Supplementary Material

A Real-World Experiment

Real-World Deployment Details. We first calibrate the intrinsic and extrinsic parameters of three Intel RealSense depth cameras, so that all captured point clouds can be transformed into a common world coordinate frame. We then scan the cluttered workspace from multiple viewpoints and merge the partial observations into a unified scene-level point cloud. Next, SAM [1] is used to segment the target object in each camera view, and the extracted object point cloud together with its surrounding scene point cloud is downsampled to meet the input requirement of $\mathcal{T}(R, O)$ Grasp++. In deployment, we test both XHand and LEAP [2] Hand in cluttered in-box and on-table settings, with five objects in each setting and randomly placed obstacles around the target. After model inference, we use Pyroki [3] toolkit for motion planning and collision avoidance, enabling the xArm7 robot to reach the predicted grasp pose in the cluttered workspace.

Experiment Result. The real-world results on XHand and LEAP Hand are reported in Tab. 1a and Tab. 1b. $\mathcal{T}(R, O)$ Grasp++ achieves overall success rates of 85% and 82% on the two hands under cluttered in-box and on-table settings, respectively. As illustrated in Fig. 1, the robot executes stable grasps on diverse daily objects under real-world settings, where observations are often partial and noisy. These experiments demonstrate the robustness of $\mathcal{T}(R, O)$ Grasp++ in challenging real-world manipulation scenes. Video results are provided on the project website https://trograspplusplus.github.io/.

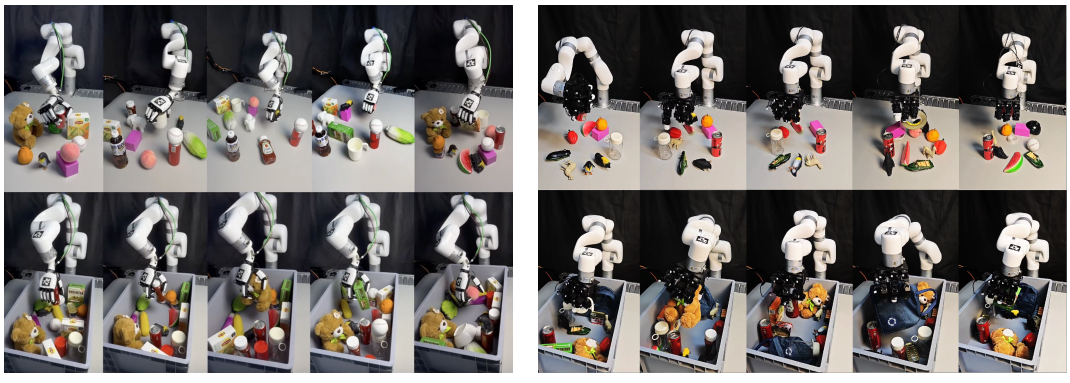
In-Box Setting				
Snack Box	**Coke Bottle**	**Peach Model**	**Ketchup**	**Bear Doll**
10/10	9/10	7/10	7/10	10/10
On-Table Setting				
Coke Bottle	**Peach Model**	**Animal Toy**	**Handled Cup**	**Snack Box**
10/10	8/10	7/10	8/10	9/10

(a) XHand

In-Box Setting				
Snack Box	**Cloth Bag**	**Handled Cup**	**Ketchup**	**Bear Doll**
9/10	10/10	7/10	7/10	9/10
On-Table Setting				
Coke Bottle	**Pepper**	**Animal Toy**	**Tape Roll**	**Orange**
8/10	7/10	7/10	10/10	9/10

(b) LEAP Hand

Table 1: **Real-world experiment results on XHand and LEAP Hand.**



(a) XHand (b) LEAP Hand

Figure 1: **Visualization of real-world experiment results on XHand and LEAP Hand.**

B Benchmark Details

B.1 Data Annotation Pipeline

As introduced in the main paper, we use a three-stage pipeline to generate dexterous grasp annotations in cluttered scenes. We first train embodiment-specific RL policies to roll out power and precision grasps on floating objects. We then sample stable tabletop object poses and transform each floating-object grasp accordingly, filtering out candidates with tabletop penetration. Finally, we construct tabletop, shelf, and box clutter by placing obstacle objects outside the 2D convex hull of the hand–object grasp configuration. In this section, we illustrate the details of each stage.

Stage 1: Floating Grasp Annotation. We first partition the complete object set into small subsets, each containing at most 128 objects. For each dexterous hand and each object subset, we train two PPO [4] policies in Isaac Gym to roll out power and precision grasps on floating target objects. The hand is initialized from sampled approaching poses around the object convex hull, and the policy controls both the 6D wrist motion and the actuated hand joints. Each episode consists of a grasping phase followed by a lifting and perturbation phase, where successful grasps are saved as floating-object grasp annotations. The two policies share the same PPO framework but use different reward designs. For the power-grasp policy, the reward encourages the hand to approach the object while regularizing the hand posture, and gives a terminal reward for successful lifting:

$$r_{\text{power}} = w_q \left\|\mathbf{q}_t - \mathbf{q}_{\text{prior}}\right\|_2 + w_d \left(\bar{d}_{t-1} - \bar{d}_t\right)\left(1 - \mathbb{I}_{\text{timeout}}\right) + w_s \mathbb{I}_{\text{succ}}, \tag{1}$$

where $\mathbf{q}_t$ is the current hand joint configuration, $\mathbf{q}_{\text{prior}}$ is the prior hand configuration, $\bar{d}_t$ is the mean distance from hand contact points to the object point cloud, and $\mathbb{I}_{\text{succ}}$ denotes final grasp success. For the precision-grasp policy, we introduce a palm-contact penalty that discourages excessive palm involvement and promotes fingertip contacts:

$$r_{\text{precision}} = r_{\text{power}} - w_p \,\text{clip}\left(\frac{\tau_p - \bar{d}_{\text{palm}}}{\tau_p}, 0, 1\right), \tag{2}$$

where $\bar{d}_{\text{palm}}$ is the mean distance from palm contact points to the object point cloud, and τ_p is the palm-contact threshold. Thus, the power-grasp policy favors stable enclosing grasps, while the precision-grasp policy biases toward fingertip contacts. To further improve annotation diversity, we maintain a memory bank of successful grasp features and add a diversity reward r_{div} that favors grasps different from previous successful samples.

Stage 2: Tabletop Grasp Filtering. After obtaining floating-object grasps, we generate stable object poses on a tabletop in simulation. For each target object, we randomly sample 128 initial object poses above the table and let the object fall under gravity until it reaches a stable resting pose. Each successful floating-object grasp is then transformed according to the resulting object pose, so that the relative transformation between the hand and the target object is preserved. We further filter out candidates that cause penetration between tabletop and the dexterous hand. After the second stage, the remaining grasp annotations provide physically plausible and collision-free tabletop grasps for the target object.

Stage 3: Cluttered Scene Construction. Finally, we construct cluttered manipulation scenes by adding obstacle objects around each transformed hand-object configuration. For each transformed grasp, we project the target-object point cloud and hand point cloud onto the tabletop plane, and compute their 2D convex hull as a forbidden placement region that approximates the occupied grasp area. Additional obstacle objects are sampled outside this region to introduce surrounding clutter without collision. We consider three scene types: tabletop, shelf, and box. The tabletop scene uses a square support plane with side length in $[0.70, 1.00]$ m; the shelf scene has length $[0.80, 1.30]$ m, depth $[0.20, 0.40]$ m, and height $[0.35, 0.50]$ m; and the box scene has side length $[0.70, 1.00]$ m and wall height $[0.25, 0.35]$ m. For each scene, we randomly place 15 to 25 obstacle objects, simulate them until they settle, and discard invalid obstacles that collide with the forbidden region

or fall outside the environment boundary. The final annotation stores the environment point cloud, target-object pose, obstacle poses, and hand joint values.

B.2 Hierarchical Difficulty Score

In the proposed $\mathcal{T}(R, O)$ Grasp++ benchmark, we quantify the execution difficulty of each grasp annotation in the cluttered scene from two complementary perspectives: object-level and environment-level difficulty. Object-level difficulty measures the intrinsic grasping challenge posed by the target object, which is determined by its geometry, shape and size. We quantify it as the number of RL optimization steps required for a policy to reach a 50% success rate on the target object. We define

$$D_{\text{obj}} = \text{Norm}\left(\min\left(N^{0.5}_{\text{power}}, N^{0.5}_{\text{precision}}\right)\right) \in (0, 1], \tag{3}$$

where $N^{0.5}_{\text{power}}$ and $N^{0.5}_{\text{precision}}$ denote the numbers of optimization steps required by the power-grasp and precision-grasp policies, respectively, to achieve a 50% success rate on the target object, and $\text{Norm}(\cdot)$ is normalization operation. If neither policy reaches a 50% success rate, we set $D_{\text{obj}} = 1$.

For environment-level difficulty, we quantify scene clutter using obstacle proximity and spatial enclosure around the target object. Let $\mathbf{P}_{\text{obj}}$ and $\mathbf{P}_{\text{env}}$ denote the point clouds of the target object and the surrounding environment, respectively. We consider environment points within a radius $r_{\max}$ from the target object center, where $r_{\max}$ defines the local neighborhood used for clutter estimation. We first compute the minimum object-environment distance and define the proximity term as a normalized exponential distance score:

$$d_{\min} = \min_{\mathbf{p}\in\mathbf{P}_{\text{obj}},\, \mathbf{q}\in\mathbf{P}_{\text{env}}} \|\mathbf{p} - \mathbf{q}\|_2, \quad D^{\text{prox}}_{\text{env}} = \max\left(0, \frac{\exp(-d_{\min}/\sigma) - \exp(-r_{\max}/\sigma)}{1 - \exp(-r_{\max}/\sigma)}\right). \tag{4}$$

To measure spatial enclosure, we use the target object center as the origin and discretize the local surrounding space into azimuth-elevation bins. The enclosure term $D^{\text{enc}}_{\text{env}}$ is defined as the fraction of bins occupied by nearby environment points. The environment-level difficulty is then defined as:

$$D_{\text{env}} = \text{Norm}\left(\alpha D^{\text{prox}}_{\text{env}} + \beta D^{\text{enc}}_{\text{env}}\right), \quad \alpha + \beta = 1. \tag{5}$$

Combining object-level difficulty D_{obj} and environment-level difficulty D_{env}, we assign each cluttered-scene grasp an overall difficulty score $D_{\text{avg}} = (D_{\text{obj}} + D_{\text{env}})/2$. The resulting score distributions over all grasp samples are shown in Fig. 2, demonstrating that our benchmark covers a broad range of grasping difficulty at both the object and environment levels. These hierarchical difficulty annotations enable structured evaluation across various clutter levels and object grasping difficulties, helping reveal the performance boundaries of different methods.

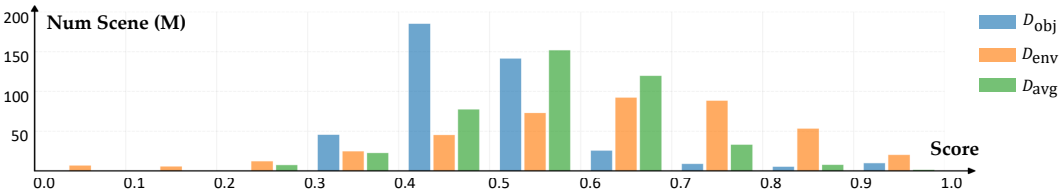


Figure 2: **Score distribution of our proposed benchmark.** Each grasp in a cluttered scene is annotated with a difficulty score derived from object geometry, obstacle proximity, and spatial enclosure.

C Implementation Details

Diffusion Model Architecture. Our diffusion model operates on $\mathcal{T}(R, O)$ Graph++ representation, which is composed of object nodes, environment nodes and robot-link nodes. Object and environment geometries are first encoded into local geometry tokens by pretrained VQ-VAE [5] encoders. Across all model scales, we use $P = 25$ object patches, $E = 25$ environment patches, and at most $L = 25$ robot-link nodes. The denoiser takes noisy robot-link poses and graph edge features as input, and predicts link-wise pose updates through stacked graph attention layers. Tab. 2

summarizes the detailed network configurations for different model scales. In our experiments, we mainly use the 0.2B model as the default configuration.

Model Params.	Layers Num.	Attn. Heads Num.	Node Dim.	Edge Dim.	Output Layer Dim.
10M	4	4	256	256	[128, 64]
50M	8	6	384	384	[256, 128]
0.2B	**8**	**12**	**768**	**768**	**[512, 256, 128]**
1B	12	24	1536	1536	[512, 256, 128]

Table 2: **Architecture configurations of diffusion models with different parameter scales.**

Training Configurations. $\mathcal{T}(R, O)$ Grasp++ takes three inputs: the target-object point cloud $\mathbf{P}_{\text{obj}} \in \mathbb{R}^{512\times3}$, the surrounding-environment point cloud $\mathbf{P}_{\text{env}} \in \mathbb{R}^{2048\times3}$, and the URDF description of an arbitrary dexterous hand. For the default 0.2B configuration, we train $\mathcal{T}(R, O)$ Grasp++ with a batch size of 48 per GPU and set the number of optimization steps to ensure full coverage of the training dataset. We use the AdamW [6] optimizer with a learning rate of 1×10^{-4}, 1,000 warmup steps, gradient clipping of 1.0, and bfloat16 mixed precision. The diffusion process follows a cosine noise schedule with $T = 1,000$ timesteps and offset $s = 0.008$. For each training batch, we sample $N_t = 5$ diffusion timesteps for supervision. For the loss function, we set loss weights $\gamma_{\text{pos}} = \gamma_{\text{rot}} = 1.0$. We train the model on 8 NVIDIA A100 80GB GPUs.

Inference Configurations. During inference, we use DDIM sampling [7] with 20 denoising steps and set $\eta = 1.0$. Given the point clouds of the target object and surrounding environment, together with the URDF description of the specified dexterous hand, $\mathcal{T}(R, O)$ Grasp++ iteratively denoises noisy robot-link poses and outputs link-wise $\mathrm{SE}(3)$ grasp poses. The predicted link poses are then retargeted to executable joint values based on Pyroki [3] toolkit. To improve the accuracy of the retargeting step, we solve IK from 8 different initializations in parallel and select the solution with the lowest average target-link position error after forward-kinematics evaluation.

D Additional Experiment

D.1 Cross-Embodiment Zero-Shot Generalization

Although $\mathcal{T}(R, O)$ Grasp++ is trained on eight dexterous embodiments covering 3-finger, 4-finger, and 5-finger hands, the diversity of available dexterous hands remains limited. Therefore, directly evaluating zero-shot generalization on entirely novel hands is challenging. To assess the cross-embodiment zero-shot generalization ability of $\mathcal{T}(R, O)$ Grasp++, we derive new hand embodiments by modifying the link lengths and joint limits of the existing training hands. We define link alignment S_{link} and joint overlap S_{joint} to measure the similarity between each derived hand and its original dexterous embodiment as follows:

$$S_{\text{link}} = 1 - \frac{1}{L}\sum_{i=1}^{L}\frac{|l_i' - l_i|}{l_i}, \quad S_{\text{joint}} = \frac{1}{L}\sum_{i=1}^{L}\frac{|j_i' \cap j_i|}{|j_i' \cup j_i|}. \tag{6}$$

where l_i and l_i' denote the original and modified link lengths, j_i and j_i' are the corresponding joint ranges. Under this definition, we conduct cross-embodiment zero-shot experiment on XHand.

Similarity	Link Alignment S_{link}		Joint Overlap S_{joint}	
	SR (%) ↑	Diversity ↑	SR (%) ↑	Diversity ↑
1.0	86.03	0.418	86.03	0.418
0.9	85.35	0.413	84.04	0.411
0.8	84.56	0.411	82.93	0.400
0.7	84.51	0.412	81.73	0.390

Table 3: **Cross-embodiment zero-shot performance.**

As shown in Tab. 3, $\mathcal{T}(R, O)$ Grasp++ maintains strong zero-shot generalization to unseen hand variants. When the embodiment similarity decreases from 1.0 to 0.7, the success rate remains above 80% for both link length and joint range perturbations. These results indicate that $\mathcal{T}(R, O)$ Grasp++ can generalize to novel dexterous embodiments with modified kinematic structures, demonstrating its effectiveness as a unified cross-embodiment grasping model.

D.2 Ablation

As discussed in the main paper, $\mathcal{T}(R, O)$ Graph++ encodes each robot link with a unique footprint that combines local link geometry and kinematic structure. This footprint allows the model to distinguish visually similar links with different kinematic roles across dexterous hands. To validate this design, we conduct an ablation study by removing the link-footprint encoding, including positional feature $\mathbf{F}^{\text{pos}}_{\text{robot}}$ and kinematic feature $\mathbf{F}^{\text{kine}}_{\text{robot}}$, while retaining only geometry feature $\mathbf{F}^{\text{BPS}}_{\text{robot}}$. We then evaluate how this geometry-only representation affects cross-embodiment grasp synthesis.

On Allegro Hand, removing the link-footprint encoding causes severe model collapse during inference. As illustrated in Fig. 3, the middle three fingers of Allegro Hand exhibit highly similar link geometries. With only the geometry link encoding $\mathbf{F}^{\text{BPS}}_{\text{robot}}$, $\mathcal{T}(R, O)$ Grasp++ model cannot distinguish links that share similar shapes but differ in kinematic positions and functional roles. Consequently, the predicted link SE(3) poses tend to collapse to similar configurations across fingers, resulting in a severe degradation in grasp synthesis performance. This ablation study demonstrates that link-footprint encoding (i.e., $\mathbf{F}^{\text{pos}}_{\text{robot}}$ and $\mathbf{F}^{\text{kine}}_{\text{robot}}$) is essential for learning distinguishable link representations in large-scale multi-hand training.

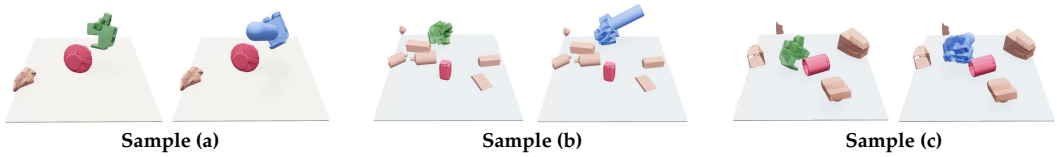


Figure 3: **Visualization of mode collapse during inference.** In each sample, the green meshes denote the per-link SE(3) predictions, while the blue mesh denotes the hand mesh reconstructed after Pyroki [3] IK.

References

[1] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

[2] K. Shaw, A. Agarwal, and D. Pathak. Leap hand: Low-cost, efficient, and anthropomorphic hand for robot learning. *arXiv preprint arXiv:2309.06440*, 2023.

[3] C. M. Kim, B. Yi, H. Choi, Y. Ma, K. Goldberg, and A. Kanazawa. Pyroki: A modular toolkit for robot kinematic optimization. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1312–1319. IEEE, 2025.

[4] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[5] A. Van Den Oord, O. Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[6] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[7] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.