GraspVLA: a Grasping Foundation Model Pre-trained on Billion-scale Synthetic Action Data

Shengliang Deng[*,1,3] **Mi Yan**[*,1,2] **Songlin Wei**[1,2] **Haixin Ma**[1] **Yuxin Yang**[1] **Jiayi Chen**[1,2]
Zhiqi Zhang[1,2] **Taoyu Yang**[2] **Xuheng Zhang**[2] **Heming Cui**[3] **Zhizheng Zhang**[†,1,4] **He Wang**[†,1,2,4]

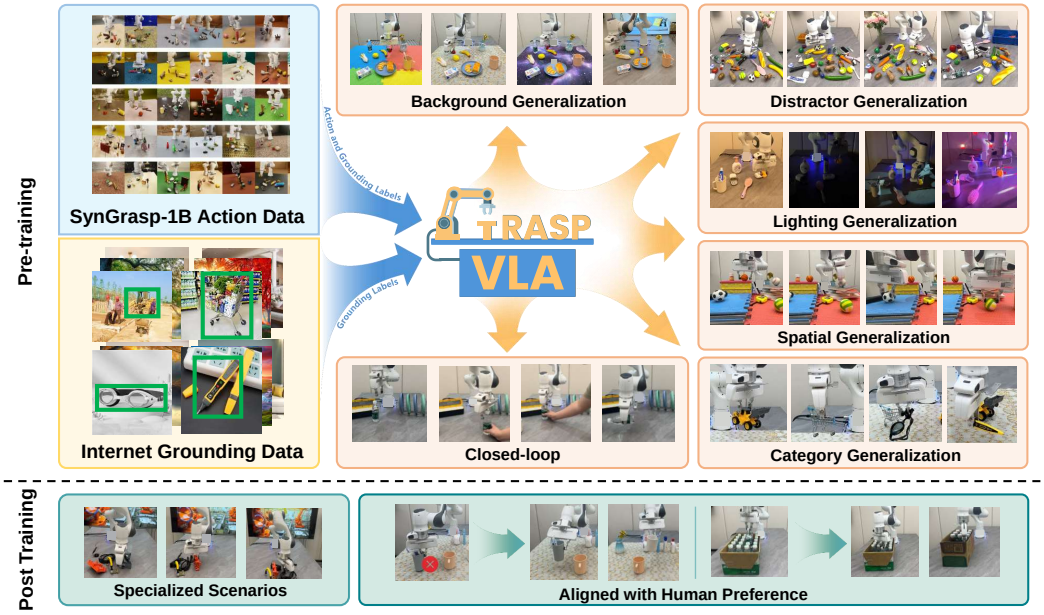


Figure 1: GraspVLA is a grasping foundation model pre-trained exclusively on billion-scale synthetic action data and co-trained with Internet semantics data. It exhibits direct sim-to-real transfer and strong zero-shot generalization across diverse aspects, as well as few-shot adaptability to specialized scenarios and human preferences.

Abstract: Embodied foundation models are gaining increasing attention for their zero-shot generalization, scalability, and adaptability to new tasks through few-shot post-training. However, existing models rely heavily on real-world data, which is costly and labor-intensive to collect. Synthetic data offers a cost-effective alternative, yet its potential remains largely underexplored. To bridge this gap, we explore the feasibility of training Vision-Language-Action (VLA) models entirely with large-scale synthetic action data. We curate SynGrasp-1B, a billion-frame robotic grasping dataset generated in simulation with photorealistic rendering and extensive domain randomization. Building on this, we present GraspVLA, a VLA model pretrained on large-scale synthetic action data as a foundational model for grasping tasks. GraspVLA integrates autoregressive perception tasks and flow-matching-based action generation into a unified Chain-of-Thought process, enabling joint training on synthetic action data and Internet semantics data. This design helps mitigate sim-to-real gaps and facilitates the transfer of learned actions to a broader range of Internet-covered objects, achieving open-vocabulary generalization in grasping. Extensive evaluations across real-world and simulation benchmarks demonstrate GraspVLA's advanced zero-shot generalizability and few-shot adaptability to specific human preferences. We will release SynGrasp-

[*] Equal contribution with the order determined by rolling dice. [†] denotes corresponding authors.
Correspondence to zhangzz@galbot.com, hewang@pku.edu.cn.
[1]Galbot, [2]Peking University, [3]The University of Hong Kong, [4]Beijing Academy of Artificial Intelligence

1B dataset and pre-trained weights to benefit the community. Our project page is at https://pku-epic.github.io/GraspVLA-web.

Keywords: Vision-Language-Action, Large-scale Robot Learning, Grasping

1 Introduction

The fields of Natural Language Processing (NLP) and Computer Vision (CV) have undergone a paradigm shift with the advent of foundation models. Large-scale models pretrained on vast amounts of Internet data exhibit zero-shot generalization to unseen scenarios [1, 2, 3] and few-shot adaptation for aligning with human preferences [4]. Inspired by this success, the foundation model for actions in the physical world has recently been introduced in Vision-Language-Action (VLA) models [5, 6, 7, 8]. These models process robotic visual observations and human instructions to directly generate robot actions. However, unlike vision and language modalities, action data is absent from existing Internet datasets, demanding a new paradigm for data collection.

Recent research mainly rely on real-world data collection through teleoperation, exemplified by community-driven efforts like Open-X-Embodiment (OXE) [9] and DROID [10] datasets. However, gathering real-world data at a large scale is both labor-intensive and costly, requiring a large number of robots and human operators, as well as diverse physical setups. In contrast, synthetic data offers a more accessible and cost-effective alternative – yet its potential remains largely underestimated.

To this end, we systematically explore the potential of synthetic data for training VLA models. As a first step in this direction, we focus on grasping, a fundamental robotic manipulation skill. We first curate a billion-frame grasping dataset, SynGrasp-1B, based on advanced ray-tracing rendering [11] and physics simulation [12], marking the first dataset of this scale globally. This dataset incorporates 10,000 unique objects from 240 categories and encompasses extensive domain randomization, ensuring broad coverage of geometric and visual variations.

To efficiently learn from this dataset, we propose GraspVLA, an end-to-end network that integrates autoregressive perception tasks and flow-matching-based action generation into a unified Chain-of-Thought (CoT) process, named Progressive Action Generation (PAG). PAG treats perception tasks, i.e., visual grounding and grasping pose prediction, as intermediate steps in action generation, forming a CoT process that causally infers actions. This design enables joint training on synthetic and Internet data in a unified framework, where Internet data is used to train the perception tasks (partial CoT process), and synthetic data is used to train the entire CoT pipeline. Synthetic data provides detailed geometric information about objects for object interactions, while Internet data offers rich semantic knowledge about objects. By leveraging these complementary sources, PAG reduces sim-to-real gaps and facilitates the transfer of learned robotic actions to semantically diverse Internet-covered objects, thereby enabling open-vocabulary grasping.

Empowered by our curated billion-scale synthetic grasping dataset and the proposed PAG mechanism, GraspVLA achieves direct sim-to-real generalization and demonstrates impressive zero-shot performance. To the best of our knowledge, this is the first work to reveal the significant potential of synthetic data in training VLA models for manipulation. Extensive experiments conducted in both real-world settings and the LIBERO [13] simulation benchmark demonstrate the model's robustness across diverse variations. In addition, GraspVLA shows excellent generalization to long-tail object categories absent from synthetic action data, such as chargers, towels, and swimming goggles. Compared to AnyGrasp [14], the state-of-the-art in traditional grasping detection algorithms, GraspVLA supports natural language instructions and delivers a robust closed-loop grasping policy. It achieves comparable performance on common objects while significantly outperforming AnyGrasp on transparent objects. Moreover, GraspVLA demonstrates strong few-shot adaptability to user preferences in specified application scenarios that extend beyond standard grasping behaviors, such as avoiding contact with the interior of drinking cups to maintain cleanliness and sequentially grasping bottles in densely packed environments.

In summary, our contributions are as follows: a) we introduce a novel pretraining paradigm that relies entirely on synthetic action data, significantly reducing the real world action data acquisition burden, b) we curate a billion-frame robotic grasping dataset, SynGrasp-1B, the first dataset of this

scale globally, c) we propose Progressive Action Generation to co-train synthetic actions with Internet data, extending GraspVLA's skills to novel object categories, and d) extensive experiments demonstrate GraspVLA's foundation capability, including strong zero-shot generalization and efficient few-shot adaptability in real-world.

2 Related Work

Vision-Language-Action (VLA) Models. Recently, a number of works[15, 16, 17, 18, 19, 20, 21, 22, 23] explored training an end-to-end VLA by learning from large-scale demonstration data. RT-2 [5] and OpenVLA [6] propose to leverage pre-trained vision-language models (VLMs) [24, 25] to exploit the rich knowledge from Internet dataset. Following the success of pre-trained VLMs, several works [26, 7, 27, 8, 28, 29] explore leveraging additional action expert to generate multi-modal actions with high fidelity. Others [30, 31, 32, 33, 34, 35] adopt generative pre-training on Internet-scale video data to learn from human videos. However, limited by the scale of real-world robotic data, existing VLA models mainly rely on in-domain post-training for deployment. Concurrent work, $\pi_{0.5}$ [36], proposes improving generalization by leveraging multimodal web data and cross-embodiment data, enabling direct out-of-the-box deployment. While our work also targets zero-shot deployment, we take a different approach—exclusively pre-training on large-scale synthetic data—and demonstrate strong zero-shot generalization.

Synthetic Data. With the fast development of GPU-accelerated simulation and photo-realistic rendering, synthetic data generation has become a popular approach to train robotic models. Previous works [37, 38, 39] pioneered the use of simulated data with domain randomization to train open-loop grasping models. Recently, several works [40, 41, 42] explore automatically augmenting human demonstrations in simulation by randomizing object configurations and leveraging motion planning to generate realistic robot trajectories. Another line of work [43, 44, 45, 46] synthesizes data from a few human demonstrations utilizing text-to-image generation models and multi-view stereoscopic rendering, without requiring any physical simulation. While these methods [47] still rely on human demonstrations to generate augmented data, our work explores direct sim-to-real transfer by leveraging large-scale synthetic data together with pre-trained vision and language backbones.

Grasping. Grasping is an essential skill [48] for embodied agents and has been actively studied in the past decade. Some works tackle this problem through open-loop grasp detection [49, 14, 50] and then control the end effector using a motion planner. Such modular-based systems usually suffer from poor depth perception [51] and lack of failure recovery behavior [52, 53]. Another line of research explores vision-based grasping systems in an end-to-end and closed-loop manner, either through reinforcement learning [54] or imitation learning [55]. With the advent of vision-language foundation models [1, 56, 57], several works aim to generalize grasping to open-vocabulary objects [58, 59, 60, 61, 62] by building a modular system that combines a grasp detection model with a VLM. While these methods achieve impressive results in standard grasping, they face challenges in adapting to specialized tasks, such as grasping with specific constraints.

3 SynGrasp-1B Dataset Generation

Training a generalizable foundation model requires a large-scale dataset encompassing diverse objects and environmental conditions. Instead of relying on costly real-world human data collection, we propose training entirely on synthetic data – which offers greater diversity at a fraction of the time and expense. We now detail the core components of our synthetic data generation pipeline.

Object Assets and Layout Generation. We utilize the LVIS subset of the Objaverse dataset [63] and carefully filter out unsuitable categories, such as weapons, resulting in a total of 240 categories and 10,680 instances. We randomly scale these objects and drop them in various poses onto a table, generating diverse and physically plausible scenes. More details can be found in the supplementary.

Grasp Synthesis and Trajectory Generation. Given initial layouts, we utilize advanced modular system to establish an expert policy for generating high-quality trajectories for grasping and lifting target objects. For each object instance, we leverage grasp synthesis algorithm [64] to generate stable antipodal grasps. We then use motion planning algorithms CuRobo [65] to plan collision-free trajectories to reach the open-loop grasp pose and lift the object. We validate all candidate trajectories in the MuJoCo physics simulator [12] to ensure successful lifting of the object.

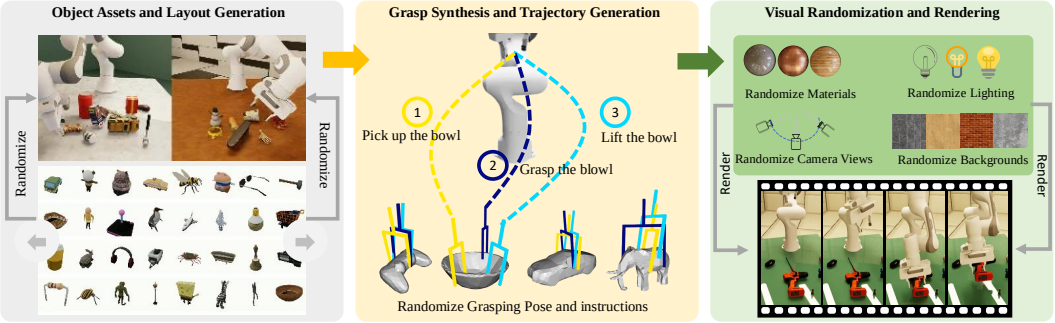


Figure 2: **Data generation pipeline:** We first curated over 10,680 object meshes from Objaverse [63] that are suitable for tabletop grasping and randomly selected and placed these objects on the table (left). Next, we used CuRobo to plan grasping trajectories with randomized grasp poses and instructions (middle). Finally, we applied domain randomization to materials (table and robot), lighting, camera views, and backgrounds to simulate and render the trajectories (right).

Visual Randomization and Rendering. Given diverse layouts and corresponding trajectories, we render high-quality RGB images with randomized lighting, backgrounds, and camera settings using Isaac Sim [66], which offers efficient photo-realistic ray-traced rendering. We employ various light sources with extensive randomization, including point, directional, and dome lights. Images are rendered from two different viewpoints to provide a comprehensive view of the scene, with randomized extrinsics around predefined centers. More details are provided in the supplementary material.

We further highlight two major considerations in the design of our data generation pipeline:

Efficient Data Generation. We develop three key strategies to improve the efficiency. High-quality meshes are often large, leading to lengthy loading times and significant memory usage. We implement a caching mechanism to avoid redundant loading while ensuring data diversity. Second, we implement asynchronous data writing, allowing images and labels to be saved in parallel, thereby improving overall efficiency in data generation. Finally, we employ parallel physics simulation and rendering to further improve efficiency. Please refer to the supplementary for more details.

Tailoring Data for Imitation Learning. To ease the difficulty of imitation learning, we introduce two improvements. First, while open-loop grasping [14] employs a two-step process (pregrasp positioning followed by grasp execution) to avoid collision, this segmented approach creates pauses. Imitation policies trained on such data often exhibit hesitation [6, 67]. Instead, we implement single-step motion planning, prioritizing trajectory smoothness over planning success rates. Second, we introduce randomized initial robot poses to improve workspace exploration and observation diversity in expert demonstrations, enhancing model robustness [68].

With this pipeline, we generate our billion-frame dataset, SynGrasp-1B, using 160 NVIDIA 4090 GPUs for 10 days. We provide data diversity analysis in the supplementary.

4 Model

Overall Architecture. GraspVLA integrates a Vision-Language Model (VLM) with an action expert [7], connected through a Progressive Action Generation (PAG) mechanism, as illustrated in Figure 3. The VLM takes observation images and a text instruction for vision-language joint perception. It comprises a trainable large language model (InternLM2 1.8B [69]), a vision encoder that fuses features from frozen DINO-v2 [70] and SigLIP [71] inspired by OpenVLA [6], and a trainable projector from the vision space to the language space. We use a conditional flow matching action expert [72] for fine-grained end-effector action generation. We further introduce PAG to efficiently transfer knowledge learned from Internet grounding dataset to grasping skills.

Progressive Action Generation. While GraspVLA learns generalizable grasping skills from our SynGrasp-1B dataset, it is constrained by the set of categories present in the synthetic dataset. To scale the grasping policy to novel categories, a straight-forward approach is to co-train with Internet grounding dataset as separate tasks, and rely on the model to implicitly generalize to object categories learned from the grounding dataset.

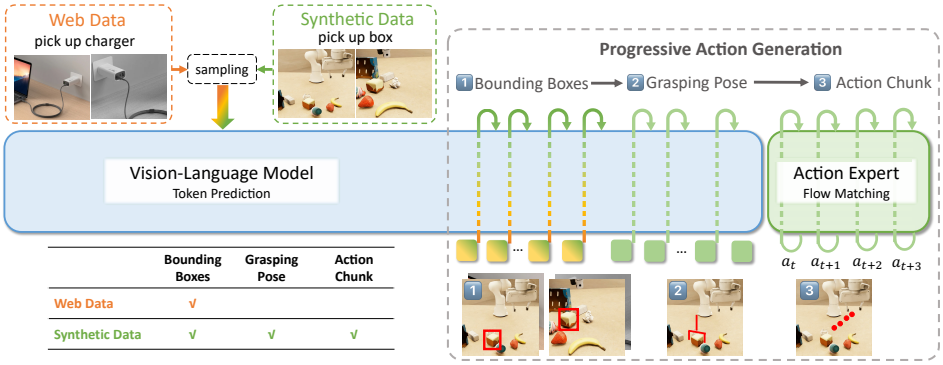


Figure 3: **GraspVLA** consists of an autoregressive vision-language backbone and a flow-matching based action expert. It exploits the synergy between Internet grounding data and synthetic action data with a Progressive Action Generation mechanism: the model first predicts 2D bounding boxes of the target object for both synthetic data and web data, and additionally generates grasp pose and chunked actions for synthetic data.

Alternatively, we formulate image grounding and grasp pose prediction as intermediate steps to generate action. Specifically, the VLM is trained to generate 2D bounding boxes for both Internet grounding dataset and synthetic action dataset in a unified format. Then, for the synthetic dataset, the VLM further predicts the target grasp pose in the robot's base frame. Finally, the action expert generates action chunk conditioned on the VLM's key-value cache of both input and intermediate reasoning tokens. To facilitate accurate 3D sensing, the proprioceptions from the latest two timesteps are tokenized and inserted before generating grasp pose. To align the Internet dataset with the dual-camera setup of SynGrasp-1B, input images are duplicated to match the number of views and independently augmented with random resizing, cropping, horizontal flipping, and color jittering. Both datasets share the same text prompt template, generating bounding box tokens first. This unified training strategy exploits the synergy between the Internet grounding and synthetic datasets, and resembles the Chain-of-Thought reasoning mechanism widely studied and proven as an effective measure to handle highly complex tasks in large language models [73].

Joint Training of VLM and action expert. In each batch, we randomly sample from the Internet dataset (GRIT [74]) and the synthetic action dataset. The former is used solely to supervise the VLM's bounding box prediction in an auto-regressive manner. The latter supervises bounding box, grasp pose, and flow-matching-based action prediction. The loss of VLM is formally defined as:

$$\mathcal{L}_{\text{S2}} = -\sum_{n=1}^{N_{\text{bbox}}} \log P_\theta(\mathbf{y}_{\text{bbox},n} \mid \mathbf{x}, \mathbf{y}_{\text{bbox},<n}) - \mathbf{1}_{\text{synthetic}} \cdot \sum_{n=1}^{N_{\text{grasp}}} \log P_\theta(\mathbf{y}_{\text{grasp},n} \mid \mathbf{x}, \mathbf{y}_{\text{bbox}}, \mathbf{y}_{\text{grasp},<n}),$$

where N_{bbox} and N_{grasp} are the lengths of the bounding box and grasp pose token sequences respectively, $\mathbf{y}_{\text{bbox},n}$ and $\mathbf{y}_{\text{grasp},n}$ are tokens at position n in their respective sequences, and $\mathbf{x}$ is the input images and text. The action expert is supervised with flow matching loss on chunked end-effector delta actions:

$$\mathcal{L}_{\text{S1}} = \|v_t(\mathbf{A}_t, \mathbf{x}, \mathbf{y}_{\text{bbox}}, \mathbf{y}_{\text{grasp}}) - u_t(\mathbf{A}_t \mid \mathbf{A}_0)\|^2,$$

where $t \in [0, 1]$ is the flow matching timestep, $\mathbf{A}_t$ is the noised action trunk at t, $v_t(\cdot)$ is the model predicted flow matching vector field, $u_t(\mathbf{A}_t \mid \mathbf{A}_0)$ is the ground-truth vector field. We empirically found a simple sum of $\mathcal{L}_{\text{S2}}$ and $\mathcal{L}_{\text{S1}}$ for the overall loss yields good performance.

5 Experiments

We evaluate GraspVLA to answer the following questions: (1) How does GraspVLA compare with existing work under various generalization factors? (2) How does GraspVLA scale with the amount of data? (3) How much do our design choices contribute to GraspVLA's performance? (4) How well does GraspVLA support few-shot post-training for specialized preferences?

	Synthetic Categories						Web Categories					
	basic↑	light↑	b.g.↑	dis.↑	height↑	SPL↑	basic↑	light↑	b.g.↑	dis.↑	height↑	SPL↑
Diffusion Policy [75]	30.0	16.6	16.6	13.3	13.3	12.3	-	-	-	-	-	-
Octo [26]	16.6	3.3	0.0	0.0	3.3	3.2	0.0	3.3	0.0	0.0	0.0	0.4
OpenVLA [6]	20.0	13.3	16.6	0.0	13.3	8.8	3.3	6.6	13.3	0.0	6.6	4.1
π_0(w/ π_0 pre-train)[7]	66.6	63.3	60.0	60.0	56.6	42.3	33.3	36.6	30.0	26.6	26.6	17.8
π_0(w/o π_0 pre-train)[7]	80.0	76.6	80.0	86.6	76.6	51.8	40.0	40.0	36.6	36.6	33.3	36.9
Ours	**93.3**	**96.6**	**93.3**	**93.3**	**90.0**	**87.2**	**93.3**	**90.0**	**93.3**	**86.6**	**86.6**	**84.7**

Table 1: **Zero-shot comparisons in real-world.** We compare our method against state-of-the-art imitation learning specialists and large VLA models. All models are fine-tuned on SynGrasp-1B dataset. Our approach achieves the highest grasping success rate on items from both synthetic and web categories using short trajectories. Detailed description of setups is provided in Section 5.1.

5.1 Zero-Shot Comparison with VLAs in Real World

Task Definition. To evaluate the effectiveness of PAG, we use two groups of objects: **synthetic categories** and **web categories**. We define synthetic categories as those present in our SynGrasp-1B dataset, while web categories refer to those exclusively present in Internet grounding dataset.

For each group of objects, we design 5 test sets: **basic**, **lighting**, **background**, **distractors**, and **height**. Each test set contains 15 objects from distinct categories randomly sampled from each group, with 2 trials per object. In other words, we test each method for $15 \times 2 \times 5 \times 2 = 300$ trials in total. We use a disco light to generate different lighting conditions. For background generalization, three distinct tablecloths are selected and interchanged. For distractor generalization, we randomly place 5 additional objects on the table as distractors. For height generalization, we increase the height of the workspace surface by 10 cm. We utilize a Franka Panda arm and employ two Intel RealSense cameras as front and side cameras. The workspace is confined to a 40 cm × 50 cm x 20 cm area in front of the robot. The initial robot and object states are fixed within each trial to ensure fair comparison.

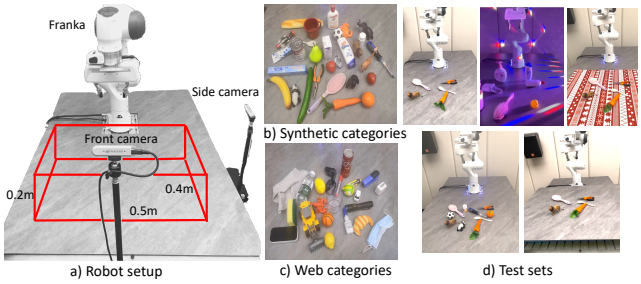


Figure 4: We show our real-world setup in (a), objects used in experiments in (b,c), and 5 test sets corresponding to basic, light, background, distractor, and height settings in (d).

Metrics. The success rate is defined as the percentage of trials in which the model successfully grasps the target object within 3 attempts. For each object group, we also report the average Success weighted by Path Length (SPL) [76], a widely used metric that weights success rate with motion efficiency by penalizing unnecessarily long paths. It is computed as: $\frac{1}{N}\sum_{i=1}^{N} S_i \frac{l_i}{max(p_i,l_i)}$, where S_i is a binary indicator of success (1 if successful), l_i is the shortest path length achieved by any method in the trial, p_i is the path length taken by the model, and N is the total number of trials.

Baselines. We compare with multiple baselines including both VLA generalists and imitation learning specialists. For generalists, we use π_0 [7], OpenVLA [6], and Octo [26], three transformer-based policies pre-trained on large-scale real-world datasets. To ensure fair comparison, we fine-tune all three models on our SynGrasp-1B dataset. Additionally, to assess the effectiveness of pre-training on **SynGrasp-1B**, we report results of direct fine-tuning π_0 from its VLM weights [77], without its cross-embodiment robotic pre-training. For specialists, we use Diffusion Policy [75], a strong diffusion baseline for visual-conditioned imitation learning. As it lacks language conditioning, we train and test it using only the elephant category. Additional details are provided in the supplementary.

Comparisons. As illustrated in Table 1, GraspVLA achieves around 90% on all test sets and significantly outperforms all baselines, demonstrating strong zero-shot generalizability. Notably, GraspVLA achieves comparable results in both synthetic and web categories, underscoring the effectiveness of PAG. Additionally, the SPL metric reveals that GraspVLA grasps objects with shorter path lengths compared to π_0 baselines which often exhibit hesitation. Interestingly, the π_0 baseline without cross-embodiment pre-training performs better than its pre-trained counterpart, suggesting

that cross-embodiment pre-training may not be optimal for this specific grasping task on the given robotic arm. We provide failure analysis in the supplementary.

5.2 Zero-Shot Comparison with VLAs in LIBERO Benchmark

Setup. LIBERO [13] is a widely used simulation benchmark for robotic manipulation, encompassing diverse tasks and object categories. We evaluate on three LIBERO suites (Long, Goal, Object), excluding Spatial, as its focus on spatial reasoning falls outside our scope. To concentrate on grasping capabilities, we omit non-prehensile tasks (e.g., 'turn on the stove') and reformulate task captions as 'pick up {object}', selecting 7-10 tasks per suite. In line with standard evaluation protocols, each task is rigorously tested with 50 randomized initial configurations, resulting in 350-500 trials per suite. More details are provided in the supplementary.

	Long	**Goal**	**Object**
OpenVLA (fine-tuned)	*33.7*	*56.6*	*65.4*
π_0 (fine-tuned)	*62.7*	*79.4*	*93.8*
Ours (zero-shot)	**82.0**	**91.2**	**94.1**

Table 2: **Comparisons with baselines in LIBERO.** The zero-shot performance of GraspVLA surpasses the fine-tuned performance of strong baselines π_0 and OpenVLA.

Comparisons. As shown in Table 2, GraspVLA demonstrates satisfactory performance when zero-shot evaluated on LIBERO. It surpasses π_0 and OpenVLA fine-tuned on the LIBERO dataset, demonstrating strong generalizability. We also observe that the format of task captions significantly affects the performance of fine-tuned models and provide detailed results in the supplementary.

5.3 Zero-Shot Comparison with AnyGrasp in Real World

Setup. We benchmark GraspVLA against AnyGrasp [14], a state-of-the-art grasp detection model specialized in grasping. For language-conditioned grasping, we integrate AnyGrasp with Grounding DINO [78], a popular open-vocabulary object detector, to filter grasp candidates. We use the same two basic test sets (Section 5.1), with metrics including overall success rate (task completion) and grasping success rate (grasping any object). To isolate grasping performance, we design two additional test sets (30 trials each): one with common household objects and another with transparent objects, where the robot can grasp any object in the scene.

	Language-Conditioned		**Arbitary Grasping**		**Speed**
	overall	grasp	common	transparent	
AnyGrasp	91.6	**96.6**	**100.0**	10.0	37 Hz
Ours	**93.3**	93.3	93.3	**86.6**	5 Hz

Table 3: **Comparison with AnyGrasp.** GraspVLA performs consistently well in both language-guided and arbitrary grasping tasks. In contrast, AnyGrasp is faster and excels at grasping common objects but struggles with transparent objects.

Comparisons. In the language-conditioned test set, both model achieve similar performance, with GraspVLA slightly outperforming AnyGrasp in grounding ability, due to its comprehensive multi-view observation. In arbitrary object grasping, while AnyGrasp achieves a 100% success rate in grasping common objects, it struggles with transparent objects due to inaccurate depth sensing and incomplete point cloud data. In contrast, GraspVLA maintains consistent performance across both test sets, highlighting its robustness to material variations. However, GraspVLA's inference speed is significantly slower than AnyGrasp's, a limitation tied to its large vision-language backbone.

5.4 Scaling Law

Figure 5 shows the scaling curve regarding the number of training frames in real world. We observe that the performance improves steadily with the number of training frames and the performance on web categories scales slower than that of synthetic categories, indicating that more training frames are needed for good generalization on web categories. For scaling law regarding the number of training categories and the number of instances per category, please refer to the supplementary.

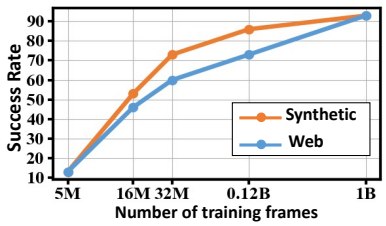


Figure 5: The performance scales with the number of training frames, especially for web categories.

5.5 Efficient Post-Training

A defining characteristic of foundation models is their ability to adapt to new tasks. To this end, we design three downstream tasks: i) Task 1 – grasping rare industrial components, ii) Task 2 – grasping a mug without touching its interior to maintain cleanliness, and iii) Task 3 – sequential grasping in a

densely packed environment. These tasks rigorously benchmark the model's adaptability capability to three critical challenges: (i) generalizing to new vocabularies, (ii) executing task-specific grasp specifications, and (iii) grasping in order. We collect 100 demos for Tasks 1–2 and 10 per bottle for Task 3. We conduct 10 trials per task and report the overall success rate (task completion) and the grasping success rate (grasping any object).

a) Post-training Task 1 b) Post Training Task 2 c) Post Training Task 3

Figure 6: **Real-world post-training.** We experimented with three different post-training tasks to showcase that our model can quickly learn to grasp new items in (a), new grasping patterns in (b), and new grasping behavior in (c).

	Training Data		Task 1		Task 2		Task 3	
	BBox	traj.	overall	grasp	overall	grasp	overall	grasp
OpenVLA	-	-	0	0	0	20	0	0
π_0	-	-	10	20	0	30	0	0
Ours	-	-	40	90	0	80	0	20
DP	-	✓	-	-	20	60	10	30
OpenVLA	-	✓	0	0	20	30	0	20
π_0	-	✓	60	80	60	70	50	60
Ours	✓	-	**90**	**100**	-	-	-	-
Ours(scratch)	✓	✓	10	30	10	30	0	20
Ours	✓	✓	**90**	**100**	**80**	**90**	**90**	**90**

Table 4: **Efficient post-training.** GraspVLA shows superior adaptability to novel tasks, surpassing the model without pre-training and all baselines.

As shown in Table 4, GraspVLA achieves a 90% success rate with only bounding box annotations in Task 1, surpassing baselines trained on full action data. This suggests that extending GraspVLA to new objects does not necessitate action annotations, thereby greatly reducing data collection effort. As shown by the last two rows, training from scratch yields lower performance, underscoring the value of our synthetic pre-training. Notably, in Task 3's dense sequential grasping, GraspVLA learns to avoid collisions with surrounding objects effectively.

5.6 Effectiveness of Design Choices

As shown in Table 5, we evaluate the effectiveness of our key design choices using both success rate and SPL metrics on the basic test set described in Sec. 5.1. The vanilla baseline, which employs co-training with Internet grounding data but excludes PAG, serves as our starting point. Introducing 2D bounding boxes as intermediate action steps (PAG-2D) yields significant improvements for web categories. Further enhancement comes with grasp pose prediction (PAG-3D), which substantially reduces hesitation behavior and improves grasping accuracy. This leads to fewer attempts and shorter trajectories, as reflected in the higher SPL scores. Together, these results demonstrate the effectiveness of our PAG approach.

	Synthetic		Web	
	SR	SPL	SR	SPL
vanilla	66.6	39.3	53.3	27.7
+ PAG-2D	80.0	59.2	76.7	48.9
+ PAG-3D	**93.3**	**90.2**	**93.3**	**91.7**

Table 5: We give a detailed ablation study of our models. With all the design choices enabled the performance boosts significantly.

6 Conclusion

In this work, we investigated building a generalizable grasping VLA model with large-scale synthetic data. First, we curated a billion-scale grasping dataset in simulation, featuring extensive randomization and photorealistic rendering. Second, we carefully designed our model to effectively learn from synthetic action data and action-free Internet grounding data, achieving strong generalizability for grasping novel-category objects in unseen environments. Extensive ablation studies and comparisons demonstrate that our method achieves state-of-the-art performance in table-top grasping. Furthermore, we observed that our model scales effectively with the amount of synthetic training data. Finally, we showcase that GraspVLA can acquire new grasping behaviors through few-shot post-training, highlighting its adaptability and potential for real-world applications.

7 Limitations and Future Work

Currently, our data generation and evaluation are conducted exclusively on the Franka Panda arm with front and side views. However, our simulation pipeline is inherently scalable and can be readily adapted to other robots and camera configurations. We leave this engineering effort as future work.

GraspVLA struggles with ambiguous instructions such as "pick up food" and "pick up the leftmost object". Addressing these challenges may require scaling vision-language pretraining and exploring architectural innovations to enhance semantic reasoning.

Like most grasping policies, we synthesize grasp labels using force-closure, which do not account for deformability—a limitation common to all such methods. Despite this, our model can still grasp certain deformable objects if their initial geometry contains convex regions enabling force closure. While previous works [79] have shown that soft-body simulation can be used to train sim2real deformable manipulation policies, we leave the integration as future work.

While current model focuses on grasping, the model design is not tailored to this specific task. We plan to extend the data generation pipeline to support other manipulation tasks, such as pick-and-place and pushing. Beyond the current modular-based expert policy used in data generation, we will explore reinforcement learning for more complex tasks like non-prehensile manipulation.

Although our PAG mechanism enables open-vocabulary grasping, it introduces additional latency. We currently achieve around 200ms latency on NVIDIA L40s utilizing Torch Compile [80]. While this is sufficient for static scenes, it may not be enough for dynamic environments, e.g., fast moving objects. Distillation and quantization techniques can be further explored.

Acknowledgments

This work was supported in part by National Key R&D Program of China 62306016 and National Key R&D Program of China 2022ZD0160201. Additionally, we extend our sincere gratitude to all our colleagues at Galbot for their assistance in collecting and annotating the post-training data.

References

[1] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

[2] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[3] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

[4] OpenAI. Chatgpt: Jan 17 version. https://openai.com/chatgpt, 2023. [Large language model].

[5] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[6] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[7] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. pi0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[8] NVIDIA, :, J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. J. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, J. Jang, Z. Jiang, J. Kautz, K. Kundalia, L. Lao, Z. Li, Z. Lin, K. Lin, G. Liu, E. Llontop, L. Magne, A. Mandlekar, A. Narayan, S. Nasiriany, S. Reed, Y. L. Tan, G. Wang, Z. Wang, J. Wang, Q. Wang, J. Xiang, Y. Xie, Y. Xu, Z. Xu, S. Ye, Z. Yu, A. Zhang, H. Zhang, Y. Zhao, R. Zheng, and Y. Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

[9] A. O'Neill, A. Rehman, A. Gupta, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[10] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[11] J. Liang, V. Makoviychuk, A. Handa, N. Chentanez, M. Macklin, and D. Fox. Gpu-accelerated robotic simulation for distributed reinforcement learning, 2018.

[12] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi:10.1109/IROS.2012.6386109.

[13] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[14] H.-S. Fang, C. Wang, H. Fang, M. Gou, J. Liu, H. Yan, W. Liu, Y. Xie, and C. Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE Transactions on Robotics*, 39(5):3929–3945, 2023.

[15] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[16] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking, 2023.

[17] L. Wang, X. Chen, J. Zhao, and K. He. Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers. *arXiv preprint arXiv:2409.20537*, 2024.

[18] M. Zawalski, W. Chen, K. Pertsch, O. Mees, C. Finn, and S. Levine. Robotic control via embodied chain-of-thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024.

[19] X. Li, M. Zhang, Y. Geng, H. Geng, Y. Long, Y. Shen, R. Zhang, J. Liu, and H. Dong. Manipllm: Embodied multimodal large language model for object-centric robotic manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18061–18070, 2024.

[20] X. Li, C. Mata, J. Park, K. Kahatapitiya, Y. S. Jang, J. Shang, K. Ranasinghe, R. Burgert, M. Cai, Y. J. Lee, et al. Llara: Supercharging robot learning data for vision-language policy. *arXiv preprint arXiv:2406.20095*, 2024.

[21] A. Goyal, V. Blukis, J. Xu, Y. Guo, Y.-W. Chao, and D. Fox. Rvt-2: Learning precise manipulation from few demonstrations. *arXiv preprint arXiv:2406.08545*, 2024.

[22] H. Zhen, X. Qiu, P. Chen, J. Yang, X. Yan, Y. Du, Y. Hong, and C. Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024.

[23] J. Zhang, K. Wang, R. Xu, G. Zhou, Y. Hong, X. Fang, Q. Wu, Z. Zhang, and H. Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*, 2024.

[24] X. Chen, J. Djolonga, P. Padlewski, B. Mustafa, S. Changpinyo, J. Wu, C. R. Ruiz, S. Goodman, X. Wang, Y. Tay, et al. Pali-x: On scaling up a multilingual vision and language model. *arXiv preprint arXiv:2305.18565*, 2023.

[25] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[26] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

[27] Q. Li, Y. Liang, Z. Wang, L. Luo, X. Chen, M. Liao, F. Wei, Y. Deng, S. Xu, Y. Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

[28] J. Wen, Y. Zhu, J. Li, M. Zhu, K. Wu, Z. Xu, N. Liu, R. Cheng, C. Shen, Y. Peng, F. Feng, and J. Tang. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation, 2024. URL https://arxiv.org/abs/2409.12514.

[29] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

[30] C.-L. Cheang, G. Chen, Y. Jing, T. Kong, H. Li, Y. Li, Y. Liu, H. Wu, J. Xu, Y. Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

[31] S. Ye, J. Jang, B. Jeon, S. Joo, J. Yang, B. Peng, A. Mandlekar, R. Tan, Y.-W. Chao, B. Y. Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024.

[32] H. Bharadhwaj, D. Dwibedi, A. Gupta, S. Tulsiani, C. Doersch, T. Xiao, D. Shah, F. Xia, D. Sadigh, and S. Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. *arXiv preprint arXiv:2409.16283*, 2024.

[33] J. Yang, B. Liu, J. Fu, B. Pan, G. Wu, and L. Wang. Spatiotemporal predictive pre-training for robotic motor control. *arXiv preprint arXiv:2403.05304*, 2024.

[34] Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. *arXiv preprint arXiv:2503.22020*, 2025.

[35] Y. Tian, S. Yang, J. Zeng, P. Wang, D. Lin, H. Dong, and J. Pang. Predictive inverse dynamics models are scalable learners for robotic manipulation, 2024. URL https://arxiv.org/abs/2412.15109.

[36] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

[37] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, and V. Vanhoucke. Using simulation and domain adaptation to improve efficiency of deep robotic grasping, 2017. URL https://arxiv.org/abs/1709.07857.

[38] C. Eppner, A. Mousavian, and D. Fox. Acronym: A large-scale grasp dataset based on simulation, 2020. URL https://arxiv.org/abs/2011.09584.

[39] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics, 2017. URL https://arxiv.org/abs/1703.09312.

[40] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations, 2023. URL https://arxiv.org/abs/2310.17596.

[41] Z. Jiang, Y. Xie, K. Lin, Z. Xu, W. Wan, A. Mandlekar, L. Fan, and Y. Zhu. Dexmimicgen: Automated data generation for bimanual dexterous manipulation via imitation learning, 2025. URL https://arxiv.org/abs/2410.24185.

[42] C. Garrett, A. Mandlekar, B. Wen, and D. Fox. Skillmimicgen: Automated demonstration generation for efficient skill learning and deployment, 2024. URL https://arxiv.org/abs/2410.18907.

[43] S. Yang, W. Yu, J. Zeng, J. Lv, K. Ren, C. Lu, D. Lin, and J. Pang. Novel demonstration generation with gaussian splatting enables robust one-shot manipulation, 2025. URL https://arxiv.org/abs/2504.13175.

[44] Z. Xue, S. Deng, Z. Chen, Y. Wang, Z. Yuan, and H. Xu. Demogen: Synthetic demonstration generation for data-efficient visuomotor policy learning, 2025. URL https://arxiv.org/abs/2502.16932.

[45] Z. Chen, S. Kiami, A. Gupta, and V. Kumar. Genaug: Retargeting behaviors to unseen situations via generative augmentation, 2023. URL https://arxiv.org/abs/2302.06671.

[46] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, D. M, J. Peralta, B. Ichter, K. Hausman, and F. Xia. Scaling robot learning with semantically imagined experience, 2023. URL https://arxiv.org/abs/2302.11550.

[47] A. Maddukuri, Z. Jiang, L. Y. Chen, S. Nasiriany, Y. Xie, Y. Fang, W. Huang, Z. Wang, Z. Xu, N. Chernyadev, S. Reed, K. Goldberg, A. Mandlekar, L. Fan, and Y. Zhu. Sim-and-real co-training: A simple recipe for vision-based robotic manipulation, 2025. URL https://arxiv.org/abs/2503.24361.

[48] R. Newbury, M. Gu, L. Chumbley, A. Mousavian, C. Eppner, J. Leitner, J. Bohg, A. Morales, T. Asfour, D. Kragic, et al. Deep learning approaches to grasp synthesis: A review. *IEEE Transactions on Robotics*, 39(5):3994–4015, 2023.

[49] H.-S. Fang, C. Wang, M. Gou, and C. Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11441–11450, 2020. doi:10.1109/CVPR42600.2020.01146.

[50] A. Mousavian, C. Eppner, and D. Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2901–2910, 2019.

[51] S. Wei, H. Geng, J. Chen, C. Deng, C. Wenbo, C. Zhao, X. Fang, L. Guibas, and H. Wang. D3roma: Disparity diffusion-based depth sensing for material-agnostic robotic manipulation. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=7E3JAys1xO.

[52] Y. Liu, A. Qualmann, Z. Yu, M. Gabriel, P. Schillinger, M. Spies, N. A. Vien, and A. Geiger. Efficient end-to-end detection of 6-dof grasps for robotic bin picking, 2024. URL https://arxiv.org/abs/2405.06336.

[53] H. Geng, S. Wei, C. Deng, B. Shen, H. Wang, and L. Guibas. Sage: Bridging semantic and actionable parts for generalizable articulated-object manipulation under language instructions. *arXiv preprint arXiv:2312.01307*, 2023.

[54] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation, 2018. URL https://arxiv.org/abs/1806.10293.

[55] S. Song, A. Zeng, J. Lee, and T. Funkhouser. Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. *IEEE Robotics and Automation Letters*, 5(3): 4978–4985, 2020.

[56] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[57] S. Karamcheti, S. Nair, A. Balakrishna, P. Liang, T. Kollar, and D. Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models, 2024. URL https://arxiv.org/abs/2402.07865.

[58] A. D. Vuong, M. N. Vu, H. Le, B. Huang, B. Huynh, T. Vo, A. Kugi, and A. Nguyen. Graspanything: Large-scale grasp dataset from foundation models, 2023. URL https://arxiv.org/abs/2309.09818.

[59] A. Stone, T. Xiao, Y. Lu, K. Gopalakrishnan, K.-H. Lee, Q. Vuong, P. Wohlhart, S. Kirmani, B. Zitkovich, F. Xia, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

[60] C. Tang, D. Huang, W. Ge, W. Liu, and H. Zhang. Graspgpt: Leveraging semantic knowledge from a large language model for task-oriented grasping. *IEEE Robotics and Automation Letters*, 2023.

[61] Y. Lu, Y. Fan, B. Deng, F. Liu, Y. Li, and S. Wang. Vl-grasp: a 6-dof interactive grasp policy for language-oriented objects in cluttered indoor scenes. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 976–983. IEEE, 2023.

[62] Y. Ding, H. Geng, C. Xu, X. Fang, J. Zhang, S. Wei, Q. Dai, Z. Zhang, and H. Wang. Open6dor: Benchmarking open-instruction 6-dof object rearrangement and a vlm-based approach. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7359–7366. IEEE, 2024.

[63] M. Deitke, D. Schwenk, J. Salvador, L. Weihs, O. Michel, E. VanderBilt, L. Schmidt, K. Ehsani, A. Kembhavi, and A. 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.

[64] J. Chen, Y. Ke, and H. Wang. Bodex: Scalable and efficient robotic dexterous grasp synthesis using bilevel optimization. *arXiv preprint arXiv:2412.16490*, 2024.

[65] B. Sundaralingam, S. K. S. Hari, A. Fishman, C. Garrett, K. Van Wyk, V. Blukis, A. Millane, H. Oleynikova, A. Handa, F. 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.

[66] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, 2023. doi:10.1109/LRA.2023.3270034.

[67] M. Dalal, A. Mandlekar, C. Garrett, A. Handa, R. Salakhutdinov, and D. Fox. Imitating task and motion planning with visuomotor transformers. *arXiv preprint arXiv:2305.16309*, 2023.

[68] F. Lin, Y. Hu, P. Sheng, C. Wen, J. You, and Y. Gao. Data scaling laws in imitation learning for robotic manipulation, 2024. URL `https://arxiv.org/abs/2410.18647`.

[69] Z. Cai, M. Cao, H. Chen, K. Chen, K. Chen, X. Chen, X. Chen, Z. Chen, Z. Chen, P. Chu, et al. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*, 2024.

[70] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[71] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023.

[72] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[73] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[74] Z. Peng, W. Wang, L. Dong, Y. Hao, S. Huang, S. Ma, and F. Wei. Kosmos-2: Grounding multimodal large language models to the world. *ArXiv*, abs/2306.14824, 2023.

[75] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[76] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, and A. R. Zamir. On evaluation of embodied navigation agents, 2018. URL https://arxiv.org/abs/1807.06757.

[77] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[78] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang. Grounding dino: Marrying dino with grounded pre-training for open-set object detection, 2024. URL https://arxiv.org/abs/2303.05499.

[79] Y. Chen, B. Xiao, and H. Wang. Foldnet: Learning generalizable closed-loop policy for garment folding via keypoint-driven asset and demonstration synthesis, 2025. URL https://arxiv.org/abs/2505.09109.

[80] Introduction to torch.compile — PyTorch tutorials 2.7.0+cu126 documentation, 2023. URL https://pytorch.org/tutorials/intermediate/torch_compile_tutorial.html.

[81] H. Walke, K. Black, A. Lee, M. J. Kim, M. Du, C. Zheng, T. Zhao, P. Hansen-Estruch, Q. Vuong, A. He, V. Myers, K. Fang, C. Finn, and S. Levine. Bridgedata v2: A dataset for robot learning at scale, 2024. URL https://arxiv.org/abs/2308.12952.

[82] AgiBot-World-Contributors, Q. Bu, J. Cai, L. Chen, X. Cui, Y. Ding, S. Feng, S. Gao, X. He, X. Hu, X. Huang, S. Jiang, Y. Jiang, C. Jing, H. Li, J. Li, C. Liu, Y. Liu, Y. Lu, J. Luo, P. Luo, Y. Mu, Y. Niu, Y. Pan, J. Pang, Y. Qiao, G. Ren, C. Ruan, J. Shan, Y. Shen, C. Shi, M. Shi, M. Shi, C. Sima, J. Song, H. Wang, W. Wang, D. Wei, C. Xie, G. Xu, J. Yan, C. Yang, L. Yang, S. Yang, M. Yao, J. Zeng, C. Zhang, Q. Zhang, B. Zhao, C. Zhao, J. Zhao, and J. Zhu. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

[83] S. Valette and J.-M. Chassery. Approximated centroidal voronoi diagrams for uniform polygonal mesh coarsening. In *Computer Graphics Forum*, volume 23, pages 381–389. Wiley Online Library, 2004.

[84] google-deepmind/envlogger, Jan. 2025. URL https://github.com/google-deepmind/envlogger. original-date: 2021-07-28T15:35:08Z.

[85] Universally unique identifier, Jan. 2025. URL https://en.wikipedia.org/w/index.php?title=Universally_unique_identifier&oldid=1272340425. Page Version ID: 1272340425.

[86] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi:10.15607/RSS.2023.XIX.016.

[87] F. R. GmbH. Ros integration for franka robotics research robots. https://github.com/frankaemika/franka_ros, 2023.

[88] J. Luo, Z. Hu, C. Xu, Y. L. Tan, J. Berg, A. Sharma, S. Schaal, C. Finn, A. Gupta, and S. Levine. Serl: A software suite for sample-efficient robotic reinforcement learning, 2024.

A Overview

In the supplementary materials, we provide details about SynGrasp-1B dataset in Section B and C. We also show that GraspVLA supports fast adaptation to new robotic arms and camera configurations in Section D. We provide details about our main experiments in Section E. We also provide additional scaling law experiments in Section F. Details about LIBERO benchmark is provided in Section G. Comprehensive comparison with AnyGrasp is provided in Section H. We ablate the number of camera views in Section I. We also analyze the sim2real gap in Section J. We provide details about the inference delay in Section K. The implementation details of our non-blocking controller is provided in Section L. We also provide details about the failure cases in Section M.

B Details about SynGrasp-1B

We present the statistics of our synthetic dataset, SynGrasp-1B, in Table 6. The dataset consists of 10 million trajectories, each containing approximately 100 frames, resulting in a total of 1 billion frames—a substantial increase in scale compared to existing open-source datasets. Our dataset encompasses a diverse array of object categories, featuring 10,680 objects across 240 categories. While real-world datasets often encounter challenges related to scene diversity and require collaboration among laboratories across different countries, synthetic data generation enables us to easily create varied scenes by altering the textures of the table, ground, and walls. We utilize around 1,000 different textures for the table and 1,200 for the ground and walls, leading to a total of 1 million unique scenes.

Unlike existing datasets, SynGrasp-1B is the first to offer precise and fine-grained annotations for camera calibration, bounding box annotations, and the 3D poses of both the target object and the gripper. Thanks to our simulation engine, we can effortlessly obtain these annotations and incorporate additional types, such as depth maps and segmentation masks, when necessary. This flexibility is a significant advantage of synthetic datasets over their real-world counterparts.

	Trajectories	Objects	Scenes	Camera Calibration	Bbox Annotation	3D Pose Annotation
RoboSet [16]	98k	< 200	11	✗	✗	✗
BridgeData V2 [81]	60k	100	24	✗	✗	✗
RT-1 [15]	130k	< 200	2	✗	✗	✗
DROID [10]	76k	< 200	2080	✓	✗	✗
AgiBot World [82]	1M	3k	106	✓	✗	✗
Open-X Embodiment [9]	1.4M	-	311	✗	✗	✗
SynGrasp-1B	10M	10k	10M	✓	✓	✓

Table 6: **Comparison of SynGrasp-1B with Existing Datasets for Robot Manipulation.** This table highlights the advantages of SynGrasp-1B in terms of scale, annotations, and scene diversity compared to other datasets.

The generation of simulation data is also much more cost-effective than real-world data collection, considering factors such as time, financial resources, space, robotic equipment, and human labor:

- **Time:** A single human operator can collect only around 1,000 trajectories per day. In contrast, we can generate 10 million trajectories in 10 days using 160 NVIDIA 4090 GPUs. This efficiency accelerates the data-model feedback loop, enabling rapid model iteration and performance improvements.
- **Space:** Real-world data collection often necessitates multiple laboratories across different countries to enhance scene diversity. Additionally, it requires significant physical space to accommodate robots and objects; for example, the AgiBot World dataset [82] utilizes a 4,000 m^2 area for data collection. In contrast, our synthetic approach does not require any physical space.
- **Robots:** Each human operator in real-world collection needs a physical robot, resulting in high costs and maintenance overhead.
- **Money:** The total cost of generating SynGrasp-1B is around $5,000—orders of magnitude cheaper than real-world alternatives.

As an initial step toward large-scale synthetic action pre-training, we focus on grasping tasks to enable detailed analysis. Our pipeline can be extended to other robotic arms (for cross-embodiment transfer) or tasks (e.g., placing, pushing, stacking), as well as large-scale camera randomization. We leave these extensions to future work.

Gallery. We randomly sample 24 trajectories from our SynGrasp-1B and visualize them in Fig. 7. Each trajectory consists of around 100 frames, and we uniformly sample 4 frames from each trajectory for visualization.

C Details about Data Generation

Object Processing and Layout Generation. To ensure that the scales of synthetic objects align with real-world counterparts and are suitable for grasping, we manually define minimum and maximum size constraints for each category. This helps our model to generalize to real-world objects with diverse scales. As shown in Fig. 8, GraspVLA can grasps all scales of dog, ranging from 2cm to 35cm. Furthermore, we simplify the object meshes using the ACVD algorithm [83] to improve the simulation efficiency. Additionally, we randomize the height of the table, ranging from -0.1 m to 0.2 m in the robot frame.

To enhance data diversity, we create different clutter layouts for each episode by randomly placing objects within a 0.4m by 0.5m area on a table. Objects are dropped in various poses to generate physically plausible scenes. For categories requiring specific orientations, such as cups, we manually define valid poses (e.g., upright).

The cameras are randomized within a 15 cm radius ball and rotated ±5° around each axis. Details are shown in Table 7.

Table 7: Camera parameters

	Position	Lookat
Front Camera	x=1.35, y=0.0, z=0.54	x=0.2, y=0.0, z=0.0
Side Camera	x=0.5, y=0.69, z=0.50	x=0.5, y=0.0, z=0.1

Asynchronous and Grouped Data Writing. We employ DeepMind EnvLogger with TFDS backend [84] for the storage of our synthetic data. Despite its clean design, considerations for high-performance large-scale simulation are necessary. Firstly, to mask the time cost of image encoding and data writing, we modified the EnvLogger implementation to perform the actual data writing operation asynchronously. Second, to avoid contention on the dataset metadata across parallel processes and to minimize data loss caused by unforeseen errors (e.g., GPU failures), the processes should not write to a single shared folder. However, if each simulation instance utilizes a unique subfolder, it results in a large number of subfolders and metadata, leading to substantial overheads in data management, transfer, and loading. As a compromise, we assign each process a subfolder with random UUIDs [85], and write all the trajectories within a process to the same subfolder.

Handling Data Corruption. During billion-scale data generation, a simulation process could hang and get killed due to hardware faults or excessive memory consumption, resulting in file corruption or loss. We handle these issues by proactively managing exceptions when loading the dataset with TFDS. Upon encountering a NotFoundError, we create an empty file at the expected file path, otherwise TFDS cannot continue loading the remaining records within the subfolder. On DataLossError, we count it as one missing record, and log the missing rate as a critical statistics for data validity. The missing rate was below 1%. FailedPreconditionError is raised when the successfully loaded number of records is smaller than that in the metadata of the subfolder, and can be safely converted to a StopIteration to facilitate the correct functioning of the data loader. The above handling of these exceptions ensures loading all the valid records. We believe these insights will benefit future large-scale dataset efforts.

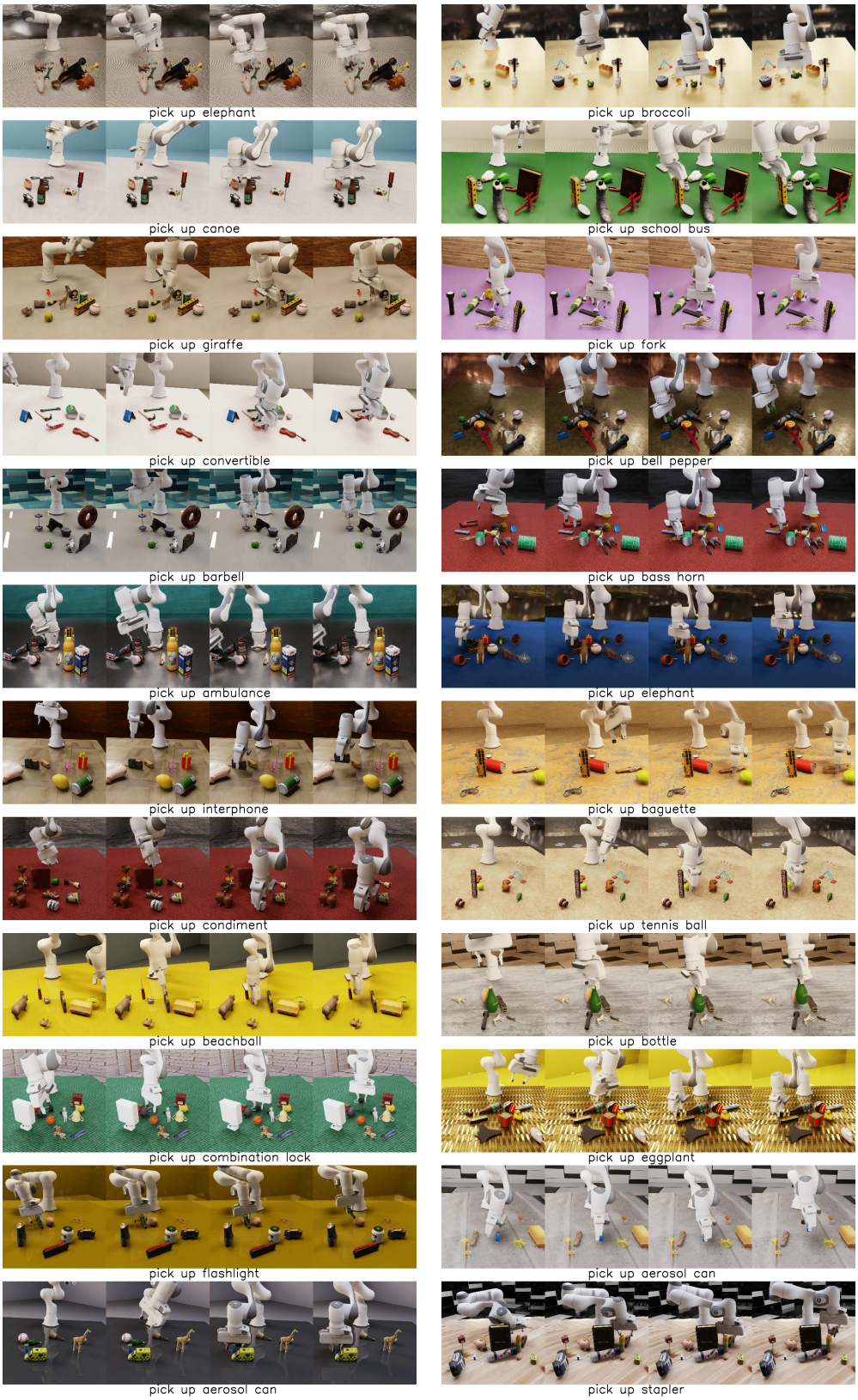


Figure 7: **Gallery of SynGrasp-1B.** 24 randomly sampled trajectories from our synthetic grasping data. For clarity, 4 frames are uniformly sampled from each trajectory for display.

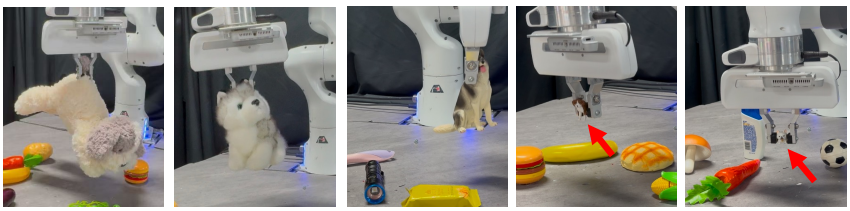


Figure 8: GraspVLA handles object with diverse scales, from 2cm to 35cm.

D Fast Adaptation to New Robotic Arm and Camera

While we focus on specific setups for in-depth analysis, GraspVLA is not specialized for this. Our model can easily adapt to new robotic arms, grippers, and camera configurations with 5k additional synthetic trajectories (generated in one day on an NVIDIA 4090 GPU). We provide the following two examples:

- **New robotic arm and gripper.** We use a UR5e arm with a Robotiq 2F-85 gripper. Since no real-world hardware is available, we test our model on the simulation environment.
- **New camera configuration.** We use front view and wrist view cameras. We conduct real-world experiments by mounting the camera on the wrist of the Franka Panda arm.

As shown in Fig. 9 and Table 8, our model shows strong performance with minimal fine-tuning, enabling rapid deployment on new setups.

Figure 9: GraspVLA supports fast adaptation to new robotic arms and camera configurations.

	Wrist camera	UR5e arm with Robotiq gripper
Success Rate	76.6	82.1

Table 8: Success rate of GraspVLA on new robotic arms and camera configurations.

E Details of Main Experiments

Metrics. In each trial, the model is allowed to attempt to grasp up to three times, with each attempt counted by the gripper closure action. Success is strictly defined as the specified object being lifted a minimum of 15 cm. The scene is not reset during each trial, even if the model knocks the object off the table.

Additionally, we introduce Success weighted by Path Length (SPL) to further account for the number of actions taken, which is a common metric in discrete navigation tasks to evaluate the efficiency of the model. While for discrete navigation tasks, the shortest path is easy to define, for grasping tasks, the shortest path is not well defined. Therefore, for each trial, if there are several methods that can successfully grasp the object, we define the shortest path as the one with the least number of action steps. If all methods fail to grasp the object, they all get zero SPL in this trial. Note that, our SynGrasp-1B dataset stores actions in 10 Hz and all methods are trained on this dataset, so the number of action steps is comparable across methods.

Baselines. As Diffusion Policy does not support language conditioning, we train it using the subset of SynGrasp-1B that grasps the elephant (around 40k trajectories), and replace the target object with

Table 9: Hyperparameters for all the methods

Baseline	Hyperparameter	Value
GraspVLA	batch_size	384
	learning_rate	1.6e-4
π_0	batch_size	256
	learning_rate	cosine schedule
	warmup_steps	1000
	peak_lr	2.5e-5
	decay_lr	2.5e-6
	decay_steps	30000
OpenVLA	lora_rank	32
	batch_size	12
	learning_rate	5e-4
	image_aug	true
Octo	batch_size	256
	learning_rate	rsqrt schedule
	warmup_steps	2000
	init_value	0.0
	peak_value	3e-4
Diffusion Policy	batch_size	256
	learning_rate	cosine schedule
	warmup_steps	500
	peak_lr	1e-4
	weight_decay	1e-6

the elephant in the real-world experiments. We train all other models with the full SynGrasp-1B dataset. We train π_0 [7] with its pre-trained weights initialization and PaliGemma initialization for comparison. Since OpenVLA [6] takes a single RGB image for visual observation, we use the front camera view for it. For Diffusion Policy [75], we train the UNet-based version as recommended by the original paper. For Octo [26], we finetune the pre-trained *octo-base-1.5* model. All the models are trained with action chunks of 4 [86], except for OpenVLA, which does not support action chunking. We provide hyperparameters of training/finetuning GraspVLA and baselines in Table 9. We run automatic evaluation in our simulation pipeline for all the baselines, continue training until the success rate converges, and select the best-performing checkpoint in the real world. We found GraspVLA achieves a consistently high success rate after 120k steps.

Real world setup and modification to robot finger. For perception, we employ an Intel RealSense D435 as the front-facing camera and a D415i as the side-facing camera. Both cameras are positioned at the center of the randomization range used in the synthetic data generation. The workspace for test objects is confined to a 40 cm x 50 cm x 20 cm area in front of the robot.

The original Franka Panda finger is too short to firmly grasp convex-shaped objects (e.g., a bottle lying on its side). This is because the hand plank collides with the top of the object, preventing the fingers from reaching deep enough to secure a stable grip. To address this issue, we extended the fingers by 2 cm in both synthetic data generation and real-world experiments.

Details about PAG-3D. For steps before the gripper closure, we use the open-loop grasp pose of this trajectory as supervision. For steps after the gripper closure, we use the next step's end-effector pose as supervision.

F Detailed Scaling Law

Simulation evaluation. While the main paper analyzes the scaling law in real-world, we extend this analysis to simulation environments. Our results show that simulation is an effective proxy for

predicting real-world performance. For these experiments, we use simulation environments with identical camera and table configurations to our data generation setup, but employ different object instances and materials to assess generalizability.

As shown in Fig. 10a), GraspVLA's performance on simulation data follows a scaling trend similar to that of real-world data, confirming the simulation's effectiveness for predicting real-world performance. However, we observe two key differences: (1) real-world performance scales more slowly (0.12B) compared to simulation, where performance saturates earlier, and (2) the sim-to-real gap decreases with more training frames, suggesting that larger datasets enable more robust representations and better transfer to real-world scenarios.

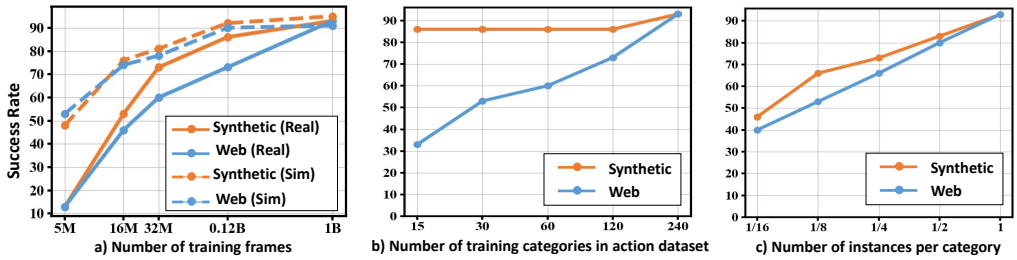


Figure 10: **Scaling laws different training regimes.** (a) Performance scaling with number of training frames in both simulation and real-world environments. (b) Impact of training category diversity while fixing instances per category. (c) Effect of varying instances per category while maintaining total category count.

We further investigate how data diversity affects GraspVLA's performance by analyzing two additional scaling factors: (1) the number of training categories and (2) the number of instances per category. For each analysis, we hold the other factor and total training frames constant.

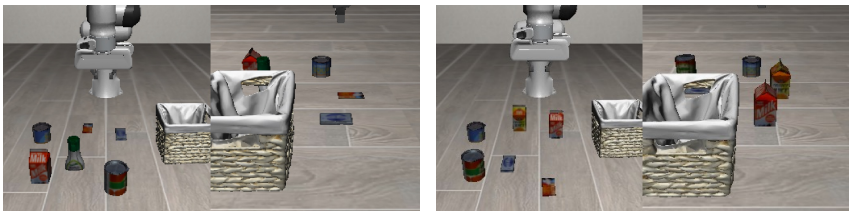
Number of Training Categories (Fig. 10b). When varying the number of categories while fixing instances per category and total frames, performance on web categories improves steadily with more training categories, whereas performance on synthetic categories saturates early. This implies that inter-category generalization (adapting to unseen categories) benefits significantly from broader categorical coverage, while intra-category generalization (recognizing diverse instances of known categories) requires less diversity.

Number of Instances per Category (Fig. 10c). With a fixed category count and total frames, increasing instances per category leads to consistent improvements across both synthetic and web categories. This underscores the importance of instance diversity within categories for robust generalization.

G Details about Experiments on LIBERO Benchmark

Setup. We consider a trial successful if the robot successfully grasps and lifts the target object to a height of 10 cm. Since our model is trained with two camera views, we modify the original camera configurations provided by the LIBERO benchmark to match our training setup, aligning the camera poses accordingly. As the basket in some tasks occludes the side view severely, we remove it. Additionally, we extend the gripper by 2 cm, as detailed in the robot finger modification in Section E. These adjustments are made exclusively for evaluating our model to ensure they do not affect the fine-tuned baselines, which are evaluated using the original camera configurations and gripper length.

The LIBERO-object test set presents a significant challenge for zero-shot models due to ambiguous target object descriptions. As illustrated in Figure 11, even humans may struggle to identify objects like "alphabet soup" and "cream cheese" in the given scene. To account for this, we relax the success criteria: a trial is deemed successful if the robot grasps any object belonging to the same category as the target. For instance, if the target is "alphabet soup," grasping any object in the "can" category



a) Pick up alphabet soup and place it in the basket b) Pick up cream cheese and place it in the basket

Figure 11: **Examples of LIBERO Benchmark.** We visualize both front and side views side by side.

is considered a success. Similarly, if the target is "cream cheese," grasping any object in the "box" category qualifies as success.

As noted in the main paper, we exclude non-prehensile tasks to focus solely on grasping capability. We also omit tasks requiring color-based distinctions (e.g., "pick up the yellow and white mug"), as reasoning about color falls outside our scope. The specific tasks deemed invalid in the original test set, along with the modified instructions, are detailed in Tables 10, 11, and 12.

Table 10: **Modification to LIBERO-Goal test set.**

Original Caption	Valid	Modified Caption
put the wine bottle on top of the cabinet	✓	pick up the wine bottle
open the top drawer and put the bowl inside	✓	pick up the bowl
turn on the stove	x	-
put the bowl on top of the cabinet	✓	pick up the bowl
put the bowl on the plate	✓	pick up the bowl
put the wine bottle on the rack	✓	pick up the wine bottle
put the cream cheese in the bowl	✓	pick up the cream cheese box
open the middle drawer of the cabinet	x	-
push the plate to the front of the stove	x	-
put the bowl on the stove	✓	pick up the bowl

Table 11: **Modification to LIBERO-Object test set.**

Original Caption	Valid	Modified Caption
pick up the alphabet soup and place it in the basket	✓	pick up the alphabet soup can
pick up the cream cheese and place it in the basket	✓	pick up the cream cheese box
pick up the milk and place it in the basket	✓	pick up the milk
pick up the tomato sauce and place it in the basket	✓	pick up the tomato sauce can
pick up the butter and place it in the basket	✓	pick up the butter box
pick up the orange juice and place it in the basket	✓	pick up the orange juice
pick up the chocolate pudding and place it in the basket	✓	pick up the chocolate pudding box
pick up the bbq sauce and place it in the basket	✓	pick up the bbq sauce bottle
pick up the ketchup and place it in the basket	✓	pick up the ketchup bottle
pick up the salad dressing and place it in the basket	✓	pick up the salad dressing bottle

Baselines. For baseline models, OpenVLA and π_0 [6, 7], we use the authors' official fine-tuned checkpoints. Both models are fine-tuned on the LIBERO demonstration dataset, processed by OpenVLA to exclude static frames and failure trajectories, and rendered in high resolution.

Impact of instruction format. While the original test sets in LEBERO mainly compose of two steps, picking up an object and placing it in a container, we focus on the first step to exclusively evaluate the grasping capabilities. Therefore, to ensure a fair comparison, we also simplify the original instruction format "pick up a object and place it in a container" to "pick up a object" for the same instructions across all models. Note that, the instructions in the fine-tuning set are not simplified due to difficulties in segmenting and removing actions related to placing objects in containers.

As shown in Table 13, the performance of both fine-tuned baselines drops significantly when the instruction format is simplified. This indicates that the models are not robust to instruction variations

Table 12: **Modification to LIBERO-Long test set.**

Original Caption	Valid	Modified Caption
turn on the stove and put the moka pot on it	✓	pick up the moka pot
put the black bowl in the bottom drawer of the cabinet and close it	✓	pick up the black bowl
put the yellow and white mug in the microwave and close it	x	-
put both moka pots on the stove	✓	pick up the moka pot
put both the alphabet soup and the cream cheese box in the basket	✓	pick up the alphabet soup can
put both the alphabet soup and the tomato sauce in the basket	✓	pick up the alphabet soup can
put both the cream cheese box and the butter in the basket	✓	pick up the cream cheese box
put the white mug on the left plate and put the yellow and white mug on the right plate	x	-
put the white mug on the plate and put the chocolate pudding to the right of the plate	x	-
pick up the book and place it in the back compartment of the caddy	✓	pick up the book

and are sensitive to the specific instruction format. Additionally, our zero-shot model, GraspVLA, outperforms the fine-tuned models in the simplified instruction format and achieves comparable performance in the original instruction format. This demonstrates the robustness of our model to generalize to unseen environments, even in the absence of fine-tuning.

	Long	**Goal**	**Object**
Format: pick up {object} and place it in {container}			
OpenVLA (fine-tuned)	*70.9*	*78.6*	*91.2*
π_0 (fine-tuned)	***88.7***	***95.4***	***98.4***
Format: pick up {object}			
OpenVLA (fine-tuned)	33.7	56.6	65.4
π_0 (fine-tuned)	62.7	79.4	93.8
Ours (zero-shot)	**82.0**	**91.2**	**94.1**

Table 13: **Impact of instruction format.** Fine-tuned baselines exhibit performance drops when the original instructions are simplified.

H Details about Comparison with AnyGrasp

Setup. To ensure a fair comparison, we run the AnyGrasp baseline with up to three attempts per trial, counting it as a success if the object is grasped in any attempt. The baseline is implemented using the authors' official SDK. For perception, we use the same Franka Emika Panda robot and a RealSense D435i camera mounted on the end-effector, with the camera calibrated for accurate depth perception. Inference speed is evaluated on an NVIDIA RTX 3090 GPU.

For the language-conditioned test set, we integrate Grounding DINO [78] to parse language instructions into bounding boxes. Grasp candidates whose 2D projections fall outside these boxes are filtered out. Given the sparse layout, this simple approach effectively eliminates irrelevant grasps. Motion planning is then used to generate trajectories for execution.

Test Sets. The language-driven task uses the same test set as in the main experiment (Table 1 in the main paper), comprising both synthetic and web categories for a total of 60 trials. For arbitrary grasping of common objects, we randomly select 30 objects (15 synthetic, 15 web), ensuring diffuse, non-reflective materials (e.g., rubber, wood). The transparent object test set consists of 5 objects, including 3 bottles, 1 cup, and 1 bowl. To focus on grasping transparent objects, we remove distractors from the scene and place the transparent objects at 6 different poses on the table, resulting in 6 trials per object. We visualize transparent objects in Figure 12.

Figure 12: **Transparent objects used for evaluation.**

Analysis. In the language-conditioned test set, the baseline fails in 5 trials. Three failures stem from incorrect bounding box predictions by Grounding DINO, largely due to ambiguities in the top-down monocular view—for example, a toy ambulance being misidentified as a charger. The remaining two failures involve flat objects (a metal fork and a plastic spoon), where the point clouds merge with the table surface, rendering the objects indistinguishable even to human observers. Transparent objects pose a similar challenge, as missing depth information leads to point-cloud-based grasping failures. However, since RGB images reliably capture these objects, our RGB-based model overcomes these limitations and succeeds where the baseline fails.

Overall, AnyGrasp and our method provide complementary solutions. AnyGrasp is a fast grasping detection model and adapting it to open-vocabulary grasping requires extra modules (segmentation, motion planner, failure recovery)—each introducing potential failures. For instance, collision-free path planning often fails in cluttered scenes like our post-train Task 3. In contrast, our model is **end-to-end**, **closed-loop**, and **easily adapts to specialized tasks** (e.g., grasping in specific poses) without requiring task-specific modules. Besides, AnyGrasp uses depth as input, which suffers from incomplete and noisy issues for transparent materials. In contrast, our model relies solely on RGBs, bypassing this issue.

I Ablation of Camera Views

π_0 natively supports multi-camera, so we fine-tune it with the same front and side views as our model to ensure fair comparison. However, OpenVLA only supports single view, so we ablate the number of views here and show that our single-view version outperforms OpenVLA by 40%.

Impact of the Number of Input Views. To ensure a fair comparison, we use only front-view images as input for our method, consistent with the single-view baseline OpenVLA. As shown in Table 14, this constraint results in approximately 30% lower performance compared to our multi-view approach. Nevertheless, our model still achieves 40% higher performance than OpenVLA, demonstrating the effectiveness of our design.

Model	Synthetic	Web
OpenVLA (single-view)	20.0	3.3
Ours (single-view)	60.0	56.6
Ours (multi-view)	**93.3**	**93.3**

Table 14: **Impact of number of input views.** Comparison of GraspVLA with different numbers of input views. The results demonstrate that while multiple views significantly improve performance, our single-view implementation still outperforms the OpenVLA baseline by 40%.

J Mitigation of Sim-to-Real Gap

In this section, we examine the sim-to-real gap in the context of training a VLA model for grasping using imitation learning. The sim-to-real gap primarily appears in two key areas: visual appearance and physical dynamics.

Visual appearance. Thanks to advances in pre-trained vision encoders and ray-traced rendering, the visual discrepancy between synthetic and real-world RGB images has significantly narrowed. By leveraging diverse material and texture datasets, we can generate realistic scenes that cover a wide range of robotic grasping scenarios—far more efficiently than collecting equivalent real-world data across varied environments (as discussed in B). Even when certain material or texture combinations appear unrealistic (e.g., a red table against a green wall), the model still learns generalizable representations from such diversity, consistent with findings in [47]. Additionally, co-training with large-scale Internet vision-language datasets further enhances the model's robustness to visual discrepancies [36].

Physical dynamics. The sim-to-real gap in physical dynamics arises mainly from inaccuracies in modeling material properties (e.g., surface friction), contact dynamics (e.g., forces, friction, deformations), and actuator/sensor behavior. In this work, we mitigate this gap through three key design choices:

- **Simplified control.** We use positional control and treat gripper actions as discrete open/close commands, avoiding complex dynamics modeling.
- **Stability filtering.** We only keep grasps that forms force-closure under low friction coefficient (0.15), ensuring the model prioritizes robust strategies.
- **Geometry-driven planning.** We focus on mesh-based grasp poses rather than dynamics-dependent policies, enhancing robustness to physical variations.

While these strategies effectively reduce the sim-to-real gap for grasping, they may not generalize to tasks requiring fine-grained dynamics understanding, such as non-prehensile manipulation. We leave the investigation of such scenarios to future work.

K Inference Delay

The combination of autoregression and flow matching in GraspVLA introduces additional inference delay. Based on Section 5.6 and Table 15, while PAG is critical for a high grasp success rate, it contributes to $\sim 63\%$ inference delay due to 14 additional tokens to generate. We leave the further improvement of the inference efficiency with PAG as future work. We additionally found that the prefill stage has a similar delay as the decode stage, which could be due to a low GPU utilization with single-sample inference.

component	**inference time (ms)**
vision encoder	9
bounding boxes (8 tokens)	72
grasp pose (6 tokens)	50
flow matching	64

Table 15: **Breakdown of inference time on NVIDIA L40s GPU.**

L Non-Blocking Control

We explore the implementation of non-blocking controller for smooth action. We implement a Cartesian-space impedance controller adapted from Franka ROS [87] and SERL Franka Controllers [88]. The architecture converts Cartesian impedance commands into joint-space control via real-time Jacobian-based transformation with singularity handling, while optimizing impedance parameters through system identification.

To mitigate abrupt target transitions, we evaluated multiple filter implementations and selected a cascaded filter design for its superior smoothing performance (Figure 13). It achieves fast convergence without overshoot while avoiding excessive initial acceleration, which is suitable for the output characteristics of the GraspVLA model. Additionally, positional interpolation was adopted instead of temporal interpolation to address synchronization mismatches between model computation latency and control pipelines.

To achieve more fluent and coherent motions, GraspVLA generates multi-step predictions at each inference cycle. These predictions are incorporated via a receding-horizon optimization scheme within the asynchronous control architecture, where filter and interpolation strategies are systematically applied to the predicted trajectory. This non-blocking control architecture proactively compensates for computational latency variations while ensuring smooth interleaving of control actions, which significantly mitigates oscillatory patterns in dynamic manipulation scenarios.

While we employ non-blocking control for demonstration recording to achieve natural trajectories, all experimental evaluations use blocking control to ensure rigorous performance measurement.

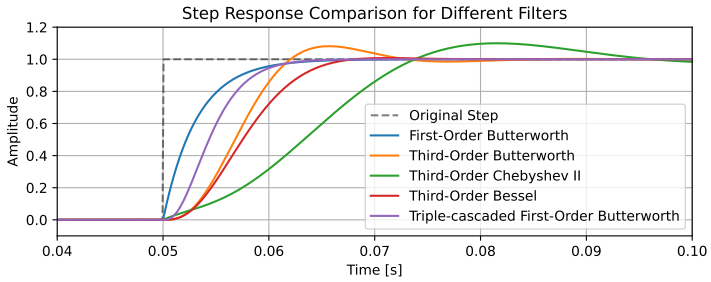


Figure 13: **Step response comparison for different filters**. In the comparison, these filters set the same sampling frequency and cutoff frequency. The first-order Butterworth filter exhibits a large initial acceleration in its step response. The third-order Butterworth filter, third-order Chebyshev II filter, and third-order Bessel filter all exhibit overshoot to varying degrees. The triple-cascaded first-order Butterworth filter can avoid excessive initial acceleration and eliminate overshoot while maintaining convergence speed, making it an ideal filter choice.

M Failure Analysis

To thoroughly assess the limitations of our approach, we conduct a detailed failure analysis. Since the test set in the main paper reveals only two failure cases—which may not be representative—we design a significantly more challenging test set featuring cluttered scenes. Specifically, we randomly place objects across the table to cover the entire workspace and stack some objects (e.g., placing a strawberry on top of a bulldozer) to create complex, occluded scenarios. We then evaluate the model's performance under these conditions and identify the primary failure modes.

The most frequent failure case (31%) occurs when the model hesitates due to ambiguous language instructions, such as when multiple objects match the description (e.g., two target bottles). This could be mitigated by incorporating longer contextual history. The second most common issue (27%) arises in highly cluttered scenes, where the model misidentifies objects, likely due to insufficient training data for such scenarios. Future work could utilize advanced data augmentation methods and generative modeling techniques to create more diverse and complex training samples. Another notable failure mode (21%) involves objects with smooth surfaces (e.g., plastic balls) slipping during grasping, which tactile feedback might help resolve. Additionally, when the target object is occluded (14%), the model struggles to grasp it precisely, suggesting a need for active perception techniques. Finally, the remaining failures (7%) include minor errors such as early gripper closure or collisions with the environment, which reinforcement learning could potentially address. We leave these potential improvements for future work.