DexGraspVLA: A Vision-Language-Action Framework Towards General Dexterous Grasping

Yifan Zhong[1,2*], **Xuchuan Huang**[1,2*], **Ruochong Li**[2,3], **Ceyao Zhang**[1,2], **Yitao Liang**[1,2], **Yaodong Yang**[1,2†], **Yuanpei Chen**[1,2†]

[1]Institute for AI, Peking University, [2]PKU-PsiBot Joint Lab, [3]Hong Kong University of Science and Technology (Guangzhou)

Figure 1 | We propose **DexGraspVLA**, a hierarchical vision-language-action framework that reaches a 90+% dexterous grasping success rate under thousands of unseen object, lighting, and background combinations in a "zero-shot" real-world environment.

Abstract

Dexterous grasping remains a fundamental yet challenging problem in robotics. A general-purpose robot must be capable of grasping diverse objects in arbitrary scenarios. However, existing research typically relies on specific assumptions, such as single-object settings or limited environments, leading to constrained generalization. Our solution is DexGraspVLA, a hierarchical framework that utilizes a pre-trained *V*ision-*L*anguage model as the high-level task planner and learns a diffusion-based policy as the low-level *A*ction controller. The key insight lies in iteratively transforming diverse language and visual inputs into domain-invariant representations, where imitation learning can be effectively applied due to the alleviation of domain shift. Thus, it enables robust generalization across a wide range of real-world scenarios. Notably, our method achieves a 90+% success rate under thousands of unseen object, lighting, and background combinations in a "zero-shot" environment. Empirical analysis further confirms the consistency of internal model behavior across environmental variations, thereby validating our design and explaining its generalization performance. We hope our work can be a step forward in achieving general dexterous grasping. Our demo and code can be found at https://dexgraspvla.github.io/.

1. Introduction

Dexterous multi-fingered hands, as versatile robotic end-effectors, have demonstrated remarkable capabilities across various manipulation tasks [1, 2, 3, 4, 5, 6, 7, 8, 9]. Among these capabilities, grasping serves as the most fundamental prerequisite, yet remains one of the most challenging problems. Existing dexterous grasping approaches are primarily evaluated on isolated objects or under simplified settings. Nevertheless, real-world applications demand more gen-

*Equal contribution. †Corresponding author(s): Yuanpei Chen{yuanpei.chen312@gmail.com},Yaodong Yang{yaodong.yang@pku.edu.cn}.

eral grasping capabilities that can function reliably in diverse scenarios such as industrial manufacturing and household environments. However, developing general dexterous grasping capabilities presents multi-faceted challenges. At the object level, the policy must generalize across diverse physical properties including geometries, masses, textures, and orientations. Beyond object characteristics, the system must also demonstrate robustness to various environmental factors, such as lighting conditions, background complexities, and potential disturbances. Compounding these challenges, multi-object scenarios introduce additional complexity that demands sophisticated reasoning capabilities. For instance, in cluttered or stacked environments, planning the optimal sequence to grasp all objects becomes a crucial cognitive task that extends beyond simple grasp execution.

Traditional approaches in dexterous grasping follow a two-stage pipeline: first predicting target grasp pose from single-frame perception, then executing open-loop motion planning to reach the pose [10, 11, 12]. However, such methods are heavily constrained by precise camera calibration and mechanical accuracy requirements. End-to-end approaches like imitation learning and reinforcement learning, enable closed-loop grasping by continuously adjusting actions based on real-time perceptual feedback, offering more robust and adaptive solutions. Recent years have witnessed remarkable progress in applying reinforcement learning to robotic systems [13, 14, 15, 16]. Leveraging large-scale parallel simulation, reinforcement learning enables robots to undergo extensive training in simulation and then deploy the learned policies to the real-world. Despite such progress, the complexity of real-world physical parameters presents significant challenges in simulation modeling, leading to an inevitable sim-to-real gap. Meanwhile, researchers have explored imitation learning approaches to learn manipulation skills [17, 18, 19]. These methods collect human demonstration data through teleoperation and directly learn the mapping from raw perceptual input to robot control commands using supervised learning. Nevertheless, such approaches often struggle with generalization beyond the demonstration data. While general grasping requires handling diverse objects and environments, collecting demonstrations for all situations is impractical. Thus, the key challenge lies in how to efficiently utilize the demonstration data to achieve broader generalization.

The rapid emergence of vision and language foundation models [20, 21, 22, 23, 24] presents promising opportunities for robotic manipulation. Leveraging vast amounts of internet-scale data in pre-training, these models demonstrate remarkable scene understanding and generalization capabilities for visual and linguistic inputs. While it appears intuitive to directly task these models with generating robotic control commands [25, 26], this straightforward strategy faces fundamental limitations. The absence of physical interaction data in their training process results in models with limited spatial intelligence. An alternative approach integrates vision-language models (VLMs) into robotic control policies, training them in an end-to-end manner [27, 28]. However, this paradigm typically demands an enormous volume of manually collected demonstrations [29, 30] in an attempt to encompass the full range of real-world diversity and complexity. Even so, these models exhibit markedly reduced performance on unseen scenarios and still require further data collection and fine-tuning to handle new conditions. In addition, the substantial disparity between robotics datasets and the massive pre-training corpora leads to catastrophic forgetting, compromising the model's valuable long-range reasoning capabilities. Effectively utilizing foundation models' world knowledge to enhance generalization in robotic policies remains challenging.

In this paper, we present **DexGraspVLA**, the first hierarchical Vision-Language-Action (VLA) framework for general dexterous grasping that integrates the complementary strengths of foundation models and imitation learning. At the high level, it utilizes a pre-trained VLM as a task planner, which interprets and reasons about language instructions, plans the overall grasping task, and provides supervisory signals. Guided by these signals and multimodal inputs, a low-level diffusion-based modularized controller produces closed-loop action sequences. The essence of DexGraspVLA lies in leveraging foundation models to iteratively transform diverse vision and language inputs into domain-invariant representations, where it then efficiently and effectively applies diffusion-based imitation learning to capture the action distribution in our dexterous grasping dataset. As a result, novel scenarios outside the training set no longer induce failures, because the foundation models translate them into representations resembling those encountered during training — thus remaining within the learned policy's domain. This approach fuses the extensive world knowledge of foundation models with the strong action modeling capacity of imitation learning, thereby enabling robust generalization performance in real-world applications.

Notably, DexGraspVLA achieves an unprecedented 90.8% success rate for grasping in cluttered scenes spanning 1,287 unseen object, lighting, and background combinations, all tested in a "zero-shot" environment. Systematic evaluations on a single-object grasping benchmark show that DexGraspVLA reaches a 98.6% aggregated success rate, outperforming exist-

ing baselines whose controller learns directly from raw visual inputs by at least 48%. Furthermore, our empirical analysis reveals that the internal representations and the attention maps within DexGraspVLA remain consistent across varying environments, thereby substantiating its framework design and explaining its performance. These results confirm that DexGraspVLA can learn effectively from a modest amount of single-domain human demonstrations while generalizing reliably to a broad range of real-world situations, marking a promising step toward general dexterous grasping.

2. Related work

2.1. Dexterous Grasping

Dexterous grasping typically falls into two categories: two-stage approaches and end-to-end methods. Two-stage approaches first generate a grasp pose and then control the dexterous hand targeting this pose. The main challenge is generating high-quality grasp poses based on visual observation. Current methods employ sample-based [31, 32], optimization-based [11, 12, 33, 34, 35, 36], or regression-based [37, 38] approaches to generate target grasp poses, followed by motion planning for robot execution. For instance, SpringGrasp [10] models uncertainties in partial observations using optimization-based methods to improve grasp pose generation quality. UGG [39] proposes a diffusion-based approach to unify the generation of grasping poses and object geometries. While these methods benefit from decoupled perception and control and simulation data generation, they typically suffer from the lack of closed-loop feedback and sensitivity to disturbances and calibration errors.

End-to-end methods directly model grasping trajectories using imitation learning or reinforcement learning. Recent works explored training dexterous manipulation using reinforcement learning in simulation environments and transfer to the real-world [40, 1, 41, 2, 42, 43, 13, 14, 15, 16, 44, 3, 4, 5, 6, 7, 45, 46, 47, 48]. DexVIP [49] and GRAFF [50] generate affordance hints using computer vision methods and use reinforcement learning to train policies based on these features. DextrAH-G [51] and DextrAH-RGB [52] demonstrate some generalization capabilities in the real-world through large-scale parallel training in simulation. However, this reliance on simulation inevitably introduces sim-to-real gaps, while direct training in the real-world suffers from poor sample efficiency. Recently, imitation learning using human demonstration has shown remarkable results in complex tasks [50, 53, 54, 17, 18, 19, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67]. These methods require human teleoperation to collect demonstration data and directly learn the distribution in the dataset. While being easier to train, this approach limits their generalization capabilities. SparseDFF [68] and Neural Attention Field [69] explore how to enhance generalization through 3D distilled feature fields.

2.2. Foundation Models for Robotics

Recent years have witnessed remarkable progress in foundation models pretrained on large-scale datasets. Vision foundation models [23, 24, 20, 70, 71] exhibit robust out-of-distribution generalization, while VLMs including GPT-4o [22] and Qwen2.5-VL [72] demonstrate sophisticated multimodal reasoning abilities. Leveraging these foundation models effectively has become a promising direction in robotics research. One prominent approach, exemplified by RT-X [30], OpenVLA [27], Pi0 [28], and more [73, 74, 75, 76, 77],involves directly fine-tuning VLMs on robotic data. However, this strategy demands a large volume of demonstrations spanning diverse real-world conditions to achieve generalization. Even the largest currently available robotic datasets [29, 30, 28] fall short of covering the full breadth of scenarios; models trained on them still struggle to match their performance on seen domains and typically require additional data collection and fine-tuning for new environments. Moreover, these models often sacrifice some of their advanced reasoning abilities due to the complexity of robotic manipulation tasks and the scarcity of specialized data. Another line of research, exemplified by VoxPoser [25] and Rekep [26], leverages VLMs to generate task-specific outputs, such as affordance maps or constraint points, that can then be integrated with conventional motion planning. While this hierarchical strategy generally preserves VLMs' inherent reasoning capacities, it relies on sufficiently robust low-level controllers to execute high-level commands, making the design of effective interfaces critical. Our work harnesses pre-trained foundation models to generate domain-invariant representations, which facilitates the learning of a dexterous grasping policy. By offloading much of the real-world complexity to the foundation models, we can significantly reduce the amount of demonstration data required and, at the same time, realize strong zero-shot generalization.

3. Problem Formulation

Our goal is to develop a vision-based control policy for language-guided dexterous grasping, formulated as a sequential decision-making problem. Initially, a language instruction l is given, *e.g.* "*grasp the toy*", to directly specify the target object. At each timestep t, the policy π receives a first-view image $\mathbf{I}_t^{\mathrm{w}} \in \mathbb{R}^{H\times W\times 3}$ from the wrist camera (H and W denote the height and

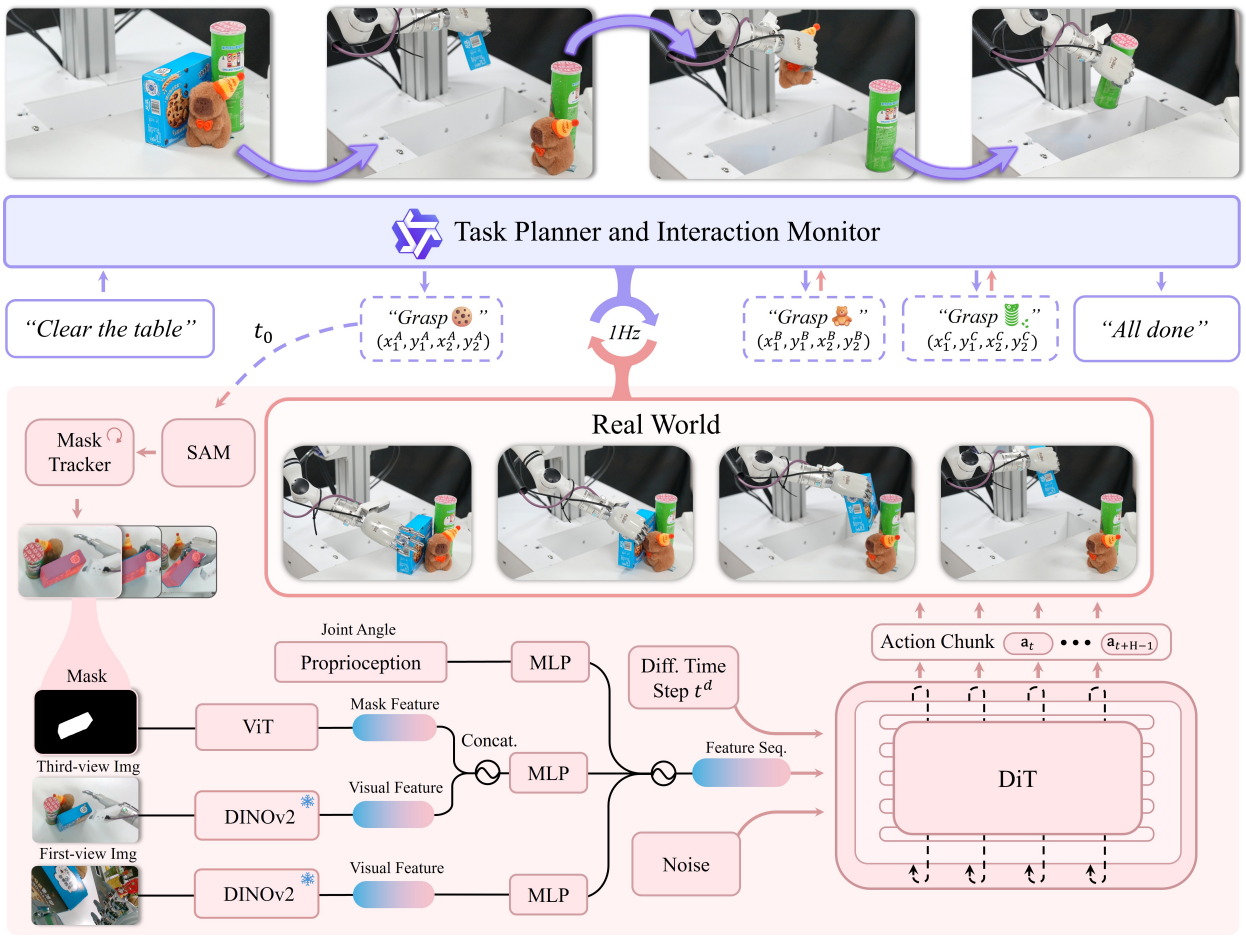

Figure 2 | **Overview of our framework.** DexGraspVLA adopts a hierarchical architecture composed of an off-the-shelf VLM-based high-level **planner** and a diffusion-based low-level **controller**. Given a cluttered scene, the **planner** reasons about the user prompt, *e.g.*, "*clear the table*", decomposing it into multiple grasping instructions when necessary. For each instruction l, *e.g.*, "*grasp the cookie*", the **planner** identifies the target object A from the head image $\mathbf{I}^{\mathrm{h}}_{t_0}$ and marks its bounding box $(x_1^A, y_1^A, x_2^A, y_2^A)$ at initial time t_0. The **controller** consists of four parts: (1) two segmentation models including SAM, which obtains the object's mask $\mathbf{m}_{t_0}$ at t_0, and Cutie, a video segmentation model that continuously tracks the mask $\mathbf{m}_t$ during each grasping process; (2) three vision encoders including two frozen DINOv2 that extract features from the third-view image $\mathbf{I}^{\mathrm{h}}_t$ and the first-view image $\mathbf{I}^{\mathrm{w}}_t$, and a trainable ViT that deals with the mask $\mathbf{m}_t$; (3) three MLP projectors that map the visual features and robot proprioceptive state into the same feature space, forming a feature sequence; and (4) a DiT that predicts an action chunk from $\mathbf{a}_t$ to $\mathbf{a}_{t+H-1}$. During the **controller**'s grasping process, the **planner** monitors the execution, checks whether grasping succeeds, and assists re-grasping when failing. This process continues until the user prompt is fully completed.

width of the image), a third-view image $\mathbf{I}^{\mathrm{h}}_t \in \mathbb{R}^{H\times W\times 3}$ from the head camera, and the robot proprioception $\mathbf{s}_t \in \mathbb{R}^{13}$ consisting of seven arm joint angles $\mathbf{s}^{\mathrm{arm}}_t \in \mathbb{R}^7$ and six hand joint angles $\mathbf{s}^{\mathrm{hand}}_t \in \mathbb{R}^6$. Conditioned on these observations, the robot produces an action $\mathbf{a}_t = (\mathbf{a}^{\mathrm{arm}}_t, \mathbf{a}^{\mathrm{hand}}_t) \in \mathbb{R}^{13}$, where $\mathbf{a}^{\mathrm{arm}}_t \in \mathbb{R}^7$ and $\mathbf{a}^{\mathrm{hand}}_t \in \mathbb{R}^6$ denote the target joint angles for arm and hand respectively, by sampling from the action distribution $\pi(\cdot|\{\mathbf{I}^{\mathrm{w}}_j\}^t_{j=0}, \{\mathbf{I}^{\mathrm{h}}_j\}^t_{j=0}, \{\mathbf{s}_j\}^t_{j=0}, l)$. This process continues until a termination condition is reached. The robot receives a binary reward $r \in \{0, 1\}$ indicating whether it has completed the instruction l successfully. The goal of the policy π is to maximize the expected reward $\mathbb{E}_{l,\{(\mathbf{I}^{\mathrm{w}}_j, \mathbf{I}^{\mathrm{h}}_j, \mathbf{s}_j, \mathbf{a}_j)\}^T_{j=0}}[r]$.

More generally, we consider the case where the user prompt p may be a long-horizon task involving multiple grasping processes, such as "*clear the table*". This requires the policy π to reason about the prompt, decompose it into individual grasping instructions $\{l_i\}$, and complete them sequentially.

4. Methods

This section introduces DexGraspVLA, the first hierarchical VLA framework for dexterous grasping. We will first elaborate DexGraspVLA framework (Section 4.1) and then detail our data collection procedure (Section 4.2), which together enable the training of a dexterous grasping policy.

4.1. DexGraspVLA Framework

As illustrated in Figure 2, DexGraspVLA adopts a hierarchical and modularized architecture composed of a planner and a controller. Below we explain how each part is designed.

Planner. We recognize that to achieve general dexterous grasping, the model needs to be able to handle multimodal inputs, perform visual grounding, and conduct reasoning about user prompts. Building upon recent advances in VLMs, we adopt an off-the-shelf pre-trained Qwen-VL-Chat [78] as a high-level planner to outline and monitor the dexterous grasping workflow. Given the user prompt p, the planner reasons about the execution plan conditioning on the head camera observation. Specifically, if p is a long-horizon task description involving multiple grasping steps, *e.g.*, "*clear the table*", the planner considers object positions and orientations on the table, and proposes a suitable grasping instruction l_1 as a first step, *e.g.*, "*grasp the cookie*". Otherwise, if p directly targets one object for grasping, the planner regards it as the instruction l.

For each instruction l, the planner guides the low-level controller by marking the target object bounding box (x_1, y_1, x_2, y_2) in the head camera image $\mathbf{I}^{\mathrm{h}}_{t_0}$ at the initial timestep t_0. While the phrasing and content of language instruction can be diverse and flexible for different users and cases, *i.e.*, showing *domain-variance*, the bounding box is a consistent format for object positioning regardless of the changes in language and visual inputs, *i.e.*, achieving *domain-invariance*. Thus, this transformation alleviates the learning challenge for the controller.

On receiving the bounding box, the controller begins execution. During this process, the planner monitors the progress by querying the current head image at a frequency of 1Hz. If it finds that the robot successfully grasps the object, the planner executes a scripted placing motion to place the object into a bag and then resets the robotic arm and hand to the initial state. Afterward, the planner proposes new grasping instruction l_2 by reasoning about the prompt and the remaining objects in its view, until the prompt p is fully completed. On the other hand, if the controller fails to grasp the target object, the planner resets the robot and re-initializes the grasping loop with a new instruction based on the current object states.

Controller. Based on the target bounding box (x_1, y_1, x_2, y_2), the controller aims to grasp the intended object in cluttered environments. We feed this bounding box as input to SAM [23] to obtain an initial binary mask $\mathbf{m}_0 \in \{0,1\}^{H\times W\times 1}$ of the target object and then use Cutie [79] to continuously track the mask over time, producing $\mathbf{m}_t$ at each timestep t. This ensures accurate identification in cluttered scenes throughout the process. The problem is to learn the policy π that effectively models the action distribution $\pi(\cdot|\mathbf{I}^{\mathrm{w}}_t, \mathbf{I}^{\mathrm{h}}_t, \mathbf{s}_t, \mathbf{m}_t)$.

To achieve general-purpose dexterous grasping capabilities, the system must generalize effectively across diverse real-world scenarios. However, the high variability in raw visual inputs $\mathbf{I}^{\mathrm{w}}_t, \mathbf{I}^{\mathrm{h}}_t$ poses a fundamental challenge to learning task-critical representations. Traditional imitation learning approaches often fail catastrophically even under minor variations in objects or environmental conditions. To address this issue, our solution is again to convert potentially *domain-varying* inputs into *domain-invariant* representations suitable for imitation learning. We recognize that while pixel-level perception can vary widely, the fine-grained semantic features extracted by large foundation models tend to be more robust and consistent. This will be empirically substantiated in Section 5.5. Thus, we utilize a feature extractor ϕ, such as DINOv2 [20] that has been pre-trained on internet-scale data, to obtain features from raw images. At each timestep t, we obtain head camera image features

$$\mathbf{z}^{\mathrm{h}}_t = \phi^{\mathrm{h}}(\mathbf{I}^{\mathrm{h}}_t) \in \mathbb{R}^{L^{\mathrm{h}}\times D^{\mathrm{h}}},$$

and wrist camera image features

$$\mathbf{z}^{\mathrm{w}}_t = \phi^{\mathrm{w}}(\mathbf{I}^{\mathrm{w}}_t) \in \mathbb{R}^{L^{\mathrm{w}}\times D^{\mathrm{w}}},$$

where $L^{\mathrm{h}}, D^{\mathrm{h}}, L^{\mathrm{w}}, D^{\mathrm{w}}$ denote length and hidden dimension of the feature sequences for head and wrist respectively. As we show in Section 5.5, these extracted features remain comparatively invariant to distracting visual factors.

Up to now, raw language and vision inputs, including instruction l and images $\mathbf{I}^{\mathrm{w}}_t, \mathbf{I}^{\mathrm{h}}_t$, have been iteratively transformed into domain-invariant representations, including mask $\mathbf{m}_t$ and features $\mathbf{z}^{\mathrm{h}}_t, \mathbf{z}^{\mathrm{w}}_t$, by leveraging foundation models. This lays the stage for imitation learning. We now learn the policy π that predicts an action chunk of horizon H conditioning on these representations.

To fuse the object mask with head camera features, we project $\mathbf{m}_t$ into the head image feature space using a randomly initialized ViT, producing $\mathbf{z}^{\mathrm{m}}_t \in \mathbb{R}^{L^{\mathrm{h}}\times D^{\mathrm{h}}}$.

We then concatenate $\mathbf{z}_t^{\mathrm{m}}$ and $\mathbf{z}_t^{\mathrm{h}}$ patch-wise to form

$$\bar{\mathbf{z}}_t^{\mathrm{h}} \in \mathbb{R}^{L^{\mathrm{h}} \times 2D^{\mathrm{h}}}.$$

Subsequently, we map $\bar{\mathbf{z}}_t^{\mathrm{h}}$, the wrist-camera features $\mathbf{z}_t^{\mathrm{w}}$, and the robot state $\mathbf{s}_t$ into a common embedding space with separate MLPs, yielding $\tilde{\mathbf{z}}_t^{\mathrm{h}}$, $\tilde{\mathbf{z}}_t^{\mathrm{w}}$, and $\tilde{\mathbf{z}}_t^{\mathrm{s}}$. These embeddings are then concatenated to form the full observation feature sequence

$$\tilde{\mathbf{z}}_t^{\mathrm{obs}} \in \mathbb{R}^{(1+L^{\mathrm{h}}+L^{\mathrm{w}}) \times D}.$$

For action prediction, we employ a diffusion transformer (DiT) [80] to generate multi-step actions, following the diffusion policy paradigm [81, 76, 28]. Concretely, at each timestep t, we bundle the next H actions into a chunk $\mathbf{A}_t = \mathbf{a}_{t:t+H} = [\mathbf{a}_t, \mathbf{a}_{t+1}, \ldots, \mathbf{a}_{t+H-1}]$. During training, a random diffusion step $t^d = k$ is sampled, and Gaussian noise $\boldsymbol{\epsilon}$ is added to $\mathbf{A}_t$, yielding the noised action tokens $\mathbf{x}_k$. Formally,

$$\mathbf{x}_k = \alpha_k \mathbf{A}_t + \sigma_k \boldsymbol{\epsilon},$$

where α_k and σ_k are the standard DDPM coefficients. We then feed $\mathbf{x}_k$ into the DiT alongside the observation feature sequence $\tilde{\mathbf{z}}_t^{\mathrm{obs}}$. Each DiT layer performs bidirectional self-attention over the action tokens, cross-attention to $\tilde{\mathbf{z}}_t^{\mathrm{obs}}$, and MLP transformations, ultimately predicting the original noise $\boldsymbol{\epsilon}$. By minimizing the discrepancy between predicted and true noise, the model learns to reconstruct the ground-truth action chunk $\mathbf{A}_t$. At inference time, iterative denoising steps recover the intended multi-step action sequence from the learned distribution, enabling robust imitation of complex, long-horizon behaviors. We also employ the receding horizon control strategy that only executes the first H_a actions before generating a new action chunk prediction, enhancing real-time responsiveness.

Overall, DexGraspVLA performs imitation learning on *domain-invariant* representations derived from *domain-varying* inputs via foundation models. This approach not only leverages the world knowledge and generalization capabilities of foundation models, but also effectively captures the mapping from these abstracted representations to the final action output. Next, we discuss our data collection that fuels DexGraspVLA training.

4.2. Data Collection

To train our dexterous grasping policy, we manually collect a dataset comprising 2,094 successful episodes of grasping in cluttered scenes. This dataset involves 36 household objects spanning a broad range of sizes, weights, geometries, textures, materials, and categories. Each episode $\tau = \{(\mathbf{I}_t^{\mathrm{h}}, \mathbf{I}_t^{\mathrm{w}}, \mathbf{s}_t, \mathbf{m}_t, \mathbf{a}_t)\}_{t=0}^{T}$ records the raw camera images $\mathbf{I}_t^{\mathrm{h}}, \mathbf{I}_t^{\mathrm{w}}$, the robot proprioception $\mathbf{s}_t$, the object mask $\mathbf{m}_t$, and the action $\mathbf{a}_t$ at each timestep t. The mask $\mathbf{m}_t$ is labeled in the same way as in the controller. For each object, we place it at nine locations arranged in a 3×3 grid and collect multiple grasping demonstrations at each position. The other objects in the cluttered scenes are randomized between episodes. These demonstrations are performed at typical human motion speeds, taking about 3.5 s each. They undergo rigorous manual inspection to ensure quality and reliability. The DexGraspVLA controller is trained on this dataset with imitation learning.

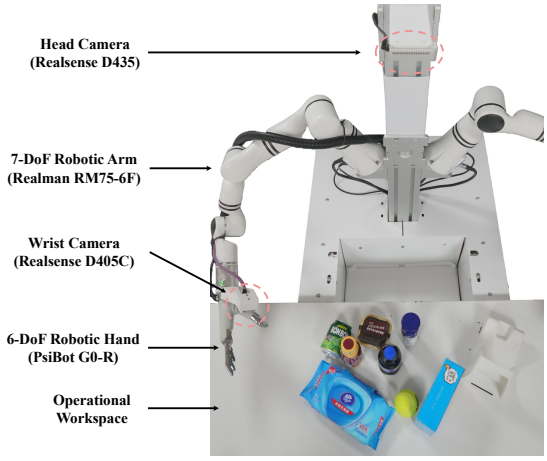

Figure 3 | The hardware platform used for dexterous grasping.

5. Experiments

In this section, we comprehensively evaluate the performance of DexGraspVLA. All experiments are conducted on a different robot and environment from the demonstration setup. This "zero-shot" setting is fundamentally more challenging than most prior imitation learning research, which relies on few-shot learning for high performance. Our experiments seek to address the following questions:

1. How effectively does DexGraspVLA generalize to thousands of diverse, previously unseen object, lighting, and background combinations in cluttered scenes? (Section 5.2)
2. How does DexGraspVLA's generalization advantage compare against baselines that do not utilize frozen feature extractors and learn directly from raw visual inputs? (Section 5.3)
3. How accurate are bounding-box predictions from the high-level planner of DexGraspVLA across different scenarios? (Section 5.4)
4. Does DexGraspVLA exhibit consistent internal

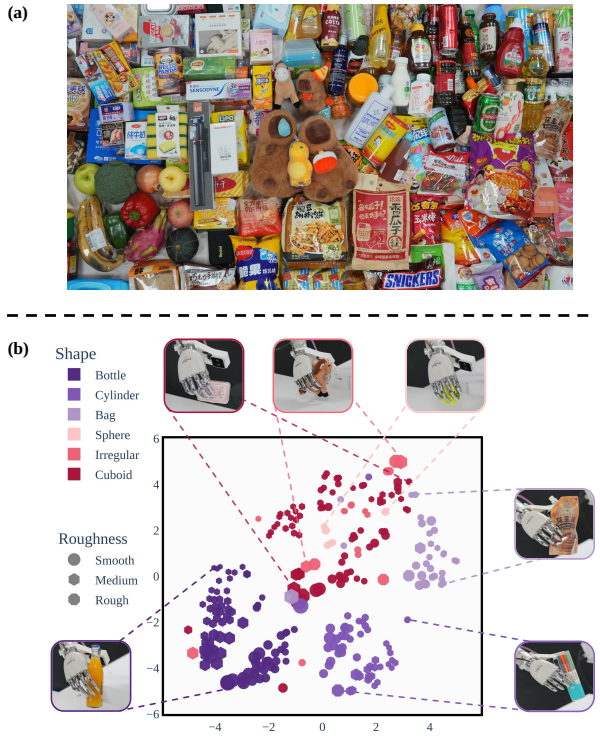

Figure 4 | **(a)** A representative part of unseen objects used in our experiments. In total, we test DexGraspVLA on 360 unseen objects that span a wide range of categories. **(b)** A t-SNE projection illustrating the diversity of all unseen objects. Each point denotes one object, where color indicates shape (e.g., bottle, cylinder, sphere), marker type denotes surface roughness (smooth, medium, rough), and point size reflects mass. The plot is derived from a high-dimensional attribute space (including length, width, height, mass, roughness, and shape), highlighting the broad coverage of sizes, weights, geometries, and textures represented by these objects.

model behaviors under varying environments? (Section 5.5)

5.1. Experiment Setups

Hardware Platform. As illustrated in Figure 3, the robot we use for dexterous grasping is a 7-DoF Realman RM75-6F arm paired with a 6-DoF PsiBot G0-R hand. A Realsense D405C camera, mounted on the arm's wrist, provides a first-person viewpoint, while a Realsense D435 camera on the robot's head offers a third-person perspective. Objects to be grasped are placed on a table in front of the robot. The control frequency of the robot is 20 Hz.

Baselines. To the best of our knowledge, there are no existing work that can directly serve as baselines for comparison. Most of dexterous grasping methods cannot process language inputs for cluttered scene, while existing VLA frameworks that accept language inputs are incompatible with dexterous hands. Therefore, we compare the following methods: (1) *DexGraspVLA (Ours)*: A full implementation of DexGraspVLA. (2) *DexGraspVLA (DINOv2-train)*: Same design as *Ours* except that the two DINOv2 models are trainable instead of frozen. (3) *DexGraspVLA (ViT-small)*: Same design as *Ours* except that the two DINOv2 models are replaced with two small trainable pretrained ViTs (the R26-S-32 ResNet-ViT hybrid from Steiner et al. [82]). Empirically, DexGraspVLA (ViT-small) represents an enhanced version of diffusion policy [81]. Implementation details of these methods are provided in Appendix A. In preliminary experiments, we find that failures may arise from randomness in policy inference and can be overcome with additional attempts. Thus, in Section 5.2, we compare *DexGraspVLA (Ours@k)*, with k ranging from 1 to 3. These are the same as *Ours* except that they are allowed k attempts for each test respectively. *Ours*@1 is equivalent to *Ours*. Note that re-grasps performed by the policy after an initial failure within a single attempt are allowed and do not count as separate attempts. In Section 5.4, we report the bounding box prediction performance of *DexGraspVLA (planner)*, which is a full implementation of the high-level planner in DexGraspVLA.

	Unseen Objects (360)	Unseen Backgrounds (6 × 103)	Unseen Lightings (3 × 103)	Aggregated (1287)
Ours@1	91.1%	90.5%	90.9%	90.8%
Ours@2	95.3%	94.2%	95.1%	94.7%
Ours@3	96.7%	96.7%	97.4%	96.9%

Table 1 | The performance of DexGraspVLA under different unseen conditions. The first row suggests that DexGraspVLA consistently achieves a 90+% success rate across thousands of unseen object, background, and lighting combinations, highlighting robustness to environmental changes and strong generalization. The second and third row shows its potential to reach even higher success rates given more chances.

5.2. Large-Scale Generalization Evaluation

Tasks. We curate 360 previously unseen objects, 6 unseen backgrounds, and 3 unseen lighting conditions. The objects are carefully selected to ensure they cover a broad range of sizes, weights, geometries, textures, materials, and categories, while also being graspable by our dexterous hand. This diversity is visualized in Figure 4. The backgrounds and lighting conditions are also chosen to be highly different. Based on this setup,

we design three types of grasping tasks in cluttered scenes, each cluttered scene involving approximately six objects: (1) *Unseen Objects*: Grasp an unseen object from a random scene on a white table under white light. Each of the 360 unseen objects is targeted once, summing up to 360 tests. (2) *Unseen Backgrounds*: We first randomly select 103 unseen objects as the *object subset* $\mathcal{S}$. For each background, we randomly arrange 103 cluttered scenes with objects in $\mathcal{S}$ under white light. Each of the 103 objects is targeted once, leading to 618 tests in total. (3) *Unseen Lightings*: For each unseen lighting, we construct 103 cluttered scenes with objects in $\mathcal{S}$ on a white table. We target each of the 103 objects once, amounting to 309 tests. Details can be found in Appendix B.

Metric. We consider a grasping attempt successful if it holds the object 10 cm above the table for 20 s. We report success rate as the evaluation metric, which is defined as the number of successful tests divided by the total number of tests. We also report the aggregated performance as a weighted sum of individual success rates based on their proportions.

Results. We present the quantitative results in Table 1. From the first row ("Ours@1"), DexGraspVLA achieves a 91.1% single-attempt success rate on 360 unseen objects, 90.5% on 6 unseen backgrounds, and 90.9% under 3 unseen lighting conditions, resulting in a 90.8% aggregated success rate. These results demonstrate that DexGraspVLA accurately controls the dexterous hand to grasp the specified object from clutter, while remaining robust to environmental changes. Notably, although the evaluation environment is novel and the tasks are previously unseen, DexGraspVLA consistently achieves high success rates without any domain-specific fine-tuning, underscoring strong generalization. This suggests that our framework substantially alleviates the longstanding challenge in imitation learning — namely, overfitting to a single domain and relying on finetuning for satisfactory performance — and is potentially meaningful for a wide range of applications. We further analyze the source of this generalization in 5.5.

Qualitatively, DexGraspVLA learns to dexterously adjust the arm and hand to accommodate varying object geometries, sizes, and positions. Although physical interference or suboptimal actions occasionally cause missed grasps, the closed-loop nature of our policy facilitates re-grasps based on updated observations, enhancing robustness. The method also tolerates human-induced perturbations, as the robot can track and follow repositioned objects until a successful grasp is achieved.

From the second and third rows ("Ours@2" and "Ours@3"), we observe that while randomness and incidental failures can occur on a single attempt, multiple attempts typically lead to success, elevating the overall performance to 96.9% at three tries. This indicates that our approach has the capacity to reach even higher success rates. Finally, our model takes around 6 s on average to grasp an object, which is close to that of humans and ensures practical usability in real-world scenarios.

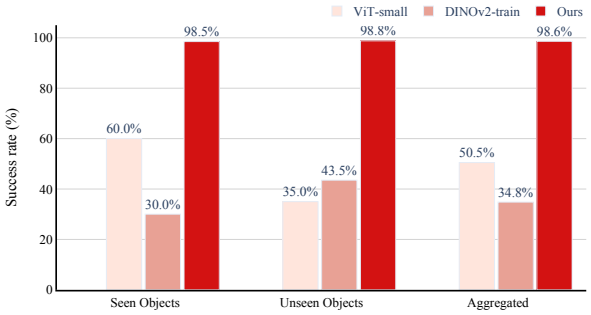

Figure 5 | DexGraspVLA significantly outperforms alternative designs in both seen and unseen object grasping experiments in the "zero-shot" environment. This confirms the effectiveness of applying imitation learning on domain-invariant features.

Overall, our large-scale evaluations confirm that DexGraspVLA can robustly handle a wide spectrum of unseen scenarios, representing a meaningful step toward general dexterous grasping and promising broader real-world deployment.

5.3. Comparison to Baselines without Frozen Vision Encoders

Tasks. To systematically compare DexGraspVLA with baselines that do not employ frozen vision encoders and learn directly from raw visual inputs, we conduct single-object grasping experiments using 13 seen objects from the training dataset and 8 unseen objects. We select five locations on the table that both span the operational workspace and remain within the robot's reach and the head camera's field of view. Each object is placed at these five points, and at each point, we let the policy grasp it twice. Note that the two grasps of the same object at the same point are counted as two separate tests rather than repeated attempts of the same test. This approach quantitatively accounts for the randomness in the experiments. In total, this involves 210 tests. The environmental conditions in these experiments are white tabletop and white light.

Metric. We report the success rates of each method in the same way as Section 5.2.

Results. Figure 5 demonstrates that DexGraspVLA (Ours) consistently reaches a more than 98% success

	No Distraction	Background Distraction	Lighting Distraction	Aggregated
Planner	96.7%	100.0%	100.0%	99.3%

Table 2 | DexGraspVLA planner robustly achieves near-perfect accuracy for bounding-box prediction in cluttered scenes across different environment conditions.

rate on both seen and unseen object grasping experiments, outperforming both DexGraspVLA (DINOv2-train) and DexGraspVLA (ViT-small) by a significant margin. The fact that the aggregated performance of our method in the "zero-shot" test environment is near-perfect shows that DexGraspVLA (ours) is not affected by domain shift in visual inputs. We also notice that the performance on unseen objects is even slightly better than on seen objects, which again confirms that our model learns to accomplish the grasping task instead of overfitting to seen data in the training set. By contrast, the alternative designs fail to work properly in a novel environment, since they directly map raw inputs to actions and the perceptual changes easily render them out of distribution.

5.4. Bounding-box Prediction Accuracy of Planner

Tasks. The bounding-box prediction accuracy of the planner is crucial to the success of grasping, as it determines the target for the controller. To evaluate this accuracy, we design three types of tasks featuring different environmental distractions: (1) *No Distraction* (1 scenario): The cluttered scene is arranged on a white table under white light; (2) *Background Distraction* (2 scenarios): The cluttered scene is placed on either a calibration board or a brightly colored tablecloth, both under white light; (3) *Lighting Distraction* (2 scenarios): The scene is set up in a dark room illuminated by either a desk lamp or a disco light. For each scenario, we randomly arrange five cluttered scenes, each containing six randomly selected objects, and then record head-camera images. For each object, we provide a textual prompt describing its appearance and location, and check whether the planner's bounding-box prediction accurately marks the target. In total, *No Distraction* accounts for 30 tests, while *Background Distraction* and *Lighting Distraction* both have 60 tests, amounting to 150 tests overall.

Metric. We define a bounding box as accurate if it tightly encloses the target object. Accuracy is then measured as the proportion of accurate bounding boxes over all tested objects.

Results. The accuracy is reported in Table 2. For 150 prompts, the planner only mislabels one bounding box while succeeding in the other 149 tests, resulting in an aggregated accuracy exceeding 99%. This proves that our planner reliably performs visual grounding of user prompts and is capable of marking the correct bounding box for the controller across varying degrees of background and lighting complexity.

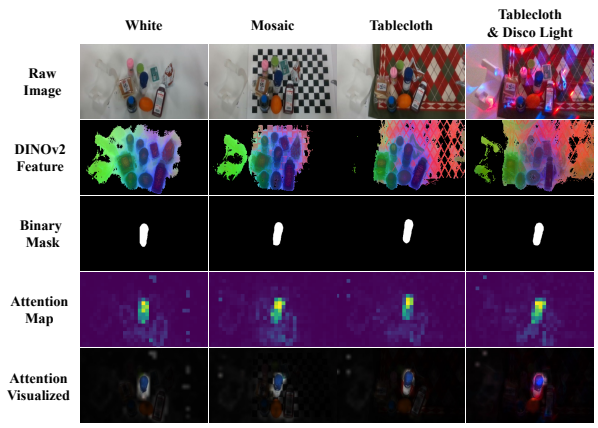

Figure 6 | **DexGraspVLA is robust to environmental variations.** The first row presents the cropped raw head images of the same cluttered objects in four different environments: a white table, a calibration board, a tablecloth, and a tablecloth under disco light. The second row shows that the DINOv2 features of images are consistent across variations. The third row is the masks of the target objects accurately tracked by Cutie. The fourth row reflects that the averaged attention maps of DiT to head image features are also consistent regardless of perceptual differences. The fifth row confirms DexGraspVLA is attending to the correct object. See Appendix B for full images.

5.5. Internal Model Behavior Analysis

To further validate our design, we empirically prove that the internal model behavior is consistent across visual variations and show the results in Figure 6. Due to space constraints, we only show the relevant portion of each image containing the tabletop workspace. The complete, uncropped images are provided in Appendix B. Specifically, we design four vastly different environmental conditions: a white table, a calibration board, a colorful tablecloth, and a colorful tablecloth with disco light. In each environment, we construct the same cluttered scene containing nine objects and let DexGraspVLA "grasp the blue yogurt in the middle". While the head images in the first row of Figure 6 appear to be markedly diverse, the DINOv2 features in the second row look rather consistent. These features are visualized by mapping principal components to RGB channels as done in Oquab et al. [20]. Across environments, the object properties are robustly main-

tained and matched, which fundamentally allows DexGraspVLA trained on a single data domain to generalize. The third row shows that Cutie accurately tracks the object, providing the correct guidance to the controller. Based on the domain-invariant mask and the DINOv2 features, the DiT action head now predicts the subsequent actions. In the fourth row, we average and normalize all cross-attentions to the head image from DiT. We find that all attention maps exhibit the same behavior of focusing on the target object instead of being distracted by environments. The fifth row overlays the attention map on the raw image to confirm the reasonable attention pattern. All visualization details are provided in Appendix B. Therefore, we substantiate that DexGraspVLA indeed transforms perceptually diverse raw inputs into invariant representations, on which it effectively applies imitation learning to model the data distribution, explaining its superior generalization performance. Expectedly, it successfully grasps the yogurt in all four environments.

6. Limitations

Although DexGraspVLA achieves high success rates across a range of unseen scenarios, certain limitations remain. First, due to the time limit, our training dataset does not encompass very small objects or extremely cluttered environments; performance on these more challenging cases could improve with dedicated data collection. Additionally, we have not yet explored functional grasping for subsequent object usage, which is a promising direction for future work.

7. Conclusion

This paper presents DexGraspVLA, the first hierarchical vision-language-action framework that advances toward general dexterous grasping. It utilizes a pre-trained VLM as the high-level planner to plan the grasping process and a diffusion-based policy as the low-level controller to perform closed-loop action prediction for grasping. Within this paradigm, DexGraspVLA capitalizes on the world knowledge of foundation models to understand diverse raw inputs and transform them into domain-invariant representations. Imitation learning is then applied to model the mapping from representation to action distribution, which is highly effective due to the alleviation of domain shift. Our large-scale evaluations show that it attains a success rate exceeding 90% across thousands of unseen cluttered scenes in a "zero-shot" test environment, demonstrating robust generalization. An empirical analysis of its internal model behavior further validates the underlying framework design. Overall, DexGraspVLA demonstrates the promise of leveraging foundation models to enhance generalization in dexterous grasping. We plan to further refine its performance and broaden its applications in future work.

Acknowledgments

We would like to express our sincere gratitude to Yijie Xu for his substantial assistance with the experiments, to Yu Li, Yue Chen, and Qianxu Wang for their insightful discussions, to Jie Chu for exploring the capabilities of foundation models, and to Chengdong Ma and Zijie Yu for their generous help. Additionally, we deeply appreciate the significant support from our colleagues at Psibot in hardware, engineering, and procurement.

References

[1] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand dexterous manipulation from depth. In *Icml workshop on new frontiers in learning, control, and dynamical systems*, 2023.

[2] Haozhi Qi, Brent Yi, Sudharshan Suresh, Mike Lambeta, Yi Ma, Roberto Calandra, and Jitendra Malik. General in-hand object rotation with vision and touch. In *Conference on Robot Learning*, pages 2549–2564. PMLR, 2023.

[3] Binghao Huang, Yuanpei Chen, Tianyu Wang, Yuzhe Qin, Yaodong Yang, Nikolay Atanasov, and Xiaolong Wang. Dynamic handover: Throw and catch with bimanual hands. In *7th Annual Conference on Robot Learning*, 2023.

[4] Toru Lin, Zhao-Heng Yin, Haozhi Qi, Pieter Abbeel, and Jitendra Malik. Twisting lids off with two hands. *arXiv preprint arXiv:2403.02338*, 2024.

[5] Yuanpei Chen, Tianhao Wu, Shengjie Wang, Xidong Feng, Jiechuan Jiang, Zongqing Lu, Stephen McAleer, Hao Dong, Song-Chun Zhu, and Yaodong Yang. Towards human-level bimanual dexterous manipulation with reinforcement learning. *Advances in Neural Information Processing Systems*, 35:5150–5163, 2022.

[6] Kelvin Xu, Zheyuan Hu, Ria Doshi, Aaron Rovinsky, Vikash Kumar, Abhishek Gupta, and Sergey Levine. Dexterous manipulation from images: Autonomous real-world rl via substep guidance. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5938–5945. IEEE, 2023.

[7] Yuanpei Chen, Chen Wang, Li Fei-Fei, and C Karen Liu. Sequential dexterity: Chaining dexterous policies for long-horizon manipulation. In *Conference on Robot Learning*, pages 3809–3829, 2023.

[8] Abhishek Gupta, Justin Yu, Tony Z Zhao, Vikash Kumar, Aaron Rovinsky, Kelvin Xu, Thomas Devlin, and Sergey Levine. Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6664–6671. IEEE, 2021.

[9] Kevin Zakka, Laura Smith, Nimrod Gileadi, Taylor Howell, Xue Bin Peng, Sumeet Singh, Yuval Tassa, Pete Florence, Andy Zeng, and Pieter Abbeel. Robopianist: A benchmark for high-dimensional robot control. *arXiv preprint arXiv:2304.04150*, 2023.

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

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

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

[13] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[14] Max Yang, Chenghua Lu, Alex Church, Yijiong Lin, Chris Ford, Haoran Li, Efi Psomopoulou, David AW Barton, and Nathan F Lepora. Anyrotate: Gravity-invariant in-hand object rotation with sim-to-real touch. *arXiv preprint arXiv:2405.07391*, 2024.

[15] Johannes Pitz, Lennart Röstel, Leon Sievers, and Berthold Bäuml. Dextrous tactile in-hand manipulation using a modular reinforcement learning architecture. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1852–1858. IEEE, 2023.

[16] Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5977–5984, 2023.

[17] Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, and C Karen Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024.

[18] Zoey Qiuyu Chen, Karl Van Wyk, Yu-Wei Chao, Wei Yang, Arsalan Mousavian, Abhishek Gupta, and Dieter Fox. Dextransfer: Real world multi-fingered dexterous grasping with minimal human demonstrations. *arXiv preprint arXiv:2209.14284*, 2022.

[19] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In *Robotics: Science and Systems*, 2017.

[20] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2023.

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

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

[23] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.

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

[25] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. In *Conference on Robot Learning*, pages 540–562, 2023.

[26] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. In *8th Annual Conference on Robot Learning*, 2024.

[27] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[28] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A Vision-Language-Action Flow Model for General Robot Control. *arXiv preprint arXiv:2410.24164*, 2024.

[29] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems*, 2024.

[30] Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[31] Jens Lundell, Francesco Verdoja, and Ville Kyrki. Ddgc: Generative deep dexterous grasping in clutter. *IEEE Robotics and Automation Letters*, 6 (4):6899–6906, 2021.

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

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

[34] Puhao Li, Tengyu Liu, Yuyang Li, Yiran Geng, Yixin Zhu, Yaodong Yang, and Siyuan Huang. Gendexgrasp: Generalizable dexterous grasping. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8068–8074. IEEE, 2023.

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

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

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

[38] Min Liu, Zherong Pan, Kai Xu, Kanishka Ganguly, and Dinesh Manocha. Deep differentiable grasp planner for high-dof grippers. In *Robotics: Science and Systems*, 2020.

[39] Jiaxin Lu, Hao Kang, Haoxiang Li, Bo Liu, Yiding Yang, Qixing Huang, and Gang Hua. Ugg: Unified generative grasping. In *European Conference on Computer Vision*, pages 414–433. Springer, 2025.

[40] Tao Chen, Jie Xu, and Pulkit Agrawal. A system for general in-hand object re-orientation. *Conference on Robot Learning*, 2021.

[41] Zhao-Heng Yin, Binghao Huang, Yuzhe Qin, Qifeng Chen, and Xiaolong Wang. Rotating without seeing: Towards in-hand dexterity through touch. In *Robotics: Science and Systems*, 2023.

[42] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-hand object rotation via rapid motor adaptation. In *Conference on Robot Learning*, pages 1722–1732. PMLR, 2023.

[43] Sudeep Dasari, Abhinav Gupta, and Vikash Kumar. Learning dexterous manipulation from exemplar object trajectories and pre-grasps. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3889–3896. IEEE, 2023.

[44] Gagan Khandate, Siqi Shang, Eric T Chang, Tristan Luca Saidi, Yang Liu, Seth Matthew Dennis, Johnson Adams, and Matei Ciocarlie. Sampling-based exploration for reinforcement learning of dexterous manipulation. In *Robotics: Science and Systems*, 2023.

[45] Yuanpei Chen, Chen Wang, Yaodong Yang, and C Karen Liu. Object-centric dexterous manipulation from human motion data. In *8th Annual Conference on Robot Learning*, 2024.

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

[47] Hui Zhang, Sammy Christen, Zicong Fan, Otmar Hilliges, and Jie Song. Graspxl: Generating grasping motions for diverse objects at scale. In *European Conference on Computer Vision*, pages 386–403. Springer, 2024.

[48] Zhecheng Yuan, Tianming Wei, Shuiqi Cheng, Gu Zhang, Yuanpei Chen, and Huazhe Xu. Learning to manipulate anywhere: A visual generalizable framework for reinforcement learning. *CoRR*, abs/2407.15815, 2024. doi: 10.48550/ARXIV.2407.15815. URL https://doi.org/10.48550/arXiv.2407.15815.

[49] Priyanka Mandikal and Kristen Grauman. Dexvip: Learning dexterous grasping with human hand pose priors from video. In *Conference on Robot Learning*, pages 651–661. PMLR, 2022.

[50] Priyanka Mandikal and Kristen Grauman. Learning dexterous grasping with object-centric visual affordances. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 6169–6176. IEEE, 2021.

[51] Tyler Ga Wei Lum, Martin Matak, Viktor Makoviychuk, Ankur Handa, Arthur Allshire, Tucker Hermans, Nathan D Ratliff, and Karl Van Wyk. Dextrah-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics. In *8th Annual Conference on Robot Learning*, 2024.

[52] Ritvik Singh, Arthur Allshire, Ankur Handa, Nathan Ratliff, and Karl Van Wyk. Dextrah-rgb: Visuomotor policies to grasp anything with dexterous hands. *arXiv preprint arXiv:2412.01791*, 2024.

[53] Zoey Qiuyu Chen, Karl Van Wyk, Yu-Wei Chao, Wei Yang, Arsalan Mousavian, Abhishek Gupta, and Dieter Fox. Learning robust real-world dexterous grasping policies via implicit shape augmentation. In *Conference on Robot Learning*, pages 1222–1232, 2022.

[54] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. Videodex: Learning dexterity from internet videos. In *Conference on Robot Learning*, pages 654–665. PMLR, 2023.

[55] Ilija Radosavovic, Xiaolong Wang, Lerrel Pinto, and Jitendra Malik. State-only imitation learning for dexterous manipulation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7865–7871. IEEE, 2021.

[56] Sridhar Pandian Arunachalam, Irmak Güzey, Soumith Chintala, and Lerrel Pinto. Holo-dex: Teaching dexterity with immersive mixed reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5962–5969. IEEE, 2023.

[57] Irmak Guzey, Ben Evans, Soumith Chintala, and Lerrel Pinto. Dexterity from touch: Self-supervised pre-training of tactile representations with robotic play. In *7th Annual Conference on Robot Learning*, 2023.

[58] Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter Fox. Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9164–9170. IEEE, 2020.

[59] Aravind Sivakumar, Kenneth Shaw, and Deepak Pathak. Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube. In *Robotics: Science and Systems*, 2022.

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

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

[62] Zichen Jeff Cui, Yibin Wang, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. From play to policy: Conditional behavior generation from uncurated robot data. In *The International Conference on Learning Representations*, 2023.

[63] Siddhant Haldar, Jyothish Pari, Anant Rai, and Lerrel Pinto. Teach a robot to fish: Versatile imitation from one minute of demonstrations. *arXiv preprint arXiv:2303.01497*, 2023.

[64] Yuzhe Qin, Hao Su, and Xiaolong Wang. From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation. *IEEE Robotics and Automation Letters*, 7(4):10873–10881, 2022.

[65] Sridhar Pandian Arunachalam, Sneha Silwal, Ben Evans, and Lerrel Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. In *2023 ieee international conference on robotics and automation (icra)*, pages 5954–5961, 2023.

[66] Irmak Guzey, Yinlong Dai, Ben Evans, Soumith Chintala, and Lerrel Pinto. See to touch: Learning tactile dexterity through visual incentives. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13825–13832. IEEE, 2024.

[67] Toru Lin, Yu Zhang, Qiyang Li, Haozhi Qi, Brent Yi, Sergey Levine, and Jitendra Malik. Learning visuotactile skills with two multifingered hands. *arXiv preprint arXiv:2404.16823*, 2024.

[68] Qianxu Wang, Haotong Zhang, Congyue Deng, Yang You, Hao Dong, Yixin Zhu, and Leonidas Guibas. Sparsedff: Sparse-view feature distillation for one-shot dexterous manipulation. In *The Twelfth International Conference on Learning Representations*, 2023.

[69] Qianxu Wang, Congyue Deng, Tyler Ga Wei Lum, Yuanpei Chen, Yaodong Yang, Jeannette Bohg, Yixin Zhu, and Leonidas Guibas. Neural attention field: Emerging point relevance in 3d scenes for one-shot dexterous grasping. In *8th Annual Conference on Robot Learning*, 2024.

[70] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10371–10381, 2024.

[71] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 37: 21875–21911, 2025.

[72] Qwen Team. Qwen2.5-vl, January 2025. URL https://qwenlm.github.io/blog/qwen2.5-vl/.

[73] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[74] Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024.

[75] Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. *arXiv preprint arXiv:2311.01977*, 2023.

[76] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

[77] Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

[78] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 1(2):3, 2023.

[79] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Joon-Young Lee, and Alexander Schwing. Putting the object back into video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3151–3161, 2024.

[80] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[81] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2023.

[82] Andreas Peter Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *Transactions on Machine Learning Research*, 2022.

[83] Yiheng Li, Heyang Jiang, Akio Kodaira, Masayoshi Tomizuka, Kurt Keutzer, and Chenfeng Xu. Immiscible diffusion: Accelerating diffusion training with noise assignment. *arXiv preprint arXiv:2406.12303*, 2024.

[84] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020.

A. Implementation Details

In this section, we present the details of DexGraspVLA implementation (Appendix A.1), baseline implementation (Appendix A.2), and dataset collection (Appendix A.3).

A.1. Details of DexGraspVLA Implementation

Planner. The high-level planner operates as described in Section 4.1. To obtain the bounding box for the low-level controller using Qwen-VL-Chat, we begin by cropping a fixed region of the head-camera image that corresponds to the tabletop workspace. We then feed this cropped area into the VLM, which identifies the target object's bounding box within it. Finally, the coordinates are mapped back to the original image to determine the bounding box used by the controller. Several labeling results are shown in Figure 10.

By leveraging an off-the-shelf VLM as the planner, our framework gains remarkable flexibility, enabling easy replacement of the original VLM with more advanced models for enhanced performance. Our observations indicate that Qwen2.5-VL-72B-Instruct [72] outperforms Qwen-VL-Chat (used in our experiments, Section 5) in reasoning and instruction following, leading to improved long-horizon task completion. Furthermore, with the 72B model, the planner can achieve near-perfect bounding box prediction accuracy without image cropping. Therefore, we provide the prompts for the DexGraspVLA planner based on Qwen2.5-VL-72B-Instruct below.

These prompts mainly instruct the VLM to function as DexGraspVLA planner, including understanding the user prompt, proposing an object as the current grasping instruction, marking the target object bounding box, checking if the grasp has succeeded, assessing whether the current instruction is completed, and evaluating whether the entire user prompt is fully fulfilled. Specifically, when a user prompt p is provided, the planner classifies its type based on whether p explicitly specifies target objects with the following prompt.

```
Analyze the following user prompt: <user prompt>

User prompt types:
- Type I (return True): User prompts with any specific descriptions
Examples:
* Color-based: "green objects"
* Position-based: "objects from the right"
* Property-based: "all cups"
* Combination: "the red cup on the left"

- Type II (return False): Abstract prompts without any object descriptions
Examples: "clear the table", "clean up", "remove everything"

Please determine:
- Is this a Type I prompt? (True/False)
- Provide your reasoning

Return format:
True/False: your reasoning

Examples:
- "grab the green cup" -> True: Contains specific object (cup) and property (green)
- "clear the table" -> False: No specific object characteristics mentioned
```

If p is classified as "Type I", meaning it explicitly

specifies the objects to grasp, the planner generates an ordered list of grasping instructions, using the following prompt and head-camera image.

> For user prompt: <user prompt>
> Process:
> 1. Analyze the user prompt and image together:
> - Match user prompt descriptions with visible objects in the image
> - If a description (e.g., "green objects") matches multiple objects, include all matching objects
> - Verify each mentioned object actually exists in the image
>
> 2. Based on the robot arm's position (right edge of the screen) and table layout
> 3. Determine the most efficient grasping sequence
> 4. Generate a reordered list of objects to grasp
>
> Requirements:
> - Only include objects mentioned in the original user prompt
> - Keep position information for each object
> - Return as a list, ordered by grasping sequence
>
> Expected output format:
> ["object with position 1", "object with position 2", ...]

Alternatively, if p is classified as "Type II", indicating that the user intends to grasp all objects on the table, the planner selects the optimal target object as the current instruction. This selection is based on the remaining objects on the table, using the following prompt and head-camera image.

> Analyze the current desktop layout and select the most suitable object to grasp, considering the following factors:
>
> Grasping Strategy:
> 1. The robotic arm is positioned on the far right (outside the frame)
> 2. Grasping Priority Order:
> - Prioritize objects on the right to avoid knocking over other objects during later operations
> - Then consider objects in the middle
> - Finally, consider objects on the left
> 3. Accessibility Analysis:
> - Relative positions between objects
> - Potential obstacles
> - Whether the grasping path might interfere with other objects
>
> Please provide your response in the following JSON format:
>
> ```
> {
>   "analysis": {
>     "priority_consideration": "
>        explanation of why this
>        object has priority",
>     "accessibility": "analysis
>        of object's accessibility
>        ",
>     "risk_assessment": "
>        potential risks in
>        grasping this object"
>   },
>   "target": "a comprehensive
>      description of the target
>      object (e.g., 'the blue
>      cube on the far right of
>      the desktop, next to the
>      red cylinder')"
> }
> ```
>
> Ensure the output is in valid JSON format.
> Note: The 'target' field should ONLY contain the object's color, shape, and position in a natural, flowing sentence. Do not include any analysis or reasoning in this field.

For each grasping instruction l, the planner marks the bounding box of the target object using the following prompt and head-camera image.

> Analyze the image and identify the best matching object with the description: <target object>.
> Instructions for object analysis:
> 1. Select ONE object that best matches the description
> 2. For the selected object, provide:
> - A concise label, object name (3-4 words max)
> - A detailed description (position, color, shape, context)
> - Accurate bbox coordinates
>
> Required JSON format with an example:
>
> ```
> {
>   "bbox_2d": [x1, y1, x2, y2],
>   "label": "green cup",  # Keep
>      this very brief (3-4 words)
>   "description": "a cylindrical
>      green ceramic cup located
> ```

```
        on the right side of the
        wooden table, next to the
        laptop"  # Detailed
        description
}
```

Critical requirements:
- Return EXACTLY ONE object
- "label": Must be brief (3-4 words) for quick reference
- "description": Must be detailed and include spatial context
- Use single JSON object format, not an array
- Ensure bbox coordinates are within image boundaries

During the controller's execution, the planner verifies whether the object has been successfully grasped, using the following prompt and head-camera image.

> Analyze the image and determine if the robotic arm has successfully grasped an object:
> 1. Observe the spatial relationship between the robotic hand and the object
> 2. Output format: explain your reasoning, then conclude with a boolean value (True=grasped, False=not grasped)

When a grasping attempt ends, the robot resets to its initial state, and the planner checks whether the current instruction has been completed, using the following prompt and head-camera image.

> Please check whether <target object> exists on the desktop. If it does not exist, output True; otherwise, output False.

For a "Type I" user prompt p, the planner considers it fulfilled if all specified objects have been successfully grasped. For a "Type II" user prompt p, the planner checks whether the prompt has been fully completed after each grasping attempt, using the following prompt and head-camera image.

> Please analyze the table in the image:
>
> Requirements:
> - Only detect physical objects with noticeable height/thickness (3D objects)
> - Exclude from consideration:
> * Flat items (papers, tablecloths, mats)
> * Light projections
> * Shadows
> * Surface patterns or textures
>
> Return format:
> - True: if the table is empty of 3D objects
> - False: if there are any 3D objects, followed by their names
>
> Example responses:
> True (for empty table)
> False: cup, bottle, plate (for table with objects)

Controller. All raw images are produced by head and wrist cameras at a resolution of $640 \times 480 \times 3$. Correspondingly, the resolution of mask is $640 \times 480 \times 1$. Through preliminary model selection, we decide to use DINOv2 ViT-B/14 as the feature extractor ϕ^{h} for head camera images and DINOv2 ViT-L/14 as the feature extractor ϕ^{w} for wrist camera images. Before feeding images into DINOv2, we resize them to $518 \times 518 \times 3$. During training, we apply domain randomization via color jittering. Finally, the images are normalized and fed into DINOv2 models. This leads to features $\mathbf{z}_t^{\mathrm{h}} \in \mathbb{R}^{1369\times768}$ and $\mathbf{z}_t^{\mathrm{w}} \in \mathbb{R}^{1369\times1024}$. By processing the mask $\mathbf{m}_t$ with a randomly initialized ViT, we extract its features $\mathbf{z}_t^{\mathrm{m}} \in \mathbb{R}^{1369\times768}$. Patch-wise concatenation of $\mathbf{z}_t^{\mathrm{h}}$ and $\mathbf{z}_t^{\mathrm{m}}$ leads to $\bar{\mathbf{z}}_t^{\mathrm{h}} \in \mathbb{R}^{1369\times1536}$. We then project $\bar{\mathbf{z}}_t^{\mathrm{h}}, \mathbf{z}_t^{\mathrm{w}}, \mathbf{s}_t$ to the same feature space of dimension 1024 with separate MLPs, yielding $\tilde{\mathbf{z}}_t^{\mathrm{h}} \in \mathbb{R}^{1369\times1024}, \tilde{\mathbf{z}}_t^{\mathrm{w}} \in \mathbb{R}^{1369\times1024}, \tilde{\mathbf{z}}_t^{\mathrm{s}} \in \mathbb{R}^{1\times1024}$, and concatenate them to form the full observation feature sequence $\tilde{\mathbf{z}}_t^{\mathrm{obs}} = (\tilde{\mathbf{z}}_t^{\mathrm{h}}, \tilde{\mathbf{z}}_t^{\mathrm{w}}, \tilde{\mathbf{z}}_t^{\mathrm{s}}) \in \mathbb{R}^{2739\times1024}$.

For action modeling, we define an action chunk horizon of $H = 64$. When we add noise to the action during training, we employ Immiscible Diffusion [83] to improve data-noise mapping. The noised action chunk $\hat{\mathbf{A}}_t$ belongs to $\mathbb{R}^{64\times13}$.

The DiT implementation is based on the original DiT paper [80], diffusion policy [81], and RDT [76]. It first embeds the diffusion timestep to the same hidden space as $\tilde{\mathbf{z}}_t^{\mathrm{obs}}$, yielding $\tilde{\mathbf{z}}_t^{\mathrm{d}} \in \mathbb{R}^{1\times1024}$, and concatenates it with $\tilde{\mathbf{z}}_t^{\mathrm{obs}}$ to form the condition sequence $\tilde{\mathbf{z}}_t = (\tilde{\mathbf{z}}_t^{\mathrm{obs}}, \tilde{\mathbf{z}}_t^{\mathrm{d}}) \in \mathbb{R}^{2740\times1024}$. We project the noised action chunk to the same hidden space, deriving $\tilde{\mathbf{z}}_t^{\mathrm{A}} \in \mathbb{R}^{64\times1024}$, and feed it into DiT. Each DiT layer performs bi-directional attention within action tokens, cross-attention to the condition sequence, and MLP projections. Finally, the output is projected back to the action space to be the model's prediction of noise. During training, we compute MSE loss between the noise prediction and ground truth, and back-propagate the gradient to update all trainable parameters. During inference, we start from Gaussian noise and iteratively denoise it using DDIM sampling [84]. At each step, the DiT model predicts the noise given the condition

Hyper-parameter	Value
epoch	84
learning rate	0.0001
learning rate scheduler	cosine
learning rate warmup steps	2000
weight decay	0.0001
AdamW betas	[0.95, 0.999]
seed	42
batch size per GPU	48
action horizon	64
number of DiT layers	12
number of DiT head	8
attention dropout	0.1
noise scheduler	DDIMScheduler
num_train_timesteps	50
beta_start	0.0001
beta_end	0.02
beta_schedule	squaredcos_cap_v2
clip_sample	True
set_alpha_to_one	True
steps_offset	0
prediction_type	epsilon
num_inference_steps	16

Table 3 | Hyper-parameters of DexGraspVLA.

sequence, and we update the action chunk using the DDIM scheduler until we obtain the final action. The controller only executes the first six actions in the predicted action chunk before making a new prediction.

In total, the controller possesses 163M trainable parameters. To accelerate training, we utilize bfloat16 mixed-precision training, reducing memory usage and improving computational efficiency. Additionally, we employ FusedAdamW as the optimizer to further speed up training through optimized memory access and fused kernel execution. With these techniques, we train the controller for 84 epochs over our dataset on an 8-A800 GPU server, which takes less than one day to complete. All hyper-parameters in our implementation are presented in Appendix A.1.

A.2. Details of Baseline Implementation

The baseline DexGraspVLA (DINOv2-train) is the same as DexGraspVLA (Ours) described in Appendix A.1 except that the two DINOv2 models are trainable instead of frozen. The baseline DexGraspVLA (ViT-small) is the same as DexGraspVLA (Ours) except that the two DINOv2 models are replaced with two small trainable pretrained ViTs (the R26-S-32 ResNet-ViT hybrid from Steiner et al. [82]). Correspondingly, we resize the images to $224 \times 224 \times 3$ to feed them into ViT-small. Each image is split into 49 patches, and the feature dimension is 384.

(a) Our data collection site.

(b) The test environment where all experiments are conducted.

Figure 7 | A comparison of the data collection and test environments, which are located in different rooms. Notably, the scenes captured by the robot's cameras vary significantly, especially for the wrist camera.

A.3. Details of Data Collection

We collect demonstrations through kinesthetic teaching. At the beginning, the robot is set to teaching mode, allowing manual guidance to grasp target objects. The operator then physically guides the robot to the target position and performs the grasping motion. The entire process follows a fixed duration (75 joint angle values recorded at 20Hz). Subsequently, we reset the environment and execute PD control using the recorded joint angles as target. At the same frequency, these target joint angles serve as actions, while images and current joint angles are collected as states. Following the same approach as the low-level controller, we post-process the collected data to generate masks, completing one demonstration sequence.

B. Experiment Details

B.1. The "Zero-Shot" Evaluation Environment

Figure 7 contrasts our data collection site and the test site, which are located in separate rooms. We gather all 2,094 human demonstrations at the data collection site (Figure 7a), whereas the experiments in Section 5

are conducted at the test site (Figure 7b). Because these sites differ in layout and background, both the head camera and the wrist camera encounter scenes not present in the training data during evaluation — particularly the wrist camera, which observes a notably altered environment, capturing a variety of front and peripheral views during operation. Despite these environmental discrepancies, we do not collect any data from the test site to fine-tune the models. Instead, the models are deployed and evaluated directly, resulting in a genuinely "zero-shot" testing environment. Even under these conditions, DexGraspVLA achieves an over 90% success rate in grasping tasks in cluttered scenes across thousands of unseen object, lighting, and background combinations, clearly demonstrating its strong generalization capability.

B.2. Additional Details of Objects, Lightings, and Backgrounds

We collect a total of 360 unseen objects, from which 103 items are randomly selected to be the *object subset* $\mathcal{S}$. In the main paper, the *Unseen Objects* experiment is conducted on all 360 objects, whereas the *Unseen Lightings* and *Unseen Backgrounds* experiments use only the objects in $\mathcal{S}$. The three unseen lighting conditions comprise disco light, lamp light, and dark light. Meanwhile, the six unseen backgrounds include a black mouse pad, a pink towel, a colorful tablecloth, a black-and-white mouse pad, a wooden board, and a calibration board. These conditions are illustrated in Figure 8.

B.3. Details of Visualization

In this part, we explain how we visualize the internal model behavior shown in Figure 6. Due to space constraints, Figure 6 only presents the relevant portion of images containing the tabletop workspace. The full version is shown in Figure 9. The first row is raw images from the head camera resized to $518 \times 518 \times 3$. The second row illustrates the DINOv2 ViT-B/14 features following the practice introduced in DINOv2 paper [20]. To make the resulting feature map recognizable for visualization purpose, we enlarge both the height and weight of images by a factor of six before feeding them into DINOv2. After obtaining the feature sequences for all four images, we combine these features, perform a PCA between all patches, and set a threshold to remove background regions. We then apply PCA again, this time to the remaining foreground features, map the top three principal components to the RGB channels, and normalize the result. This yields the visualization shown in the second row. The third row showcases the binary masks $\mathbf{m}_t \in \mathbb{R}^{518\times518\times1}$ tracked by Cutie. The fourth row displays the averaged DiT attention maps over the head image features. This is computed by summing attention weights to each head image patch across all diffusion steps, DiT layers, DiT heads, and action tokens, and normalize the sum to one. The shape of the averaged attention map is $37 \times 37 \times 1$. Finally, we upsample the attention map to $518 \times 518 \times 1$, multiply it by 2 to increase brightness, and use it to scale the value channel of head images in HSV space, resulting in the visualization shown in the fifth row.

C. Additional Results

This section provides additional results for the experiments in the main paper. In Table 4, we report the detailed success rates for our large-scale generalization evaluation under each environment condition, corresponding to Table 1 in Section 5.2. From the first row ("Ours@1"), it is evident that DexGraspVLA maintains consistently high success rates across various unseen object, lighting, and background combinations. Many observed failures stem from randomness in policy inference; allowing additional attempts often recovers these failed cases. Accordingly, the second and third rows ("Ours@2" and "Ours@3") show further improvements in performance, highlighting the potential for DexGraspVLA to reach even higher success rates. In Figure 10, we present examples of bounding-box predictions produced by the DexGraspVLA planner. Despite substantial variation in environmental conditions, the planner consistently grounds grasping instructions in cluttered scenes and provides the correct bounding boxes. Notably, we can label objects by names such as "Coca Cola" or "milk," reflecting the system's extensive common sense and world knowledge. By drawing on the broad knowledge embedded in each of its foundation models, DexGraspVLA achieves robust generalization across diverse scenarios.

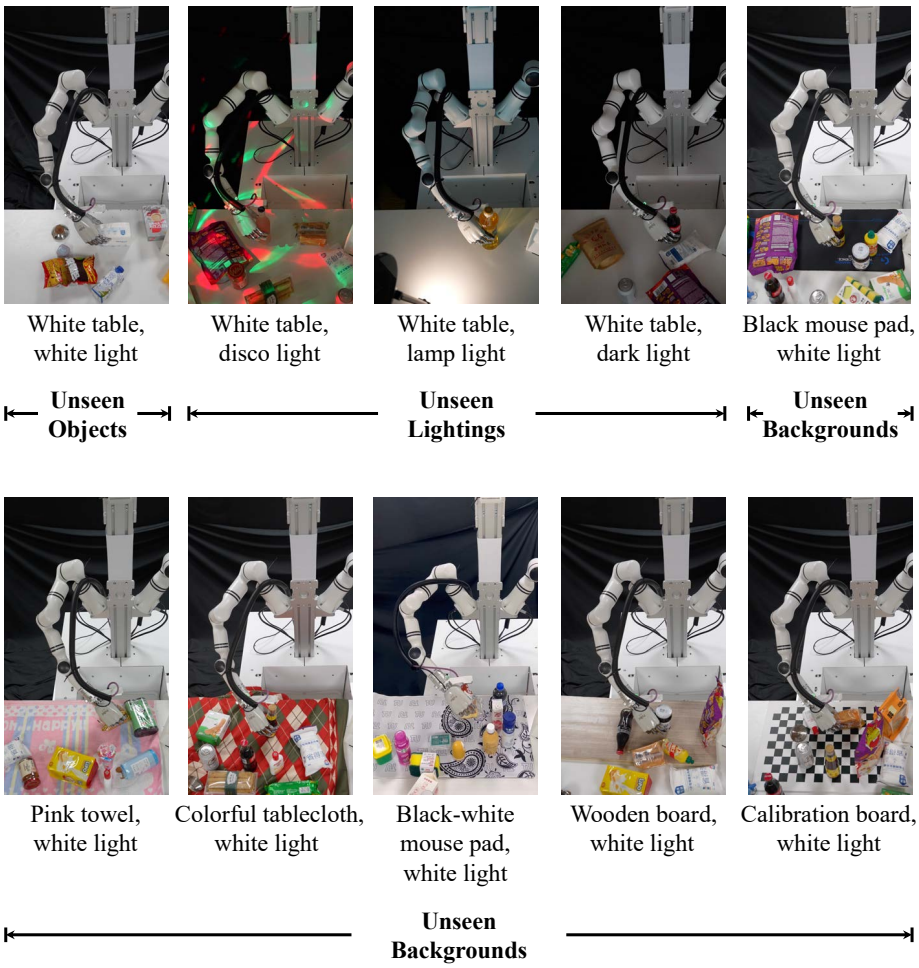

Figure 8 | Environment conditions used in our large-scale generalization evaluation (Section 5.2).

Tasks	Unseen Objects (360)	Unseen Lightings (3 × 103)			Unseen Backgrounds (6 × 103)						Aggregated (1287)
Lighting Conditions	White Light	Disco Light	Lamp Light	Dark Light	White Light	White Light	White Light	White Light	White Light	White Light	
Background Conditions	White Table	White Table	White Table	White Table	Black Mouse Pad	Pink Towel	Colorful Tablecloth	Black-White Mouse Pad	Wooden Board	Calibration Board	
Ours@1	91.1%	92.2%	89.3%	91.2%	94.2%	84.5%	90.3%	92.2%	93.2%	88.3%	90.8%
Ours@2	95.3%	97.0%	95.1%	93.2%	97.1%	90.3%	91.3%	95.1%	98.1%	93.2%	94.7%
Ours@3	96.7%	98.1%	98.1%	96.1%	98.1%	91.3%	94.2%	98.1%	100.0%	98.1%	96.9%

Table 4 | The detailed performance of DexGraspVLA under different unseen conditions, which indicates that our approach consistently achieves high success rates across various objects, lightings, and backgrounds. The second and third rows highlight its potential to reach even higher success rates given more chances.

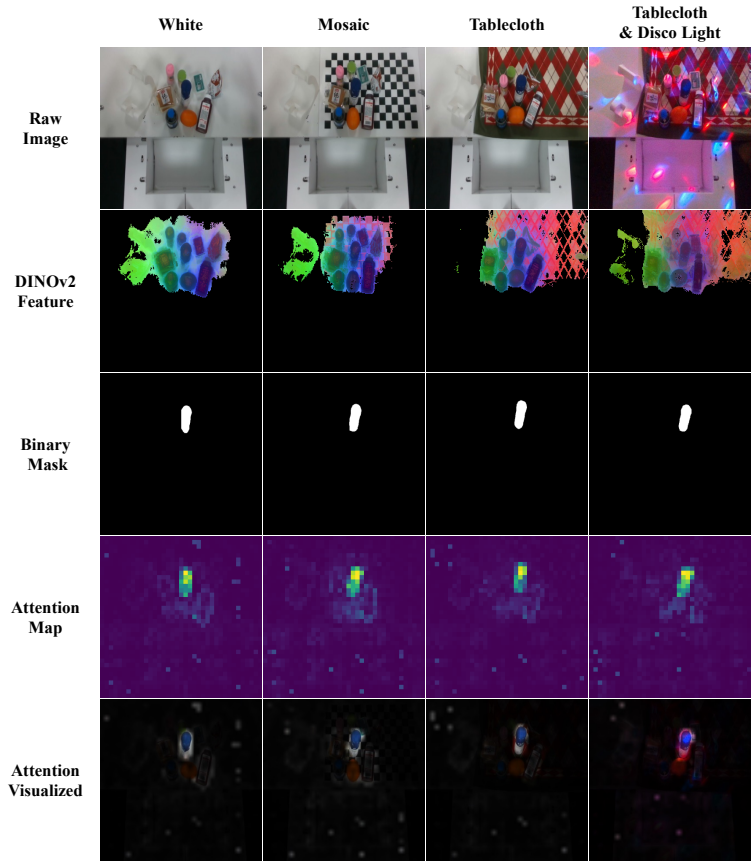

Figure 9 | The complete, uncropped version of Figure 6.

Figure 10 | Bounding-box predictions made by DexGraspVLA planner based on Qwen-VL-Chat. Across diverse lighting and background conditions, it accurately grounds the language instruction to the target object in cluttered scenes and marks the correct bounding box.