VQ-VLA: Improving Vision-Language-Action Models via Scaling Vector-Quantized Action Tokenizers

Yating Wang[1,2] Haoyi Zhu[1,3] Mingyu Liu[1,4] Jiange Yang[1,5] Hao-Shu Fang[6] Tong He[1†]
[1]Shanghai AI Lab [2]Tongji [3]USTC [4]ZJU [5]NJU [6]SJTU
{wangyating,liumingyu,yangjiange,hetong}@pjlab.org.cn
{hyizhu1108,fhaoshu}@gmail.com
[†]Corresponding Author

Abstract

In this paper, we introduce an innovative vector quantization based action tokenizer built upon the largest-scale action trajectory dataset to date, leveraging over 100 times more data than previous approaches. This extensive dataset enables our tokenizer to capture rich spatiotemporal dynamics, resulting in a model that not only accelerates inference but also generates smoother and more coherent action outputs. Once trained, the tokenizer can be seamlessly adapted to a wide range of downstream tasks in a zero-shot manner, from short-horizon reactive behaviors to long-horizon planning. A key finding of our work is that the domain gap between synthetic and real action trajectories is marginal, allowing us to effectively utilize a vast amount of synthetic data during training without compromising real-world performance. To validate our approach, we conducted extensive experiments in both simulated environments and on real robotic platforms. The results demonstrate that as the volume of synthetic trajectory data increases, the performance of our tokenizer on downstream tasks improves significantly—most notably, achieving up to a 30% higher success rate on two real-world tasks in long-horizon scenarios. These findings highlight the potential of our action tokenizer as a robust and scalable solution for real-time embodied intelligence systems, paving the way for more efficient and reliable robotic control in diverse application domains. Project website.

1. Introduction

Tokenization plays a critical role in recent generative models, including large language models (LLMs) [1, 29], image and video generation models [15, 21, 37], and vision-language-action (VLA) models [6, 24]. One of the key benefits of tokenization is that it compresses the input space. By reducing high-dimensional continuous data into a compact sequence of tokens, the complexity of the learning task is substantially reduced. Compared to image patches and language tokens, action sequences are inherently easier to compress because of their spatio-temporal continuity.

Recently, several studies have started to explore action quantized tokenization for Vision-Language-Action (VLA) models [4, 33], demonstrating promising potential. Effective action tokenization not only significantly enhances the downstream performance of VLA models, especially for tasks involving long-horizon planning, but also markedly improves their training and inference efficiency. In this paper, we delve deeper into the potential of action tokenization, with a specific emphasis on its scalability and accuracy. We first find that the more precise the tokenization, the more pronounced the improvements in long-horizon action modeling. This motivates us to train the VQ tokenizer with well-scaled action trajectories to cover various tasks. Additionally, we observe that action trajectories, unlike visual and physical modalities, exhibit minimal domain gaps between real-world and simulated environments. This characteristic enables effective scaling of action tokenizers through the extensive use of synthetic action data. Moreover, training action tokenizers is computationally lightweight compared to scaling entire VLA models. Consequently, focusing on scaling action tokenization emerges as a highly cost-effective strategy, requiring fewer computational and data resources while delivering significant performance enhancements.

Specifically, we propose a convolutional residual VQ-VAE [4, 26, 48] framework for training action tokenizers. To effectively train the model, we propose a progressive training strategy: Initially, we train the tokenizer on real-world robotic datasets, such as OpenX-Embodiment [32], which typically contain noisy and jittery trajectories. Subsequently, we gradually integrate cleaner and smoother synthetic data from large-scale simulated robotic datasets, such as LIBERO [27] and ManiSkill [31]. This progressive approach allows the VQ model to converge toward smoother and more stable representations. Compared to previous approaches that typically rely on training with single-task datasets, our method expands the tokenizer training dataset by more than 100 times, effectively covering a broad spectrum of downstream tasks.

We conduct extensive experiments in both simulated and real-world environments. First, we evaluate our VQ-VAE action tokenizers in the LIBERO simulator, where the results demonstrate the effectiveness of the convolutional residual VQ-VAE and provide preliminary validation for the hypothesis that the VQ-VAE action tokenizer can leverage synthetic data for scaling. Additionally, real-world experiments with a Franka Research 3 robot further validate the superiority of our approach. Specifically, our findings are as follows: (1) as the amount of simulated action data increases, the VQ-VAE tokenizers exhibit linear scaling properties in improving VLA success rates; (2) the tokenizers significantly enhance inference speed and smoothness of VLA models; and (3) the tokenizers effectively reduce cumulative errors, enabling better performance in long-horizon tasks.

In summary, our contributions are as follows:

- We propose a general convolutional residual VQ-VAE-based framework for action tokenizers.
- We demonstrate that action tokenizers can be effectively scaled by leveraging large-scale simulated action data.
- We prove that our action tokenizers improve the performance, inference speed, and long-horizon capabilities of VLA models.

2. Releated Works

Vision-Language-Action Models. Vision-Language-Action (VLA) models[5, 7, 9, 16, 24, 38, 42, 43, 52, 53] bridge visual-language understanding with robot control by mapping multimodal inputs to action outputs. Based on vision-language models (VLM) [3, 8, 9, 13], VLAs represent robot actions (e.g., 6DoF motion, gripper control) as discrete tokens compatible with text-based output. For example, RT-1[5] and RT-2[6] showed that dividing continuous actions into discrete bins allows integration with VLMs, enabling zero-shot generalization using web-scale pretraining. Recent works[24, 32] further leverage this approach, demonstrating improved task generalization through large VLM backbones. Building on this foundation, recent works have explored various approaches to improve VLA, such as leveraging 3D visual inputs [53], integrating chain-of-thought reasoning [47], and employing parallel decoding strategies [25]. In our work, we aim to simultaneously improve the performance and execution speed of VLA.

Action Representation. Action representation is one of the core components in robot policy learning. Classic action representation schemes include high-level sub-tasks [2, 13], action primitives [20, 40], keypoints [12, 18], as well as low-level continuous end-effector pose [27, 32] or joint state [19]. Recently, imitation learning-based end-to-end approaches [5, 11, 51, 54, 55] often utilize the latter. In order to represent complex multi-modal distributions and adapt to discrete generative models, many works further discretize continuous low-level actions, including per-dimension and per-timestep binning discretization [5, 24], VQ-VAE [4, 26, 30], and cosine transform [33]. Additionally, some works extract corresponding predictive signals from action-less video data to represent actions, such as pixels [7, 44], trajectories [41, 45], and latent motion [10, 46].

Tokenization. The tokenizer is one of the fundamental components in language and vision generation tasks, especially for autoregressive transformer-based generation models. For language generation tasks, mainstream methods adopt byte pair encoding (BPE) [17, 34] to compress input text. For vision generation tasks, recent works [15, 28, 35, 37, 49, 50] mainly use vector quantization [39] for tokenization to map continuous visual signals into a discrete token sequence. Similarly, in robot learning, many works [5, 6, 26, 30, 33] model action prediction as a generative problem and tokenize the robot action modality to capture the multi-modal distribution within skills and adapt it to autoregressive generative policies. In our work, we propose a general convolutional residual VQ-VAE to tokenize robot action. Different from several releated works [26, 30] that use VQ-VAE to tokenize robot actions, our VQ-VAE is trained on large-scale data and is applicable to all tasks.

3. Methods

3.1. Preliminaries: VLA models.

We use OpenVLA[24] as our backbone model. OpenVLA's origin formulation is to adopt a discrete tokenization strategy for robot action prediction through fine-tuning the Prismatic-7B VLM backbone. The method frames action prediction as a vision-language task, mapping input observation images and natural language instructions to discrete robot action sequences. Specifically, continuous robot actions are discretized into 256 bins per dimension, with bin boundaries determined by the 1st and 99th percentiles of training data distributions rather than min-max ranges to mitigate outlier effects. This discretization converts N-dimensional actions into N discrete integers (0-255). To embed these into the LLM's vocabulary, OpenVLA overwrites the 256 least-used tokens in the Llama tokenizer (last 256 tokens) rather than using special tokens, as the original tokenizer only reserves 100 special tokens.

3.2. Action Tokenizer via Residual VQ-VAE

Based on Residual VQ-VAE [4, 26, 48], we design our encoder and decoder inspired by VAE in pyramidal flow matching [23]. Instead of the simple Multi-Layer Perceptron (MLP) used in Residual VQ-VAE [26, 48], we integrate 2D

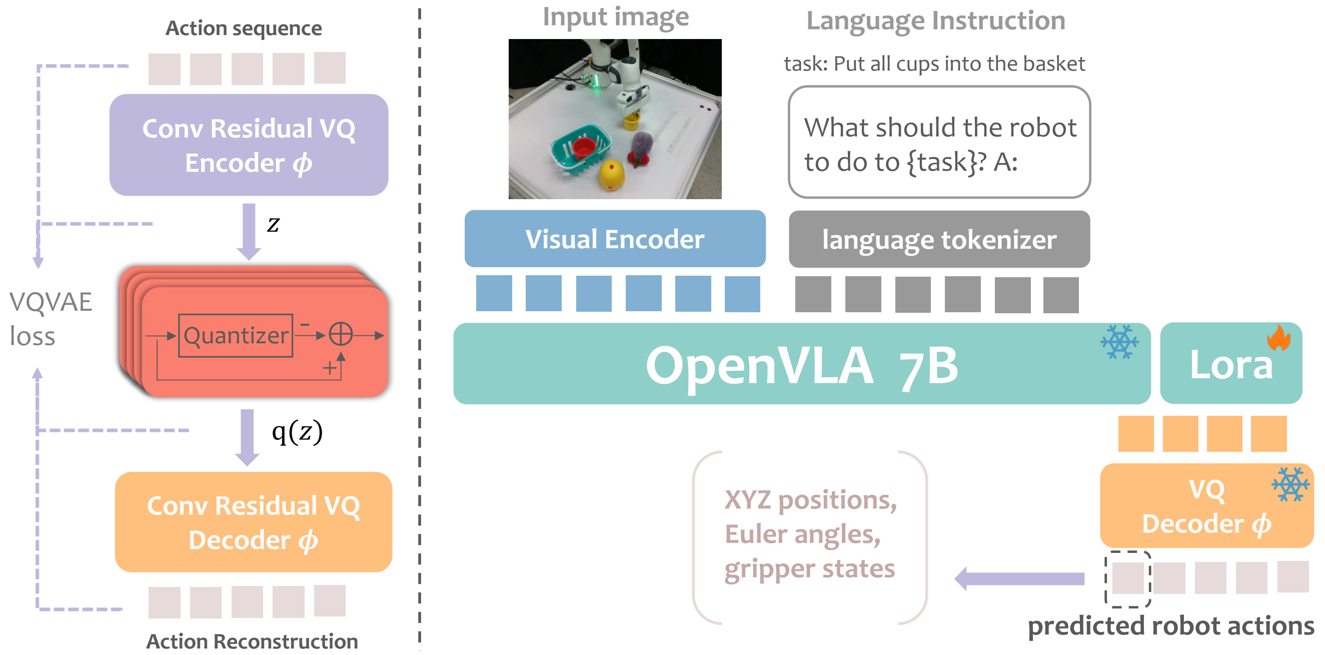

Figure 1. **The VQ-VLA pipeline**, consisting of two main stages: (1) training a general convolutional residual VQ-VAE and (2) fine-tuning OpenVLA using the LoRA approach. Specifically, a general convolutional residual VQ-VAE is first trained on the Open X-Embodiment dataset, LIEBRO, and ManiSkill datasets. The trained VQ-VAE is then frozen and serves as an action tokenizer for OpenVLA, replacing the simple binning method. In the second stage, OpenVLA is fine-tuned using the LoRA technique to optimize its performance.

temporal convolutional layers, motivated by their ability to efficiently capture local relationships and hierarchical temporal dependencies, addressing the scaling limitations of MLPs.

Given an input action sequence $\mathbf{a}_{t:t+n} \in \mathbb{R}^{n\times d}$, where n is the sequence length and d the action dimensionality, the encoder ϕ_{enc}, composed of 2D temporal convolutional layers, transforms the sequence into a latent embedding $\mathbf{x} \in \mathbb{R}^k$, expressed as $\mathbf{x} = \phi_{\text{enc}}(\mathbf{a}_{t:t+n})$. To compress $\mathbf{x}$, we apply Residual Vector Quantization (RVQ) [48], decomposing $\mathbf{x}$ into quantized residuals: $\mathbf{q}(\mathbf{x}) = \sum_{i=1}^{N_q} \mathbf{q}_i(\mathbf{r}_i)$, where $\mathbf{r}_1 = \mathbf{x}$, $\mathbf{r}_{i+1} = \mathbf{r}_i - \mathbf{q}_i(\mathbf{r}_i)$, and N_q denotes the number of quantization stages.

The quantized embedding $\mathbf{q}(\mathbf{x})$ is passed through the decoder ϕ_{dec}, which uses 2D temporal deconvolutional layers to reconstruct the sequence $\hat{\mathbf{a}}_{t:t+n}$, ensuring temporal structure preservation: $\hat{\mathbf{a}}_{t:t+n} = \phi_{\text{dec}}(\mathbf{q}(\mathbf{x}))$.

To train the framework, we minimize the total loss $\mathcal{L}$, a weighted combination of reconstruction loss $\mathcal{L}_{\text{rec}}$, vector quantization (VQ) loss $\mathcal{L}_{\text{codebook}}$, and commitment loss $\mathcal{L}_{\text{commit}}$:

$$\begin{aligned}\mathcal{L} =&\|\mathbf{a}_{t:t+n} - \hat{\mathbf{a}}_{t:t+n}\|_2^2 \\ &+ \lambda\big(\|\text{sg}(\mathbf{x}) - \mathbf{q}(\mathbf{x})\|_2^2 + \|\mathbf{x} - \text{sg}(\mathbf{q}(\mathbf{x}))\|_2^2\big),\end{aligned} \tag{1}$$

where $\text{sg}(\cdot)$ denotes the stop-gradient operation, and λ balances the loss components. We set $\lambda = 4$ in our experiments. This design ensures efficient, scalable, and robust encoding of structured temporal data.

3.3. Training Residual VQ-VAE

We train three versions of the Residual VQ-VAE to evaluate the scaling capability of our action tokenizer: (1) using only the Open X-Embodiment dataset[32], (2) combining Open X-Embodiment and Libero datasets[27], and (3) combining Open X-Embodiment, Libero, and ManiSkill[31] datasets. These experiments were designed to test our hypothesis that the action tokenizer can effectively scale with simulated ata.

To improve the encoder's ability to process temporal and spatial information, we introduced two types of embeddings before the action sequences are passed into the encoder:

- **Time Embedding**: A sinusoidal time embedding was added to encode temporal information at varying frequencies. This embedding allows the model to capture both low-frequency and high-frequency temporal patterns in the input actions, improving its ability to represent fine-grained temporal details.
- **Action-Type Embedding**: Learnable embeddings were added for the different components of the action sequence (e.g., XYZ positions, Euler angles, and gripper states). Since the 7 dimensions of the action vector have distinct meanings, this embedding provides a strong prior for the model, helping it distinguish and process the unique roles of each dimension more effectively.

The use of these embeddings enhances the encoder's ability to process structured data, improving the quality of the latent representations and the overall performance of the tokenizer.

To train a more universal robot action tokenizer and reduce computational overhead, the model is trained using only action sequences as input, without additional conditional inputs. This design reduces complexity while maintaining the generalizability of the tokenizer. All models are trained on a single A100 GPU. For example, training on the Open X-Embodiment dataset requires just one A100 GPU and is completed in one week.

3.4. Integrating Residual VQ-VAE as Action Tokenizer in VLA

In this work, we replace the simple bin-based action tokenizer used in OpenVLA with a Residual VQ-VAE-based action tokenizer to improve the expressiveness and precision of action tokenization. Instead of discretizing action sequences into uniform bins, the action sequence $\mathbf{a}_{t:t+n}$ is first processed through a pre-trained and frozen Residual VQ-VAE encoder $\phi(\cdot)$, generating latent representations. These representations are then quantized into discrete codebook indices (tokens) $\{z_q^i\}_{i=1}^{N_q}$ corresponding to N_q Residual VQ layers. The quantized tokens z_q^i are used as the action tokens for training and prediction in the Vision-Language Model (VLM).

Unlike OpenVLA, where all token IDs are mapped to the range $[0, 255]$, the token IDs generated by different VQ layers in the Residual VQ-VAE are assigned unique, non-overlapping ranges. Specifically, tokens from the i-th VQ layer are offset by $(i-1) \times 256$, ensuring that:

$$z_q^i \in [256 \times (i-1), 256 \times i - 1], \quad \forall i \in \{1, \dots, N_q\}.$$

For example, tokens from the first layer are in the range $[0, 255]$, those from the second layer are in $[256, 511]$, and so on. This design avoids conflicts between token IDs from different layers, as tokens with the same ID in different layers represent semantically different features of the action space.

The Vision-Language Model (VLM) is trained to predict these tokens directly. The loss function is the standard next-token prediction loss, computed as the cross-entropy between the predicted token distribution $\hat{z}_q^i$ and the ground truth token z_q^i produced by the frozen Residual VQ-VAE:

$$L_{\text{VLM}} = -\sum_{i=1}^{N_q} \log P(\hat{z}_q^i = z_q^i | o_{t-h:t}),$$

where $o_{t-h:t}$ represents the input observation sequence, and N_q is the number of Residual VQ layers.

To further enhance computational efficiency, similar to OpenVLA, the least-used tokens in the vocabulary are replaced during fine-tuning. However, this replacement process respects the layer-specific token ID ranges, ensuring that the tokenization remains consistent and interpretable across all layers.

This integration of a pre-trained and frozen Residual VQ-VAE as the action tokenizer in VLA enables a more expressive and scalable representation of actions. By leveraging hierarchical quantization with non-overlapping token ID ranges, the model achieves better action representation, avoids semantic confusion between layers, and ensures stable loss convergence during training. Furthermore, by using the VQ-VAE tokens to represent longer action sequences instead of predicting one action at a time, the model significantly reduces the number of tokens required for training and inference. This mechanism allows the model to predict more complex behavior sequences with fewer steps, leading to a substantial improvement in inference speed and computational efficiency.

4. Experiments

In our experiments, we first validate the effectiveness and scalability of the action tokenizer in VLA models using LIBERO simulator[27]. Subsequently, real-world experiments are conducted to further verify our hypothesis. We also investigate the impact of action tokenizers on the performance, inference speed, and long-horizon capabilities of VLA models, alongside ablation studies to evaluate key design choices.

4.1. Simulation Experiments

4.1.1. Experiment Setup

We utilize the LIBERO benchmark[27] to validate and evaluate the effectiveness and scalability of the action tokenizer, using the Franka Panda robot. Specifically, the entire LIBERO task suite—including LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, LIBERO-10, and LIBERO-90—is used as the entire LIBERO dataset. Among these, LIBERO-90 comprises 90 short-horizon tasks, while the other task suites each contain 10 tasks, with 50 demonstrations per task. For evaluation, testing is conducted on LIBERO-goal, LIBERO-10 and LIBERO-90.

Following the approach in OpenVLA[24], we filter out all ”no-op” actions from the dataset. To preliminarily validate the effectiveness of using VQ-VAE as an action tokenizer in VLA models and its scalability, we train two versions of a convolutional residual VQ-VAE. The first version VQ_M is trained solely on the Maniskill dataset[31], and the second $\text{VQ}_\text{M+R}$ is trained on a mixture of Maniskill dataset and RLBench dataset[22]. Both models are trained on a single A100 GPU with a batch size of 1024, which takes about only 1 week.

We use two pre-trained VQ-VAE models as frozen action tokenizers for OpenVLA: VQ_M and $\text{VQ}_\text{M+R}$. LoRA is then applied to fine-tune OpenVLA on the LIBERO-90 dataset.

Additionally, we fine-tune the original OpenVLA model on the LIBERO-90 dataset using LoRA as a baseline for comparison.For a fair comparison, all fine-tuning on the LIBERO-90 task suite is conducted for 400K gradient steps with a batch size of 4, using 4 A100-80GB GPUs and an action chunk length of K=5.

4.1.2. Effectiveness of Conv Residual VQ-VAE

In the early stages of the experiment, we selected two task suites, LIBERO-10 and LIBERO-GOAL, to conduct preliminary scaling validation experiments. Specifically, we used two variants of Residual VQ-VAE models: one with a simple MLP as the encoder and decoder, and the other with a larger 2D temporal convolutional network as the encoder and decoder. The results are summarized in Tab. 1. The findings indicate that using temporal convolutional networks as the encoder and decoder in the Residual VQ-VAE significantly outperforms the MLP-based architecture in terms of success rate. These results suggest that temporal convolutional networks can serve as an effective action tokenizer, capable of capturing temporal dependencies more effectively.

Furthermore, when we increased the training data for the VQ-VAE model, transitioning from data solely derived from individual LIBERO tasks to the entire LIBERO dataset, the success rate consistently improved. This observation provides preliminary evidence supporting the scalability of our action tokenizer design.

	Training Dataset of VQ	**LIBERO-10 (%)**	**LIBERO-GOAL (%)**
Original OpenVLA	-	51.0	**75.8**
MLP Residual VQ-VAE	ALL-LIBERO	53.4	72.6
	LIBERO-10	53.2	-
	LIBERO-GOAL	-	65.2
Conv Residual VQ-VAE	ALL-LIBERO	**60.0**	75.2
	LIBERO-10	54.0	-
	LIBERO-GOAL	-	72.4

Table 1. **The evaluation results of residual VQ-VAE architectures.**The results demonstrate that the Conv Residual VQ-VAE outperforms the MLP-based version, particularly when trained on the full LIBERO dataset (ALL-LIBERO), highlighting its ability to better capture temporal dependencies and improve success rates.

4.1.3. Scaling Data Improves VQ-VAE Action Tokenizer Performance

In the experiment, to avoid potential performance inflation from in-domain data during evaluation on LIBERO-90, we trained the VQ-VAE Action Tokenizer on completely out-of-domain ManiSkill and RLBench simulation data, rather than LIBERO data. we compare three models: the baseline, which fine-tunes the original OpenVLA; $\mathrm{VQ_M}$, which uses a VQ-VAE trained on the Maniskill dataset as the action tokenizer for OpenVLA; and $\mathrm{VQ_{M+R}}$, which uses a VQ-VAE trained on a mixture of the Maniskill and RLBench datasets as the action tokenizer for OpenVLA.

On LIBERO-90, $\mathrm{VQ_{M+R}}$ achieved 80.98%, a 7.45% improvement over the OpenVLA baseline (73.53%), Tab. 2. Furthermore, an ablation study using only ManiSkill data for training $\mathrm{VQ_M}$ resulted in substantially lower performance, underscoring the critical role of sufficient synthetic data scale.

	baseline(%)	**$\mathrm{VQ_M}$**(%)	**$\mathrm{VQ_{M+R}}$**(%)
LIEBRO-90	73.53	14.38	**80.98**

Table 2. **Effectiveness of VQ-VAE Action Tokenizers in Scaling Simulation Data.**The results demonstrate $\mathrm{VQ_{M+R}}$ reached 80.98%, outperforming the OpenVLA baseline by 7.45%

4.2. Real-Word Experiment

4.2.1. Experiment Setup

Our robotic platform consists of a single Franka Research3 arm equipped with a third-person-view RealSense D435 camera mounted in a fixed position to capture environmental observations. The system operates at 20 Hz (moderately reduced from the native 100 Hz control frequency to balance training efficiency and motion continuity), with actions defined as absolute end-effector poses in SE(3) space (3D position + quaternion orientation).

Our experimental benchmark comprises six manipulation tasks (4 short-horizon tasks, 2 long-horizon tasks) designed to evaluate the model's ability to handle varying task complexities. For each task, we collect 50 demonstrations and evaluate performance over 20 trials:

1) **Pull out a tissue paper**: The robot need to grasp and pull out a single tissue paper.

2) **Pick up the [TOY NAME]**:The robot is required to pick up the specific toy, including the toy snake, the toy eggplant, and the toy chicken, resulting in a total of three tasks (with no other distractions).

3) **Put the toy into the basket**: The robot need to pick up the toy(no other distractions) and put it into the basket.

4) **Flip the pot upright**: We set a flipped pot on the platform, the robot need to flip and upright a fallen cooking pot.

5) **Put all cups into the basket**: We place two different cups on the table and set a few other things(toys) as distractions. The robot need to sequentially put two cups into the basket, testing long-horizon task.

6) **Put the toy into the drawer**: We place a toy on the table and set a few other things as distractions. The robot need to sequentially open the drawer,pick up the specific toy and put it into the drawer, and close the drawer, testing long-horizon task.

We finetune each task separately for 100K gradient steps (batch size 4 across 4 A100-80GB GPUs) with action chunk length K=5. During deployment.

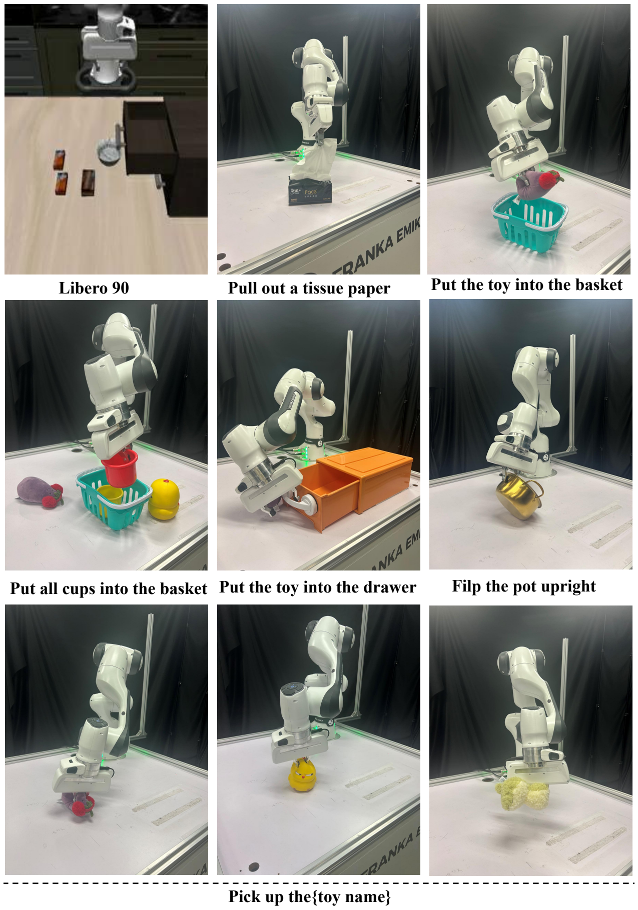
Libero 90
Pull out a tissue paper
Put the toy into the basket
Put all cups into the basket
Put the toy into the drawer
Filp the pot upright
Pick up the{toy name}

Figure 2. **All Evaluation environments**:We conduct comprehensive evaluations of VQ-VLA in both simulation and real-world settings. In simulation, evaluations are performed on the LIBERO-90 benchmark within the LIBERO dataset. And six diverse tasks are designed for real-world testing.

4.2.2. Performance on Short-Horizon Tasks

Based on the pre-trained VQ$_{\text{O}}$ and VQ$_{\text{O+L}}$ in the simulation experiments, we further incorporated 120k synthetic trajectories from the ManiSkill dataset, co-training VQ$_{\text{O+L+M}}$ with Open X-Embodiment dataset and LIBERO dataset to evaluate the scaling capability of our action tokenizer. We evaluated the performance of the action tokenizer trained with different datasets on four carefully designed short-horizon tasks, focusing on multi-task learning, precision manipulation, and dynamic adjustment capabilities. The results are shown in Fig. 3.

It can be observed that when synthetic trajectories are added for co-training, the performance of the action tokenizer improves significantly. The average success rate increased from the baseline of 23% to 46.25%, with a notable improvement of 30% in the ”Flip the pot upright” task. In the ”Pull out a tissue paper” task, which tests the robot's performance in high-precision dynamic operations (as this task requires continuous, fine-grained grasping and pulling motions), the baseline model achieved only a 5% success rate, failing almost entirely. In contrast, models incorporating VQ achieved success rates of 20% or higher.

Furthermore, comparing VQ$_{\text{O+L}}$ with VQ$_{\text{O}}$, the average success rate on short-horizon tasks increased by only 0.5%. This may be attributed to the limited size of the LIBERO dataset, which has minimal impact on short-horizon tasks. However, the ManiSkill dataset is 50 times larger than LIBERO, leading to a significant improvement in success rates when used for training.

4.2.3. Performance on Long-Horizon Tasks

VQ-VLA demonstrates outstanding performance on long-horizon tasks (”Put all cups in the basket” and ”Put the toy into the drawer”), significantly outperforming baseline model in both success rate and efficiency. In scenarios where baseline models achieve success rates as low as 15% or nearly 0, the VQ$_{O+L+M}$ model achieves significantly higher success rates of 50% and 30%, respectively (see Fig.3). For the ”Put the toy into the drawer” task, a representative long-horizon scenario, the baseline model was only able to complete the first step of opening the drawer in 15 out of 20 trials, failing to proceed further in most cases. In contrast, the VQ$_{O+L+M}$ model successfully opened the drawer in all test cases, demonstrating its robustness and reliability in handling complex sequential tasks.

A key advantage of VQ-VLA in long-horizon tasks lies in its use of VQ-VAE as an action tokenizer, which enables the model to predict multiple actions in a single inference step. This design reduces the accumulation of errors over extended task sequences, making VQ-VLA particularly advantageous for tasks requiring long-term planning and execution. Additionally, in scenarios involving the consecutive execution of multiple subtasks, VQ-VLA not only achieves higher success rates but also significantly reduces task completion time compared to baseline models, highlighting its superior execution efficiency and ability to handle long-horizon challenges effectively.

4.2.4. Sim&Real Domain Gap Analysis

To quantify the domain gap between synthetic data and real-world data, we trained a VQ-VAE model exclusively on the LIBERO dataset, referred to as VQ$_{\text{L}}$, for OpenVLA. The model was tested in three real-world tasks (one long-horizon and two short-horizon tasks), and the results are shown in Tab. 3. The results indicate that the performance of VQ$_{\text{L}}$ is comparable to that of both VQ$_{\text{O+L}}$ and VQ$_{\text{O}}$, suggesting that the domain gap between synthetic and real-world data is minimal.

Although real-world data may contain noise, the inclusion of Open X-Embodiment data as a real-world dataset expands the data sources and enriches the diversity of data types,

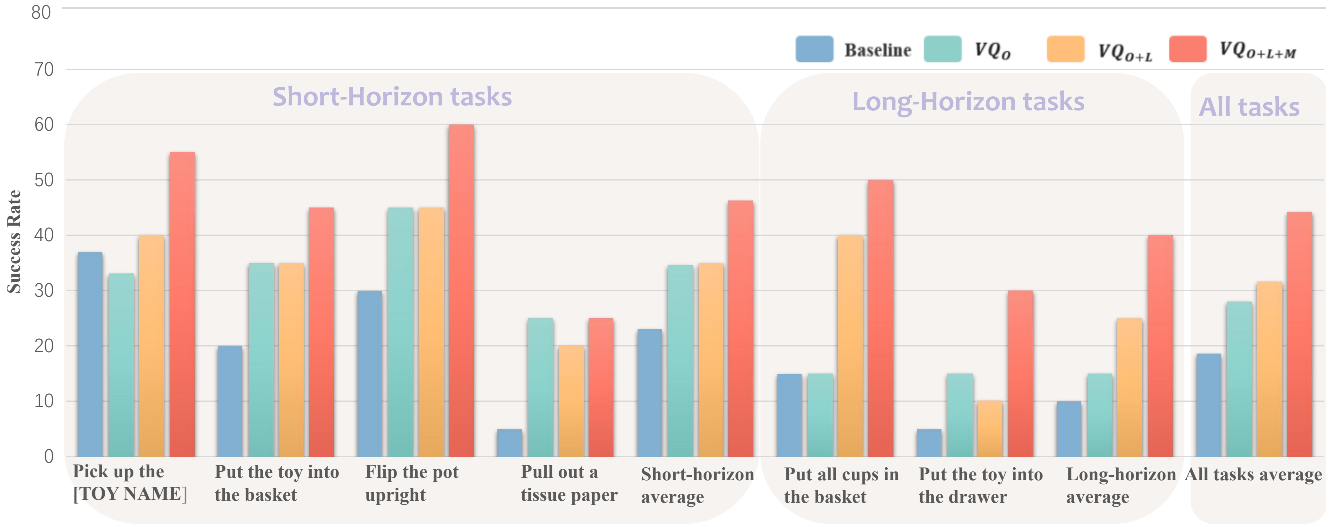

Figure 3. **Real-world experimental results:** We compare the performance of Baseline, VQ_{O}, $\text{VQ}_{\text{O+L}}$, and $\text{VQ}_{\text{O+L+M}}$ on both short-horizon and long-horizon tasks. In terms of the average success rate, all VQ-based models outperform the Baseline. The best-performing model, $\text{VQ}_{\text{O+L+M}}$, achieves a success rate that is 23.25% higher than the Baseline on both short-horizon and long-horizon tasks. Additionally, the results show that $\text{VQ}_{\text{O+L+M}}$ outperforms $\text{VQ}_{\text{O+L}}$, which in turn outperforms VQ_{O}, indicating the effectiveness of incorporating synthetic data during training without compromising real-world performance.

which effectively enhances the model's generalization and robustness.

	Put the toy into the drawer (%)	**Flip the pot upright (%)**	**Put the toy into the basket (%)**
baseline	5.0	30.0	20.0
VQ_{O}	15.0	45.0	35.0
VQ_{L}	10.0	55.0	35.0
$\text{VQ}_{\text{O+L}}$	10.0	45.0	35.0
$\text{VQ}_{\text{O+L+M}}$	**25.0**	**60.0**	**45.0**

Table 3. **Performance Comparison Across Real-World Tasks:** We observe that the performance of VQ_{L} is comparable to that of both $\text{VQ}_{\text{O+L}}$ and VQ_{O}, indicating that the domain gap between synthetic and real-world data is minimal.

4.2.5. Inference Speed Comparison

During the real-world experiments, we measured the action execution frequency of VQ-VLA and compared it with the original OpenVLA. The results are summarized in Tab. 4. As shown in the table, with a compression ratio of 5 in VQ-VAE, the inference speed is nearly tripled. This significant improvement greatly facilitates real-time performance in practical applications.

4.3. Ablation Studies

In this section, we report some ablation studies to show the effectiveness of the design choices of our method.

	Frequency (Hz)
VQ-VLA	**11.84**
OpenVLA	4.16

Table 4. **The Results of Frequencies.** We report the comparison results of our VQ-VLA and baseline OpenVLA.

4.3.1. Action Chunking via VQ-VAE and Autoregressive Output

The original OpenVLA model generates a single action per inference step based solely on the current observation, resulting in a step-by-step execution approach. To improve efficiency and handle longer action sequences, we extend OpenVLA to output a sequence of five actions in an autoregressive manner, similar to the action chunking mechanism in VQ-VLA. Specifically, the model predicts the next action based on the current observation and previously generated actions, allowing it to generate a sequence of actions in a single inference step. This autoregressive approach serves as an alternative action chunking strategy by grouping multiple actions for execution.

To evaluate the effectiveness of different action chunking strategies, we design ablation experiments comparing the autoregressive output of OpenVLA to the VQ-based action chunking method in VQ-VLA. The results, shown in Tab. 5, indicate that the use of the autoregressive approach for action chunking in the original OpenVLA leads to a significant drop in success rate compared to the baseline. In real-world experiments, when using the autoregressive approach as a form of action chunking, the spatial magnitude of each action chunk is relatively small, resulting in slower overall execution of actions. Compared to the Action Chunking via VQ-VAE method, even with the same action chunk size, the time required to reach the same target location is significantly longer. Additionally, this approach tends to exhibit shortcut learning, where the model directly copies previous actions within the same action chunk. By analyzing the output values of each action chunk, it is observed that multiple actions within a single chunk have remarkably similar values, indicating a lack of diversity in the predicted actions.

Overall, Action Chunking via VQ-VAE not only improves inference speed but also enhances the performance of VLA by generating more effective and diverse action sequences. This makes it a more suitable approach for addressing real-world long-horizon tasks.

Action Chunking	**LIBERO-90 (%)**	**Flip the pot upright (%)**	**Put the toy into the basket (%)**
baseline	74.76	30.0	20.0
Autoregressive Output	66.53	10.0	0.0
VQ-based (VQ$_{O+L+M}$)	**86.61**	**60.0**	**45.0**

Table 5. **Comparison of action chunking methods:** We evaluate the performance on three tasks (one simulator task and two real-world tasks). The results show that the Autoregressive Output used in OpenVLA performs poorly as an action chunking method, with a significant gap compared to the VQ-based approach. This indicates that the VQ-based method is more effective.

4.3.2. Embedding Integration Effectiveness

To evaluate the impact of embeddings, we conducted an ablation study comparing the model's performance with and without embeddings. Specifically, the baseline model processes raw action sequences directly, while the enhanced model incorporates both time embedding and action-type embedding.

As shown in Tab. 6, the model with embeddings significantly outperforms the baseline in terms of success rate. This demonstrates that the integration of embeddings improves the encoder's ability to represent structured action sequences, leading to better overall performance.

	LIBERO-90 (%)	**Flip the pot upright (%)**	**Put the toy into the basket (%)**
VQ$_{O+L}$ (w.o. Embeddings)	85.17	40.0	35.0
VQ$_{O+L}$ (w Embeddings)	**86.16**	**45.0**	**35.0**

Table 6. **Embedding integration improves performance.** The table compares models with and without embeddings across three tasks, showing that embeddings enhance success rates, especially for "Flip the pot upright."

5. Limitations and Future Works

In our work, extensive experiments demonstrate that our proposed action tokenizers can improve VLA performance and inference speed while achieving scalability on simulated data. However, there still remain some limitations and opportunities for future work. First, our action tokenizers can be further extended to larger-scale simulated datasets, such as RLbench [22] based on CoppeliaSim. Second, our work improves inference speed by decoding multi-step action sequences, which can be further combined with techniques such as distillation [36] and quantization [14] of VLMs in the future. Fially, the architecture design of our action tokenizers can be further improved, for example, by incorporating the frequency of action data as a extra condition. In summary, we hope that our work can serve as a strong baseline and inspire future research.

6. Conclusions

In this paper, we propose a general convolutional residual VQ-VAE framework for action tokenizers that can seamlessly integrate with state-of-the-art VLA models. Our VQ-VAE is trained on 100 times more data than previous methods and can be directly transferred to downstream both real-world and simulated robotic manipulation tasks. Extensive experiments conducted in both simulated and real-world environments validate that the proposed convolutional residual VQ-VAE framework not only enhances the performance of VLA policies but also accelerates inference. Finally, our VQ-VAE also demonstrates the ability to scale effectively with large-scale simulated data.

Acknowledgments

This work is supported by the National Key R&D Program of China (NO.2022ZD0160102) and Shanghai Artificial Intelligence Laboratory.

References

[1] 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. 1

[2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022. 2

[3] Jean-Baptiste Alayrac, Jeffrey Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: A visual language model for few-shot learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:23716–23736, 2022. 2

[4] Suneel Belkhale and Dorsa Sadigh. Minivla: A better vla with a smaller footprint. *https://ai.stanford.edu/blog/minivla/*, 2024. 1, 2

[5] 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. 2

[6] 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. 1, 2

[7] 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. 2

[8] Wei Chen, Yifeng Zhang, Hao Li, Xufeng Wang, and Ming Liu. Anyvlm: Unified vision-language model for any robot morphology. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8. IEEE, 2024. 2

[9] 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

[10] Xiaoyu Chen, Junliang Guo, Tianyu He, Chuheng Zhang, Pushi Zhang, Derek Cathera Yang, Li Zhao, and Jiang Bian. Igor: Image-goal representations are the atomic control units for foundation models in embodied ai. *arXiv preprint arXiv:2411.00785*, 2024. 2

[11] 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

[12] Norman Di Palo and Edward Johns. Keypoint action tokens enable in-context imitation learning in robotics. *arXiv preprint arXiv:2403.19578*, 2024. 2

[13] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: an embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, pages 8469–8488, 2023. 2

[14] Kazuki Egashira, Mark Vero, Robin Staab, Jingxuan He, and Martin Vechev. Exploiting llm quantization. *Advances in Neural Information Processing Systems*, 37:41709–41732, 2025. 8

[15] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021. 1, 2

[16] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Chenxi Wang, Junbo Wang, Haoyi Zhu, and Cewu Lu. Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot. *arXiv preprint arXiv:2307.00595*, 2023. 2

[17] Philip Gage. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38, 1994. 2

[18] Theophile Gervet, Zhou Xian, Nikolaos Gkanatsios, and Katerina Fragkiadaki. Act3d: 3d feature field transformers for multi-task robotic manipulation. In *7th Annual Conference on Robot Learning*. 2

[19] Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019. 2

[20] Matthew Hausknecht and Peter Stone. Deep reinforcement learning in parameterized action space. *arXiv preprint arXiv:1511.04143*, 2015. 2

[21] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022. 1

[22] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020. 4, 8

[23] Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. *arXiv preprint arXiv:2410.05954*, 2024. 2

[24] 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, 4

[25] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025. 2

[26] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. *arXiv preprint arXiv:2403.03181*, 2024. 1, 2

[27] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023. 1, 2, 3, 4

[28] Zhuoyan Luo, Fengyuan Shi, Yixiao Ge, Yujiu Yang, Limin Wang, and Ying Shan. Open-magvit2: An open-source project toward democratizing auto-regressive visual generation. *arXiv preprint arXiv:2409.04410*, 2024. 2

[29] Ben Mann, N Ryder, M Subbiah, J Kaplan, P Dhariwal, A Neelakantan, P Shyam, G Sastry, A Askell, S Agarwal, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 1:3, 2020. 1

[30] Atharva Mete, Haotian Xue, Albert Wilcox, Yongxin Chen, and Animesh Garg. Quest: Self-supervised skill abstractions for learning continuous control. *Advances in Neural Information Processing Systems*, 37:4062–4089, 2025. 2

[31] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with

large-scale demonstrations. *arXiv preprint arXiv:2107.14483*, 2021. 1, 3, 4

[32] Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024. 1, 2, 3

[33] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025. 1, 2

[34] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 2

[35] Fengyuan Shi, Zhuoyan Luo, Yixiao Ge, Yujiu Yang, Ying Shan, and Limin Wang. Taming scalable visual tokenizer for autoregressive image generation. *arXiv preprint arXiv:2412.02692*, 2024. 2

[36] Sharath Turuvekere Sreenivas, Saurav Muralidharan, Raviraj Joshi, Marcin Chochowski, Ameya Sunil Mahabaleshwarkar, Gerald Shen, Jiaqi Zeng, Zijia Chen, Yoshi Suhara, Shizhe Diao, et al. Llm pruning and distillation in practice: The minitron approach. *arXiv preprint arXiv:2408.11796*, 2024. 8

[37] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024. 1, 2

[38] 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. 2

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

[40] Sai H Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. Chatgpt for robotics: Design principles and model abilities. *Ieee Access*, 2024. 2

[41] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023. 2

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

[43] Jiange Yang, Wenhui Tan, Chuhao Jin, Keling Yao, Bei Liu, Jianlong Fu, Ruihua Song, Gangshan Wu, and Limin Wang. Transferring foundation models for generalizable robotic manipulation. *arXiv preprint arXiv:2306.05716*, 2023. 2

[44] Jiange Yang, Bei Liu, Jianlong Fu, Bocheng Pan, Gangshan Wu, and Limin Wang. Spatiotemporal predictive pre-training for robotic motor control. *arXiv preprint arXiv:2403.05304*, 2024. 2

[45] Jiange Yang, Haoyi Zhu, Yating Wang, Gangshan Wu, Tong He, and Limin Wang. Tra-moe: Learning trajectory prediction model from multiple domains for adaptive policy conditioning. *arXiv preprint arXiv:2411.14519*, 2024. 2

[46] Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024. 2

[47] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024. 2

[48] 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. 1, 2, 3

[49] Canyu Zhao, Mingyu Liu, Wen Wang, Weihua Chen, Fan Wang, Hao Chen, Bo Zhang, and Chunhua Shen. Moviedreamer: Hierarchical generation for coherent long visual sequence. *arXiv preprint arXiv:2407.16655*, 2024. 2

[50] Canyu Zhao, Mingyu Liu, Huanyi Zheng, Muzhi Zhu, Zhiyue Zhao, Hao Chen, Tong He, and Chunhua Shen. Diception: A generalist diffusion model for visual perceptual tasks. *arXiv preprint arXiv:2502.17157*, 2025. 2

[51] 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. 2

[52] Wei Zhao, Pengxiang Ding, Min Zhang, Zhefei Gong, Shuanghao Bai, Han Zhao, and Donglin Wang. Vlas: Vision-language-action model with speech instructions for customized robot manipulation. *arXiv preprint arXiv:2502.13508*, 2025. 2

[53] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024. 2

[54] Haoyi Zhu, Yating Wang, Di Huang, Weicai Ye, Wanli Ouyang, and Tong He. Point cloud matters: Rethinking the impact of different observation spaces on robot learning. *Advances in Neural Information Processing Systems*, 37: 77799–77830, 2024. 2

[55] Haoyi Zhu, Honghui Yang, Yating Wang, Jiange Yang, Limin Wang, and Tong He. Spa: 3d spatial-awareness enables effective embodied representation. *arXiv preprint arXiv:2410.08208*, 2024. 2