DexGraspNet 2.0: Learning Generative Dexterous Grasping in Large-scale Synthetic Cluttered Scenes

Jialiang Zhang[1,2,*] Haoran Liu[1,2,*] Danshi Li[2,*] Xinqiang Yu[2,*]
Haoran Geng[1,2] Yufei Ding[1,2] Jiayi Chen[1,2] He Wang[1,2,3,†]

I. Experiment Details

We provide additional details on the experiment settings due to space constraints in the main paper. Sec. I-A delineates how we evaluate a grasp in a simulator and enumerates some of the physics parameters involved. Sec. I-B outlines the three baseline methods benchmarked in the main paper.

A. Evaluation Metric

We evaluate various grasping models by measuring their simulation success rates in the Isaac Gym simulator. For each test scene, a model is expected to take a single-view depth point cloud as input and output one grasp pose G_p. If capable of generating multiple grasps, the model must select the best proposal, as required in the main paper. Following this, the evaluator determines whether G_p constitutes a successful grasp. Specifically, a predefined rule is applied to calculate a pregrasp pose, squeeze pose, and lift pose, thereby establishing a complete action trajectory T. Subsequently, T is executed within the simulator, and success is determined by its ability to lift an object off the table without any initial intersection with the table or surrounding objects. Consistency is ensured across all experiments by maintaining the same trajectory generation rule and physics parameters. Some of the important physics parameters are listed in Tab. I.

Parameter	Value	Parameter	Value
friction coeff	0.2	object mass	0.1 kg
joint stiffness	800	joint damping	20

TABLE I: Physics Parameters

B. Baseline Details

We outline the three baselines compared in the main paper and detail how we adapted two of them from their original setting of single-object grasping to our cluttered scenarios.

HGC-Net [16]. HGC-Net is a two-stage method for grasping in cluttered scenes. Initially, a segmentation model divides the scene point cloud into graspable points and ungraspable points. Following this, a deterministic model predicts a grasp pose near each graspable point. Given that this method already focuses on cluttered scenes, minimal modifications were required. The only change made was switching their end effector from the HIT-DLR II hand to the LEAP hand.

ISAGrasp [4]. ISAGrasp is a regressive method designed for grasping single objects. It employs a PointNet++ encoder [22] to encode the object point cloud into a global feature vector. Subsequently, an MLP is utilized to predict the wrist translation, wrist quaternions, and joint angles. We extensively modified this method to adapt it for cluttered scenes: (1) We replaced their PointNet++ encoder with a ResUNet14 encoder-decoder and incorporated a seed point proposal module based on point-wise graspness prediction, similar to our method. (2) During inference, this modified model predicts the grasp parameters from the local feature vector of the proposed seed point, instead of the global feature vector obtained from their original point cloud encoder. (3) During training, each grasp label is associated with its corresponding point rather than its target object. We designate the modified model as ISAGrasp†. It is worth noting that this adaptation already rectifies a major suboptimal aspect of their original baseline by integrating one of our key designs: replacing global conditioning with local conditioning. Consequently, the adapted method differs from our model solely in the use of a regressive model to predict the wrist pose, whereas we employ a conditional generative model.

GraspTTA [14]. GraspTTA utilizes a CVAE for grasping single objects. It leverages PointNet [21] to encode the object point cloud into a global feature vector, which serves as conditioning for the CVAE to predict the distribution of the wrist translation, wrist axis angles, and joint angles. We adapt it for cluttered scenes using the same approach as ISAGrasp†, and denote the adapted version as GraspTTA†. Furthermore, we discard the test-time optimization of the original method because it relies on the full point cloud, which is an invalid assumption in our task settings.

II. Additional Experiments

A. Ablation Studies on Model Design and Training Data

Our method aims to model the complex distribution of dexterous grasping while achieving higher generalization efficiency by leveraging a generative model that conditions on local features. To better analyze the effectiveness of each module, we construct three ablated versions of our method: (1) ablate **local feature**, during training, each grasp label corresponds to the global feature vector of the scene point cloud (output by the encoder) instead of the local feature

[1]CFCS, School of CS, Peking University
[2]Galbot
[3]Beijing Academy of Artificial Intelligence
[*]Equal contribution
[†]Corresponding author: hewang@pku.edu.cn

Method	GraspNet-1Billion			ShapeNet		
	Dense	Random	Loose	Dense	Random	Loose
local feature	16.8	10.9	4.8	21.3	17.6	10.9
decomposed model	84.2	80.7	71.3	74.9	72.5	66.4
random scene	90.0	**84.1**	68.2	78.9	78.8	71.3
Ours	**90.6**	83.7	**73.2**	**81.0**	**85.4**	**74.2**

TABLE II: **Ablation studies on model design and training data.** Ablation studies are conducted on three aspects as shown in the lower half of the table. Each **Dense** scene contains 8-11 obiects, and each **Random** scene contains 1-10 objects, obtained by deleting objects from Dense scenes, and each **Loose** scene contains 1-2 objects.

vector of the grasp's corresponding point (one of the point-wise vectors output by the decoder); (2) ablate **decomposed pose modeling**, using a single conditional generative model to fit the joint distribution $p(T, R, \theta | f_s)$; (3) ablate **randomly-packed training scenes**, training solely on densely-packed scenes. As shown in Tab. II, substituting local feature conditioning with global ones yields poor performance. To illustrate, global conditioning relies on scene-level variations to generalize, which are limited in number, whereas point-wise local features derive numerous diverse geometry patches (with paired grasp labels), greatly enhancing generalization efficiency. Moreover, replacing decomposed pose modeling with the combined modeling of (R, T, θ) also leads to a perceivable decline in performance due to the incompatibility of the Euclidean space with $\mathrm{SO}(3)$ space. Finally, our dataset design that incorporates scenes with random object numbers proves to be effective in improving performance.

B. Ablate Rotation Representation

Our method employs the rotation matrix to represent wrist rotation and applies SVD [15] to orthogonalize network predictions. We compared this design against several alternatives: **Euler Angle** (representing rotation as 3D Euler angles), **Axis Angle** (rotation represented by the angle of rotation multiplied by the rotation axis), **Quaternion** (represented as a 4D quaternion), and **6D** (using the first two rows of the rotation matrix). The results in Tab. III demonstrate that our choice outperforms all other methods across the evaluated task. As discussed in [15], rotation representations in Euclidean space with fewer than five dimensions, such as Euler angles, axis-angle, and quaternions, are inherently discontinuous. Although the 6D representation circumvents this issue, it is coordinate-dependent. Introducing small noises in different directions to the rotation in a 6D representation results in changes of varying magnitudes. In contrast, our 9D representation is both continuous and coordinate-independent, thereby outperforming other rotation representations.

C. Ablate Ranking Strategy

During inference, we rank all predicted samples to identify the best one using a linear combination of the graspness scores of the seed points and the estimated log probabilities of the wrist poses. We ablate this ranking strategy by removing the graspness score, the log probability, or both. Tab. IV presents the results. Our method (**Ours**), which ranks samples based on a combination of graspness scores and log probabilities, consistently outperforms the other strategies. Ranking solely by graspness scores (**Graspness**) or log probabilities (**Log Probability**) yields moderate performances, while selecting samples randomly (**Random**) results in the lowest success rates. These findings underscore the efficacy of our proposed ranking strategy in identifying optimal grasp poses.

Interesting to note, despite the theoretical challenges in defining a probability density function $p(T, R | f_s)$ on a 6-dimensional data manifold embedded within a higher-dimensional parameter space (12D), experiments demonstrate that our estimated log probabilities consistently enhance the performance of our ranking strategy. Nevertheless, we acknowledge this theoretical inelegance and defer the solution to future studies, such as exploring the use of normalizing flows on $\mathrm{SE}(3)$ or employing manifold diffusion methods.

D. Scaling the Dataset for Grippers

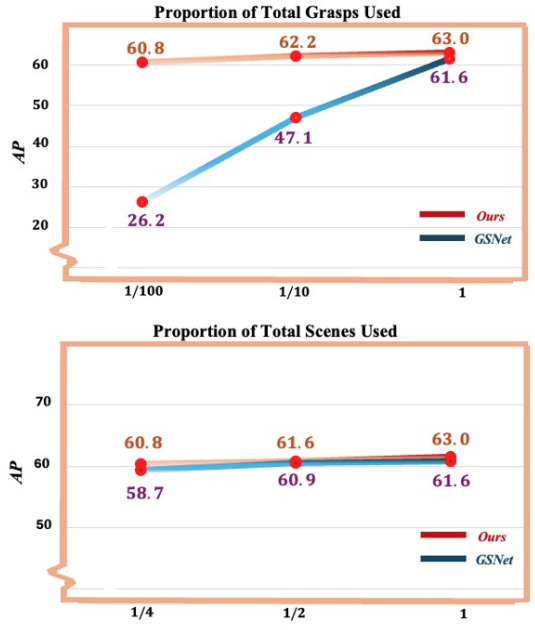

Fig. 1: **AP metric** evaluated on models trained with down-scaled dataset. **Top**: downscaling the number of grasp labels in each scene. **Bottom**: downscaling number of scenes trained on

We scale down the training data of parallel gripper by (1) reducing the number of grasps in each scene, and (2)

	Method	GraspNet-1Billion			ShapeNet		
		Dense	Random	Loose	Dense	Random	Loose
Ablation	Euler Angle	87.6	82.0	73.0	78.0	76.4	**75.2**
	Axis Angle	86.4	81.7	70.5	79.0	76.4	74.1
	Quaternion	87.9	81.5	72.0	78.6	77.0	72.9
	6D	88.2	81.5	71.9	80.2	79.0	73.0
	Ours	**90.6**	**83.7**	**73.2**	**81.0**	**85.4**	74.2

TABLE III: **Ablation studies for representations of rotation. Euler Angle** represents rotation as 3D Euler angle; **Axis Angle** represents rotation in 3D as the angle of rotation multiplies the rotation axis; **Quaternion** represents rotation as 4D quaternion; **6D** represents rotation with the first two rows of the rotation matrix. **Ours** represents the rotation as the rotation matrix.

	Method	GraspNet-1Billion			ShapeNet		
		Dense	Random	Loose	Dense	Random	Loose
Ablation	Graspness	81.8	76.6	68.0	73.7	71.3	64.4
	Log Probability	78.1	78.4	75.1	72.4	71.6	**74.6**
	Random	65.1	62.0	57.2	61.7	58.9	56.4
	Ours	**90.6**	**83.7**	**73.2**	**81.0**	**85.4**	74.2

TABLE IV: **Ablation studies for sampling strategy. Graspness** ranks samples by graspness score only; **Log Probability** ranks samples by log probability only; **Random** randomly draws from sampled poses; **Ours** ranks samples by combination of graspness scores and log probabilities.

Fraction of Grasps	Success Rate
1/100(42k)	81.3
1(4.2M)	92.4

TABLE V: **Success Rate of real-world experiment on Ours model trained over downscaled dataset**. We train Ours model with random 1/100 fraction of grasp labels and the entire grasp pose dataset, amounting 42k and 4.2M labels, respectively.

Depth Restoration	Diffuse	Trans	Hybrid
With	94.1	80.0	90.7
Without	94.1	50.0	86.4

TABLE VI: Real-world cluttered scene dexterous grasping with/without depth restoration. **Diffuse** includes only diffuse objects, **Trans** comprises only transparent or specular objects, and **Hybrid** includes scenes used in the main paper, consisting of a mixture of diffuse, transparent, and specular objects for comparison.

decreasing the number of training scenes. We evaluate the AP metric in simulation for each setting and success rate in real world.

As shown in Fig. 1, although under the full-data setting our generative model only slightly outperforms GSNet by +1.4 AP, the AP metric of GSNet drops by a significant amount of 35.4 as we downscale the number of grasps by 100, whereas our generative pipeline drops by only 2.2. This suggests that our generative pipeline is significantly more sample-efficient than GSNet. Both methods are robust to downscaling of number of training scenes at the scope of our experiment, with only slightly dropped AP.

The resulting statistics in terms of AP is much to our surprise, as being trained with 1/100 total grasp labels, namely only 42k grasp labels, our generative model seems to still reatin strong performance. In order to validate this counter-intuitive result, we carry out real-robot experiments with Ours models trained with downscaled number of grasps and report success rate in Tab. V. With 42k training labels, our generative model achieve 81.5% success rate in real-woorld cluttered scenes as shown in Fig. 2, which is affirmative to the AP statistics.

In summary, the experiments in this section give strong evidence that the distribution of valid grasp poses does exist and the amount of data required to simulate at least a valid support of such a distribution may prove to be much smaller than previously been conjectured.

E. Using Raw Depth in the Real World

In our real-world experiments, we integrated depth restoration techniques [23] to facilitate grasping transparent and specular objects amidst cluttered scenes. Here, we conduct additional experiments to demonstrate that our method do not rely on depth restoration when grasping diffuse objects. We constructed four additional cluttered scenes in the real world: two scenes (**Diffuse**, as shown in Fig. 2) consisting solely of diffuse objects and two scenes (**Trans**, as shown in Fig. 3) containing only transparent and specular objects. The original five test scenes from the main paper, which include a mixture of objects, are denoted as **Hybrid**. We then evaluated our model on all test groups both with and without the application of depth restoration techniques. The results in Tab. VI demonstrate two key findings: firstly, our model's effectiveness in real-world grasping is independent of depth restoration for **Diffuse** scenes; secondly, our model exhibits enhanced robustness to object texture, particularly transparent and specular surfaces, when depth restoration is applied.

F. Simulation Experiments on Parallel Grasping

To evaluate our method's performance in parallel grasping within cluttered scenes, we use the widely adopted Average Precision (AP) metric from [11] and conduct experiments on the 90 test scenes from GraspNet-1Billion [11], which are uniformly divided into three categories: seen, similar,

Fig. 2: Two **Diffuse** scenes in real world.

Fig. 3: Two **Trans** scenes in real world.

and novel. Due to some imperfect grasping poses in the dataset, we perform filtering and refinement on poses with scores $\geq$ 0.9, resulting in 4.2 million grasps. Additionally, with recent advancements in test-time depth restoration [23], the necessity for models to learn to grasp from noisy point clouds is diminishing. Thus, we render ground truth single-view point clouds and subsequently train and evaluate our method alongside GSNet [27]. Fig. VII demonstrates that with the same data, our method is comparable to GSNet and outperforms other methods. This indicates that our method is effective for robots with varying morphologies. Additionally, it shows that as a generative model, we can significantly enhance model performance by improving data quality, and GSNet can also benefit from some of our data refinement methods.

G. Discussion on Dexterous Hands vs Parallel Grippers

While grasping systems utilizing parallel grippers have already achieved impressive robustness in the real world [27, 12], we advocate that dexterous hands can further enhance performance. In addition to the 5 test scenes (**Normal**, as shown in Fig. 4) demonstrated in the main paper, we also construct an additional scene (**Large**) consisting of 4 large objects, as shown in main paper. Real-world experiment results in Tab. VIII indicate that the dexterous hand can grasp each object in this scene, whereas the parallel gripper cannot grasp any object. This is because the dexterous hand possesses strong envelopment capabilities, allowing it to grasp larger objects effectively.

III. Benckmark Specifications

This Section presents further details about the DexGraspNet 2.0 benchmark proposed by this work. Sec. III-A provides statistics of the DexGraspNet 2.0 benchmark, including both the **Training Set** that contains ground truth grasp pose annotations and the **Test Set** with no ground truth provided.

Method	**AP Seen**	**AP Similar**	**AP Novel**
GG-CNN [20]	15.5/16.9	13.3/15.1	5.5/7.4
Chu *et al.* [6]	16.0/17.6	15.4/17.4	7.6/8.0
GPD [26]	22.9/24.4	21.3/23.2	8.2/9.6
PointGPD [17]	26.0/27.6	22.7/24.4	9.2/10.7
GraspNet1B[11]	27.6/29.9	26.1/27.8	10.6/11.5
GSNet [27]	67.1/63.5	54.8/49.2	24.3/19.8
GSNet [27]*	**68.7**	59.8	24.6
Ours*	56.0	53.2	23.2
Ours # *	66.6/61.5	**61.7**/53.3	**27.4**/23.2
GSNet (render)*	**78.4**	69.7	36.7
Ours (render) # *	77.4	**72.5**	**39.1**

TABLE VII: **Experiment results on GraspNet-1Billion.** * means using grasps that scores over $\geq$ 0.9. # means using refined poses, and render means using rendered depth images. Statistics in the table follows a **RealSense/Kinect** format, where results with a single number uses Realsense setting.

End Effector	Normal	Large
Parallel Gripper	92.4	0.0
Dexterous Hand	81.5	100.0

TABLE VIII: Comparison of real-world grasping performance using a parallel gripper or a dexterous hand across different scene types. The five **Normal** scenes consist of typical cluttered environments, while the **Large** scene includes 4 large objects.

Sec. III-B identifies the objects used to generate our benckmark. Sec. III-C presents the pipeline used to generate training scenes with selected objects. Sec. III-D elaborates the protocol of generating test scenes and how we divide them into different splits.

A. Benchmark Statistics

Tab.IX illustrates the overall statistics. The entire benchmark encompasses two components: a **Training Set** used to train our models and a **Test Set** to evaluate dexterous grasping pose generation models on. Note that ground truth grasp pose annotations are only provided for training set. In total, the benchmark contains 8270 scenes, 1319 objects and 426.6M grasp pose annotations.

Training Set contains 7600 scenes and 60 objects in total. all training objects are from the GraspNet-1Billion [11] dataset

Test Set contains 670 scenes and 1319 objects in total. the 88 objects from the GraspNet-1Billion [11] dataset are used to compose 450 of the test scenes, and 1231 objects picked from ShapeNet [1] are used to compose the remaining 220 test scenes

B. Object Selection

The 60 objects in Training Set are those appeared in GraspNet-1Billion [11] scenes 0000-0099. The Test Set contains 1319 objects, 88 of them are all the objects in GraspNet-1Billion [11], and the remaining 1231 objects are picked from ShapeNetSem [1].

Splits	number of objects	number of scenes
Training	60(GraspNet1B)	100(seminal)+7500(augmented)
Test	88(GraspNet1B) + 1231(ShapeNet)	670
Total	88(GraspNet1B) + 1231(ShapeNet)	8270

TABLE IX: Statistics of the DexGraspNet 2.0 Benchmark

Fig. 4: Five **Normal** test scenes for gripper in the main paper.

C. Training Scenes Specification

In the 7600 training scenes, 100 are called **seminal scenes**, which corresponds to the Scenes 0000-0099 in the GraspNet-1Billion [11] dataset composed and rendered using their official meshes and annotations. We augment each seminal scenes 75 times by randomly deleting objects in the scene. In each augmented scene, the number of objects deleted is uniformly sampled from [1,k-1], where k is the number of objects in the original scene. In total, we generate 7500 augmented training scenes with 100 seminal scenes, totalling 7600 scenes in the entire training set.

D. Test Set Scenes Specification

As shown in Tab. 1 of the main paper, the Test Set is divided into 6 splits. In the following, we specify each of these splits.

GraspNet-1Billion Dense composes of 90 scenes that correspond to the Scenes 0100-0189 in the GraspNet-1Billion [11] dataset. Each scene contains 8-11 objects.

GraspNet-1Billion Random composes of 180 scenes. This split is generated by augmenting each GraspNet-1Billion Dense split scenes twice with the process as described in Sec.III-C

GraspNet-1Billion Loose composes of 180 scenes by augmenting each GraspNet-1Billion Dense split scenes twice with the process as described in Sec.III-C, with only 1-2 random objects remaining in the scene.

The three ShapeNet splits are generated by dropping objects on a 30cm×50cm physics paratop. In specific, we follow the scene generation process of DREDS [9] with the material randomization function disabled. We run the scene generation process in PyBullet [7] and filter physically sphysics para ones in IsaacGym [18]. The Dense/Random/Loose splits are divided according to the number of objects appearing in each scenes.

ShapeNet Dense composes of 100 scenes, each containing 8-11 objects

ShapeNet Random composes of 90 scenes, each containing 5-9 objects

ShapeNet Loose composes of 30 scenes, each containing 1-2 objects

IV. Grasp Label Generation

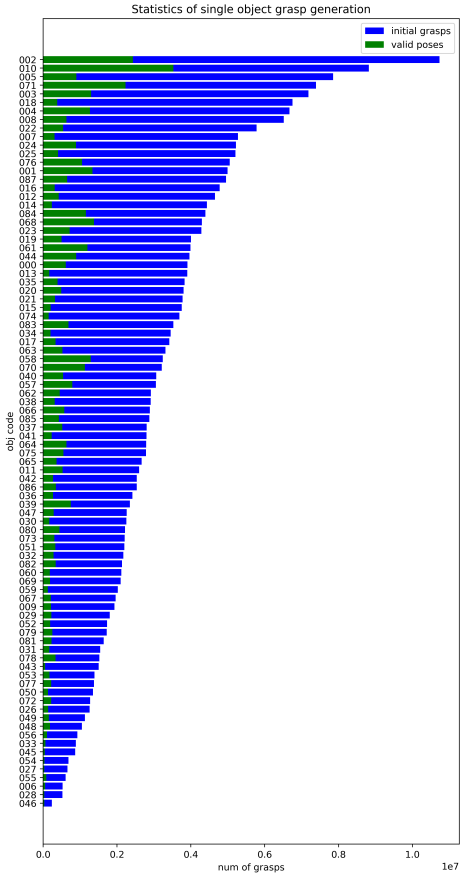

Fig. 5: **Number of per-object initial grasp poses**. The proportion corresponding to valid grasps after optimization are colored green.

This section elaborates our pipeline for generating dexterous grasping poses on single objects. First, we define initial hand poses by retargeting GraspNet-1Billion [11] annotations to dexterous hand. Then we run physics-based optimization to generate stable grasps. To maximally diversify the produced data, we adopt two different methods, [2] which targets Grasp Wrench Space (GWS) optimality, and [28] which targets force-closure, as optimization algorithms, each generating half of the dataset. Lastly, we filter stable and collision-free grasps via

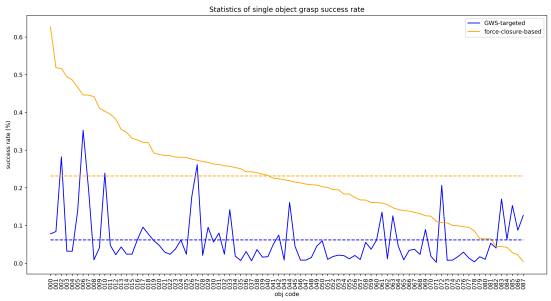

Fig. 6: **Valid Rate** of single object grasp synthesis in sorted order. **Yellow** and **Blue** curves present per-object valid rates for our force-closure based optimization method (Sec.IV-B2) and GWS-based optimization method (Sec.IV-B1), respectively. Averaged success rates are drawn in dotted line, with values 24.19% and 7.91% respectively.

simulation in the IsaacGym [18] simulator. As shown in Fig. 5, in total we generate 44.9M stable grasp poses for 88 objects from 280M initial poses. Even in the face of our very strict friction coefficient μ=*0.2*, our method still maintains overall success rate of 16.07%. In the following subsections we detail each of these components.

A. Hand Pose Initialization

As discovered in [28], the success rate of dexterous grasp generation is very sensitive to initial hand pose. Moreover, we aim to cover valid grasp modes for each object as comprehensively as possible. Therefore, we initialize dexterous hand poses by retargeting the exhaustive GraspNet-1Billion [11] gripper anntations.

In specific, we filter points where stable gripper grasp poses are annotated in [11] as grasp points. As shown in Fig. 7, for each grasp point, we align the +y axis (pointing forward out of the palm) of dexterous hand with the +x axis of gripper pose annotation, retreat the center of palm a fixed distance from grasp point in the approacting direction, initialize hand joint qpos with a set of predifined values and exhaustively apply transformations corresponding to 256 approaching directions, 4 depths and 12 in-plane angles as defined in [11].

B. Grasp Pose Optimization

1) GWS-based optimization (adapted version of [2]): We reimplement [2] on the CuRobo [25] framework for better computation parallelism. We set the target Task Wrench Space (TWS) as a unit sphere in 6D wrench space such that the task objective is identical to forming a force-closure grasp, and run 600 iterations with naive gradient descent.

2) force-closure-based optimization (adapted version of [28]): We adopt [28] with modification in its definition of force-closure energy, and reimplement the modified algorithm on the CuRobo [25] framework as well.

We observe that the force-closure energy used in [28] assumes unit contact force is applied to each contact point, whereas human naturally adjust contact forces applied to different contact points in order to maintain a firm grasp. The above assumption limits the objective of optimization in [28] onto a submanifold of the space of all valid grasp poses, hurting the quality and diversity of generated data. Following the notations in [28], we relax the unit-contact force assumption by reformulating the force closure energy as the following bilevel form:

- At each timestep, given the current hand pose, we solve the optimal contact forces applied to current contact points such that the total wrench imposed on the object is minimized. We formulate this intuition into the following linear program:

$$\begin{aligned} P_t = &\min_{\lambda_t} \|G(\lambda_t \odot c)\|_2 \\ s.t. &\max_i(\lambda_t)_i = 1 \\ &(\lambda_t)_i \geq 0, i = 1, 2, ..., n \end{aligned}$$

Where P_t has the physical meaning as the total wrench applied to the object when the combination of contact force magnitude, λ_t, is applied to the contact points. $\odot$ means element-wise product. Note this linear program admits closed-form solution therefore imposes neglectable computation burden.

- Across timesteps, we optimize the differentiable force-closure metric in awareness of the plausibility of the current hand pose:

$$E_{FC} = \begin{cases} \|G(\lambda_t \odot c)\|_2, & \text{if } P_t < \tau_{FC}, \\ & \min_i(\lambda_t)_i \geq \tau_\lambda, \\ & \text{and } B = 1 \\ \|Gc\|_2, & \text{otherwise} \end{cases} \quad (1)$$

Where τ_{FC}, τ_λ are predefined thresholds, and B is a binary random variable with $P(B = 1) = 0.9$.

If the current hand pose is already capable of forming a force-closure grasp on the object, mathematically defined as $P_t < \tau_{FC}$ (total wrench acceptably small) and $\min_i(\lambda_t)_i \geq \tau_\lambda$ (a minimum contact force is applied to each contact point), then we decide the current pose is good enough in terms of force-closure property. In this case, we scale the force closure energy to prevent overoptimization. In effect, the force closure energy now works as a regularization term. Otherwise, if the current hand pose is not stable enough, we keep searching for more stable poses by optimizing the force closure metric with original energy term. In addition, even for the former case, we stochastically use the original energy term with probability 0.1 to encourage forming more robust grasp poses.

Note in the above formulation, the global minimum set of hand poses for E_{FC} are the poses for which there exists a non-trivial contact force combination such that the total wrench executed to the object is zero. This global minimum set exactly corresponds to the original definition of force closure in [8].

C. Filtering Stable and Collision-Free grasps

We perform grasp filtering in the IsaacGym simulator. First, we check for each grasp pose if the penetration between hand

mesh and object mesh is below 2 mm. For all collision-free grasps, we execute the grasp with a predefined heuristic and simulate for 60 timesteps at 60Hz. The grasp pose is validated as stable if it can deny gravity in all 6 axis-aligned directions. The friction coefficient μ for both hand and objects are set to 0.2, making the filtering process very strict.

Fig. 5 shows the **Valid Rate** for each object, which is defined as the portion of generated grasps that are both collision-free and stable. The overall success rate is 16.07%, as we generate in total 44.9M valid grasp poses out of 280M grasp pose initializations. The method-specific valid rate for [2] and [28] are 7.91% and 24.19% respectively.

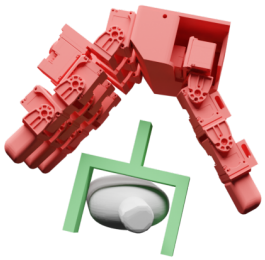

Fig. 7: **Initial dexterous hand pose** superimposed with gripper grasp label at the same grasp point. We retarget gripper annotation in GraspNet-1Billion [11] to initial 6D wrist pose of dexterous hand, and use a predefined set of joint qpos for initialization.

V. Implementation Details for Dexterous Hands

In this section, we elaborate on the data organization (Sec. V-A) and model architecture (Sec. V-B) of our method for dexterous grasping.

A. Data

Data Reblancing In each training scene, the numbers of grasp labels on graspable objects may be uneven. Randomly sampling grasp labels uniformly across all valid ones in each scene could slow down the learning of grasping objects that have fewer labels. To address this, we implement a two-stage sampling approach to rebalance the training process: first, we randomly sample a graspable object, and then we randomly sample one of its labels.

Data Augmentation. We implement data augmentation by rotating the scene point cloud and grasp labels around the camera axis with a random angle uniformly sampled from the interval $[0, 2\pi)$. No further augmentations are needed.

Ground-truth Graspness Definition. For each training scene, we define a graspness score for the surface points of each object to represent its graspability. This score is determined by identifying a seed point and then assigning graspness to the nearby points. For an object o in this scene, we denote all valid grasp labels that target o as $G_o = \{g_o^i\}$, and the surface points of o as $P_o = \{p_o^j\}$. We then define a grasp cone with c being the apex, vector cm being the axis and an aperture of $60°$, as shown in Fig. 8. Subsequently, we compute the projected distance of vector cp_o^j along cm, denoted as d, and the spanning angle θ between cp_o^j and cm. Using these quantities, the value of $f(g_o^i, p_o^j)$ is defined in Eq. 2. Numerically, this function is designed to attenuate exponentially with response to θ and d, halving at $10°$ or 1.5 cm. Then the seed point is defined as the point with the largest f as shown in Eq. 3.

Finally, the seed point assigns graspness to nearby points with exponential decay and the graspness score of p_o^j is computed as the logarithm of the sum of all contributed graspness, as in Eq. 5. Empirically, this score reflects the number of valid grasp labels near p_o^j.

From another perspective, this correspondence implicitly defines a grasp distribution conditioned on a point within a scene. Although articulating this distribution in precise mathematical terms is difficult, we contend that it objectively exists. This distribution represents the target distribution that the grasp generation module approximates.

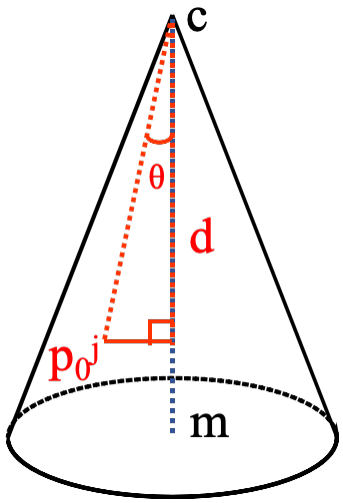

Fig. 8: Grasp cone for the graspness definition.

$$f(g_o^i, p_o^j) = \begin{cases} 0 & p_o^j \notin \text{this cone} \\ \exp\left(-\frac{\ln 2}{10}\frac{180}{\pi}\theta - \frac{\ln 2}{0.015}d\right) & p_o^j \in \text{this cone} \end{cases} \tag{2}$$

$$\text{seed\_point}(g_o^i) = \arg\max_{p_o^j \in P_o} f(g_o^i, p_o^j) \tag{3}$$

$$h(g_o^i, p_o^j) = 10^{-150||\text{seed\_point}(g_o^i) - p_o^j||_2} \tag{4}$$

$$\text{graspness\_score}(p_o^j) = \ln\left(0.001 + \sum_{g_o^i \in G_o} h(g_o^i, p_o^j)\right) \tag{5}$$

B. Model

Network Structure. In the following paragraph, we elaborate on the network structures of our feature extractor, denois-

ing model, graspness MLP, and joint MLP. First, our feature extractor employs the ResUNet14 architecture implemented with MinkowskiEngine [5] to derive point-wise feature vectors $f_p \in \mathbb{R}^{512}$ from a scene point cloud P, which is quantized into sparse voxels. This network resembles the one utilized in GSNet [27]. Second, our denoising model $v_\Theta(\hat{g}_E^t, f_s, t)$ is implemented as an MLP with layer sizes (524, 512, 256, 12) and Mish activations [19]. This model embeds t into $\mathbb{R}^{512}$ using sinusoidal position embedding, adds this embedding with f_s, concatenates the resulting sum with $\hat{g}_E^t$, and feeds this concatenation into the MLP to predict the velocity. Third, our graspness MLP comprises a single-layer linear transformation, which maps f_p to three values. The first two are interpreted as binary classification logits indicating whether this point is an object point, while the thid value represents the predicted graspness score GP_p. Fourth, our joint MLP is a 6-layered MLP with ReLU activations and residual block designs following [10].

Detailed Diffusion Dynamics. The forward and backward processes of the diffusion each consist of T_{train} and $T_{\text{inference}}$ time steps, respectively, evenly distributed within the interval $[0, 1]$. Additionally, the number of time steps of the backward process is required to be a divisor of that of the forward process. We denote the interval between two neighboring time steps of the backward process as $\mathrm{d}t = 1/T_{\text{inference}}$. The DDPM [13] scheduler is employed to schedule the forward process variances β_t for each time step $t = i/T_{\text{train}}, i = 1, 2, \ldots, T_{\text{train}}$:

$$\beta_t = \beta_{\text{min}} + \frac{i-1}{T_{\text{train}}-1}(\beta_{\text{max}} - \beta_{\text{min}}) \tag{6}$$

where $\beta_{\min}, \beta_{\max}$ are hyper-parameters. Then we define $\alpha_t = 1 - \beta_t$ and its cumulative product as $\overline{\alpha}_t = \prod_{j=1}^{i} \alpha_{j/T_{\text{train}}}$. At each training step, $\overline{\alpha}_t$ is utilized to determine the magnitude of noise to be added to the sample, as detailed in the main paper. At each inference step, we denoise a noisy sample $\hat{g}_E^t$ into a less noisy sample $\hat{g}_E^{t-\mathrm{d}t}$ by solving the following ODE with t from 1 to 0:

$$\hat{g}_E^t - \hat{g}_E^{t-\mathrm{d}t} = \mathrm{d}\hat{g}_E^t = \frac{T_{\text{train}}\beta_t\sqrt{\overline{\alpha}_t}}{2\sqrt{1-\overline{\alpha}_t}} v_\Theta(\hat{g}_E^t, f_s, t)\mathrm{d}t \tag{7}$$

Moreover, [3, 24] introduce a PDE to estimate the probability $p(g_E|f_s)$:

$$\frac{\partial \log p(\hat{g}_E^t|f_s)}{\partial t} = -\operatorname{Tr}\left(\frac{\partial \bar{v}_t}{\partial \hat{g}_E^t}\right) \tag{8}$$
$$\text{where } \bar{v}_t = \frac{T_{\text{train}}\beta_t\sqrt{\overline{\alpha}_t}}{2\sqrt{1-\overline{\alpha}_t}} v_\Theta(\hat{g}_E^t, f_s, t)$$

Based on the above equation, we can approximate a sample's probability $p(g_E|f_s)$ with numerical integration during the backward process. We rank each output g of the grasp generation module using a linear combination of the estimated probability $p(g_E|f_s)$ of the wrist pose g_E and the predicted graspness GS_s of the seed point s:

$$\text{rank}(g) = p(g_E|f_s) + \eta GS_s \tag{9}$$

Inference Speed and Memory Cost. Our model efficiently processes a scene point cloud comprising 40,000 points, generating 128 grasp poses and ranking them all within **0.5 seconds**. The maximum memory usage during this inference is approximately **3 GB**. These evaluations were conducted on an NVIDIA 4090 graphics card.

VI. Implementation Details for Parallel Grippers

A. Data Filtering and Refinement

As our generative model considers all grasping poses from the dataset as successful, and since the original GraspNet-1Billion dataset [11] includes some imperfect poses, we introduce a data filtering and refinement process before training. We retain only the grasping poses with a score of ≥ 0.9 to ensure that all can successfully grasp the object with a friction coefficient of 0.2. To simplify motion planning, we assume that all grasps can be achieved by moving along the approaching vector and filtering out poses that would result in collisions during this movement. We also fix the depth to 4 cm and adjust the translation accordingly.

To handle poses that collide with the object and the table, we calculate the upper (u) and lower (l) bounds of the distance between the fingers along the original approaching vector. If the distance between any finger and the object is $u - l < 1.5$ cm, we discard the pose. We then uniformly sample new finger positions from the adjusted lower bound $l' = l + s$ and the adjusted upper bound $u' = l' + \min(0.01, (u - l - 0.01) - 2s)$, where $s = \min(0.01, \frac{u-l-0.01}{2})$. This ensures the fingers maintain a safe distance from the object without being too far. Finally, we calculate the intersection point of the object mesh and the new approaching vector, setting it as the seed point. Poses without a valid seed point are filtered out.

B. Graspness Definition for Gripper

For parallel grippers, after we define the intersection point as the seed point, we assign the graspness to nearby points with Eq. 4 and compute the total graspness for each point with Eq. 5, same as the dexterous hand experiments.

C. Sampling Poses from Prediction

Given the variability in graspness among different objects, we developed a new sampling strategy to maintain diversity and select high-quality grasping poses. First, we identify all seed points within the top 1% for graspability. For each of these seed points, we collect all points within a 2 cm radius. We then select the top 10% of these points based on graspability as new seed points and calculate grasping poses with them.

D. Real-World Experiments

As a lot of the objects in the LEAP Hand's experiment are too large for our parallel gripper, we use different scenes in those two experiments as shown in Fig. 4.

Hyper-parameter	Value	Hyper-parameter	Value	Hyper-parameter	Value
Scene in each Batch	8	Grasp in each Scene	64	Init LR	1e-3
LR Scheduler	Cosine	Iter	50000	Point Num	40000
Voxel side length	0.005 m	k_{trans}	25	T_{train}	1000
$T_{\text{inference}}$	200	β_{min}	0.0001	β_{min}	0.02
λ_o	1	λ_g	1	λ_d	10
λ_θ	1	η	10		

TABLE X: Hyper-parameter Setup

VII. Additional Visualizations

In Fig. 9 we present more scenes with the predictions of our network. All point clouds are colored with heatmap of model predicted graspness, with lighter color meaning higher graspness. Each scene is also dubbed with the predicted grasping pose corresponding to highest rank.

In Fig. 10 we show some renderings of test scenes composed of objects from ShapeNet [1].

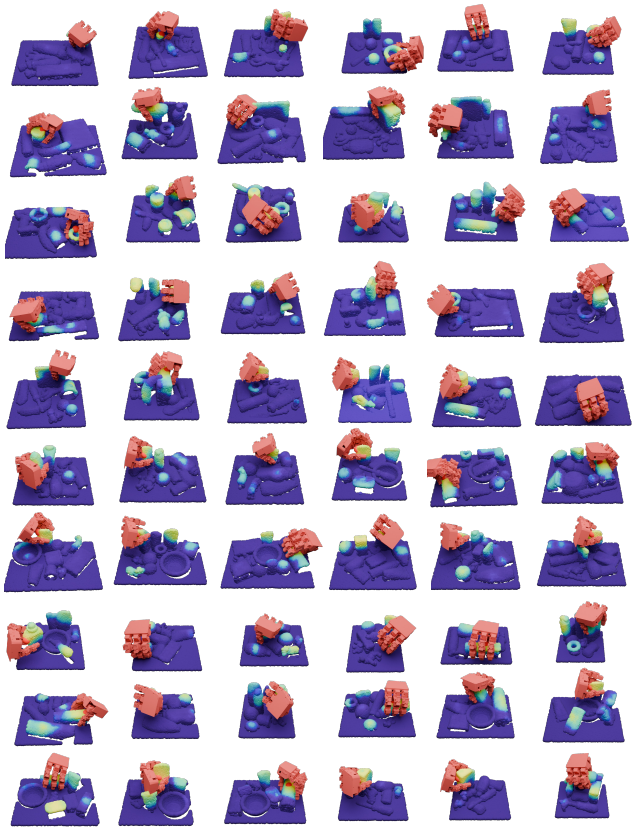

Fig. 9: **Gallery visualization** of test scenes in our benchmark, corresponding to scenes 0100-0159 in GraspNet-1Billion [11]. All point clouds are colored with heatmap of model predicted graspness, with lighter color meaning higher graspness. Each scene is also dubbed with the predicted grasping pose corresponding to highest rank.

Fig. 10: Test scenes composed of objects from ShapeNet [1].

References

[1] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012, 2015.

[2] 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.

[3] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. Advances in neural information processing systems, 31, 2018.

[4] Zoey Qiuyu Chen, Karl Van Wyk, Yu-Wei Chao, Wei Yang, Arsalan Mousavian, Abhishek Gupta, and Dieter Fox. Learning robust real-world dexterous grasping policies via implicit shape augmentation. arXiv preprint arXiv:2210.13638, 2022.

[5] 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.

[6] Fu-Jen Chu, Ruinian Xu, and Patricio A Vela. Real-world multiobject, multigrasp detection. IEEE Robotics and Automation Letters, 3(4):3355–3362, 2018.

[7] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021.

[8] Hongkai Dai, Anirudha Majumdar, and Russ Tedrake. Synthesis and optimization of force closure grasps via sequential semidefinite programming. Robotics Research: Volume 1, pages 285–305, 2018.

[9] Qiyu Dai, Jiyao Zhang, Qiwei Li, Tianhao Wu, Hao Dong, Ziyuan Liu, Ping Tan, and He Wang. Domain randomization-enhanced depth simulation and restoration for perceiving and grasping specular and transparent objects. In European Conference on Computer Vision, pages 374–391. Springer, 2022.

[10] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. nflows: normalizing flows in PyTorch, November 2020. URL https://doi.org/10.5281/zenodo.4296287.

[11] 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.

[12] Hao-Shu Fang, Chenxi Wang, Hongjie Fang, Minghao Gou, Jirong Liu, Hengxu Yan, Wenhai Liu, Yichen Xie, and Cewu Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. IEEE Transactions on Robotics, 2023.

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. Advances in neural information processing systems, 33:6840–6851, 2020.

[14] 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.

[15] Jake Levinson, Carlos Esteves, Kefan Chen, Noah Snavely, Angjoo Kanazawa, Afshin Rostamizadeh, and Ameesh Makadia. An analysis of svd for deep rotation estimation. arXiv preprint arXiv:2006.14616, 2020.

[16] Yiming Li, Wei Wei, Daheng Li, Peng Wang, Wanyi Li, and Jun Zhong. Hgc-net: Deep anthropomorphic hand grasping in clutter. In 2022 International Conference on Robotics and Automation (ICRA), pages 714–720. IEEE, 2022.

[17] Hongzhuo Liang, Xiaojian Ma, Shuang Li, Michael Görner, Song Tang, Bin Fang, Fuchun Sun, and Jianwei Zhang. Pointnetgpd: Detecting grasp configurations from point sets. In 2019 International Conference on Robotics and Automation (ICRA), pages 3629–3635. IEEE, 2019.

[18] 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.

[19] Diganta Misra. Mish: A self regularized non-monotonic activation function. arXiv preprint arXiv:1908.08681, 2019.

[20] Douglas Morrison, Peter Corke, and Jürgen Leitner. Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. arXiv preprint arXiv:1804.05172, 2018.

[21] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 652–660, 2017.

[22] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems, 30, 2017.

[23] Jun Shi, Yixiang Jin, Dingzhe Li, Haoyu Niu, Zhezhu Jin, He Wang, et al. Asgrasp: Generalizable transparent object reconstruction and grasping from rgb-d active stereo camera. arXiv preprint arXiv:2405.05648, 2024.

[24] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456, 2020.

[25] 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 minimum-jerk robot motion generation. arXiv preprint arXiv:2310.17274, 2023.

[26] Andreas Ten Pas, Marcus Gualtieri, Kate Saenko, and Robert Platt. Grasp pose detection in point clouds. The International Journal of Robotics Research, 36(13-14): 1455–1473, 2017.

[27] Chenxi Wang, Hao-Shu Fang, Minghao Gou, Hongjie Fang, Jin Gao, and Cewu Lu. Graspness discovery in clutters for fast and accurate grasp detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 15964–15973, 2021.

[28] 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.