CogACT: A Foundational Vision-Language-Action Model for Synergizing Cognition and Action in Robotic Manipulation

Qixiu Li[1*†] Yaobo Liang[2*‡] Zeyu Wang[1*†] Lin Luo[2] Xi Chen[2] Mozheng Liao[3†] Fangyun Wei[2]
Yu Deng[2] Sicheng Xu[2] Yizhong Zhang[2] Xiaofan Wang[4†] Bei Liu[2] Jianlong Fu[2] Jianmin Bao[2]
Dong Chen[2] Yuanchun Shi[1] Jiaolong Yang[2‡] Baining Guo[2]
[1]Tsinghua University [2]Microsoft Research Asia [3]USTC [4]Institute of Microelectronics, CAS

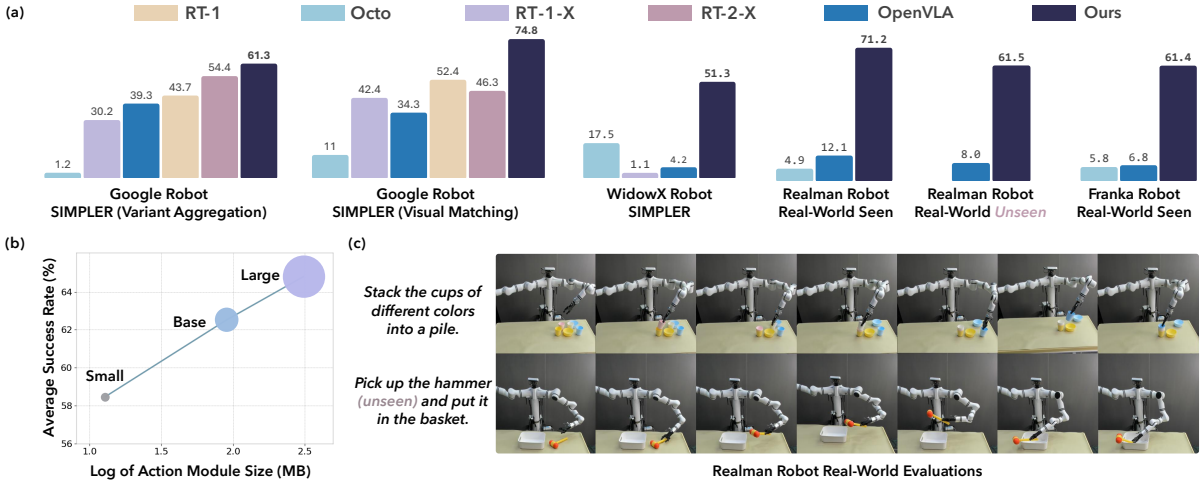

Figure 1. (a) Success rate (%) comparison of our model against RT-1 [7], RT-1-X [48], RT-2-X [48], Octo [62], and OpenVLA [30] across simulated benchmarks (first thee charts) and real-world evaluations (last three charts), using various robots: Google robot, WidowX robot, Realman robot, and Franka robot. All models are trained on the expansive Open X-Embodiment dataset [48] (expect for RT-1 that trains only on the Google Robot subset) and finetuned on a small amount of data for real robot experiments. (b) Scaling behavior: averaged success rate on SIMPLER [33] with respect to action module size. (c) Examples of Realman Robot executing tasks involving sequentially stacking multiple cups and picking and placing unseen objects.

Abstract

The advancement of large Vision-Language-Action (VLA) models has significantly improved robotic manipulation in terms of language-guided task execution and generalization to unseen scenarios. While existing VLAs adapted from pre-trained large Vision-Language-Models (VLM) have demonstrated promising generalizability, their task performance is still unsatisfactory as indicated by the low tasks success rates in different environments. In this paper, we present a new advanced VLA architecture derived from VLM. Unlike previous works that directly repurpose VLM for action prediction by simple action quantization, we propose a componentized VLA architecture that has a specialized action module conditioned on VLM output. We systematically study the design of the action module and demonstrate the strong performance enhancement with diffusion action transformers for action sequence modeling, as well as their favorable scaling behaviors. We also conduct comprehensive experiments and ablation studies to evaluate the efficacy of our models with varied designs. The evaluation on five robot embodiments in simulation and real work shows that our model not only significantly surpasses existing VLAs in task performance but also exhibits remarkable adaptation to new robots and generalization to unseen objects and backgrounds. It exceeds the average success rates of OpenVLA which has similar model size (7B) with ours by over 35% in simulated evaluation and 55% in real robot experiments. It also outperforms the large RT-2-X model (55B) by 18% absolute success rates in simulation. Code and models can be found on our project page.

*Equal Contributions
†Interns at Microsoft Research
‡Project Leads. Email: {yalia, jiaoyan}@microsoft.com

1. Introduction

In recent years, there has been a surge of interest in robotic control models equipped with visual capabilities [7, 8, 15, 30, 34, 45, 48, 58, 60, 62, 67, 69]. Among them, the development of large-scale Vision-Language-Action (VLA) models [8, 30, 32] are particularly promising, which empowers robots to perform complex tasks guided by natural language instructions and potentially manage objects or environments that deviate from the training distribution. Additionally, they exhibit rapid adaptability to new tasks and embodiments through finetuning.

The notable generalization capability of large VLAs can be attributed to both their substantial model size and the potent Vision-Language-Models (VLM) [13, 28, 35] that serve as their foundation. These VLMs are typically pretrained on massive, Internet-scale image-text pairs, which play a crucial role in enhancing VLA generalization to novel objects and semantically diverse instructions [8].

Existing large VLAs often adapt VLMs for action prediction in simple ways, leading to several issues that hinder task performance. For instance, works like [8, 30] directly quantize the continuous spectrum of robot actions into discrete bins in accordance to the next token prediction scheme of VLMs. However, such a simple quantization, unlike sophisticated tokenizers such as those designed for images [65, 72] and audio [19, 73], poses difficulties in action learning and limits action precision. [32] introduces additional action heads, such as LSTMs, to transform VLM output into actions. The shift to a regression-based learning scheme, however, overlooks the probabilistic and multimodal[1] nature of actions.

In this paper, we propose a new VLA model architecture derived from VLM. Instead of repurposing pretrained VLMs for action prediction, *we use the cognitive information extracted by VLM to guide the action prediction process of a specialized action module*. To handle the inherent characteristics of action signals – continuous, multimodal, temporally correlated, and requiring high precision – we employ advanced diffusion-based transformers (DiT) [51] as our action modules, preconditioned on VLM output via the attention mechanism.

The intuition behind our design is *the decoupling of "cognition" and "action" capabilities*. While the large VLMs amass broad visual and semantic knowledge learned from vast amounts of text and images, the cognitive capability and the output language modality have fundamental gaps to dense robot actions. Rather than directly repurposing the VLMs, we advocate the design of componentized VLAs with a dedicated action module.[2] This action module is specialized for action signal modeling with cognition model output as preconditions. We synergize the cognition and action capabilities via end-to-end training or finetuning. Hence, our approach is named *CogACT*.

We systematically study the different backbone architectures for the action module as well as their scalability on model size, and several notable insights have emerged. For example, it is found that sequential modeling with a diffusion transformer significantly outperforms single-step action prediction. More crucially, *we identified a favorable scaling behavior of the action module with diffusion transformers*: adding several hundred million parameters, which is relatively minor compared to a 7B VLM base, results in sizable performance enhancements. This finding suggests the advantages of a specialized action module and a more efficient approach for VLA model scaling.

In addition to our study on action module design, we also introduce some accompanying algorithms of independent interest. An Adaptive Action Ensemble (AAE) algorithm is proposed to fuse the past action predictions in an adaptive manner, which brings notable performance improvement. We train our VLA models on the Open X-Embodiment dataset [48], and evaluate them on both simulation [33] and real-robot benchmarks. The comprehensive evaluation and comparisons show that our model performs remarkably well, surpassing existing VLAs by a wide margin.

The contributions of this work are summarized below:

- We introduce the integration of the action diffusion process into large-scale VLA models.
- We propose a componentized VLA model architecture and study the design of large action modules[3] as well as their scaling behaviors.
- We propose an adaptive action ensemble algorithm which is simple yet effective for temporal fusion.
- Our model achieves significantly better performance than previous VLAs, exhibiting quick adaptation to new robots and tasks and effective generation to unseen objects and backgrounds, as shown in Figure 1. All our code and models are publicly released.

2. Related Works

Vision-Language-Action Models. The success of Large Language Models (LLMs) [2, 9, 63, 64] and Vision-Language Models (VLMs) [1, 14, 28, 37, 61] has inspired the development of Vision-Language-Action (VLA) models, which extend the capabilities of VLMs by integrating action generation. For instance, RoboFlamingo [32] extends OpenFlamingo [3] by incorporating a head network to predict actions and optimizing with MSE loss. RT-2 [8]

[1] A robot can follow multiple possible trajectories to accomplish a task.

[2] As an interesting analogy, our human brain has visual cortex [25], language-relevant cortex [21], and motor cortex [71] – the last being dedicated to the control of human body movements.

[3] Our largest action DiT module is of 300M parameters. Although this may seem modest, especially in comparison with large LLMs/VLMs, it is considered large given the 7D vector space of robot actions we address.

tokenizes 7D actions into discrete tokens and uses the VLM PaLI-X [13] to predict them autoregressively like language tokens. OpenVLA adopts a similar approach, tokenizing actions and training the Prismatic VLM [28] on the Open-X-Embodiment dataset [48]. While these models benefit from VLMs' capabilities and demonstrate promising performance and impressive generalization, they lack the consideration that actions are inherently continuous and temporal, a modality distinct from language. A group of methods [5, 11, 68] employ large-scale video generative pretraining to enhance visual robot manipulation learning without leveraging pretrained VLMs, and promising results have been demonstrated.

Large Action Models. There are some recent attempts *concurrent to ours* that explored large action models for generalist robots. For example, [24] trained a diffusion transformer with 221M parameters and [38] further scaled the action model size to 1B. Both works apply separate vision and language encoders that are pretrained and frozen to process language instructions and images, and they train the action model to integrate these inputs and predict actions with VLA data. Different from ours, these works cannot leverage the generalization and instruction following capability of powerful VLMs pretrained on Internet-scale vision-language aligned data.

Diffusion-Based Robot Policies. Recent studies [15, 50, 53] have introduced diffusion models as an innovative approach for modeling robotic actions. These diffusion policies have demonstrated strong capabilities to capture the multi-mode nature of robotic action distributions and effectively model the various feasible trajectories that a robot can take to accomplish a given task [15]. Inspired by diffusion policies, Octo [62] supplements a transformer-based backbone architecture with compact diffusion heads of 3M parameters to adapt the action output across different robots. However, the small diffusion head falls short in capturing the precise action distributions and the overall approach does not benefit from strong vision-language models pretrained on web-scale data. In contrast, our work studies large, dedicated action modules (rather than "heads") with the diffusion transformer architecture. Besides, unlike [15, 50, 53], we are interested in large VLAs derived from VLM foundation models with strong generalization capability.

3. Method

Problem Formulation. Our goal is to develop a VLA model that enables different robots to physically execute diverse tasks while receiving visual observations and language instructions. Formally, given the language instruction $\boldsymbol{l}$ and visual observation $\boldsymbol{o}_t$ at time t, a model $\boldsymbol{\pi}$ predicts a temporal action sequence $(\boldsymbol{a}_t, \boldsymbol{a}_{t+1}, ..., \boldsymbol{a}_{t+N})$ for executing the desired task:

$$\boldsymbol{\pi} : (\boldsymbol{l}, \boldsymbol{o}_t) \rightarrow (\boldsymbol{a}_t, \boldsymbol{a}_{t+1}, ..., \boldsymbol{a}_{t+N}). \tag{1}$$

While in general, $\boldsymbol{a}_t$ can describe various robot actions with different control modes and end-effectors, we consider the action space of a gripper with 7 degrees of freedom (DoF) in this work:

$$\boldsymbol{a}_t = [\Delta x, \Delta y, \Delta z, \Delta\phi, \Delta\theta, \Delta\psi, g], \tag{2}$$

where $\Delta x, \Delta y, \Delta z$ are the relative translation offsets of the end effector, $\Delta\phi, \Delta\theta, \Delta\psi$ denote the rotation changes, and $g \in \{0, 1\}$ indicates the gripper's open/close state.

Overall architecture. To effectively handle complex visual observations and language instructions and collaboratively transform them into precise actions, we componentize the model $\boldsymbol{\pi}$ into three parts: a *vision* module, a *language* module, and an *action* module, as shown in Fig. 2. We describe each part in details below.

3.1. Vision and Language Modules

Our vision and language modules are adapted from an existing VLM from [28] that has about 7B parameters in total, similar to [30]. We briefly describe them below for completeness.

Vision Module. The vision module processes raw images input into a set of perceptual tokens. It consists of powerful vision transformers, DINOv2 [49] and SigLIP [74], pretrained on Internet-scale image data, to capture rich visual features and a comprehensive semantic understanding of the observations. At each timestep t, the image observation $\boldsymbol{o}_t$ is fed into the two models, producing two downsampled feature maps $\boldsymbol{f}_t^{\text{DINO}}$ and $\boldsymbol{f}_t^{\text{Sig}}$, respectively. These feature maps are then concatenated along the channel dimension, passed through a linear projection layer, and serialized into a set of visual perceptual tokens, $\mathcal{V} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_{N_\mathcal{V}}\}$ with a length $N_\mathcal{V}$ (we use 256 by default).

Language Module. The language module is responsible for integrating visual information and language instructions and conducting cognitive reasoning. Here, a LLAMA-2 model [64] is applied as the backbone. The language instruction $\boldsymbol{l}$ is converted into a set of linguistic tokens, $\mathcal{T} = \{\boldsymbol{l}_1, \boldsymbol{l}_2, ..., \boldsymbol{l}_{N_\mathcal{T}}\}$, using LLAMA-2's tokenizer. These tokens are then concatenated with the visual tokens $\mathcal{V}$ and an additional learnable cognition token $\boldsymbol{c}$, and processed by the model using a causal attention mechanism. The resulting output feature $\boldsymbol{f}_t^{\boldsymbol{c}}$, corresponding to the cognition token, encodes integrated information that determines the action to be executed for the current task. This serves as a condition for the subsequent action module to interpret and derive the desired actions.

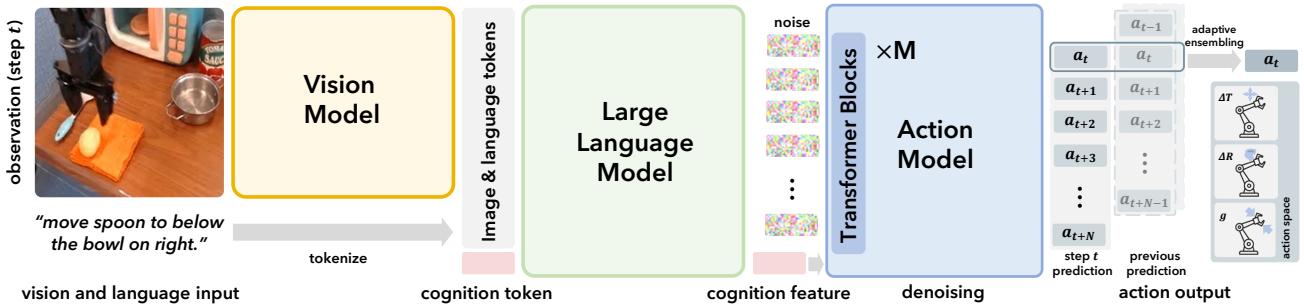

Figure 2. Overview of our architecture. Our model is componentized into three parts: 1) a vision module encoding information from the current image observation into visual tokens; 2) a language module that integrates the visual tokens with the language instructions, and produces a cognition feature determining the desired action to be executed; 3) a diffusion action module, which predicts a sequence of multi-step actions conditioned on the cognition feature. An adaptive ensemble strategy is applied for trajectory ensemble at inference.

3.2. Diffusion Action Module

The action module receives the cognition feature as an input condition to generate a series of actions, as defined in Eq. (1) and (2). Given that real-world physical actions are continuous and often multi-modal, we predict them using a diffusion modeling process [47]. To model complex and temporally-correlated actions, we apply a diffusion transformer (DiT) [51] as a powerful backbone for the action decoding process.

Specifically, our action module takes the cognition feature $\boldsymbol{f}_t^{\boldsymbol{c}}$ along with a series of noisy actions $(\boldsymbol{a}_t^i, \boldsymbol{a}_{t+1}^i, ..., \boldsymbol{a}_{t+N}^i)$ as input, where i denotes the current denoising step. It predicts the final actions $(\boldsymbol{a}_t, \boldsymbol{a}_{t+1}, ..., \boldsymbol{a}_{t+N})$ through multiple denoising steps. The cognition feature and the noisy actions serve as input tokens to the transformer blocks, while the step information i is added to the cognition feature with a sinusoidal positional encoding. We enforce the action model to predict not only the current action $\boldsymbol{a}_t$ but also multiple future actions $(\boldsymbol{a}_{t+1}, ..., \boldsymbol{a}_{t+N})$. This approach enhances the overall smoothness of the predicted actions at each time step and increases the final success rates for task execution, as observed similarly in previous studies [15, 75]. In practice, the number of predicted future actions is set to a small value ($N = 15$ by default), leading to a context length of $N + 2 = 17$ for the action module. This makes the diffusion process highly efficient and does not introduce much computational cost to the overall framework.

3.3. Training Objective

Our vision module, language module, and action module are trained/finetuned end-to-end by minimizing the mean-squared error (MSE) between the predicted noises from the action module and the ground truth noises. The loss function is defined as:

$$\mathcal{L}_{\text{MSE}} = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,1),i}||\hat{\boldsymbol{\epsilon}}^i - \boldsymbol{\epsilon}||_2, \tag{3}$$

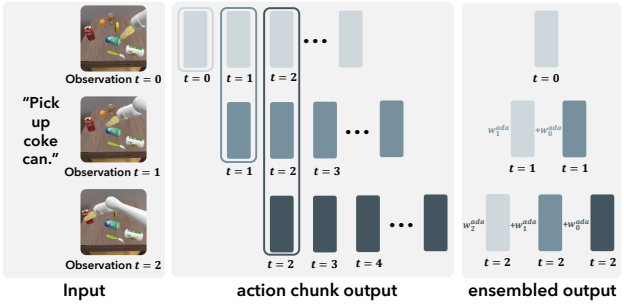

Figure 3. Illustration of our action ensemble strategy with $K = 2$ (using the last 2 historical action predictions) as an example. The action ensemble aggregates historical predictions with the current prediction to jointly determine the final action to be executed.

where $\hat{\boldsymbol{\epsilon}}^i$ is the predicted noise for the noisy action sequence $(\boldsymbol{a}_t^i, \boldsymbol{a}_{t+1}^i, ..., \boldsymbol{a}_{t+N}^i)$ at the i's denoising step, and $\boldsymbol{\epsilon}$ is the corresponding ground truth.

3.4. Adaptive Action Ensemble

During inference, our model predicts actions for multiple time steps. One straightforward strategy is to execute these actions consecutively (Action Chunking [75]) based on the current observation, $\boldsymbol{o_t}$. However, this does not fully leverage the visual information available at each time step and may result in jerky motion, as discussed in [75]. Alternatively, executing only the action for the current time step (*i.e.*, $\boldsymbol{a}_t$) also leads to a less smooth trajectory and diminished performance.

To alleviate these, [75] introduced a temporal ensemble strategy that combines actions predicted for the current time step from both present and past predictions using preset aggregation weights. However, feasible actions for task execution can belong to different modes [15], and simply aggregating them could result in an action that does not align with any mode, which is suboptimal.

We propose an adaptive ensemble strategy which consid-

ers similarities between actions to be aggregated, as shown in Fig. 3. This approach avoid unreasonable aggregation of actions from different modes. Specifically, let $\boldsymbol{a}_t|\boldsymbol{o}_t$ represent the action prediction for the current time step t given the observation $\boldsymbol{o}_t$, and $\{\boldsymbol{a}_t|\boldsymbol{o}_{t-K}, ..., \boldsymbol{a}_t|\boldsymbol{o}_{t-1}\}$ denote the corresponding action predictions based on historical observations $\{\boldsymbol{o}_{t-K}, ..., \boldsymbol{o}_{t-1}\}$. We derive the final action $\hat{\boldsymbol{a}}_t$ to be executed at time step t as:

$$\hat{\boldsymbol{a}}_t = \sum_{k=0}^{K} w_k^{\text{ada}} \cdot \boldsymbol{a}_t|\boldsymbol{o}_{t-k}. \tag{4}$$

Here, w_k^{ada} is an adaptive weighting scalar that assigns greater importance to past predictions that are more similar to the current prediction $\boldsymbol{a}_t|\boldsymbol{o}_t$:

$$w_k^{\text{ada}} = \exp(\alpha \cdot < \boldsymbol{a}_t|\boldsymbol{o}_t, \boldsymbol{a}_t|\boldsymbol{o}_{t-k} >), \tag{5}$$

where $< \cdot, \cdot >$ calculates the cosine similarity between two actions, and α is a hyperparameter set to 0.1 in practice.

Our empirical results show that this adaptive action ensemble strategy effectively boosts the success rate of task executions while adding minimal extra cost to inference, since past predictions can be readily cached.

4. Experiment

Training Dataset. We use the Open X-Embodiment (OXE) [48] dataset as our primary training dataset. It includes over 1 million real-world robotic trajectories collected from 60 datasets, covering 22 different robot embodiments. We use the similar subset of OXE as in Octo [62] and OpenVLA [30] for training, which comprises 22.5 million frames. For details regarding data distributions, please refer to [30, 62].

Implementation Details. The model is trained with a batch size of 256 and 8 diffusion steps per sample, initialized with pre-trained vision and language module weights from [30]. The vision module (*i.e.*, DINOv2 and SigLIP), language module (*i.e.*, LLAMA-2), and action module are all trained end-to-end, following a constant learning rate of $2e-5$ for over 135K iterations. Training is conducted on 16 NVIDIA A100 GPUs with approximately 5 days using PyTorch's Fully Sharded Data Parallel (FSDP) framework. By default, we use DiT-Base as our action model. The ensemble window K is set to be inversely proportional to the moving distance per frame, which can be inferred from robots' motion speed and observation frequency. In practice, we use the standard deviation of actions from training set to determine K, *i.e.*, 2 for RT-1 dataset with Google Robot, 7 for BridgeDataV2 with WidowX Robot.

4.1. Simulated Evaluation

Evaluation Environment. After training, we evaluate our model within the SIMPLER [33] evaluation environment. This simulation platform is designed to bridge the real-to-sim control and visual gap by faithfully replicating real-world conditions for robots like the Google robot and the WidowX robot. Extensive testing of various VLA models has shown a strong correlation between performance in SIMPLER and real-world outcomes [33].

Evaluation Settings. SIMPLER offers two evaluation settings: *Visual Matching*, which closely replicates real-world tasks by minimizing discrepancies between the simulated and real environments, and *Variant Aggregations*, which introduces variations to *Visual Matching* by modifying elements such as background, lighting, distractors, and table texture. For the Google robot, SIMPLER provides both the two evaluation settings, each featuring the same four tasks: 1) Pick coke can; 2) Move near; 3) Open/close drawer; and 4) Open top drawer and place apple. For the WidowX robot, SIMPLER provides only the *Visual Matching* setting, with four tasks: 1) Put spoon on towel; 2) Put carrot on plate; 3) Stack green block on yellow block; and 4) Put eggplant in yellow basket. Success rate is used as the evaluation metric.

Experiments on Google Robot. Table 1 shows the success rates of our model and compares our approach with existing VLA models on the Google robot across four tasks in both SIMPLER settings. Our model achieves the highest average success rate in both settings, with 74.8% in *Visual Matching* and 61.3% in *Variant Aggregation*. Remarkably, our model even outperforms RT-1 [7], which is trained on a Google robot-specific dataset, by an average success rate margin of 22.4% in *Visual Matching* and 17.6% in *Variant Aggregation*. Additionally, the average success rate of our model significantly surpasses that of RT-2-X [48], despite our model being much smaller, with 7.6B parameters compared to RT-2-X that has 55B parameters.

Experiments on WidowX Robot. Table 2 presents the evaluation results of our model compared with the other methods on the WidowX robot using the SIMPLER environment in the *Visual Matching* setting. Our model also attains the highest average success rate of 51.3%, outperforming the other models by a significant margin.

4.2. Real-World Evaluation with Realman Robot

Robot. We perform real-world experiments using the Realman Arm[4], which has 7 DoFs and is equipped with a 1-DoF gripper. Inverse kinematics is applied to determine joint angles, enabling the end effector to align with the model's predicted pose. **Task Definitions.** We design three tasks for real-world experiments, named "Pick", "Stack", and "Place":

- Pick up the `Object` and place it onto the `Color` plate, where `Object` $\in$ {Banana, Lemon, Avocado},

[4]https://www.realman-robotics.com/rm75-b.html

Table 1. Comparison of our approach with existing VLA models on the Google robot across four tasks in two SIMPLER settings. All models are trained on the Open X-Embodiment dataset, except for RT-1 which is trained exclusively on the Google robot subset.

Google Robot	Method	Pick Coke Can	Move Near	Open/Close Drawer	Open Top Drawer and Place Apple	*Average*
SIMPLER (Visual Matching)	RT-1 [7]	85.7	44.2	73.0	6.5	52.4
	RT-1-X [48]	56.7	31.7	59.7	21.3	42.4
	RT-2-X [48]	78.7	77.9	25.0	3.7	46.3
	Octo-Base [62]	17.0	4.2	22.7	0.0	11.0
	OpenVLA [30]	18.0	56.3	63.0	0.0	34.3
	Ours	**91.3**	**85.0**	**71.8**	**50.9**	**74.8**
SIMPLER (Variant Aggregation)	RT-1 [7]	89.8	50.0	32.3	2.6	43.7
	RT-1-X [48]	49.0	32.3	29.4	10.1	30.2
	RT-2-X [48]	82.3	79.2	**35.3**	20.6	54.4
	Octo-Base [62]	0.6	3.1	1.1	0.0	1.2
	OpenVLA [30]	60.8	67.7	28.8	0.0	39.3
	Ours	**89.6**	**80.8**	28.3	**46.6**	**61.3**

Table 2. Evaluation results on the WidowX robot in the SIMPLER *Visual Matching* setting. For these tests, we repeat each task 5 times to improve the statistical significance (see *suppl. material* for details), and thus the results of Octo may slightly differ from those in [33].

WidowX Robot	Method	Put Spoon on Towel	Put Carrot on Plate	Stack Green Block on Yellow Block	Put Eggplant in Yellow Basket	*Average*
SIMPLER (Visual Matching)	RT-1-X [48]	0.0	4.2	0.0	0.0	1.1
	Octo-Base [62]	15.8	12.5	0.0	41.7	17.5
	Octo-Small [62]	41.7	8.2	0.0	56.7	26.7
	OpenVLA [30]	4.2	0.0	0.0	12.5	4.2
	Ours	**71.7**	**50.8**	**15.0**	**67.5**	**51.3**

and `Color` $\in$ {White, Blue, Yellow}.
- Stack the `Color Object` into the `Color Object`, where `Object` $\in$ {Cup, Bowl} and `Color` $\in$ {Pink, White, Blue, Yellow}.
- Place the `Color` block onto the `Color` block, where `Color` $\in$ {Red, Orange, Blue, Green, Yellow}.

Fine-Tuning Data. We manually collect 48, 67 and 79 demonstrations for the "Pick", "Stack", and "Place" tasks, respectively. Additionally, we capture 197 demonstrations for various other tasks like "Pick up the orange can and drop it into the basket". In total, we gather 391 demonstrations for fine-tuning. Notably, demonstrations for tasks identical to the evaluation settings are very few. For example, there are only two demonstrations for the specific task "put the banana on the yellow plate." Details of the fine-tuning process are provided in the *suppl. material*.

Evaluation. Each task is evaluated as follows:

- "Pick": 24 evaluations in total, with 8 trials per object (Banana, Lemon and Avocado). We report the success rate for each object, as well as the average success rate across the three objects.
- "Stack": 48 evaluations in total, with 24 trials per object (Cup and Bowl). We report the success rate for each object, along with the average success rate across both objects.
- "Place": 24 evaluations in total. This task is divided into two steps: 1) picking up the block, and 2) placing the block onto another block. We report the success rate for each step, and the average success rate across both steps.

In each evaluation, we randomly choose the `Color`, and place both the target and at least three distractor objects in random positions on the table. We vary the table height within the robot's accessible range, and randomly set the robot's position around the table.

Results. Table 3 presents a comparison of our model with Octo-Base [62] and OpenVLA [30]. For a fair evaluation, all models are pre-trained on the OXE dataset and subsequently fine-tuned using our collected demonstrations. Our model achieves a notable improvement, outperforming OpenVLA by a margin of 59.1% in success rate.

Generalization Evaluation. We assess the generalization capability of the fine-tuned models on environments, distractors, tables, and objects that are not present in the fine-tuning dataset. We exclude Octo-Base [62] from this evaluation since its success rate in the seen task is already close to zero.

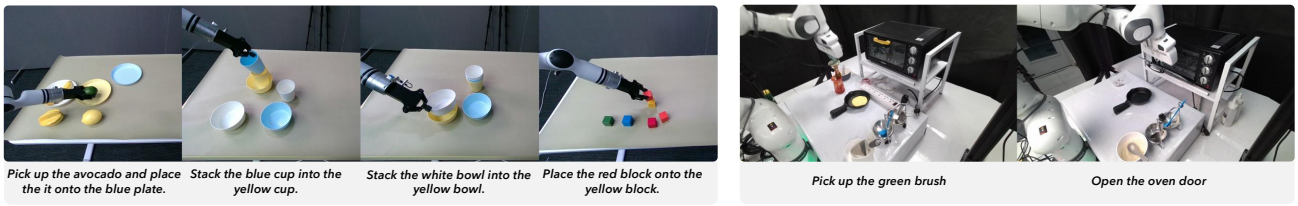

Figure 4. Real-world evaluation environments of Realman robot (left) and Franka robot (right).

Table 3. Real-world evaluation with the Realman Robot across three tasks. All models are pre-trained on OXE and then fine-tuned on our collected data.

Method	Pick				Stack			Place			Task (All)
	Banana	Lemon	Avocado	Avg.	Cup	Bowl	Avg.	Pick	Stack	Avg.	Avg.
Octo-Base [62]	25.0	0.0	0.0	8.3	0.0	0.0	0.0	12.5	0.0	6.3	4.9
OpenVLA [30]	12.5	12.5	0.0	8.3	25.0	6.3	15.6	25.0	4.2	12.5	12.1
Ours	**75.0**	**50.0**	**87.5**	**70.8**	**95.8**	**68.8**	**82.3**	**87.5**	**33.3**	**60.4**	**71.2**

Table 4. Real-world generalization evaluation with the Realman Robot on *unseen* tables with additional *unseen* distractors.

Method	Pick				Stack			Place			Task (All)
	Banana	Lemon	Avocado	Avg.	Cup	Bowl	Avg.	Pick	Stack	Avg.	Avg.
OpenVLA [30]	12.5	0.0	12.5	8.3	12.5	0.0	6.3	29.2	0.0	14.6	9.7
Ours	**75.0**	**62.5**	**87.5**	**75.0**	**83.3**	**45.8**	**64.6**	**54.2**	**16.7**	**35.5**	**58.4**

Table 5. Real-world generalization evaluation with the Realman Robot on unseen colors, shapes and categories.

Method	Unseen Colors	Unseen Shapes	Unseen Categories	Avg.
OpenVLA [30]	0.0	6.3	12.5	6.3
Ours	**87.5**	**81.3**	**25.0**	**64.6**

- *Unseen Tables with Unseen Distractors.* The table color is altered to closely resemble the colors of the background and floor, increasing visual complexity. Additionally, three new unseen objects are introduced as distractors. The results for each task is presented in Table 4.
- *Unseen Colors, Shapes, and Categories.* We assess performance on unseen colors in the "Pick" task, using seen objects (Banana, Avocado) with new colors (Green, Pink) of plates. Each object-color combination is evaluated twice, for a total of 8 trials.
  For evaluating on unseen shapes, we define four new tasks: (1-2) "Pick up the `Shape` block and put it into the basket", where `Shape` $\in$ {Triangular, Arched}; (3) "Pick up a small can and drop it into the basket"; (4) "Stack one cylindrical block on top of another cylindrical block". Each task is evaluated 4 times, resulting in a total of 16 trials.
  For evaluating unseen categories, we introduce a new task: "Pick up the `Object` and place it into the basket", where `Object` $\in$ {Eggplant, Hammer}. This task is evaluated 8 times in total, with 4 trials per object.
  Table 5 presents the results, demonstrating that our approach exhibits strong generalization capabilities, particularly in handling unseen colors and shapes.

Table 6. Real-world evaluation with the Franka Robot across four tasks. All models are pre-trained on OXE and then fine-tuned on our collected data.

Method	Close Oven	Open Oven	Pick Bowl	Pick Brush	Avg.
Octo-Base [62]	0.0	0.0	27.3	0.0	5.8
OpenVLA [30]	18.2	0.0	9.1	0.0	6.8
Ours	**63.6**	**72.7**	**72.7**	**36.4**	**61.4**

4.3. Real-World Evaluation with Franka Robot

Robot. To further demonstrate the generalization capability of our approach, we employ an additional robot, the Franka Arm[5], which features 7 DoFs and a 1-DoF gripper, for real-world evaluation.

Task Definitions. We define four tasks for evaluation: 1) Close the oven door; 2) Open the oven door; 3) Pick up the green brush; 4) Pick up a bowl containing food.

[5]https://franka.de/

Table 7. Performance comparison of different action model network structures.

Action Model	Params	GR (VM)	GR (VA)	WR (VM)	Average
MLP (3-Layer)	3M	52.2	52.4	47.1	50.6
MLP (7-Layer)	89M	61.4	48.0	48.1	52.5
DiT-Small	13M	73.3	51.3	51.0	58.5
DiT-Base	89M	74.8	**61.3**	51.3	62.5
DiT-Large	308M	**76.7**	59.3	**58.3**	**64.8**

Fine-Tuning Data. For each task, we collect 100 demonstration samples, resulting in a total of 400 samples used to fine-tune all the models.

Results. Each task is tested over 11 trials. Table 6 presents a comparison between our model, Octo-Base, and OpenVLA, demonstrating that our approach achieves significantly higher performance. Real-world results with the Realman Robot (Table 3) and the Franka Robot (Table 6) consistently validate the generalization capability of our approach across different robotic platforms.

4.4. Ablation Study

We employ SIMPLER evaluation on both the Google robot and the WidowX robot for all ablation studies. We use the following abbreviations: GR for Google Robot, WR for WidowX Robot, VM for the SIMPLER Visual Matching setting, and VA for the SIMPLER Visual Aggregation setting.

Action Model Architectures. We evalute various action model architectures on GR and WR. The architectures examined include Multi-Layer Perceptrons (MLPs) with depths of 3 and 7 layers, respectively, as well as a series of diffusion transformers of varying scales. The hidden state dimensions are set to 256 and 1024 for the two MLPs.

Table 7 shows that both MLP and transformer structures show improved success rate with increased model size. With the same number of parameters, transformers outperform MLPs, likely due to the attention mechanism's superior sequence modeling capabilities. Notably, DiT-Large achieves the highest average success rate of 64.8%. As shown in Figure 1, the average success rate of transformers is approximately linearly related to the logarithm of the model size. This indicates a promising scaling behavior of the action module with diffusion transformers.

Multi-Step Action Prediction. During training, our model predicts the action for the current step along with N future steps. In Table 8, we examine the effect of varying N (0, 3, 15 and 31 future steps) on performance. Our results indicate that predicting 15 future steps leads to the highest performance.

Table 8. Impact of multi-step action prediction on model performance. A future step setting of 0 indicates no multi-step action prediction during training.

Future Steps	GR (VM)	GR (VA)	WR (VM)	Average
0	73.4	49.0	6.3	42.8
3	70.4	58.9	37.1	55.5
15	**74.8**	**61.3**	51.3	**62.5**
31	54.3	47.6	**51.7**	51.2

Table 9. Comparison of our proposed action ensemble strategy, Adaptive Ensemble, against two strategies introduced in [75].

Strategy	GR (VM)	GR (VA)	WR (VM)	Average
Action Chunking	67.4	52.5	32.1	50.7
Temporal Ensemble	**75.0**	59.9	41.9	58.9
Adaptive Ensemble	74.8	**61.3**	**51.3**	**62.5**

Adaptive Action Ensemble. In Section 3.4, we present an action ensemble strategy, termed Adaptive Ensemble, as formulated in Eq. (4). We evaluate this approach against the two ensemble strategies introduced in ACT [75]: Action Chunking and Temporal Ensemble. Given a sequence of action predictions of current observation, Action Chunking executes the first two predictions directly. We use the same ensemble windows for Temporal Ensemble and our Adaptive Ensemble. Table 9 presents the success rates of these strategies. Our proposed Adaptive Ensemble outperforms others, and we attribute this to its integration of similarity weighting between current and historical predictions.

5. Conclusion

We have presented a large VLA model designed with a specialized focus on action modeling. Unlike previous VLAs that employ simple adaptations of vision-language models for action prediction, CogACT separates cognitive and action capabilities, using advanced diffusion transformers as a dedicated action module. This approach effectively addresses the continuous, multimodal, and temporally correlated nature of robot actions, leading to substantial improvements in performance and generalization. Our findings highlight the advantages of a componentized VLA model, where a large VLM serves as the cognitive foundation while the DiT action module handles precise, sequential action prediction. Favorable scaling behaviors of the action module have been observed, where modest parameter increases yield significant performance gains. The extensive experiments show that our model not only significantly surpasses existing VLAs in task performance and but also exhibits remarkable generation capability to unseen objects and backgrounds.

References

[1] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024. 2

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 2

[3] Anas Awadalla, Irena Gao, Josh Gardner, Jack Hessel, Yusuf Hanafy, Wanrong Zhu, Kalyani Marathe, Yonatan Bitton, Samir Gadre, Shiori Sagawa, et al. Openflamingo: An open-source framework for training large autoregressive vision-language models. *arXiv preprint arXiv:2308.01390*, 2023. 2

[4] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Hydra: Hybrid robot actions for imitation learning. *arxiv*, 2023. 13

[5] Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. *arXiv preprint arXiv:2409.16283*, 2024. 3

[6] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. 13

[7] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. 1, 2, 5, 6

[8] 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. 2

[9] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 2

[10] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015. 14

[11] 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. 3

[12] Lawrence Yunliang Chen, Simeon Adebola, and Ken Goldberg. Berkeley UR5 demonstration dataset. https://sites.google.com/view/berkeley-ur5/home. 13

[13] Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, et al. Pali-x: On scaling up a multilingual vision and language model. *arXiv preprint arXiv:2305.18565*, 2023. 2, 3

[14] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198, 2024. 2

[15] 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*, page 02783649241273668, 2023. 2, 3, 4

[16] Nikolaus Correll, Kostas E Bekris, Dmitry Berenson, Oliver Brock, Albert Causo, Kris Hauser, Kei Okada, Alberto Rodriguez, Joseph M Romano, and Peter R Wurman. Analysis and observations from the first amazon picking challenge. *IEEE Transactions on Automation Science and Engineering*, 15(1):172–188, 2016. 14

[17] Zichen Jeff Cui, Yibin Wang, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. From play to policy: Conditional behavior generation from uncurated robot data. *arXiv preprint arXiv:2210.10047*, 2022. 13

[18] Shivin Dass, Jullian Yapeter, Jesse Zhang, Jiahui Zhang, Karl Pertsch, Stefanos Nikolaidis, and Joseph J. Lim. Clvr jaco play dataset, 2023. 13

[19] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022. 2

[20] Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021. 13

[21] Angela D Friederici. The brain basis of language processing: from structure to function. *Physiological reviews*, 91 (4):1357–1392, 2011. 2

[22] Minho Heo, Youngwoon Lee, Doohyun Lee, and Joseph J. Lim. Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation. In *Robotics: Science and Systems*, 2023. 13

[23] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 14, 16

[24] Zhi Hou, Tianyi Zhang, Yuwen Xiong, Hengjun Pu, Chengyang Zhao, Ronglei Tong, Yu Qiao, Jifeng Dai, and Yuntao Chen. Diffusion transformer policy. *arXiv preprint arXiv:2410.15959*, 2024. 3

[25] Trevor Huff, Navid Mahabadi, and Prasanna Tadi. Neuroanatomy, visual cortex. 2018. 2

[26] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022. 13

[27] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly,

Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018. 13

[28] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. *arXiv preprint arXiv:2402.07865*, 2024. 2, 3, 13

[29] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. Droid: A large-scale in-the-wild robot manipulation dataset. 2024. 13

[30] 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. 1, 2, 3, 5, 6, 7, 13, 15

[31] Eric Krotkov, Douglas Hackett, Larry Jackel, Michael Perschbacher, James Pippine, Jesse Strauss, Gill Pratt, and Christopher Orlowski. The darpa robotics challenge finals: Results and perspectives. *The DARPA robotics challenge finals: Humanoid robots to the rescue*, pages 1–26, 2018. 14

[32] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. *arXiv preprint arXiv:2311.01378*, 2023. 2

[33] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024. 1, 2, 5, 6, 14, 15

[34] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. In *The Twelfth International Conference on Learning Representations*, 2024. 2

[35] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2023. 2

[36] Huihan Liu, Soroush Nasiriany, Lance Zhang, Zhiyao Bao, and Yuke Zhu. Robot learning on the job: Human-in-the-loop autonomy and learning during deployment. In *Robotics: Science and Systems (RSS)*, 2023. 13

[37] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024. 2

[38] 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. 3

[39] Jianlan Luo, Charles Xu, Xinyang Geng, Gilbert Feng, Kuan Fang, Liam Tan, Stefan Schaal, and Sergey Levine. Multi-stage cable routing through hierarchical imitation learning. *arXiv pre-print*, 2023. 13

[40] Jianlan Luo, Charles Xu, Fangchen Liu, Liam Tan, Zipeng Lin, Jeffrey Wu, Pieter Abbeel, and Sergey Levine. Fmb: a functional manipulation benchmark for generalizable robotic learning. *arXiv preprint arXiv:2401.08553*, 2024. 13

[41] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023. 13

[42] Ajay Mandlekar, Jonathan Booher, Max Spero, Albert Tung, Anchit Gupta, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1048–1055. IEEE, 2019. 13

[43] Oier Mees, Jessica Borja-Diaz, and Wolfram Burgard. Grounding language with visual affordances over unstructured data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, London, UK, 2023. 13

[44] Russell Mendonca, Shikhar Bahl, and Deepak Pathak. Structured world models from human videos. *CoRL*, 2023. 13

[45] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, pages 892–909, 2023. 2

[46] Soroush Nasiriany, Tian Gao, Ajay Mandlekar, and Yuke Zhu. Learning and retrieval from prior data for skill-based imitation learning. In *Conference on Robot Learning (CoRL)*, 2022. 13

[47] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 4

[48] 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. 1, 2, 3, 5, 6, 13, 15

[49] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. 3, 13

[50] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023. 3

[51] 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. 2, 4

[52] Gabriel Quere, Annette Hagengruber, Maged Iskandar, Samuel Bustamante, Daniel Leidner, Freek Stulp, and Joern Vogel. Shared Control Templates for Assistive Robotics. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, page 7, Paris, France, 2020. 13

[53] Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023. 3

[54] Erick Rosete-Beas, Oier Mees, Gabriel Kalweit, Joschka Boedecker, and Wolfram Burgard. Latent plans for task agnostic offline reinforcement learning. 2022. 13

[55] Saumya Saxena, Mohit Sharma, and Oliver Kroemer. Multi-resolution sensing for real-time control with vision-language models. In *7th Annual Conference on Robot Learning*, 2023. 13

[56] Nur Muhammad Mahi Shafiullah, Anant Rai, Haritheja Etukuru, Yiqian Liu, Ishan Misra, Soumith Chintala, and Lerrel Pinto. On bringing robots home, 2023. 13

[57] Rutav Shah, Roberto Martín-Martín, and Yuke Zhu. MUTEX: Learning unified policies from multimodal task specifications. In *7th Annual Conference on Robot Learning*, 2023. 13

[58] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023. 2

[59] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 14

[60] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, et al. Open-world object manipulation using pre-trained vision-language models. In *Conference on Robot Learning*, pages 3397–3417, 2023. 2

[61] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 2

[62] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024. 1, 2, 3, 5, 6, 7, 13, 15

[63] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 2

[64] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 2, 3, 13

[65] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 2

[66] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023. 13

[67] Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Kun Wu, Zhiyuan Xu, Ran Cheng, Chaomin Shen, Yaxin Peng, Feifei Feng, et al. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2409.12514*, 2024. 2

[68] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. *arXiv preprint arXiv:2312.13139*, 2023. 3

[69] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *The Twelfth International Conference on Learning Representations*, 2024. 2

[70] Ge Yan, Kris Wu, and Xiaolong Wang. ucsd kitchens Dataset. 2023. 13

[71] Derek W Yip, Ayoola O Awosika, and Forshing Lui. Physiology, motor cortical, 2024. 2

[72] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023. 2

[73] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-

to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021. 2

[74] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11975–11986, 2023. 3, 13

[75] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 4, 8

[76] Gaoyue Zhou, Victoria Dean, Mohan Kumar Srirama, Aravind Rajeswaran, Jyothish Pari, Kyle Hatch, Aryan Jain, Tianhe Yu, Pieter Abbeel, Lerrel Pinto, et al. Train offline, test online: A real robot learning benchmark. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9197–9203. IEEE, 2023. 13, 14

[77] Xinghao Zhu, Ran Tian, Chenfeng Xu, Mingyu Ding, Wei Zhan, and Masayoshi Tomizuka. Fanuc manipulation: A dataset for learning-based manipulation with fanuc mate 200id robot. 2023. 13

[78] Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. Viola: Imitation learning for vision-based manipulation with object proposal priors. *6th Annual Conference on Robot Learning (CoRL)*, 2022. 13

[79] Yifeng Zhu, Peter Stone, and Yuke Zhu. Bottom-up skill discovery from unsegmented demonstrations for long-horizon robot manipulation. *IEEE Robotics and Automation Letters*, 7(2):4126–4133, 2022. 13

CogACT: A Foundational Vision-Language-Action Model for Synergizing Cognition and Action in Robotic Manipulation

Supplementary Material

A. Training Data Details

A.1. Pretraining Data

Vision-Language Data. Our VLA model is built upon a pretrained VLM, Prismatic [28]. Here, we briefly describe the training data of [28] for self-containedness purposes and refer the readers to [28, 49, 64, 74] for more details. The Prismatic model uses DINOv2 [49] and SigLIP [74] as the vision modules, which were trained with 1.2 billion images and 40 billion image-text pairs, respectively. A LLaMa-2 [64] is employed as the LLM backbone, which was trained on 2 trillion language tokens. The vision and LLM modules were further finetuned with 1.2 million multimodal instruct tuning examples by [28].

Vision-Language-Action Data. We pretrain our model on 25 VLA datasets from Open X-Embodiment [48]. As in Octo [62] and OpenVLA [30], we restrict our training on datasets with single-arm end-effector control and at least one third-person camera perspective. Our data mixture strategy primarily follows [30, 62], except that we do not use the Language Table [41] and Droid [29] datasets in our entire training process due to their significant distribution disparities with other data. The detailed data mixture is listed in Table I. In total, we use 0.4 million robot trajectories containing 22.5 million frames as our training data.

Table I. Our training data mixture using datasets from the Open X-Embodiment dataset [48].

Dataset	Ratio
Fractal [6]	27.1%
Kuka [27]	14.7%
Bridge [20, 66]	15.3%
Taco Play [43, 54]	3.4%
Jaco Play [18]	0.6%
Berkeley Cable Routing [39]	0.3%
Roboturk [42]	2.7%
Viola [78]	1.1%
Berkeley Autolab UR5 [12]	1.4%
Toto [76]	2.3%
Stanford Hydra Dataset [4]	5.1%
Austin Buds Dataset [79]	0.2%
NYU Franka Play Dataset [17]	1.0%
Furniture Bench Dataset [22]	2.8%
UCSD Kitchen Dataset [70]	<0.1%
Austin Sailor Dataset [46]	2.5%
Austin Sirius Dataset [36]	2.0%
DLR EDAN Shared Control [52]	<0.1%
IAMLab CMU Pickup Insert [55]	1.0%
UTAustin Mutex [57]	2.6%
Berkeley Fanuc Manipulation [77]	0.9%
CMU Stretch [44]	0.2%
BC-Z [26]	8.6%
FMB Dataset [40]	2.4%
DobbE [56]	1.6%

A.2. Finetuning Data for Real Robot Experiments

Realman Robot Setup. Our hardware setup for the Realman robot experiments is presented in Figure I. We built a robot with two Realman arms on its shoulders, each having 7 degrees of freedom (DoF). We connect the left arm with a 1-DoF gripper and only use this arm in all experiments. We use Intel RealSense camera to capture the RGB image. We perform hand-eye calibration each time we move the camera. We convert all actions in the training data to the camera coordinate system, and the actions generated by the model to the robot coordinate system. To evaluate the generalization ability of our model across different viewpoints, we randomly move the camera before our experiments. Our robot is equipped with a mobile base, which was repositioned randomly before each experiment in order to assess the model's generalization ability.

Franka Robot Setup. The Franka Robot setup is shown in Figure II. For this embodiment, a robot arm which has 7 DoF is rigidly attached onto a table. We use a Kinect DK camera on the right to capture RGB images. The actions throughout the entire pipeline are in the robot coordinate system.

Data Collection. We use the touch controller from a Meta Quest 2 device to teleoperate the robot and collect demonstration data for both robots. The translation and rotation of the touch controller are mapped to the gripper's 3D motion, and a button controls the opening and closing of the gripper. We record video frames at 30 Hz for the Realman robot and 5$\sim$6 Hz for the Franka robot. We manually trim the video segments from the recorded full videos and append the language instructions describing the tasks.

Data Preprocessing. We follow the format of the Open X-Embodiment dataset [48] to process both the two fine-

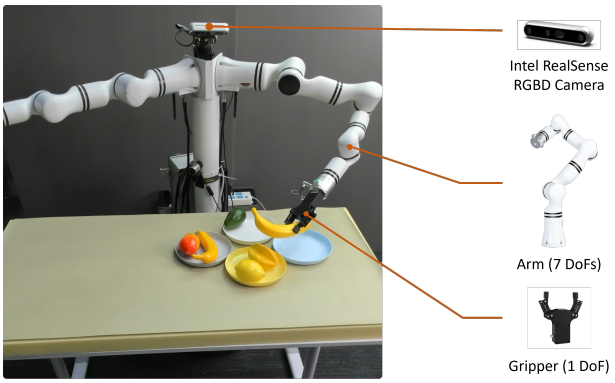

Figure I. Realman robot setup (right arm not used).

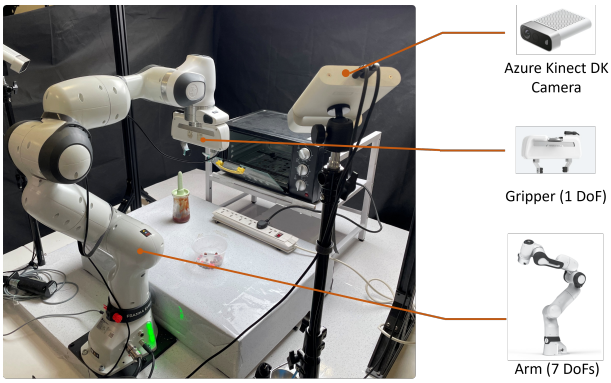

Figure II. Franka robot setup.

tuning datasets of the Realman robot and Franka robot. We crop and resize the image to 224×224, and transform the actions to relative translation offset and rotation changes. We calculate the rotation changes in rotation matrix and transform them to Euler angle for training. During training, the Realman robot data is randomly subsampled to 5 Hz with a time interval of 0.2 seconds between frames. The Franka robot data is used as is.

A.3. Data Augmentation and Normalization

We utilize data augmentations for images like random crop, random brightness, random contrast, random saturation and random hue. When predicting future actions, for those that exceed the length of the demonstration, we use a stationary action for padding. For the normalization of actions, we normalize them to the range of $[-1, 1]$. We also tried normalizing the actions to a Gaussian distribution $\mathcal{N}(0, 1)$, but did not observe any performance improvement.

B. Evaluation Details

B.1. Simulated Evaluation

Evaluating robotic manipulation tasks is challenging due to the varying hardware, environments, and tasks used across different studies [33]. While standardized real-world setups can be beneficial, they often require substantial time and resources [10, 16, 31, 76]. To address these issues while preserving the accuracy of real-world performance evaluations, SIMPLER [33] has developed a system for evaluating manipulation policies trained on real data in simulated environments. Therefore, we employs SIMPLER for our evaluations, offering an easily reproducible and completely fair assessment framework.

Table II. Simulated evaluation tasks and their number of trials in SIMPLER.

Task	# Trials (VM)	(VA)
Pick Coke Can	300	825
Move Near	240	600
Open/Close Drawer	216	378
Open Top Drawer and Place Apple	108	189
Put Spoon on Towel	120 (24×5)	N/A
Put Carrot on Plate	120 (24×5)	N/A
Stack Green Block on Yellow Block	120 (24×5)	N/A
Put Eggplant in Yellow Basket	120 (24×5)	N/A

Task Definitions. We utilize all task variants provided in SIMPLER for our evaluations. The Google robot setup includes the following tasks: 1) "pick Coke can", 2) "move {obj1} near {obj2}", 3) "(open / close) (top / middle / bottom) drawer", and 4) "open top drawer; place apple into top drawer". The WidowX robot setup includes: 1) "put the spoon on the towel", 2) "put carrot on plate", 3) "stack the green block on the yellow block", and 4) "put eggplant into yellow basket". For the Google robot setup, both *Visual Matching* (VM) and *Variant Aggregations* (VA) evaluations are provided (See Sec. 4.1 in the main paper), while only the *Visual Matching* (VM) evaluation is provided for the WidowX robot setup.

The total number of trials for each sub-task is shown in Table II. Note that to accommodate the limited number of the original trials (24 per task) on the WidowX robot, we repeat each original task trial five times (with different random initialization seeds) to enhance the statistical significance. For more detailed information on the tasks and evaluation protocols, we refer the readers to the SIMPLER paper [33].

Implementation Details. All simulated evaluations are conducted on a single NVIDIA A6000 GPU or a single NVIDIA A100 GPU. During inference, we employ DDIM [59] sampling with 10 sampling steps and a classifier-free guidance (CFG) [23] coefficient of 1.5.

We apply a unified adaptive action ensemble strategy (described in Sec. 3.4 in the main paper) across all our models, where the hyperparameter α is set to 0.1. The ensemble window size K determines the number of historical obser-

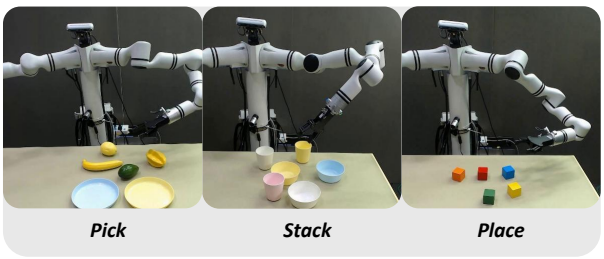

Figure III. Seen task setups on the Realman robot.

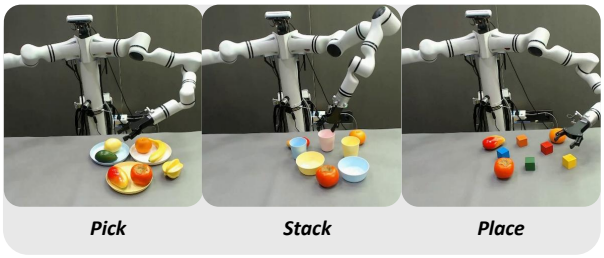

Figure IV. Unseen task setups on the Realman robot.

vations and their predicted actions to be used. Since different dataset have varying control frequency and robot speed, the window size K should be adaptive. We select the window size K such that the product of K and the average standard deviation (std) of the 6D action per timestep remains constant across different datasets. Formally, this relationship can be expressed as $C = K \times std$, where C is a constant representing the distance and angle traversed by the robot over the last K steps. Empirically, we set the C to 0.2 for all experiments and derive their K accordingly.

For the experiments on the Google robot, the task success rates of the pervious methods were evaluated by [33] and incorporated into this paper, except for OpenVLA [30], which was not included in [33]. For OpenVLA, we directly evaluate it using the weight from its official repository[6]. On the WidowX robot, the performance of RT-1-X [48] is directly reported as in [33] and OpenVLA is evaluated as described above. For Octo-Base and Octo-Small [62], we incorporate more random seeds and conduct five retests to mitigate volatility in their probabilistic sampling. For all configurations of our models, we evaluate them using the same number of tests as those used for Octo. Note that the performance of RT-2-X on WidowX has not been reported [33], and no model weights are publicly available for evaluation.

Visualization Results. Figure V provides the successful examples of each task executed by our default model on the Google robot, while Figure VI presents those on the WidowX robot.

[6] https://huggingface.co/openvla/openvla-7b-prismatic

Table III. The hyperparameters of different action modules.

Action Model	# Layers	Emb Dim	# Heads	# Params
MLP (3-Layer)	3	256	N/A	3M
MLP (7-Layer)	7	1024	N/A	89M
DiT-Small	6	384	4	13M
DiT-Base	12	768	12	89M
DiT-Large	24	1024	16	308M

B.2. Real-World Evaluation

Task Definitions. Section 4.2 in the main paper defines three tasks "Pick", "Stack" and "Place" on the Realman robot. The setups of these tasks in the seen environment are illustrated in Figure III, while those on the unseen table are illustrated in Figure IV. Section 4.3 in the main paper defines four tasks: "Close the oven door", "Open the oven door", "Pick up the green brush" and "Pick up a bowl containing food". The setups of these tasks are presented in Figure IX.

Implementation Details. The finetuning process was conducted on 16 NVIDIA A100 GPUs, with all models involved in the comparison utilizing PyTorch FSDP for full finetuning and a batch size of 256. For our method, we simply tested one finetuned checkpoint at 10K finetuning steps, which takes 7.5 hour's finetuning time. For Octo [62] and OpenVLA [30], we tested multiple finetuned checkpoints at different steps and select the ones that worked best in the real-world tasks. Specifically, for the Realman robot, Octo uses the 20K checkpoint, while OpenVLA uses the 30K checkpoint; both models use the 30K checkpoint on the Franka robot. All finetuning data follows the same data augmentation as used during pretraining.

Visualization results. Figure VII and VIII present evaluation examples of each tasks executed by our default model on the Realman robot, while Figure IX provides evaluation examples of each tasks on the Franka robot.

C. Ablation Study Details

C.1. Action Model Architectures

As described in Sec. 4.4 in the main paper, we study the performances of different action models on the Google robot and WidowX robot. Table III presents the hyperparameters of all the action modules mentioned in Sec. 4.4. The MLP components of these models are structured such that each MLP block expands the embedding dimension by four times and then scales it back to the original embedding dimension. The structure of the MLP (3-layer) model is consistent with the model architecture in Octo [62]. Table 7 in the main paper illustrates that such small action models have significant limitations in modeling actions and perform far worse than the DiT models.

Figure V. Visual examples of each task on the Google robot driven by our model.

Table IV. Comparison of different classifier-free guidance coefficients [23] in simulated evaluations.

CFG Scale	GR (VM)	GR (VA)	WR (VM)	Average
1.0	66.9	54.0	46.3	55.7
1.5	74.8	61.3	**51.3**	62.5
3.0	**76.6**	**63.1**	48.3	**62.7**

C.2. Classifier-Free Guidance Scale

During inference, the classifier-free guidance (CFG) is always applied with a default scale of 1.5, as the experiments show that incorporating the CFG technique significantly improves success rates. Different CFG scales are evaluated in Table IV.

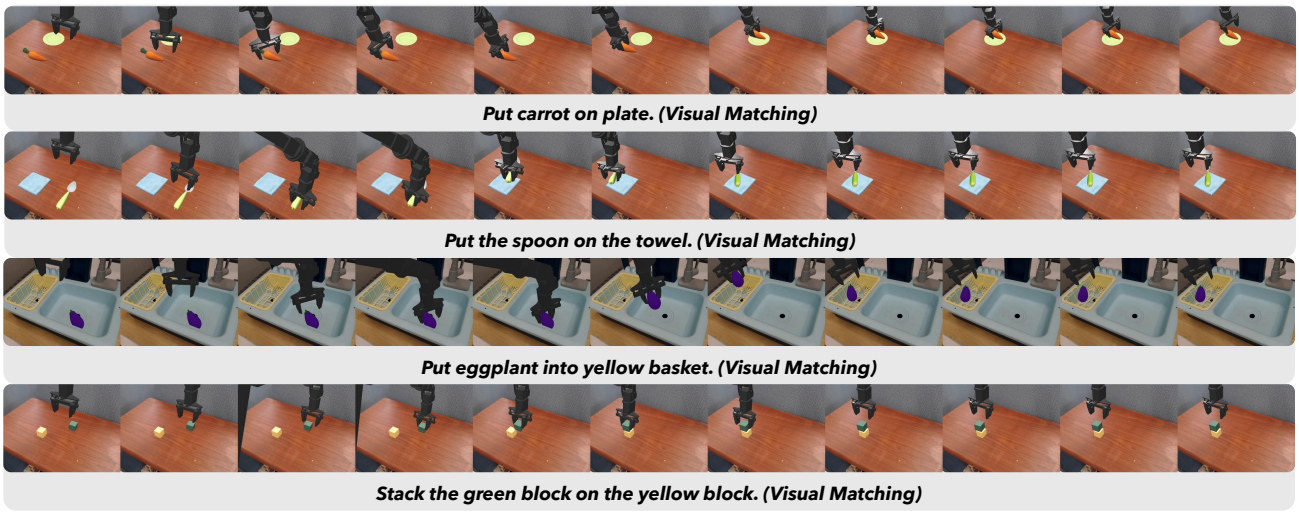

Figure VI. Visual examples of each task on the WidowX robot driven by our model.

Figure VII. Examples of real-world seen tasks and unseen-table-with-unseen-distractors tasks on the Realman robot driven by our model.

Figure VIII. More examples of different real-world tasks on the Realman robot using our model.

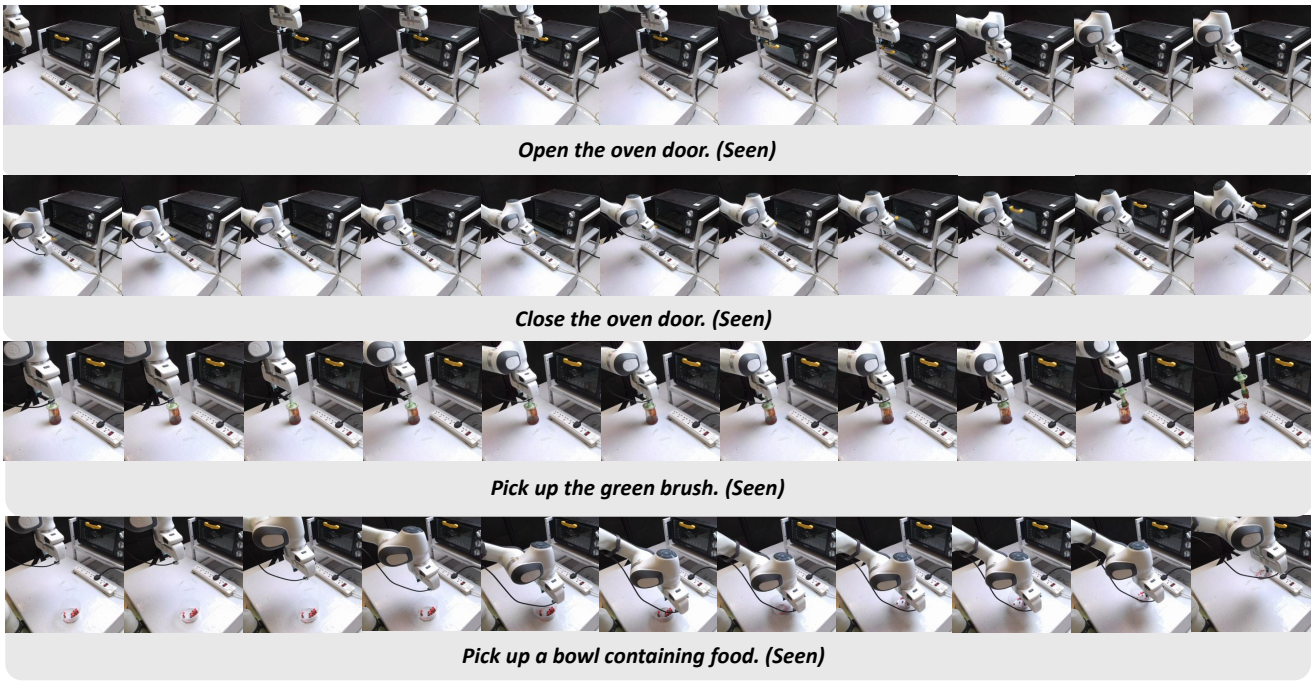

Figure IX. Visual examples of real-world tasks on the Franka robot using our model.