic Grasping

Rigid-body physics simulators, such as Isaac PhysX [34], MuJoCo [42], Bullet [15], are widely used to validate a grasp or perform RL for dexterous hands. However, prior work rarely uses simulators for local refinement of the hand pose to improve contact with the object. Most prior work [24, 47, 8, 7] develops custom energy functions and optimizers to refine hand-object contact, but they often lead to centimeter-level penetrations and require careful hyperparameter tuning. In contrast, our approach utilizes MuJoCo to refine the hand pose, achieving submillimeter-level contact convergence with minimal parameter tuning in different experiment settings. This is made possible by MuJoCo's highly optimized framework and its second-order Newton optimizer.

III. Preliminary of Grasp Quality Metric

In this section, we summarize a unified formulation of grasp quality metrics widely used in recent work [27, 47, 51, 7]. While prior work focuses on their own formulations, our summary provides a cohesive view across them. Although our pipeline does not directly optimize this metric—unlike prior analytical methods—it plays an important role in several stages of our pipeline, such as post-filtering (Sections IV-B and IV-C) and the contact-aware control strategy (Section IV-D).

We assume that the object O is grasped by a robot hand with m contact points. For each contact i, let $\mathbf{p}_i \in \mathbb{R}^3$ denote the contact position, $\mathbf{n}_i \in \mathbb{R}^3$ the inward-pointing surface unit normal, and $\mathbf{d}_i \in \mathbb{R}^3$ and $\mathbf{c}_i \in \mathbb{R}^3$ be two unit tangent vectors satisfying $\mathbf{n}_i = \mathbf{d}_i \times \mathbf{c}_i$. The Coulomb friction cone $\mathcal{F}_i$ and the contact Jacobian $\mathbf{J}_{o,i}$ for the object at contact i are defined as follows:

$$\mathcal{F}_i = \left\{ \mathbf{x}_i \in \mathbb{R}^3 \mid 0 \le x_{i,1} \le 1, x_{i,2}^2 + x_{i,3}^2 \le \mu^2 x_{i,1}^2 \right\} \quad (1)$$

$$\mathbf{J}_{o,i}^T = \begin{bmatrix} \mathbf{n}_i & \mathbf{d}_i & \mathbf{c}_i \\ \mathbf{p}_i \times \mathbf{n}_i & \mathbf{p}_i \times \mathbf{d}_i & \mathbf{p}_i \times \mathbf{c}_i \end{bmatrix} \in \mathbb{R}^{6\times 3} \quad (2)$$

where μ is the friction coefficient. The friction cone $\mathcal{F}_i$ represents all feasible contact forces at contact i, and $\mathbf{J}_{o,i}$ maps a contact force $\mathbf{x}_i$ to a wrench $\mathbf{w}_i = \mathbf{J}_{o,i}^T \mathbf{x}_i$.

To balance an external wrench $\mathbf{g} \in \mathbb{R}^6$ (e.g., object gravity), the optimal contact forces $\{\mathbf{f}_i\}_{i=1}^m$ are obtained by solving the following quadratic program (QP):

$$(\mathbf{f}_1, ..., \mathbf{f}_m) = \underset{(\mathbf{x}_1,...,\mathbf{x}_m)}{\arg\min} \quad \| \sum_{i=1}^m \mathbf{J}_{o,i}^T \mathbf{x}_i - \mathbf{g} \|^2 \quad (3)$$

$$\text{s.t.} \quad \mathbf{x}_i \in \mathcal{F}_i, \quad i \in \{1, ..., m\} \quad (4)$$

$$\Sigma_{i=1}^m x_{i,1} \ge \lambda \quad (5)$$

where λ is a hyperparameter enforcing a minimum total normal force to avoid trivial solutions. To reduce computational complexity, the friction cone $\mathcal{F}_i$ is approximated by a pyramid, converting the problem into a linearly-constrained quadratic program (LCQP) that can be efficiently solved [2].

Finally, the grasp quality metric e is defined as:

$$e = \| \sum_{i=1}^m \mathbf{J}_{o,i}^T \mathbf{f}_i - \mathbf{g} \|^2 \quad (6)$$

where a lower e indicates a more stable or robust grasp.

The formulation in Eq. 3 is intuitive for testing one external wrench on the object. However, to test force-closure grasps, we typically need to consider six orthogonal gravities [7]. This requires solving the LCQP six times for each grasp and thus is inefficient. The energy function used in DexGraspNet [27, 47] assumes equal contact forces and zero friction, simplifying the problem by setting $\mathbf{f}_i = [1, 0, 0]$ and $\mathbf{g} = \mathbf{0}$. While this approach eliminates the need for optimization and is extremely fast, it sacrifices accuracy. In contrast, [51] only sets $\mathbf{g} = \mathbf{0}$, providing a better balance between computational cost and accuracy. Therefore, in this paper, we adopt the variation proposed by [51] as the grasp quality metric.

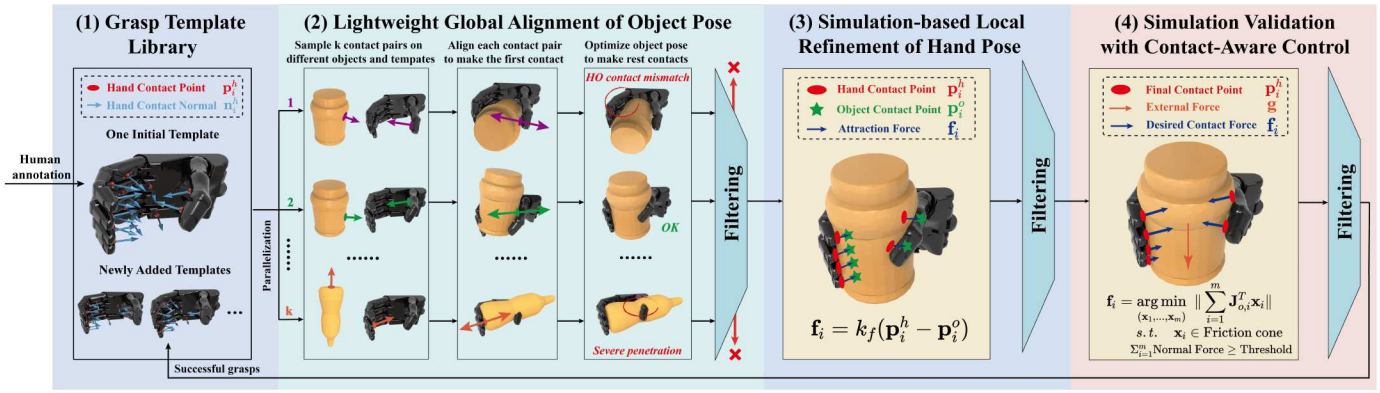

Fig. 2: **The pipeline of Dexonomy.** (1) *Grasp Template Library* initially requires one human-annotated template. (2) *Lightweight Global Alignment* stage samples and optimizes the object poses in parallel on a GPU, to match the contact points and normals of the selected grasp templates. (3) *Simulation-based Local Refinement* stage adjusts the hand pose to improve hand-object contacts. (4) *Simulation Validation* tests force-closure grasps using our proposed contact-aware control strategy. (5) New templates are constructed from successful grasps and added to the *Grasp Template Library*, used in the following iterations.

IV. Grasp Synthesis Method

In this section, we present our proposed dexterous grasp synthesis pipeline, with an overview shown in Figure 2.

A. Grasp Template Definition

A grasp template consists of several components: the hand joint configuration $\mathbf{q} \in \mathbb{R}^q$, hand contact points $\mathbf{p}_i^h \in \mathbb{R}^3$, corresponding normals $\mathbf{n}_i^h \in \mathbb{R}^3$, and the link name for each contact point ($i = 1, 2, \ldots, m$). Our algorithm requires a single human-annotated grasp template for each hand and grasp type as initialization.

B. Lightweight Global Alignment of Object Pose

In this stage, we simultaneously sample and optimize the object pose to align with the selected template's hand contacts while keeping the hand pose fixed. The optimization variable is the object's transformation, parameterized by its (optional) scale $s_o \in \mathbb{R}$, rotation $\mathbf{R}_o \in \mathcal{S}^3$, and translation $\mathbf{t}_o \in \mathbb{R}^3$.

Before optimization, we begin with dense sampling. First, a random grasp template is selected from the *Grasp Template Library*, and a random hand contact from the template is chosen. Then, a random object is selected, and a random surface point on the object is chosen. The object is initialized by aligning the sampled hand and object contacts, where contact points are matched and the contact normal directions are set opposite. The object's scale and in-plane rotation perpendicular to the normal direction are randomly sampled. Our pipeline supports parallelizing massive samples of different contacts, objects, and grasp templates on a single GPU.

During each optimization iteration, each hand contact point $\mathbf{p}_i^h$ calculates the nearest point $\mathbf{p}_i^o$ on the object's surface using the differentiable library `Warp` [31]. To penalize the mismatch between hand and object contacts, we optimize the object pose by minimizing the following energy function:

$$L = k_p \sum_{i=1}^{m} \|\mathbf{p}_i^h - \mathbf{p}_i^o\|^2 + k_n \sum_{i=1}^{m} \|\mathbf{n}_i^h - \mathbf{n}_i^o\|^2 \tag{7}$$

where k_p and k_n are hyperparameters. There is no other energy used for optimization except Eq. 7.

After optimization, results are filtered using four criteria. First, the final energy function must be below a threshold to ensure a good match between hand and object contacts. Second, severe penetration between the hand and object should be avoided, which we efficiently detect using our proposed hand collision skeletons parameterized by line segments (details in Appendix A). Third, the object contact quality, as measured by Eq. 6, must exceed a threshold. Finally, we apply a process similar to farthest point sampling to filter out duplicate object transformations, further detailed in Appendix A.

Our design, using only one energy during optimization and leaving other checks for post-filtering, provides several advantages. First, it reduces computational costs, enabling maximized parallelization to benefit from dense sampling to avoid local optimum traps. Second, it reduces sensitivity to hyperparameters, as post-filtering criteria are applied sequentially, while optimization energies need to be applied together.

C. Simulation-based Local Refinement of Hand Pose

In this stage, the object is fixed, and the hand pose is locally refined to improve the hand-object contact. A virtual force $\mathbf{f}_i$ is needed at each hand point $\mathbf{p}_i^h$ toward the corresponding nearest object point $\mathbf{p}_i^o$. To apply these virtual forces in MuJoCo, they are transferred to the hand's joint torque via simplified transposed Jacobian control:

$$\mathbf{f}_i = k_f(\mathbf{p}_i^h - \mathbf{p}_i^o), \quad \tau = \sum_{i=1}^{m} \mathbf{J}_{h,i}^T \mathbf{f}_i \tag{8}$$

where k_f is a hyperparameter and $\mathbf{J}_{h,i}^T \in \mathbb{R}^{q \times 3}$ is the transpose of the hand contact Jacobian that maps force vectors from the world to joint coordinates.

While Eq. 8 is a simplified control strategy with many assumptions (e.g., no dynamics or gravity; joint torques mapped from each contact force are independent and additive), it serves

Dataset	Hand	Sim./Real	Objects	Grasps	Grasp Types	Force Closure	Data Type	Method
DexGraspNet [47]	Shadow	IsaacGym	5.4k	1.32M	Random	✓	Grasp pose	Optimization
RealDex [29]	Shadow	Real	52	59k	Random	✗	Motion	Teleoperation
GraspXL [59]	Multiple	RaiSim	500k	10M	Random	✗	Motion	RL
BODex [7]	Shadow	MuJoCo	2.4k	3.62M	Fingertip	✓	Pre-grasp, grasp poses	Optimization
Dexonomy (Ours)	Shadow	MuJoCo	10.7k	9.5M	31 types	✓	Pre-grasp, grasp, squeeze poses	Sampling+opt.

TABLE I: **Dexterous Grasp Dataset Comparison.** Our large-scale dataset aims to support the study of data-driven methods for type-aware grasp synthesis.

our need for synthesizing contact-rich grasps in simulation. This is easy to implement and theoretically works for other physics simulators. Eq. 8 is iteratively applied for 200 steps, where $\mathbf{p}_i^o$ is fixed in the world frame to avoid drift, while $\mathbf{p}_i^h$ is fixed in the hand link frame (and moves in the world frame).

To ensure the hand remains strictly penetration-free with respect to the object, we apply a 1mm contact margin in MuJoCo—meaning the hand is repelled if it comes within 1mm of the object surface. Since MuJoCo reliably resolves penetration at the millimeter scale, the resulting contact distance between the hand and the object typically falls within the range of $[0, 2\text{mm}]$. This behavior motivates our use of a 2mm contact tolerance: a hand link is considered to be in contact if its distance to the object is less than 2mm.

After optimization, we filter the results based on three criteria. First, there should be no penetration between the hand and the object, measured using collision meshes. Second, those fingers with at least one annotated contact are required to be in contact with the object. This prevents the synthesized grasps from deviating significantly from the initial template. Finally, the grasp quality, as measured by Eq. 6, must exceed a threshold, which is needed again because the final contact points may slightly differ from the ones in the previous stage.

D. Simulation Validation with Contact-Aware Control

To validate the synthesized grasps in MuJoCo, the hand should squeeze to hold the object stably, controlled by a control signal of joint torques. Our contact-aware control strategy first calculates the desired forces on each contact using Eq. 3 where $\mathbf{g} = \mathbf{0}$, and then converts these forces into joint torques using the same transposed Jacobian control as in Eq. 8.

The $\mathbf{g}$ is set to zero to reduce computational cost and standardize the control signals across different external forces. Despite the resultant force of all contacts being approximately zero, friction naturally adjusts to prevent object movement. Given the large strength of the normal force (i.e., around $1 \sim 10$N in our experiment for a 100g object), constrained by Eq. 5, there is typically sufficient room for friction to adjust, if the grasp pose is physically plausible. A grasp is considered to succeed only if the object remains stable under all six orthogonal external forces for 2 seconds in simulation.

E. Construction of New Grasp Templates

Once a grasp successfully passes the simulation validation, a new grasp template is constructed and added to the template library. The joint configuration of the new template is taken from the successful grasp, while the contact information (i.e., points and normals) is updated only if an actual contact is detected near the original contact on the same hand link. This strategy prevents the new template's contact information from deviating too much from the original. Newly added templates can be randomly selected during the global alignment stage in subsequent iterations of the whole pipeline.

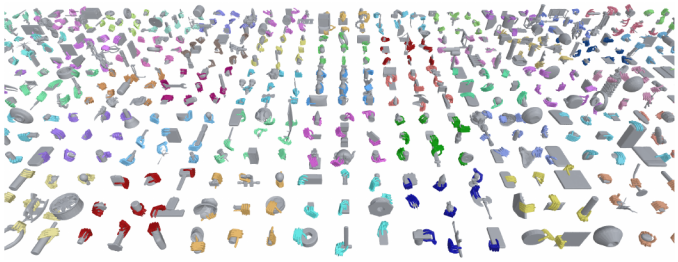

Fig. 3: **Dexonomy Dataset Visualization.** Each color corresponds to a different grasp type.

V. Dexonomy Dataset

Using our proposed grasp synthesis pipeline, we construct a large-scale dataset for Shadow hand covering 31 grasp types from the GRASP taxonomy [20]. This dataset is designed to support research on data-driven methods for type-aware grasp synthesis. Two grasp types in the taxonomy, *Distal Type (#19)* and *Tripod Variation (#21)*, are excluded due to their specificity to object categories, namely scissors and chopsticks, respectively.

As shown in Table I, our dataset comprises 10.7k object assets, including 5,697 objects from DexGraspNet [47] and 5,000 new objects randomly selected from Objaverse [18]. All objects are normalized such that the diagonal of their axis-aligned bounding box is 2 meters, with scales ranging between $[0.05, 0.2]$. Only successful grasps are retained, resulting in 9.5M data points. The entire dataset was synthesized in less than 3 days on a server with 8 NVIDIA RTX 3090 GPUs. Additional statistics are provided in Table IV and Appendix B.

Each data point includes three key poses:

- **Grasp pose**, obtained via local refinement (Section IV-C).
- **Pre-grasp pose** for collision-free motion planning, generated after the grasp pose by enforcing a 2cm contact margin in MuJoCo—pushing the hand away if it is within 2cm of the object.
- **Squeeze pose**, derived from the control signal used for simulation validation (Section IV-D), to apply force through hand-object contacts.

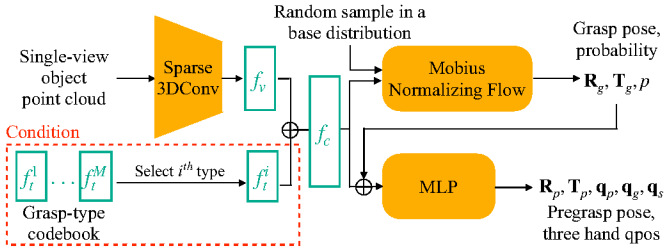

Fig. 4: **Type-Conditional Grasp Generative Model.** Without the grasp-type codebook in the red dashed box, the model becomes type-unconditional and is similar to BODex [7].

These poses provide the minimal requirements for generating a complete grasping trajectory (including reaching and squeezing) and are compatible with diverse robot arms and initial hand configurations.

VI. Type-Conditional Grasp Generative Model

To generate grasps from partial observations for real-world deployment, data-driven methods are essential. Although learning is not the main focus of this paper, we present a simple model as an initial try. The model architecture, illustrated in Figure 4, is similar to previous work [60, 7], with the key difference being the grasp-type codebook added as a conditional input to specify a grasp type.

The input to the model consists of a single-view object point cloud and a type feature f_t^i selected from the grasp-type codebook. The point cloud is encoded into a feature f_v using a Sparse3DConv network with MinkowskiEngine [12]. This vision feature f_v, along with the type feature f_t, are concatenated to form a conditional feature f_c. Conditioned on f_c, the Mobius normalizing flow [28] maps a random sample in a base distribution to a grasp pose $\mathbf{R}_g$ and $\mathbf{T}_g$, and calculates a probability p indicating the pose quality. The predicted grasp pose is then concatenated with f_c and passed through an MLP to predict a pre-grasp pose $\mathbf{R}_p$, $\mathbf{T}_p$, and three hand qpos $\mathbf{q}_p$, $\mathbf{q}_g$, and $\mathbf{q}_s$ for the pre-grasp, grasp, and squeeze poses, respectively. The whole model is trained end-to-end and the type feature f_t^i is also optimizable.

VII. Experiment

A. Evaluation Metrics

The following metrics are used for a comprehensive evaluation of the synthesis pipeline and grasp quality. All distances are measured using collision meshes in MuJoCo.

Grasp Success Rate (GSR) (unit: %): The percentage of successful grasps relative to the attempt number. For our method, one attempt is defined as one valid result output by the global alignment stage. A grasp succeeds only if it resists six external forces in MuJoCo and does not have severe penetrations (> 1 cm), since the penetration may cause simulation failure and prevent the object from moving. The object mass is 100g, and the success criteria for the object pose are 5cm and 15°.

Object Success Rate (OSR) (unit: %): The percentage of objects that have at least one successful grasp. If the object scales are fixed, as in Sections VII-B and VII-E, different scales of the same object are treated as separate objects.

Speed (S) (unit: second^{-1}): The maximum number of attempts completed per second on a server with 8 NVIDIA RTX 3090 GPUs and 2 Intel Xeon Platinum 8255C CPUs (48 cores, 96 threads). We report the time running on a server because our method utilizes both GPUs and CPUs. This metric excludes simulation validation.

Contact Link Number (CLN): The number of hand links whose distance to the object surface is within 2 mm.

Contact Distance Consistency (CDC) (unit: mm): The delta between the maximum and minimum signed distances across all fingers. This metric quantifies the variation in contact distance across different fingers and is invariant to penetration.

Penetration Depth (PD) (unit: mm): The maximum intersection distance between the hand and object for each grasp.

Self-Penetration Depth (SPD) (unit: mm): The maximum self-intersection distance among different hand links.

Diversity (D) (unit: %): The proportion of total variance explained by the first principal component in PCA, computed as the ratio of the first eigenvalue to the sum of all eigenvalues. PCA is performed on data points that include grasp translation $\mathbf{T}_g$, rotation $\mathbf{R}_g$ (in the axis-angle representation), and joint angles $\mathbf{q}_g$.

B. Type-Unaware Grasp Synthesis

1) Comparison with analytical methods: Four open-source baselines are compared: DexGraspNet [47], FRoGGeR [25], SpringGrasp [8], and BODex [7].

Experiment settings. Most of the settings follow the benchmark provided by BODex [7], with some modifications on the object set to increase difficulty. The Allegro hand is used, as it is the only hand type supported by all baselines. 5697 object assets from DexGraspNet are used, with six scales applied to each normalized object: 0.05, 0.08, 0.11, 0.14, 0.17, and 0.20. Although our method supports optimizing object scales, we fix the object scale to ensure a fair comparison. The attempt number for each method is set to 20, and we manually annotate two grasp templates, each with 10 attempts.

Quantitative result analysis. As shown in Table II, our method achieves the highest grasp success rate and best performance on contact and penetration. Our speed is slightly lower than that of BODex, as their pipeline is highly optimized for GPUs, while our local refinement stage uses MuJoCo's CPU version. Detailed time analysis of our pipeline is available in Appendix F. Our diversity is somewhat lower, as we use only two initial templates and our hand refinement stage makes only local adjustments. However, the overall diversity of our Dexonomy dataset is much better, as reported in Table IV. The success rates of baselines are lower than those reported in BODex [7], primarily because our objects have a higher mass (100g vs. 30g), a larger scale range ($[0.05, 0.2]$ vs. $[0.06, 0.12]$), and more diverse shapes (5697 instances vs.

	GSR (%) ↑	OSR (%) ↑	S (s^{-1}) ↑	CLN ↑	CDC (mm) ↓	PD (mm) ↓	SPD (mm) ↓	D (%) ↓
DexGraspNet [47]	12.10	57.01	3.25	3.22	7.58	4.85	1.20	29.03
FRoGGeR [25]	10.34	55.70	2.98	2.51	4.95	0.22	**0.00**	**27.01**
SpringGrasp [8]	7.83	35.44	5.47	2.79	23.59	16.58	1.06	70.18
BODex [7]	49.23	**96.56**	**403.9**	3.85	3.03	0.63	0.02	32.50
Ours	**60.50**	96.53	323.4	**4.38**	**0.21**	**0.00**	**0.00**	34.17

TABLE II: **Comparison with Type-Unaware Grasp Synthesis Baselines for Allegro Hand.** Most baselines, except DexGraspNet, only synthesize fingertip grasps, so we also synthesize fingertip grasps for a fair comparison.

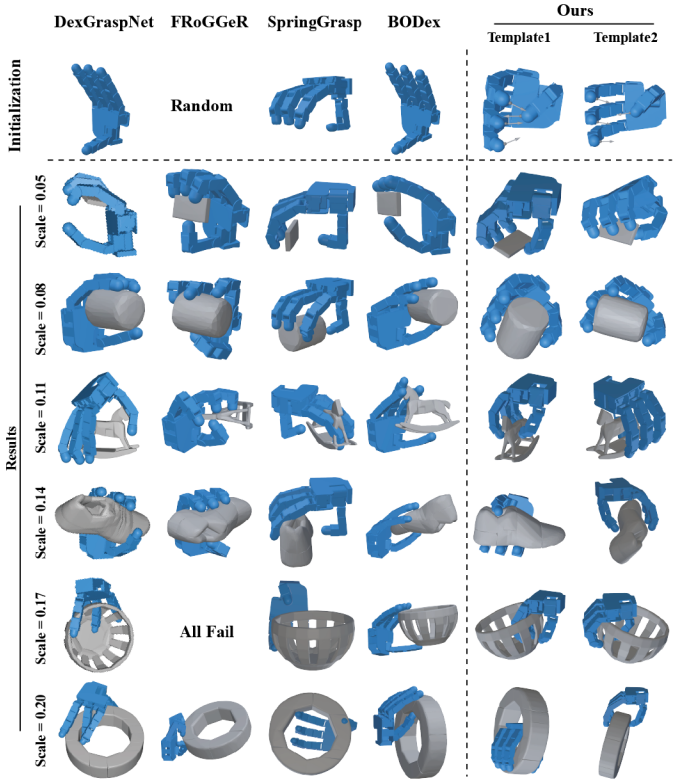

Fig. 5: **Visualization of Type-Unaware Grasps.** Our method synthesizes human-like and stable grasps, even for objects with complex geometries (e.g., object scales = 0.11, 0.17, and 0.20).

2397). More details about the performance drop of baselines are provided in Appendix C.

Visualization analysis. Figure 5 illustrates the initial hand pose and some synthesized grasps for each method. Our method consistently synthesizes human-like and stable grasps, even for objects with complex geometries (e.g., for scales 0.11, 0.17, and 0.20). Notably, the synthesized grasp for template 1 and object scale 0.05 requires high precision and is challenging for previous methods. Furthermore, the grasp for template 2 and object scale 0.14 shows a much larger thumb-to-other-tip distance than the initial human-annotated template, demonstrating our method's ability to adjust hand joint angles across a large range.

For baseline methods, DexGraspNet [47] shows high uncertainty, partly due to its randomness in selecting contact points. While it occasionally generates good grasps (e.g., for scales 0.08 and 0.14), it often results in twisted fingers

Method	Attempt Number	DGN object [47]		Objaverse [18]	
		GSR↑	OSR↑	GSR↑	OSR↑
BODex	20	14.79	71.30	6.92	43.48
BODex	100	14.80	89.84	6.91	73.53
Ours	20	27.16	91.28	18.25	84.17
Ours	100	**27.18**	**95.13**	**18.34**	**94.63**

TABLE III: **A Harder Benchmark for Fingertip Grasp Synthesis.** This benchmark uses smaller friction coefficients and more diverse objects, and our method consistently outperforms the baseline. DGN indicates DexGraspNet.

(e.g., for scales 0.17 and 0.20) or large thumb-to-object distance (e.g., for scale 0.05). FRoGGeR [25] performs well on simple objects but almost always fails on objects with complex geometries. It also tends to generate grasps with different contact normals for each fingertip (e.g., for scales 0.05 and 0.08), an issue encouraged by many previous force closure metrics. SpringGrasp [8] suffers from severe penetration and inconsistent contact distances, especially for the thumb. Additionally, their grasps lack diversity, and their thumb joint frequently exceeds the feasible range, which is not executable in both MuJoCo and the real world. Although BODex [7] demonstrates high success rates in simulation, their synthesized grasps rarely involve finger bending, resulting in unnatural poses.

2) Comparison using a harder benchmark: The benchmark in Table II uses large friction coefficients and many simple objects, which do not fully reflect the ability of each method to synthesize very high-quality grasps in more complex scenarios. To address this, we introduce a more challenging benchmark by reducing the tangential and torsional friction coefficients from 0.6 and 0.02 to 0.3 and 0.002, respectively, and randomly selecting 5000 additional objects from Objaverse [18] for testing. To mitigate the increased difficulty, we allow each method more attempts per object (from 20 to 100). We compare only with BODex, as other baselines exhibit significantly lower success rates and slower speeds.

As shown in Table III, our method significantly outperforms BODex. Notably, our method achieves an object success rate exceeding 94%, successfully grasping nearly all scaled objects, while BODex fails on about 27% of the Objaverse objects. This highlights our method's stronger generalizability to complex in-the-wild objects. Additionally, our grasp success rate continues to improve with more attempts, benefiting from continuous updates to our template repository, whereas the performance of BODex remains unchanged. This further demonstrates the adaptability of our approach.

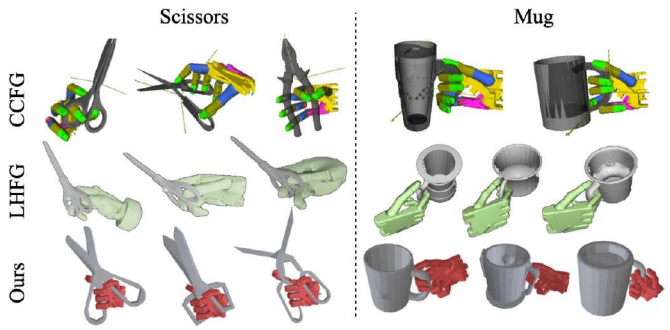

Fig. 6: **Comparison with Functional Grasp Transfer Baselines.** Our grasps involve more contact points without any penetration, indicating higher stability, especially for scissors.

	GSR(%)↑		OSR(%)↑		CLN↑	D(%)↓
	Normal	Hard	Normal	Hard		
Power	24.2	**12.8**	81.9	68.3	**9.1**	**24.7**
Intermediate	23.0	6.6	79.9	69.4	4.8	27.6
Precision	**36.0**	11.4	**95.9**	**85.6**	4.2	25.8

TABLE IV: **Statistics of Grasp Synthesis for the GRASP Taxonomy.** The success rate is lower than fingertip grasps because many flexible grasp types are suitable only for specific objects, e.g., *Lateral (#16)* grasps for flat objects.

C. Type-Aware Grasp Synthesis

1) Comparison with functional grasp transfer baselines: Unfortunately, we did not acquire the code of suitable baselines for comparison, as existing methods either do not support robotic hands (e.g., Oakink [55]) or have not made their code publicly available (e.g., LHFG [48] and CCFG [52]). Consequently, we can only perform qualitative comparisons using figures from their papers. As shown in Figure 6, previous baselines mainly use fingertips to grasp the object, particularly for scissors. In contrast, our method achieves significantly more contact points (approximately 10 links in contact for scissors and 7 for mugs), resulting in more stable and human-like grasps. Additionally, CCFG's grasps show noticeable penetrations, especially with mugs, while LHFG reports a maximum penetration of around 1 cm in their paper. In contrast, our grasps do not have any penetration.

2) Statistics analysis of our pipeline: In the absence of a suitable baseline for comparison, we provide some quantitative results in Table IV, which were gathered while synthesizing our Dexonomy dataset in Section V. The grasp types are categorized into three large groups, namely power, intermediate, and precision grasps, according to the GRASP taxonomy [20].

The overall success rate is considerably lower than that of fingertip grasp synthesis, as many flexible grasp types are designed for specific object shapes. For instance, the *Lateral (#16)* grasp is only used for flat and small objects. Among different grasp types, precision grasps exhibit the highest success rate under *normal* test conditions (i.e., with friction coefficients of 0.6 and 0.02), since these grasps typically involve only the fingertips and suit more objects. However, the success rate of precision grasps drops more rapidly than that of power grasps when the friction coefficients are reduced to 0.3 and 0.002 (i.e., the *hard* test conditions), indicating that power grasps offer higher stability due to more contact with the object. Additionally, the overall diversity of grasps is better than previous work reported in Table II, owing to the inclusion of many distinct grasp types.

		GSR↑	OSR↑	CDC↓	PD↓
Global stage	w/o opt.	45.7	88.4	0.22	**0.00**
	w/o filter	18.6	80.3	0.34	**0.00**
Local stage	w/o opt.	17.8	72.6	10.24	4.95
	w/o filter	**62.8**	**96.7**	0.82	0.34
Template library	w/o update	41.5	88.8	0.24	**0.00**
Ours		60.5	96.5	**0.21**	**0.00**

TABLE V: **Ablation Study on Pipeline Modules for Fingertip Grasp Synthesis of Allegro Hand.**

GSR (%) ↑	Shadow			Allegro
	Power	Intermediate	Precision	Fingertip
Grasp only	17.64	9.81	13.79	24.60
w/ pre-grasp	**25.85**	19.93	34.56	45.08
Ours	24.23	**23.03**	**36.00**	**60.50**

TABLE VI: **Ablation on Control Strategy for Simulation Validation**. Power grasps are robust likely due to rich contacts.

D. Ablation Study

1) Modular ablation: Table V shows the impact of each module on fingertip grasp synthesis for the Allegro hand. For the global alignment stage, optimization provides a slight improvement, while post-filtering plays a more significant role. We also take a deeper look at the post-filtering of this stage, and find that a significant portion ($>$ 50k out of 80k) is filtered out due to a high loss (Eq. 7). In contrast, collision and grasp quality filters only remove fewer than 10k grasps, indicating that using them as new losses for optimization is not effective. In the local refinement stage, optimization is crucial for improving grasp quality, while post-filtering primarily ensures penetration-free grasps and negatively affects the success rate. Additionally, constructing new grasp templates significantly enhances the synthesis success rate.

2) Control strategy ablation: In Table VI, we evaluate the impact of different control strategies during simulation testing. The *Grasp only* method computes the pre-grasp and squeeze joint angles by scaling the grasp joint angles with fixed factors: $0.9\times$ for the pre-grasp and $1.1\times$ for the squeeze pose. The *w/ pre-grasp* method calculates the squeeze joint angles as $squeeze = 2 \times grasp - pregrasp$, following BODex [7]. Note that these two strategies only calculate the joint angles, while the 6DoF hand root pose is the same as the one of the grasp pose. During execution, the hand transitions sequentially from the pre-grasp to the grasp pose, and then to the squeeze pose.

The results demonstrate that our contact-aware control strategy consistently improves the grasp success rate, except the power grasps. Power grasps are particularly robust to the control strategy, likely due to the rich contacts.

3) Robustness to initial human-annotated templates: To investigate the robustness of our pipeline to different initial

Grasp Success Rate(%)↑	6.Prismatic 4 Finger			1.Large Diameter		
Template ID	1	2	3	1	2	3
w/o new templates	45.6	12.0	1.7	22.9	22.5	7.2
w/ new templates (Ours)	**57.1**	**56.7**	**46.2**	**30.1**	**34.7**	**26.8**

TABLE VII: **Robustness to Initial Templates**. Our strategy, adding new templates from successful grasps, is the key to robustness.

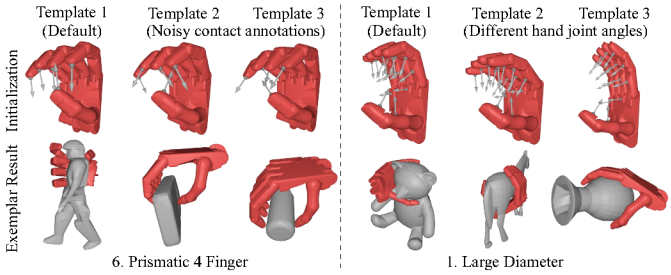

Fig. 7: **Robustness to Initial Templates**. Our method generates good grasps even for very noisy contact annotations as in columns 2 and 3.

Method	Dataset	GSR↑	OSR↑	CDC↓	PD↓	D↓
Type-uncond.	DGN [47]	8.32	44.3	20.5	15.9	29.1
	BODex [7]	54.0	84.4	11.7	**6.2**	32.0
	Ours-type1	55.5	85.9	**10.8**	8.4	31.5
	Ours-all	24.5	73.2	15.6	11.6	28.0
Type-cond.	Ours-all	**63.9**	**91.3**	13.9	8.6	**25.7**

TABLE VIII: **Learning-based Grasp Synthesis from Single-View Object Point Clouds in Simulation.** Our type-conditional model trained on our Dexonomy dataset significantly outperforms baselines.

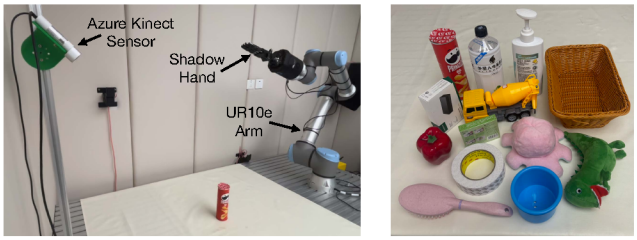

Fig. 8: **Real-World Experiment.** (Left) Hardware setup. (Right) 13 varied objects for testing.

templates, we annotated two additional templates for two common grasp types of the Shadow Hand and evaluated on 1000 random objects, with 100 attempts per object. As shown in Table VII and Figure 7, our algorithm is robust to both noisy contact annotations and variations in hand joint angles, thanks to our template-adding strategy. As long as the grasp success rate is not 0%, the template-adding strategy will construct high-quality templates from successful grasps and greatly reduce the impact of bad initial templates.

A more challenging case is using random initialization. However, this setting fails catastrophically, likely due to the high dimensionality of our template (including hand pose and contact annotations), where valid grasps are extremely sparse. This highlights the usage of human-annotated templates.

E. Learning-based Grasp Synthesis from Partial Observation

In this section, we compare the influence of both the grasping dataset and the learning method in simulation. The 10.7k objects in our Dexonomy dataset are randomly split into training and test sets with a 4:1 ratio. While the object scales used for training vary, we fix the scales during testing, using the same six scale levels as described in Section VII-B. To ensure a fair comparison, we also regenerate a dataset for BODex using our objects and scales, resulting in 0.7M valid grasps. *Ours-type1* includes only the *Large Diameter (#1)* grasp type from the Dexonomy dataset and contains 0.4M data points, while *Ours-all* uses the full 9.5M dataset. For the type-conditional model, we additionally train a classifier to select the best grasp type based on each object's point cloud; more details are provided in Appendix D. For each object, 100 candidate grasps are predicted and ranked by their associated probabilities, with the top 10 selected as the final outputs.

As shown in Table VIII, our type-conditional model trained on the Dexonomy dataset significantly outperforms the BODex baseline by around 10%, further highlighting the value of our dataset. Notably, even when using only a single grasp type with less data, the learned model still outperforms its counterpart trained on BODex. Without type-conditional features, the model struggles to learn from the diverse grasp data and performs poorly. In contrast, the type-conditional model successfully synthesizes the intended grasp types, as visualized in Figure 9. The model trained on our dataset exhibits slightly higher penetration, likely due to the fact that our grasps are more contact-rich. Contact distance consistency is also higher for *Ours-all*, as this metric considers all fingers, while some grasp types do not involve every finger.

F. Real-World Experiment

Finally, we validate the trained model in the real world. The experimental setup and 13 unseen objects for testing are shown in Figure 8. To simplify the testing process, we select 12 grasp types from the taxonomy that are distinct and suitable for grasping from a table. For 3 simple objects, namely the red cylinder, the white box, and the red apple, we use all 12 grasp types to grasp them. For the other 10 objects, we let GPT-4o choose 3 suitable grasp types by providing pictures of the grasp types and the objects. Each object and grasp type combination is tested in 3 trials, resulting in around 200 total testing trials.

To perform a grasp, a single-view object point cloud segmented by SAM2 [39] and the specified grasp type are taken as input to the trained type-conditional generative model. The model generates 100 candidates and we use the pre-grasp poses as the target for collision-free motion planning with CuRobo [41], filtering out failed ones. The remaining grasps are ordered by the output probability of the normalizing flow, and the top 3 are executed. In this way, we prevent the success rate of motion planning from affecting the results, since it is not the focus of this paper. After reaching the pre-grasp pose,

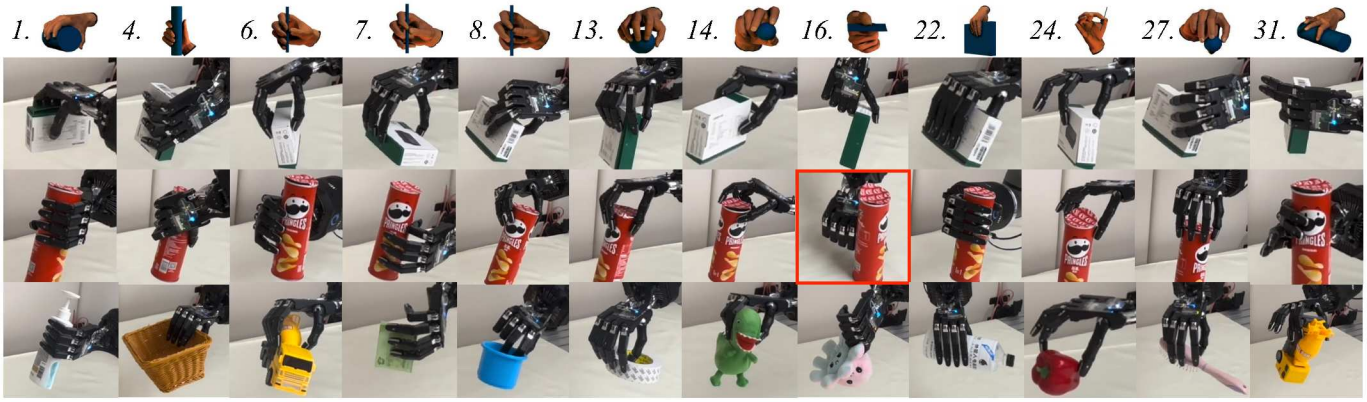

Fig. 9: **Real-World Gallery.** Our trained type-conditional generative model successfully synthesizes desired grasp types from single-view object point clouds. All grasps succeed in lifting the object except the one in the red box, where the grasp type is unsuitable for the object.

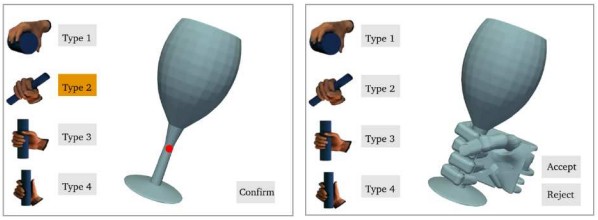

Fig. 10: **An Annotation UI based on Our Algorithm for Collecting Functional Grasp.** (Left) The user *clicks twice* to specify a contact point on the object and a grasp type. (Right) A high-quality grasp is synthesized according to the user's needs within seconds.

the hand moves to the grasp pose and then the squeeze pose to grasp the object stably, and finally lifts it.

As shown in Figure 9, our model can correctly generate physically plausible grasps for the specified types and achieves an overall success rate of 82.3%. The most common failure mode is the unsuitable grasp types for the object, especially for *Lateral (#16)*. Moreover, some grasp types, such as *Tip Pinch (#24)*, are not very robust and easily fail due to real-world noise and prediction errors. More details are provided in Appendix E.

VIII. Application

Although our algorithm is semantic-unaware and cannot directly synthesize grasps to touch object regions specified by human language commands, it can be used to develop an efficient annotation system for collecting semantic dexterous grasp data. Unlike widely used teleoperation methods, which often require well-trained annotators and hardware dependencies like data gloves, our annotation system has minimal requirements, relying only on simple mouse clicks.

As shown in Figure 10, the annotator only needs to click twice: once to specify a contact point on the object and once to select a desired grasp type. Our algorithm will automatically sample nearby object points and grasp templates from existing libraries, and synthesize valid grasps, with the best results displayed in the GUI within seconds. For a full demonstration, please refer to our supplementary video. We plan to continue improving this tool and hope it facilitates future research on semantic grasping.

IX. Limitations and Future Work

First, our synthesized grasps occasionally fail due to unsuitable or unstable grasp types, as shown in Figure 9. Future work could explore a more suitable taxonomy for robotic grasping, as well as improved strategies for grasp type selection. Second, our method focuses on synthesizing static grasp poses rather than generating sequential trajectories that involve contact changes. A promising direction for future research is to incorporate trajectory generation for dynamic grasping, potentially leveraging reinforcement learning[9, 57] and differentiable physics simulators [16, 17, 54]. Finally, our work focuses solely on grasping a single object, and the extension to cluttered scenes remains an open challenge for future exploration.

X. Conclusion

In this work, we present a novel pipeline to efficiently synthesize high-quality dexterous grasps for any grasp type, object, and articulated object hand, starting from just one human-annotated template. Our pipeline greatly outperforms previous type-unaware grasp synthesis baselines in the simulation benchmark. Using our proposed method, a large-scale dataset covering 31 grasp types is constructed, enabling the training of a type-conditional grasp generative model. The trained model successfully generates desired grasp types from single-view object point clouds, achieving a real-world success rate of 82.3%.

Acknowledgments

We thank Danshi Li for assisting with the integration of the Inspire Hand and the Unitree G1 Hand.

References

[1] Ananye Agarwal, Shagun Uppal, Kenneth Shaw, and Deepak Pathak. Dexterous functional grasping. In *7th Annual Conference on Robot Learning*, 2023.

[2] Arun L Bishop, John Z Zhang, Swaminathan Gurumurthy, Kevin Tracy, and Zachary Manchester. Reluqp: A gpu-accelerated quadratic programming solver for model-predictive control. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13285–13292. IEEE, 2024.

[3] Samarth Brahmbhatt, Chengcheng Tang, Christopher D Twigg, Charles C Kemp, and James Hays. Contactpose: A dataset of grasps with object contact and hand pose. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, pages 361–378. Springer, 2020.

[4] Hanwen Cao, Hao-Shu Fang, Wenhai Liu, and Cewu Lu. Suctionnet-1billion: A large-scale benchmark for suction grasping. *IEEE Robotics and Automation Letters*, 6(4): 8718–8725, 2021.

[5] Yu-Wei Chao, Wei Yang, Yu Xiang, Pavlo Molchanov, Ankur Handa, Jonathan Tremblay, Yashraj S Narang, Karl Van Wyk, Umar Iqbal, Stan Birchfield, et al. Dexycb: A benchmark for capturing hand grasping of objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9044–9053, 2021.

[6] Jiayi Chen, Yuxing Chen, Jialiang Zhang, and He Wang. Task-oriented dexterous grasp synthesis via differentiable grasp wrench boundary estimator. *arXiv preprint arXiv:2309.13586*, 2023.

[7] Jiayi Chen, Yubin Ke, and He Wang. Bodex: Scalable and efficient robotic dexterous grasp synthesis using bilevel optimization. *arXiv preprint arXiv:2412.16490*, 2024.

[8] Sirui Chen, Jeannette Bohg, and C Karen Liu. Springgrasp: An optimization pipeline for robust and compliant dexterous pre-grasp synthesis. *arXiv preprint arXiv:2404.13532*, 2024.

[9] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84):eadc9244, 2023.

[10] Zerui Chen, Shizhe Chen, Etienne Arlaud, Ivan Laptev, and Cordelia Schmid. Vividex: Learning vision-based dexterous manipulation from human videos. *arXiv preprint arXiv:2404.15709*, 2024.

[11] X Cheng, J Li, S Yang, G Yang, and X Wang. Open-television: Teleoperation with immersive active visual feedback, 2024. *URL https://arxiv. org/abs/2407.01512*.

[12] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3075–3084, 2019.

[13] Sammy Christen, Muhammed Kocabas, Emre Aksan, Jemin Hwangbo, Jie Song, and Otmar Hilliges. D-grasp: Physically plausible dynamic grasp synthesis for hand-object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20577–20586, 2022.

[14] Matei Ciocarlie, Corey Goldfeder, and Peter Allen. Dexterous grasping via eigengrasps: A low-dimensional approach to a high-complexity problem. In *Robotics: Science and systems manipulation workshop-sensing and adapting to the real world*, 2007.

[15] Erwin Coumans. Bullet physics simulation. In *ACM SIGGRAPH 2015 Courses*, page 1. 2015.

[16] Filipe de Avila Belbute-Peres, Kevin Smith, Kelsey Allen, Josh Tenenbaum, and J Zico Kolter. End-to-end differentiable physics for learning and control. *Advances in neural information processing systems*, 31, 2018.

[17] Jonas Degrave, Michiel Hermans, Joni Dambre, and Francis Wyffels. A differentiable physics engine for deep learning in robotics. *Frontiers in neurorobotics*, 13:6, 2019.

[18] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023.

[19] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11444–11453, 2020.

[20] Thomas Feix, Javier Romero, Heinz-Bodo Schmiedmayer, Aaron M Dollar, and Danica Kragic. The grasp taxonomy of human grasp types. *IEEE Transactions on human-machine systems*, 46(1):66–77, 2015.

[21] Carlo Ferrari, John F Canny, et al. Planning optimal grasps. In *ICRA*, volume 3, page 6, 1992.

[22] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[23] Juntao Jian, Xiuping Liu, Manyi Li, Ruizhen Hu, and Jian Liu. Affordpose: A large-scale dataset of hand-object interactions with affordance-driven hand pose. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14713–14724, 2023.

[24] Hanwen Jiang, Shaowei Liu, Jiashun Wang, and Xiaolong Wang. Hand-object contact consistency reasoning for human grasps generation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11107–11116, 2021.

[25] Albert H Li, Preston Culbertson, Joel W Burdick, and Aaron D Ames. Frogger: Fast robust grasp generation via the min-weight metric. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*,

pages 6809–6816. IEEE, 2023.

[26] Yuyang Li, Wenxin Du, Chang Yu, Puhao Li, Zihang Zhao, Tengyu Liu, Chenfanfu Jiang, Yixin Zhu, and Siyuan Huang. Taccel: Scaling up vision-based tactile robotics via high-performance gpu simulation, 2025.

[27] Tengyu Liu, Zeyu Liu, Ziyuan Jiao, Yixin Zhu, and Song-Chun Zhu. Synthesizing diverse and physically stable grasps with arbitrary hand structures using differentiable force closure estimator. *IEEE Robotics and Automation Letters*, 7(1):470–477, 2021.

[28] Yulin Liu, Haoran Liu, Yingda Yin, Yang Wang, Baoquan Chen, and He Wang. Delving into discrete normalizing flows on so (3) manifold for probabilistic rotation modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21264–21273, 2023.

[29] Yumeng Liu, Yaxun Yang, Youzhuo Wang, Xiaofei Wu, Jiamin Wang, Yichen Yao, Sören Schwertfeger, Sibei Yang, Wenping Wang, Jingyi Yu, et al. Realdex: Towards human-like grasping for robotic dexterous hand. *arXiv preprint arXiv:2402.13853*, 2024.

[30] Yunze Liu, Yun Liu, Che Jiang, Kangbo Lyu, Weikang Wan, Hao Shen, Boqiang Liang, Zhoujie Fu, He Wang, and Li Yi. Hoi4d: A 4d egocentric dataset for category-level human-object interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21013–21022, 2022.

[31] Miles Macklin. Warp: A high-performance python framework for gpu simulation and graphics. https://github.com/nvidia/warp, March 2022. NVIDIA GPU Technology Conference (GTC).

[32] Jeffrey Mahler, Florian T Pokorny, Brian Hou, Melrose Roderick, Michael Laskey, Mathieu Aubry, Kai Kohlhoff, Torsten Kröger, James Kuffner, and Ken Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1957–1964. IEEE, 2016.

[33] Jeffrey Mahler, Matthew Matl, Xinyu Liu, Albert Li, David Gealy, and Ken Goldberg. Dex-net 3.0: Computing robust vacuum suction grasp targets in point clouds using a new analytic model and deep learning. In *2018 IEEE International Conference on robotics and automation (ICRA)*, pages 5620–5627. IEEE, 2018.

[34] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

[35] Andrew T Miller and Peter K Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004.

[36] Ken Museth, Jeff Lait, John Johanson, Jeff Budsberg, Ron Henderson, Mihai Alden, Peter Cucka, David Hill, and Andrew Pearce. Openvdb: an open-source data structure and toolkit for high-resolution volumes. In *Acm siggraph 2013 courses*, pages 1–1. 2013.

[37] Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *European Conference on Computer Vision*, pages 570–587. Springer, 2022.

[38] Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dieter Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. *arXiv preprint arXiv:2307.04577*, 2023.

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

[40] Máximo A Roa and Raúl Suárez. Grasp quality measures: review and performance. *Autonomous robots*, 38: 65–88, 2015.

[41] Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, et al. Curobo: Parallelized collision-free robot motion generation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8112–8119. IEEE, 2023.

[42] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[43] Dylan Turpin, Liquan Wang, Eric Heiden, Yun-Chun Chen, Miles Macklin, Stavros Tsogkas, Sven Dickinson, and Animesh Garg. Grasp'd: Differentiable contact-rich grasp synthesis for multi-fingered hands. In *European Conference on Computer Vision*, pages 201–221. Springer, 2022.

[44] Dylan Turpin, Tao Zhong, Shutong Zhang, Guanglei Zhu, Eric Heiden, Miles Macklin, Stavros Tsogkas, Sven Dickinson, and Animesh Garg. Fast-grasp'd: Dexterous multi-finger grasp generation through differentiable simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8082–8089. IEEE, 2023.

[45] Sébastien Valette, Jean Marc Chassery, and Rémy Prost. Generic remeshing of 3d triangular meshes with metric-dependent discrete voronoi diagrams. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):369–381, 2008.

[46] Weikang Wan, Haoran Geng, Yun Liu, Zikang Shan, Yaodong Yang, Li Yi, and He Wang. Unidexgrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3891–3902, 2023.

[47] Ruicheng Wang, Jialiang Zhang, Jiayi Chen, Yinzhen Xu, Puhao Li, Tengyu Liu, and He Wang. Dexgraspnet: A large-scale robotic dexterous grasp dataset for general objects based on simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11359–11366. IEEE, 2023.

[48] Wei Wei, Peng Wang, Sizhe Wang, Yongkang Luo, Wanyi Li, Daheng Li, Yayu Huang, and Haonan Duan. Learning human-like functional grasping for multi-finger hands from few demonstrations. *IEEE Transactions on Robotics*, 2024.

[49] Xinyue Wei, Minghua Liu, Zhan Ling, and Hao Su. Approximate convex decomposition for 3d meshes with collision-aware concavity and tree search. *ACM Transactions on Graphics (TOG)*, 41(4):1–18, 2022.

[50] Zehang Weng, Haofei Lu, Danica Kragic, and Jens Lundell. Dexdiffuser: Generating dexterous grasps with diffusion models. *arXiv preprint arXiv:2402.02989*, 2024.

[51] Albert Wu, Michelle Guo, and C Karen Liu. Learning diverse and physically feasible dexterous grasps with generative model and bilevel optimization. *arXiv preprint arXiv:2207.00195*, 2022.

[52] Rina Wu, Tianqiang Zhu, Xiangbo Lin, and Yi Sun. Cross-category functional grasp tansfer. *arXiv preprint arXiv:2405.08310*, 2024.

[53] Yinzhen Xu, Weikang Wan, Jialiang Zhang, Haoran Liu, Zikang Shan, Hao Shen, Ruicheng Wang, Haoran Geng, Yijia Weng, Jiayi Chen, et al. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4737–4746, 2023.

[54] Gang Yang, Siyuan Luo, Yunhai Feng, Zhixin Sun, Chenrui Tie, and Lin Shao. Jade: A differentiable physics engine for articulated rigid bodies with intersection-free frictional contact. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16915–16922. IEEE, 2024.

[55] Lixin Yang, Kailin Li, Xinyu Zhan, Fei Wu, Anran Xu, Liu Liu, and Cewu Lu. Oakink: A large-scale knowledge repository for understanding hand-object interaction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20953–20962, 2022.

[56] Shiqi Yang, Minghuan Liu, Yuzhe Qin, Runyu Ding, Jialong Li, Xuxin Cheng, Ruihan Yang, Sha Yi, and Xiaolong Wang. Ace: A cross-platform visual-exoskeletons system for low-cost dexterous teleoperation. *arXiv preprint arXiv:2408.11805*, 2024.

[57] Zhao-Heng Yin, Changhao Wang, Luis Pineda, Francois Hogan, Krishna Bodduluri, Akash Sharma, Patrick Lancaster, Ishita Prasad, Mrinal Kalakrishnan, Jitendra Malik, et al. Dexteritygen: Foundation controller for unprecedented dexterity. *arXiv preprint arXiv:2502.04307*, 2025.

[58] Kevin Zakka, Yuval Tassa, and MuJoCo Menagerie Contributors. MuJoCo Menagerie: A collection of high-quality simulation models for MuJoCo, September 2022.

[59] Hui Zhang, Sammy Christen, Zicong Fan, Otmar Hilliges, and Jie Song. Graspxl: Generating grasping motions for diverse objects at scale. In *European Conference on Computer Vision*, pages 386–403. Springer, 2025.

[60] Jialiang Zhang, Haoran Liu, Danshi Li, XinQiang Yu, Haoran Geng, Yufei Ding, Jiayi Chen, and He Wang. Dexgraspnet 2.0: Learning generative dexterous grasping in large-scale synthetic cluttered scenes. In *8th Annual Conference on Robot Learning*, 2024.

[61] Tianqiang Zhu, Rina Wu, Xiangbo Lin, and Yi Sun. Toward human-like grasp: Dexterous grasping via semantic representation of object-hand. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15741–15751, 2021.

APPENDIX

A. Post-Filtering for Global Alignment Stage

Detecting severe penetration: We propose a skeleton representation for the hand, consisting of several line segments, as shown in Figure 11. Since the skeleton is parameterized by line segments, collision detection is efficiently performed through an intersection query between the line segment and the object's mesh. If the posed skeleton collides with the object, the hand is considered to have penetrated too much and is discarded. Note that the skeleton only needs to be defined once for each robotic hand.

Filter out duplicate transformations: This process iteratively selects the farthest transformations until reaching a predefined number or the farthest distance to other transformations is less than a predefined threshold. The distance metric between two transformations is computed as a weighted sum of: (1) the angular difference between rotations, (2) the Euclidean distance between translations, and (3) optionally, the difference in scales.

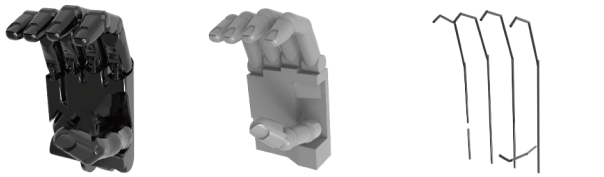

Fig. 11: **Geometric Representations for Shadow Hand.** Our skeleton representation, based on line segments, efficiently detects severe penetrations with the object in the post-filtering of the global alignment stage. The fingertip's skeleton is designed to be shorter, because the fingertip's penetration is often easy to resolve in the local refinement stage.

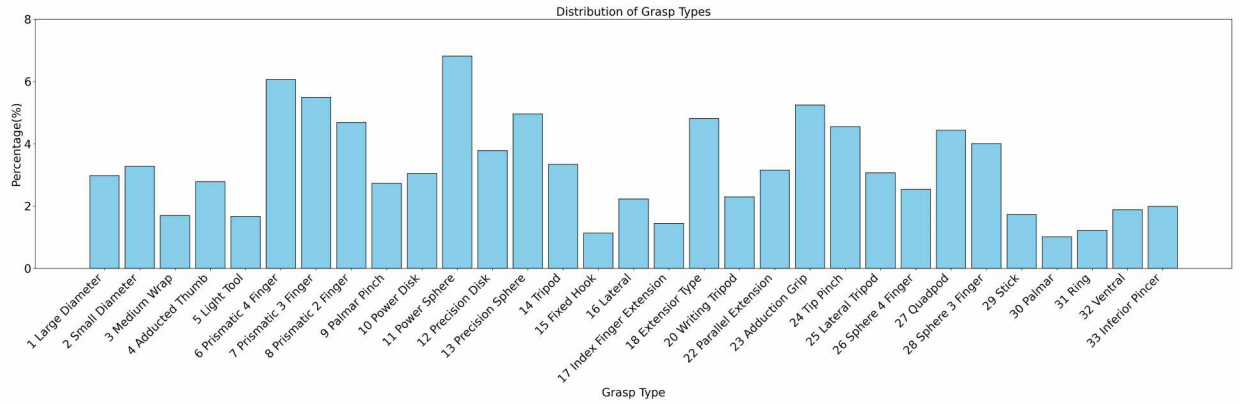

Fig. 12: **Distribution of Grasp Types in the Dexonomy Dataset.** The distribution of grasp types exhibits some bias, which is inevitable due to the varying suitability and stability of different types.

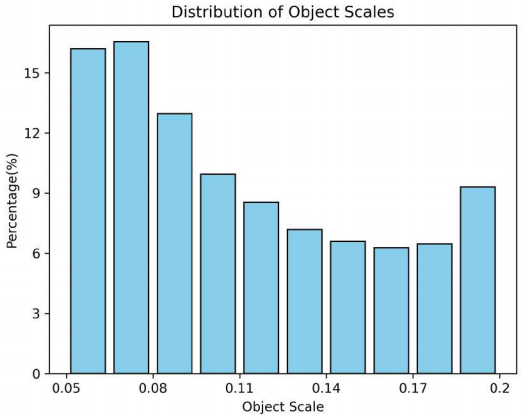

Fig. 13: **Distribution of Object Scales in the Dexonomy Dataset.** The distribution is slightly skewed toward smaller object scales, as smaller objects tend to yield higher grasp success rates—a trend also observed in BODex [7].

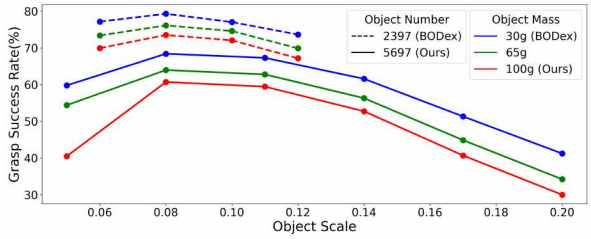

Fig. 14: **Performance Discrepancy of BODex**. The performance of BODex reported in Table II is different from the original paper due to the object set, scale, and mass.

B. Details of Dexonomy Dataset

All objects are pre-processed using CoACD [49] for convex decomposition, OpenVDB [36] for internal face removal, and ACVD [45] for mesh simplification. The OpenVDB and ACVD are essential to support nearest-point queries. Our object pre-processing is very robust, with a success rate over 95% for in-the-wild objects, such as those from Objaverse [18].

The dataset generation process starts with one manually annotated grasp template per grasp type. Our algorithm runs for 10 epochs, iterating through all objects with 10 attempts per object. Compared to using 1 epoch with 100 attempts, this approach can try more grasp templates for each object, as new templates are actively constructed during the process.

The resulting distributions of the object scales and grasp types are shown in Figure 13 and Figure 12, respectively.

C. Performance Drop of BODex

As shown in Figure 14, the performance of the type-unaware grasp synthesis baselines in Table II is different from those reported in BODex [7] because of three reasons:

- **Object set**: BODex selects 2397 objects from DexGraspNet [47] using a heuristic about the minimum AABB length to filter out flat objects, while this work uses all 5697 objects from DexGraspNet.
- **Object scale**: our paper uses a larger object scale range ($[0.05, 0.2]$) than BODex ($[0.06, 0.12]$).
- **Object mass**: our paper uses a larger mass ($100g$) than BODex ($30g$), which is more challenging.

D. More Experimental Results on Learning Method

In this section, we present additional preliminary experiments on learning-based grasp synthesis.

First, we compare the impact of different generative model architectures, as shown in Table IX. While diffusion models is more popular [60], their performance on our dataset is inferior to that of the Mobius Normalizing Flow [28, 7]. Additionally, the predicted grasp poses are significantly worse without incorporating an MLP head—a phenomenon also observed in [60]. We hypothesize that this is because the hand joint configurations (qpos) for grasps of the same type are relatively similar and thus easier for an MLP to learn, whereas the 6-

	Diffusion	Diffusion + MLP	Flow	Flow + MLP
GSR (%) ↑	16.0	41.8	NaN	55.5

TABLE IX: **Different Learning Architectures Trained on Grasp Type 1**. *Flow* denotes for mobius normalizing flow. NaN means that the training fails with NaN gradients.

Dataset	GSR↑	OSR↑	CDC↓	PD↓	D↓
Ours-type1	55.5	85.9	**10.8**	8.4	**31.5**
Ours-type6	48.8	83.8	11.4	7.0	34.1
Ours-type9	48.0	79.8	15.5	**6.6**	34.8
Ours-type18	**61.2**	**87.0**	13.3	8.6	34.0
Ours-type22	52.0	79.1	12.6	9.0	35.4
Ours-type26	51.3	82.0	14.0	10.3	33.5
Ours-type31	47.8	81.8	14.9	6.9	32.9
Ours-type33	46.3	77.7	15.7	6.7	33.8

TABLE X: **Type-unconditional Model Trained on High-Quality Grasp Types**. The selected grasp types exhibit high success rates during the synthesis of the Dexonomy dataset.

DoF root pose of the hand has a more complex distribution that requires the generative model to capture.

Second, we select several grasp types with high success rates from the Dexonomy dataset and train type-unconditional models on them. As shown in Table X, although the performance varies across grasp types, the overall results remain strong.

Finally, we evaluate different testing strategies for our type-conditional model trained on the full Dexonomy dataset. In the *Average* setting, we generate all 31 grasp types for each testing object point cloud and report the average success rate. In the *Classifier* setting, we train a separate classifier to predict the most suitable grasp type for each object point cloud. *Classifier A* uses the grasp type with the highest success rate per training object in the original Dexonomy dataset as the ground-truth label. In contrast, *Classifier B* synthesizes grasps on training objects using the trained type-conditional model and selects the type with the highest success rate as the ground truth. The classifier employs a 3D sparse convolutional backbone (copied and frozen from the trained type-conditional model) followed by an MLP that outputs grasp type probabilities. The *Top 1* setting directly records the highest success rate across all grasp types for each test object, serving as an oracle upper bound.

As shown in Table XI, the performance of the *Average* strategy is poor, likely because many objects are not compatible with all grasp types. In contrast, the *Top 1* oracle achieves very high performance, indicating the strong potential of our dataset when paired with an effective grasp type selection mechanism. Between the two classifiers, *Classifier B* outperforms *Classifier A*, likely because the grasp type preferences of the trained model differ from those of our pipeline used to synthesize the Dexonomy dataset. Notably, the *Top 1* oracle and using *Classifier B* outperform all type-unconditional models in Table X, further highlights the advantage of studying different types for grasping.

	Average	Top 1 (oracle)	Classifier A	Classifier B
GSR (%)	28.9	**78.2**	46.1	63.9

TABLE XI: **Type-Conditional Model with Different Testing Methods.** The performances of *Top 1* oracle and *Classifier B* outperform type-unconditional models in Table X, highlighting the potential of studying different types for grasping.

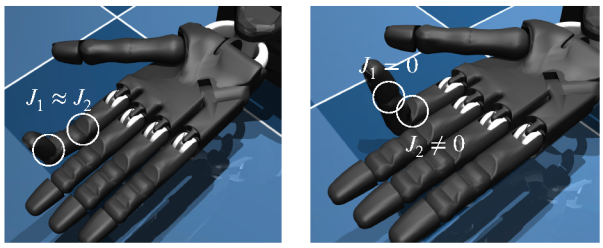

Fig. 15: **Sim-Real Misalignment of Shadow Hand**. (Left) The widely used Shadow hand assets in simulation. (Right) The real-world behavior.

E. Details of Real-World Experiment

One may find that the farthest joint of all fingers, except the thumb finger, never bends in our real-world experiments. This is because the under-actuation mechanism of the real Shadow hand differs from the commonly used one in simulation from MuJoCo Menagerie [58], as shown in Figure 15. In simulation, the two farthest joints of each finger, namely J_1 and J_2, are always equal to each other. However, in the real world, the farthest joint J_1 only bends after the second joint J_2 reaches around 90 degrees. At the time of this work, we were unsure how to accurately simulate the phenomenon, so we fixed the farthest joint to zero, annotated new templates, synthesized new data, and trained a new model for real-world deployment. However, we have since found a potential solution and plan to explore it in future work.

Another noticeable issue is related to motion planning. First, the success rate of motion planning is low (less than 30%). Second, even when the motion planning succeeds, there are sometimes unexpected collisions while moving to the pre-grasp pose. We suspect that CuRobo may not be well-suited for grasping tasks involving contact, as the planned trajectories are consistently close to the obstacle at all timesteps. Ideally, trajectories should maintain a larger hand-object distance, except in the final few timesteps, which would make them more robust to real-world noise and partial observations. This can be explored in future work.

F. Detailed Time Analysis

The times reported in Table II represent the maximum speed for synthesis without testing. This section provides a more detailed time breakdown of our proposed grasp synthesis pipeline.

First, the *lightweight global alignment* stage processes over 100,000 initial samples in approximately 3 seconds on a single 3090 GPU. The maximum number of intermediate results

generated for the next stage can be controlled via a hyperparameter. We typically process 10 objects in parallel, with 10 results per object. For grasp types that are commonly suitable for many objects, this stage is usually not the bottleneck, as it can synthesize over 200 intermediate results per second using 8 GPUs. However, for more challenging grasp types that are hard to match, this stage can become the bottleneck, because there may be less than 20 results per second.

The optimization step consistently takes around 1.2 seconds, while the time cost of calculating the grasp quality metric for post-filtering varies significantly, ranging from 0.3 to 1.5 seconds. When many samples are filtered out, leaving only around 5,000 for energy calculation, the process takes approximately 0.3 seconds. This speed is achieved by using the batched Relu-QP [2] algorithm as in BODex [7], whereas traditional CPU-based QP solvers are significantly slower. The other operations are very fast.

Next, the *simulation-based local refinement* stage requires 200 simulation steps, which take less than 0.1 seconds. This stage is highly efficient, easily synthesizing more than 200 grasps per second when utilizing 32 threads.

Finally, the *simulation validation* stage often becomes the speed bottleneck, as it involves approximately 3,000 simulation steps (6 external force directions, with 500 steps per direction). Despite employing early-stop strategies to handle failure cases, this stage can only process about 40 grasps per second using 48 threads. The slow speed has nothing to do with our proposed contact-aware control strategy and is consistent for other synthesis baselines if they want to test in MuJoCo. Future work may try to use GPU-based MuJoCo or other faster physics simulators [26] for testing.