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 [6], **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, 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 [7], 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 [6]. However, collecting 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 [17, 16, 9, 3, 2] 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 [21, 18, 19] 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 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 [15] — 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 sample and optimize 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.

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, but also ensures that the final grasp remains similar to the initial grasp template and thus remains natural. In contrast to most prior work [8, 17, 3, 2] 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 [17, 23] 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. 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 hand, starting from one human-annotated template per hand and grasp type.
- A large-scale dataset with 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. Method

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 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

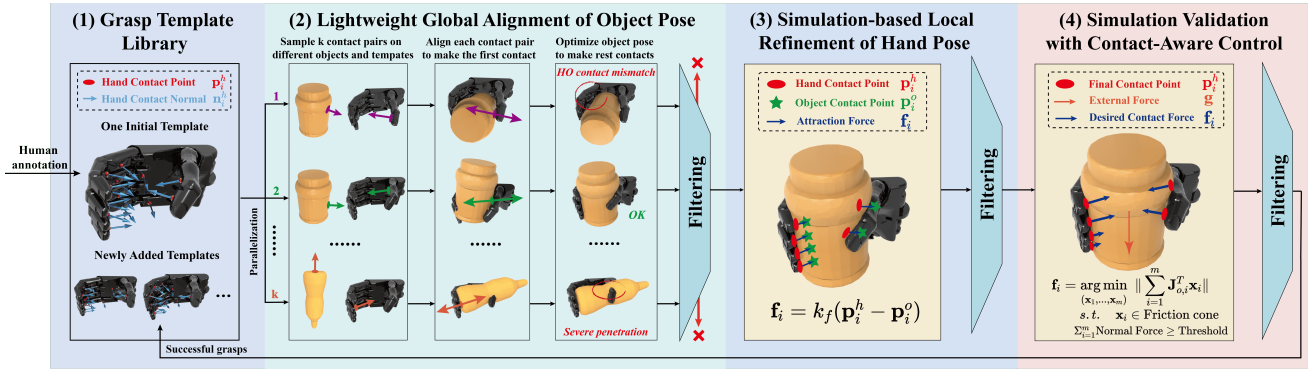

Fig. 2: **The pipeline of Dexonomy.** (1) *Grasp Template Library* initially requires one human-annotated template. (2) *Lightweight Global Alignment* stage samples and optimizes the object poses in parallel on a GPU, to match the contact points and normals of the selected grasp templates. (3) *Simulation-based 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.

normal direction are sampled randomly. 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` [12]. To penalize the mismatch between hand and object contacts, we optimize the object pose by minimizing the following energy function:

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

where k_p and k_n are hyperparameters. There is no other energy used for optimization except Eq. 1.

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-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 SUPP). Third, the object contact quality, as measured by the QP-based grasp energy from BODex [2], must exceed a threshold. Finally, we apply a process similar to farthest point sampling to filter out duplicate object transformations.

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 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:

$$\mathbf{f}_i = k_f(\mathbf{p}_i^h - \mathbf{p}_i^o), \quad \tau = \sum_{i=1}^{m} \mathbf{J}_{h,i}^T \mathbf{f}_i \qquad (2)$$

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. 2 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 our need for synthesizing contact-rich grasps in simulation. This is easy to implement and works for other physics simulators. Eq. 2 is iteratively applied for 200 steps, with $\mathbf{p}_i^h$ remaining static in the hand link frame and $\mathbf{p}_i^o$ remaining static in the world frame to avoid drift.

After optimization, we filter the results based on three criteria. First, there should be no hand-object penetration, measured using collision meshes. Second, all fingers that have at least one annotated contact should touch the object, meaning the minimal distance between hand links and object meshes should be within 2 mm. Finally, the grasp quality must exceed a threshold, as 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 the quadratic programming (QP) [2], and then converts these forces into joint torques using the same transposed Jacobian control as in Eq. 2. 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

	GSR (%) ↑	OSR (%) ↑	S (s^{-1}) ↑	CLN ↑	CDC (mm) ↓	PD (mm) ↓	SPD (mm) ↓	D (%) ↓
DexGraspNet [17]	12.10	57.01	3.25	3.22	7.58	4.85	1.20	29.03
FRoGGeR [9]	10.34	55.70	2.98	2.51	4.95	0.22	**0.00**	**27.01**
SpringGrasp [3]	7.83	35.44	5.47	2.79	23.59	16.58	1.06	70.18
BODex [2]	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 I: **Comparison with Type-Unaware Grasp Synthesis Baselines for Allegro Hand in Simulation.** Most baselines, except DexGraspNet, only synthesize fingertip grasps, so we also synthesize fingertip grasps for a fair comparison.

Fig. 3: **Real-World Gallery.** Our trained type-conditional generative model synthesizes desired grasp types from single-view object point clouds. All grasps succeed except the one in the red box, where the grasp type is unsuitable for the object.

from the successful grasp, while the contact information 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 in the global alignment stage of the following loops.

III. Experiment

A. Type-Unaware Grasp Synthesis in Simulation

Although our work focuses on type-aware dexterous grasp synthesis, there is no suitable baseline available for direct comparison. Therefore, we conduct experiments on type-unaware grasp synthesis in simulation to demonstrate the effectiveness of our pipeline.

Evaluation metrics. Eight metrics similar to BODex [2] are used for a comprehensive evaluation: Grasp Success Rate (GSR), Object Success Rate (OSR), Speed (S), Contact Link Number (CLN), Contact Distance Consistency (CDC), Penetration Depth (PD), Self-Penetration Depth (SPD), Diversity (D). The detailed description of each metric is in SUPP.

Experiment setup. We use the Allegro hand and 5689 object assets from DexGraspNet, with six scales applied to each normalized object: 0.05, 0.08, 0.11, 0.14, 0.17, and 0.20. Each method allows 20 attempts, where for our method one attempt is defined as one valid result output by the global alignment stage. The reported speed does not include simulation validation, and the detailed time is in SUPP.

Result analysis. As shown in Table I, our method achieves the highest grasp success rate and best performance on contact and penetration. The penetration for our grasps is consistently 0 because we set a 1mm contact margin in MuJoCo, and MuJoCo can resolve millimeter-level penetration. Our speed is slightly lower than that of BODex, as their pipeline mainly runs on GPUs, while our local refinement stage uses MuJoCo's CPU version. Our diversity is somewhat lower, as we use only two similar templates and a smaller step number for refining hand poses compared to the baselines. However, the overall diversity of our synthesized grasps for all grasp types is much better, as reported in SUPP. The success rates of baselines are lower than those reported in BODex [2], primarily because our objects have a higher mass (100g vs. 30g) and a larger scale range ($[0.05, 0.2]$ vs. $[0.06, 0.12]$).

B. Learning Type-Aware Grasp Synthesis

Using our proposed method, we generate a large-scale dataset for Shadow Hand covering 31 grasp types in the GRASP taxonomy. We also propose a type-conditional generative model based on normalizing flow [20, 2]. The main idea is just to add a conditional codebook, where each grasp type corresponds to a code in it. Since the learning model is just used as proof-of-concept and not the main contribution of this paper, the details are left in SUPP.

To perform a grasp, a single-view object point cloud segmented by SAM2 [13] 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 [14], 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, the hand moves to the grasp pose and then the squeeze pose to grasp the object stably, and finally lift it. As shown in Fig. 3, our model can correctly generate physically plausible grasps for the specified types and achieves an overall success rate of 82.3% on 13 test objects.

References

[1] 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.

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

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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).

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

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

[20] 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.

[21] 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.

[22] 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.

[23] 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.

APPENDIX

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, 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. Experiment: Type-unaware Grasp Synthesis

1) Visualization Comparison: Figure 4 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.

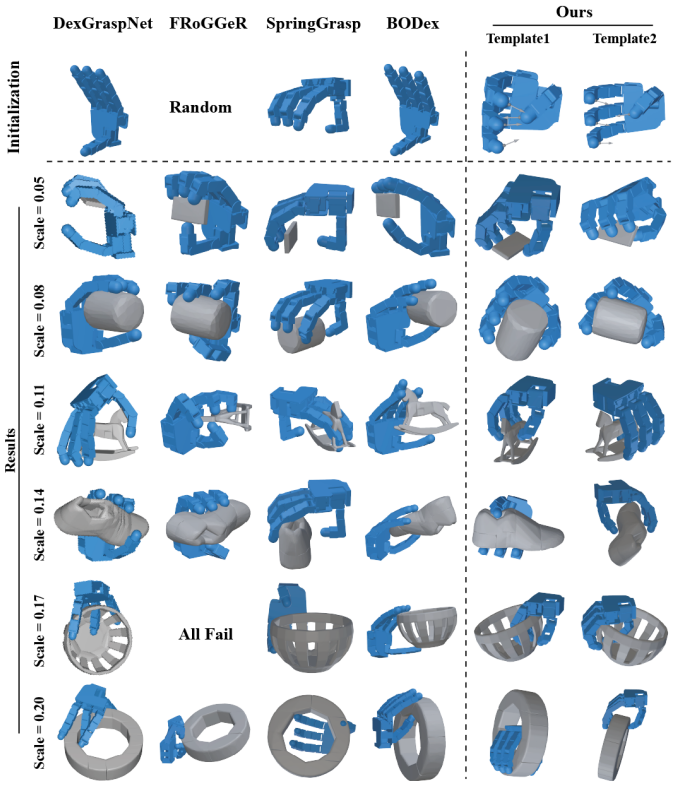

Fig. 4: **Visualization of Synthesized Fingertip Grasps.** Our method synthesizes human-like and stable grasps, even for objects with complex geometries (e.g., object scales = 0.11, 0.17, and 0.20).

For baseline methods, DexGraspNet [17] 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 (e.g., for scales 0.17 and 0.20) or large thumb-to-object distance (e.g., for scale 0.05). FRoGGeR [9] 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 [3] 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 [2] 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 the main paper uses large friction coefficients and many

Method	Attempt Number	DGN object [17]		Objaverse [5]	
		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 II: **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.

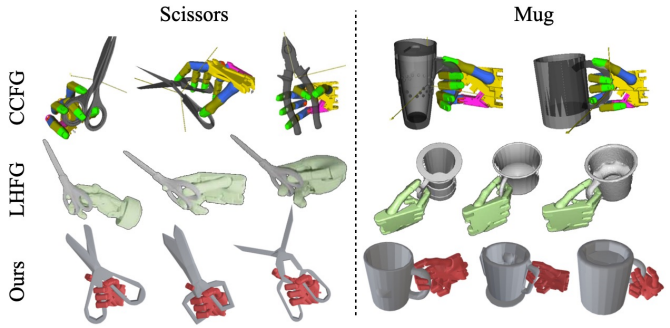

Fig. 5: **Comparison with Functional Grasp Transfer Baselines.** Our grasps involve more contact points while ensuring no penetration, indicating higher stability, particularly for scissors.

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 [5] 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 II, 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.

C. Experiment: Type-aware Grasp Synthesis

1) Visual Comparison: Unfortunately, we didn't acquire the code of suitable baselines for comparison, as existing methods either do not support robotic hands (e.g., Oakink [21]) or have not made their code publicly available (e.g., LHFG [18] and CCFG [19]) despite our requests through emails. Consequently, we can only perform qualitative comparisons using

	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 III: **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.

figures from their papers. As shown in Fig. 5, previous baselines mainly use fingertips to grasp the object, particularly for scissors. In contrast, our method achieves significantly more contact points (approximately 10 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 about 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 III, which were gathered while synthesizing our Dexonomy dataset in Section F. The grasp types are categorized into three large groups, namely power, intermediate, and precision grasps, according to the GRASP taxonomy [6].

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 the main paper, owing to the inclusion of many distinct grasp types.

D. Experiment: Learning-based Grasp Synthesis in Simulation

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 **??**. 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. For

Dataset	Hand	Sim./Real	Objects	Grasps	Grasp Types	Force Closure	Data Type	Method
DexGraspNet [17]	Shadow	IsaacGym	5.4k	1.32M	Random	✓	Grasp pose	Optimization
RealDex [11]	Shadow	Real	52	59k	Random	✗	Motion	Teleoperation
GraspXL [22]	Multiple	RaiSim	500k	10M	Random	✗	Motion	RL
BODex [2]	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 IV: **Dexterous Grasp Dataset Comparison.** Our large-scale dataset aims to support the study of data-driven methods for type-aware grasp synthesis.

Method	Dataset	GSR↑	OSR↑	CDC↓	PD↓	D↓
Type-uncond.	DGN [17]	8.32	44.3	20.5	15.9	29.1
	BODex [2]	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 V: **Learning-based Grasp Synthesis from Single-View Object Point Clouds in Simulation.** Our type-conditional model trained on our Dexonomy dataset significantly outperforms baselines.

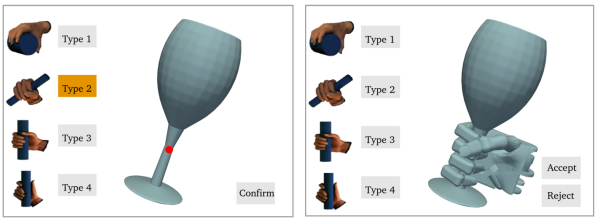

Fig. 6: **An Annotation UI based on Our Algorithm for Collecting Functional Grasp.** (Left) The user *clicks twice* to specify a contact point on the object and a grasp type. (Right) A high-quality grasp is synthesized according to the user's needs within seconds.

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 V, 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 3. 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.

E. Application: Annotation UI

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 6, 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.

F. 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 [6]. 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 IV, our dataset comprises 10.7k object assets, including 5,697 objects from DexGraspNet [17] and 5,000 new objects randomly selected from Objaverse [5]. 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 III.

Each data point includes three key poses:

- **Grasp pose**, obtained via local refinement.
- **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, to apply force through hand-object contacts.

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.

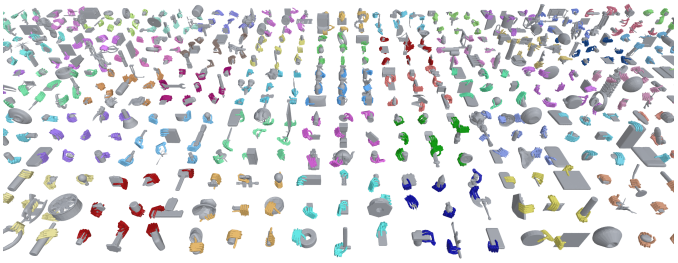

Fig. 7: **Dexonomy Dataset Visualization.** Each color corresponds to a different grasp type.

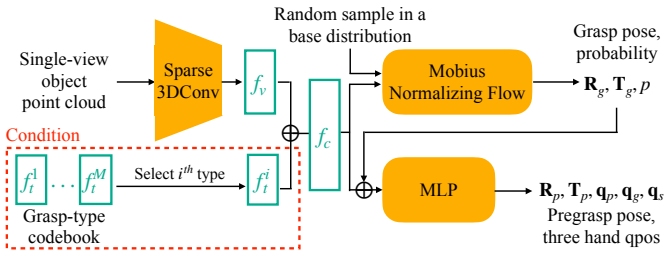

Fig. 8: **Type-Conditional Grasp Generative Model.** Without the grasp-type codebook in the red dashed box, the model becomes type-unconditional and is similar to previous works [2, 23].

G. 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 is very similar to previous works [23, 2], 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 [4]. 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 [10] 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.

H. Time Analysis

The times reported in Table 1 of the main paper represent the maximum speed for synthesis **without simulation validation**. 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 [1] algorithm as in BODex [2], 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 for testing.