AsyncVLA: Asynchronous Flow Matching for Vision-Language-Action Models

Yuhua Jiang[1,2], Shuang Cheng[1,3], Yan Ding[4], Feifei Gao[2†], Biqing Qi[1†]
[1] Shanghai AI Laboratory, [2] Tsinghua University, [3] Zhejiang University, [4] Lumos Robotics

Abstract

Vision-language-action (VLA) models have recently emerged as a powerful paradigm for building generalist robots. However, traditional VLA models that generate actions through flow matching (FM) typically rely on rigid and uniform time schedules, i.e., synchronous FM (SFM). Without action context awareness and asynchronous self-correction, SFM becomes unstable in long-horizon tasks, where a single action error can cascade into failure. In this work, we propose asynchronous flow matching VLA (AsyncVLA), a novel framework that introduces temporal flexibility in asynchronous FM (AFM) and enables self-correction in action generation. AsyncVLA breaks from the vanilla SFM in VLA models by generating the action tokens in a non-uniform time schedule with action context awareness. Besides, our method introduces the confidence rater to extract confidence of the initially generated actions, enabling the model to selectively refine inaccurate action tokens before execution. Moreover, we propose a unified training procedure for SFM and AFM that endows a single model with both modes, improving KV-cache utilization. Extensive experiments on robotic manipulation benchmarks demonstrate that AsyncVLA is data-efficient and exhibits self-correction ability. AsyncVLA achieves state-of-the-art results across general embodied evaluations due to its asynchronous generation in AFM. Our code is available at https://github.com/YuhuaJiang2002/AsyncVLA.

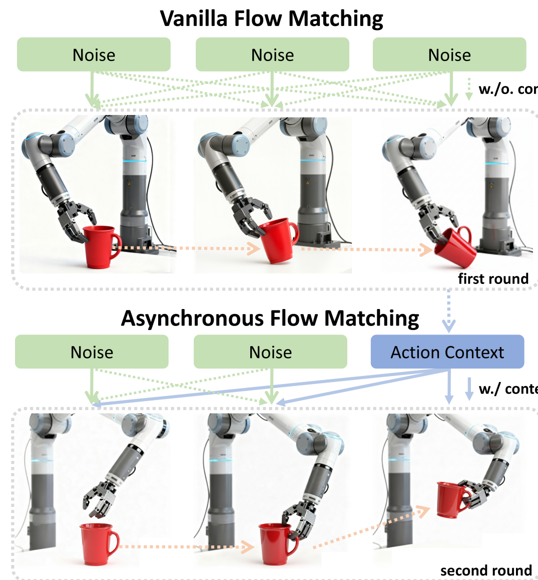

Figure 1. **Comparison of vanilla flow matching and asynchronous flow matching in VLA models. Top:** Vanilla flow matching employs a uniform time schedule for all action tokens, generating them synchronously from noise to actions, i.e., synchronous flow matching. **Bottom:** Asynchronous flow matching dynamically assigns individual time steps to regenerate action tokens. The first-round generated actions provide context information that allows for selective and non-uniform self-correction in the second-round action generation.

1. Introduction

Training generalist robot policies that integrate perception, language, and low-level control remains one of the core challenges for embodied intelligence [12, 22, 26, 35, 38, 45, 56, 59]. To address it, vision-language-action (VLA) models leverage heterogeneous vision-language (VL) corpora and robot demonstrations, grounding broad semantics into executable control [5, 7, 17, 28, 36, 46, 50, 74, 75]. This paradigm achieves strong instruction-following performance across both simulated and real systems [19, 30, 31, 44, 47, 60, 65, 66, 79], with representative large-scale agents such as RT-2 [6], PaLM-E [15], RoboCat [49], and Mobile ALOHA [17, 79]. In order to improve the task success rates and the efficiency of VLA models, subsequent work advances architecture and action generation mechanisms, including exploitation of spatial and temporal information [10, 37, 48, 61, 72], parameter-efficient adaptation [62, 67], improved tokenizer [21, 53], high-throughput execution [31, 62], and strengthening interpretability by interleaving intermediate reasoning with action prediction [27, 68, 69, 78, 82]. Building on these advances, recent VLA models incorporate self-correction to improve reliability under uncertainty. CollabVLA [63] integrates self-correction reasoning and seeks human guidance when its confidence is low. ReflectVLM [16] removes the need for human involvement and iteratively refines long-horizon plans via test-time reflection. Imitating the fast and slow systems in the human brain [29, 40], SC-VLA [34] pairs a fast action head with a slow self-correction module to detect failures. Enhanced by reinforcement learning (RL), RB-VLA [33] applies a dual-pathway loop for insitu adaptation. Inspired by diffusion large language models (DLLMs) [11, 20, 23, 51, 58, 64, 71], discrete-diffusion VLA [42] brings masked-token denoising and secondary remasking into a single model for adaptive decoding and self-correction. LLaDA-VLA [70], dVLA [68], and UD-VLA [9] further employ multi-modal chain of thought (CoT) with prefix attention and KV-cache.

Despite these successes, a fundamental limitation exists in mainstream VLA architectures: their reliance on a rigid and synchronous action generation process [3, 4, 54, 76]. VLA models based on vanilla flow matching (FM) employ a uniform time schedule across all action tokens, generating them synchronously from noise to the final actions, i.e., synchronous FM (SFM). SFM employs fixed action-generation time schedules, regardless of the task's current complexity or the model's internal confidence [13, 14, 77]. Without the utilization of action context information and the mechanism for self-correction, SFM's monolithic generation method is inherently unstable. In Fig. 1, we show the SFM's generation process with unawareness of action context. Consequently, a single inaccurate action prediction can cascade into an unrecoverable error, critically hindering performance in long-horizon or precision-demanding scenarios.

In order to utilize the action context information in action generation, we find that temporal asynchrony—the ability to non-uniformly and dynamically decide the action generation time schedule—is the key to unlocking robust robotic control with self-correction ability. Our core insight is to reframe action generation, particularly within the framework of FM [18, 43, 80], not as a fixed procedure, but as a deliberative denoising process with asynchronous time schedule, where the model can reconsider those parts of the first-round generated actions with low confidence. By regenerating a subset of actions while keeping others unchanged, temporal asynchrony exploits the context information of first-round generated actions to refine potentially inaccurate actions, and thus realizes self-correction.

In this paper, we propose asynchronous flow matching VLA (AsyncVLA), a novel VLA framework that employs the initial SFM and the subsequent asynchronous FM (AFM), enabling confidence-aware robot action generation with self-correction. Instead of a fixed and uniform time schedule in the denoising process, AsyncVLA adaptively schedules its AFM time steps, performing regeneration on action tokens with low confidence. Specifically, we propose a confidence rater that evaluates the confidence of each action token generated by SFM. AsyncVLA leverages these confidence signals to trigger asynchronous self-correction, enabling the model to selectively revisit and refine low-confidence parts of its action plan before execution. Moreover, the first-round generated actions with relatively high confidence provide context information that facilitates correcting actions with relatively low confidence. Therefore, AsyncVLA possesses an introspective capability to dynamically modulate its generation process and selectively reconsider its generated actions based on confidence.

Our contributions can be summarized as follows:

- We propose AsyncVLA, a novel VLA framework that introduces AFM into the action generation process, breaking from the rigid and synchronous time schedules in vanilla SFM.
- We introduce the confidence rater to estimate confidence of the first-round actions generated by SFM, enabling dynamic regeneration and selective self-correction in AFM according to the confidence of actions.
- We demonstrate through extensive experiments in simulated robot tasks that AsyncVLA enhances the model's robustness to perturbation from first-round erroneous actions with large deviation, and significantly improves task success rates compared to state-of-the-art VLA models.

2. Related Work

Vision-Language-Action Models VLA models adapt the vision-language model (VLM) backbones to map visual inputs and natural language instructions to low-level robot actions. Early-stage VLA models employ auto-regressive decoding with discretized action tokens [5, 19, 21, 30]. Inspired by CoT in VLM, CoT-VLA [78] and FlowVLA [82] generate future sub-goal images as a visual CoT before predicting short action chunks, which enhances long-horizon success and interpretability. To enhance inference efficiency, OpenVLA-OFT [31] introduces parallel decoding and chunked control, achieving both high throughput and strong performance. For continuous action modeling, π_0 [3], $\pi_{0.5}$ [4], WALL-OSS [76], and EO-1 [54] utilize FM to generate actions, but these models adopt synchronous generation schedule based on SFM. In order to move beyond the limitation of fixed-step and synchronous schedule in SFM, we propose AsyncVLA, which couples AFM with confidence-driven self-correction to enable calibrated action regeneration only when necessary.

Self-Correction in VLA Models Self-correction mechanism is introduced in recent VLA models to improve task success rates. CollabVLA [63] integrates self-correction reasoning with diffusion-based action generation and proactively seeks human guidance under uncertainty. Without the need for human involvement, ReflectVLM [16] combines

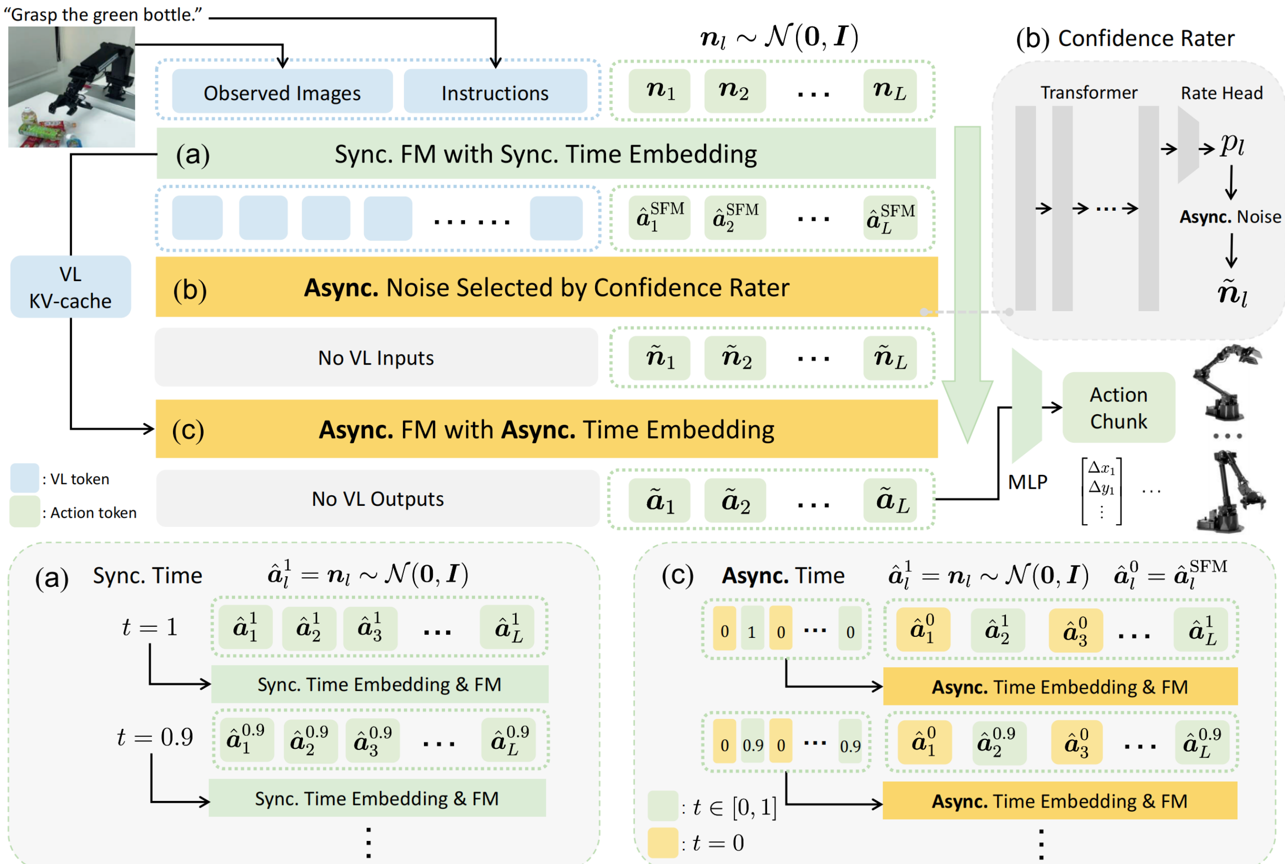

Figure 2. Overview of the AsyncVLA framework that comprises three components: (a) SFM applies a uniform time schedule t across all action tokens, generating them synchronously from noise ($t = 1$) to action ($t = 0$). (b) Confidence rater estimates the actions' token-level confidence and mask the low-confidence actions by selecting asynchronous noise for AFM. (c) AFM dynamically assigns individual FM time to each action token, allowing for selective and non-uniform regeneration based on the actions' confidence. SFM and AFM share a single unified model with the same parameters, enabling the VL KV-cache produced by SFM to be reutilized in AFM.

test-time reflection with diffusion-based imagination to iteratively revise long-horizon plans. Inspired by the fast and slow systems in the human brain [40], SC-VLA [34] couples a fast action head with a slow self-correction module to detect failures and issue fixes within one policy. Benefiting from RL, RB-VLA [33] uses a dual-pathway loop, i.e., failure-driven RL combined with success-driven supervised fine-tuning (SFT), for autonomous in-situ adaptation. Built upon DLLMs [11, 20, 51, 58, 64, 71], discrete-diffusion VLA [42] applies masked-token denoising for action chunk generation inside a single transformer, which allows for adaptive decoding and secondary remasking for error correction. LLaDA-VLA [70], dVLA [68], and UD-VLA [9] employ multi-modal CoT and introduce acceleration techniques such as prefix attention and KV-cache to achieve real-time control. However, the above work mainly focuses on self-correction in discrete action token generation. In order to empower the model's self-correction ability in continuous action generation without supervision from humans or large reward models, AsyncVLA introduces a unified model with SFM and AFM generation, enhancing the model's self-correction ability through confidence-guided regeneration.

3. Methodology

We introduce AsyncVLA, a VLA model enhanced by AFM. We start by introducing the self-correction mechanism of AFM in Sec. 3.1, followed by the confidence rater that determines the positions of masked action tokens in Sec. 3.2, and the overall training procedures in Sec. 3.3.

3.1. Asynchronous Flow Matching

We formulate the robot policy as a VLA model in a synergistic structure of VLM backbone and FM action head. The model can flexibly generate continuous action chunks whose length is denoted by L. The FM velocity for action generation can be written as $V_\theta\left(o_t, \ell, \hat{a}^\tau_{t:t+L}\right)$, where $o_t = \left[I_t^{(1)}, \ldots, I_t^{(n)}, q_t\right]$ consists of multi-view image ob-

Algorithm 1 Asynchronous Flow Matching Inference

Require: $\boldsymbol{o}_t,\ \ell,\ \hat{\boldsymbol{a}}^{\text{SFM}}_{t:t+L},\ \boldsymbol{m}\in\{0,1\}^L$; step size δ

For all l with $m_l = 0$: set $\hat{\boldsymbol{a}}_l^1 \leftarrow \hat{\boldsymbol{a}}_l^{\text{SFM}}$;
For all l with $m_l = 1$: set $\hat{\boldsymbol{a}}_l^1 \leftarrow \boldsymbol{n}_l \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
Set $\tau \leftarrow 1$
while $\tau > 0$ **do**
  Predict velocity: $\hat{\boldsymbol{v}}_{t:t+L} \leftarrow V_\theta\big(\boldsymbol{o}_t, \ell, \hat{\boldsymbol{a}}^\tau_{t:t+L}\big)$
  For all l with $m_l = 0$: $\hat{\boldsymbol{a}}_l^{\tau-\delta} \leftarrow \hat{\boldsymbol{a}}_l^{\tau}$
  For all l with $m_l = 1$: $\hat{\boldsymbol{a}}_l^{\tau-\delta} \leftarrow \hat{\boldsymbol{a}}_l^{\tau} - \delta\, \hat{\boldsymbol{v}}_l$
  $\tau \leftarrow \max(0,\ \tau - \delta)$
end while
return Final action $\tilde{\boldsymbol{a}}_{t:t+L} = \hat{\boldsymbol{a}}^0_{t:t+L}$

servations and robot state at time t, the language context ℓ is the embodied task instructions, and $\hat{\boldsymbol{a}}^\tau_{t:t+L}$ is the partially denoised action chunk at FM time τ.

As shown in Fig. 2, AsyncVLA consists of 3 sequential parts: SFM, the confidence rater, and AFM. SFM and AFM share the same model that is trained in a unified training procedure. For input, each token may correspond to a text token, an image patch token, a robot state token, or a partially denoised action token. For output, we employ an FM head to generate continuous action tokens.

Asynchronous Flow Matching Inference During inference of AFM, the model masks part of the action tokens generated by SFM, whose positions are denoted by the mask $\boldsymbol{m} \in \mathbb{R}^L$. The element of $\boldsymbol{m}$ is 1 if the corresponding action token is masked and is 0 otherwise. In AFM generation, the unmasked tokens remain unchanged, while the masked tokens are updated using the forward Euler rule as:

$$\hat{\boldsymbol{a}}^{\tau-\delta}_{t:t+L} \odot \boldsymbol{m} = \hat{\boldsymbol{a}}^{\tau}_{t:t+L} \odot \boldsymbol{m} - \delta V_\theta\left(\boldsymbol{o}_t, \ell, \hat{\boldsymbol{a}}^\tau_{t:t+L}\right), \quad (1)$$

where $\odot$ denotes token-wise Hadamard product, and δ denotes the time step size. For $\tau = 1$, we design the starting asynchronous noise $\hat{\boldsymbol{a}}^1_{t:t+L} = [\tilde{\boldsymbol{n}}_{t+1}, \cdots, \tilde{\boldsymbol{n}}_{t+L}]$ as:

$$\tilde{\boldsymbol{n}}_l = \begin{cases} \hat{\boldsymbol{a}}_l^{\text{SFM}}, & \text{if } m_l = 0, \\ \boldsymbol{n}_l \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), & \text{if } m_l = 1, \end{cases} \quad (2)$$

where m_l is the l-th element of $\boldsymbol{m}$ with $l = t+1, \cdots, t+L$, and $\hat{\boldsymbol{a}}_l^{\text{SFM}}$ is the action predicted by the previous SFM. In AFM, the model incorporates information from SFM-estimated action tokens even when the regenerated tokens remain in an early noisy stage, thereby producing more accurate actions. Since SFM can be regarded as a fully-masked special case of AFM, we employ the same model for both SFM and AFM modes. Thus, the VL KV-cache generated in SFM can be directly reutilized in AFM. In this way, we save the burden of repeatedly processing the VL tokens and thus significantly improve the model's inference efficiency for real-time control. The procedure of AFM inference is summarized in Algorithm 1.

Algorithm 2 Unified Training for AFM and SFM

Require: Dataset $\mathcal{D}$; FM model V_θ; batch size B

repeat
  Sample $\{(\boldsymbol{o}_t^{(i)}, \boldsymbol{a}^{(i)}_{t:t+L}, \ell^{(i)})\}_{i=1}^B \sim \mathcal{D}$
  for $i = 1, \ldots, B$ **do**
    Sample $y^{(i)} \sim \mathcal{U}(0,1)$; $m_l^{(i)} \sim \text{Bernoulli}(y^{(i)})$
    Sample $\tau^{(i)} \sim \text{Beta}(1.5, 1)$; $\boldsymbol{n}^{(i)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
    $\boldsymbol{u}^{(i)} \leftarrow \boldsymbol{n}^{(i)} - \boldsymbol{a}^{(i)}$ ▷ GT velocity in Eq. (4)
    $\hat{\boldsymbol{a}}^{\tau^{(i)}} \leftarrow \boldsymbol{a}^{(i)}_{t:t+L} - \tau^{(i)}\left(\boldsymbol{a}^{(i)}_{t:t+L} - \boldsymbol{n}^{(i)}_{t:t+L}\right) \odot \boldsymbol{m}^{(i)}$
    $\hat{\boldsymbol{v}}^{(i)} \leftarrow V_\theta\big(\boldsymbol{o}_t^{(i)}, \ell^{(i)}, \hat{\boldsymbol{a}}^{\tau^{(i)}}\big)$
  end for
  $\mathcal{L} \leftarrow \frac{1}{B}\sum_{i=1}^B \left\| [\hat{\boldsymbol{v}}^{(i)} - \boldsymbol{u}^{(i)}] \odot \boldsymbol{m}^{(i)} \right\|^2$ ▷ Eq. (4)
  Update θ by back-propagation on $\mathcal{L}$
until converged

Asynchronous Time Embedding in AFM To distinguish masked from unmasked action tokens, we propose the asynchronous time embedding module. Denote the dimension of the VLM's hidden states as d. At FM time τ, we first apply the sinusoidal encoding function $\mathcal{S}(\cdot)$ to map $\tau\boldsymbol{m}$ to the asynchronous time-embedding matrix $\mathcal{S}(\tau\boldsymbol{m}) \in \mathbb{R}^{L\times d}$. We then concatenate $\mathcal{S}(\tau\boldsymbol{m})$ and the linearly projected noisy action $\mathcal{P}(\hat{\boldsymbol{a}}^\tau_{t:t+L}) \in \mathbb{R}^{L\times d}$ along the last dimension and yield $\hat{\boldsymbol{h}}^\tau_{t:t+L} \in \mathbb{R}^{L\times 2d}$. Finally, a multi-layer perceptron (MLP) is utilized to project $\hat{\boldsymbol{h}}^\tau_{t:t+L}$ to the asynchronous time-embedded action hidden state $\hat{\boldsymbol{x}}^\tau_{t:t+L} \in \mathbb{R}^{L\times d}$. With the same hidden dimension as the VLM, $\hat{\boldsymbol{x}}^\tau_{t:t+L} \in \mathbb{R}^{L\times d}$ can be sent into the VLM's transformer backbone. Following [78], full attention is employed for action generation.

3.2. Confidence Rater

Since AsyncVLA lacks a dedicated output head for action-token logits, it is hard to directly estimate the model's confidence based on token probability. Thus, we individually design a confidence rater to estimate the confidence of the actions. The confidence rater takes the embeddings of VL tokens as well as the first-round actions generated by SFM as input, and evaluates the confidence of the l-th action token as $p_l \in (0,1)$, $l = t+1, \cdots, t+L$.

The confidence rater consists of several transformer layers and a final linear layer as its rate head. The action tokens are projected into the embedding space of VL tokens using a linear layer. The transformer layers apply full attention, such that the confidence can be calculated according to the VL information and the context actions before or after the evaluated action token. The rate head projects the hidden states to a scalar and employs Sigmoid function to generate

p_l. Using p_l, we generate the l-th element of the mask as:

$$m_l = \mathbb{1}\{p_l < T\}, \quad l = t+1, \dots, t+L, \tag{3}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function, and $T \in (0,1)$ is a predefined threshold that controls the number of masked tokens. In Eq. (3), a self-adaptive number of action tokens will be masked according to the actions' confidence, which offers better flexibility than other strategies such as Top-K selection. The confidence rater ensures that only actions with relatively large deviation are regenerated, and the unmasked actions providing context information are relatively accurate.

3.3. Training Procedures

Unified Training for SFM and AFM In order to employ a single model to realize both SFM and AFM inference, we propose a unified training procedure that treats SFM as a fully-masked special case of AFM. The VLM backbone and FM head are jointly trained by minimizing the following end-to-end AFM velocity prediction loss on masked tokens:

$$\mathcal{L} = \mathbb{E}_\tau \left\{ \left\| \left[V_\theta \left(\boldsymbol{o}_t, \ell, \hat{\boldsymbol{a}}^\tau_{t:t+L} \right) - \boldsymbol{u}_{t:t+L} \right] \odot \boldsymbol{m} \right\|^2 \right\}, \tag{4}$$

where $\boldsymbol{u}_{t:t+L} = \boldsymbol{n}_{t:t+L} - \boldsymbol{a}_{t:t+L}$ denotes the ground truth velocity with Gaussian noise $\boldsymbol{n}_{t:t+L} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, and $\hat{\boldsymbol{a}}^\tau_{t:t+L}$ denotes the intermediate asynchronous noisy action that is computed as:

$$\hat{\boldsymbol{a}}^\tau_{t:t+L} = \boldsymbol{a}_{t:t+L} - \tau \left(\boldsymbol{a}_{t:t+L} - \boldsymbol{n}_{t:t+L} \right) \odot \boldsymbol{m}. \tag{5}$$

In training, each element of $\boldsymbol{m}$ is identically and independently sampled from $\mathrm{Bernoulli}(y)$ with the pre-sampled probability $y \sim \mathcal{U}(0,1)$. Following [3], we sample the FM time τ from the Beta distribution $\mathrm{Beta}(1.5, 1)$ which emphasizes noisier time steps that are close to 1. Note that when the sampled $\boldsymbol{m}$ is an all-1 vector, the AFM loss in Eq. (4) degenerates to the vanilla SFM loss. Such training samples guarantee the unified model's ability of both AFM and SFM inference. The randomly sampled $\boldsymbol{m}$ also equivalently plays the role of data augmentation that improves the training data efficiency. The unified training procedure for AFM and SFM is summarized in Algorithm 2.

Training Confidence Rater After the VLA backbone is fully trained for SFM and AFM using Eq. (4), we train the confidence rater with the frozen VLA backbone in an end-to-end manner. Since the actions generated by SFM do not provide direct signals that indicate the confidence of the model, we need to deliberately design the pseudo labels of the confidence rater. We first compute the mean-squared error (MSE) of the action chunk generated by SFM denoted as $e_{t:t+L}$. Since the model should have higher confidence on tokens with smaller MSE and vice versa, we define the pseudo labels of the confidence rater as:

$$q_{t:t+L} = 1 - \alpha - \beta \frac{e_{t:t+L} - \min\{e_l\}}{\max\{e_l\} - \min\{e_l\} + \epsilon}, \tag{6}$$

where α and β are hyper-parameters that control the region of the pseudo labels in roughly $[1-\alpha-\beta, 1-\alpha]$, e.g., $[0.01, 0.99]$ if $\alpha = 0.01$ and $\beta = 0.98$, and ϵ is a small scalar that prevents the denominator being 0. In Eq. (6), we alleviate the gradient vanishing problem caused by the final Sigmoid function, by preventing the labels from being extremely close to 0 or 1. To evaluate the confidence in a relative manner, $\max\{e_l\}$ and $\min\{e_l\}$ denote the maximum and minimum MSE in the action chunk, respectively. We thus assign the relatively accurate action tokens high confidence and utilize their context information to regenerate the action tokens with low confidence. When training the confidence rater, we set the loss function to the MSE between the confidence rater's output and $q_{t:t+L}$.

4. Experiments

4.1. Experimental Setup

We adopt Qwen2.5-VL-3B-Instruct [1] as the VLM backbone and augment it with an FM action head along with a confidence rater. Our AsyncVLA is pretrained on the Open X-Embodiment dataset [52] and is subsequently finetuned for different evaluation tasks on the corresponding datasets, including LIBERO [44], Bridge-V2 [66], and Fractal [5]. In the data pre-processing stage, we mark the pause intervals in trajectories and exclude those action tokens from loss computation. The learning rate is set as 1×10^{-4} for both the language model backbone and the FM action head, and is set as 2×10^{-5} for the vision encoder.

4.2. Evaluation Results

LIBERO Benchmark We finetune and evaluate AsyncVLA and baselines on the LIBERO benchmark [44]. Results are presented in Tab. 1, where we evaluate the models over 500 trials per task suite (10 tasks × 50 episodes). It is seen that AsyncVLA performs well in the LIBERO environment, achieving the highest success rates in all 4 tasks. By analyzing rollout videos of successful cases and comparing them with the trajectory generated by SFM in the same task, we find that AsyncVLA demonstrates the ability of self-correction, particularly in challenging tasks where first-round actions contain errors.

Self-Correction Ability We illustrate AsyncVLA's self-correction ability on LIBERO-Long task in Fig. 3. The top row shows the actions generated by SFM, and the bottom row shows the actions regenerated by the following AFM. Since the confidence rater gives a low confidence on the "drop now" action, this action token is remasked while others are not. AsyncVLA takes the other high-confidence action tokens into account as context and finds that before the

Model	LIBERO-Spatial	LIBERO-Object	LIBERO-Goal	LIBERO-Long	Avg.
MDT [57]	78.5	87.5	73.5	64.8	76.1
OpenVLA [30]	84.7	88.4	79.2	53.7	76.5
WorldVLA [8]	87.6	96.2	83.4	60.0	81.8
Dita / DiT Policy [24]	84.2	96.3	85.4	63.8	82.4
TraceVLA [81]	84.6	85.2	75.1	54.1	74.8
SpatialVLA [55]	88.2	89.9	78.6	55.5	78.1
π_0-FAST [53]	96.4	96.8	88.6	60.2	85.5
π_0 [3]	96.8	98.8	95.8	85.2	94.2
OpenVLA-OFT (Con.) [31]	96.9	98.1	95.5	91.1	95.4
OpenVLA-OFT (Dis.) [31]	96.2	98.2	95.6	92.0	95.5
GR00T-N1 [2]	94.4	97.6	93.0	90.6	93.9
UD-VLA [9]	94.1	95.7	91.2	89.6	92.7
Discrete Diffusion VLA [42]	97.2	98.6	97.4	92.0	96.3
dVLA [68]	97.4	97.9	98.2	92.2	96.4
AsyncVLA (Ours)	**98.4**	**99.2**	**98.6**	**93.4**	**97.4**

Table 1. LIBERO task performance results evaluated by success rates. OpenVLA-OFT (Con./Dis.) refers to OpenVLA-OFT with continuous or discrete action. We finetune AsyncVLA on the combined 4 tasks as a whole, instead of training 4 individual VLA models.

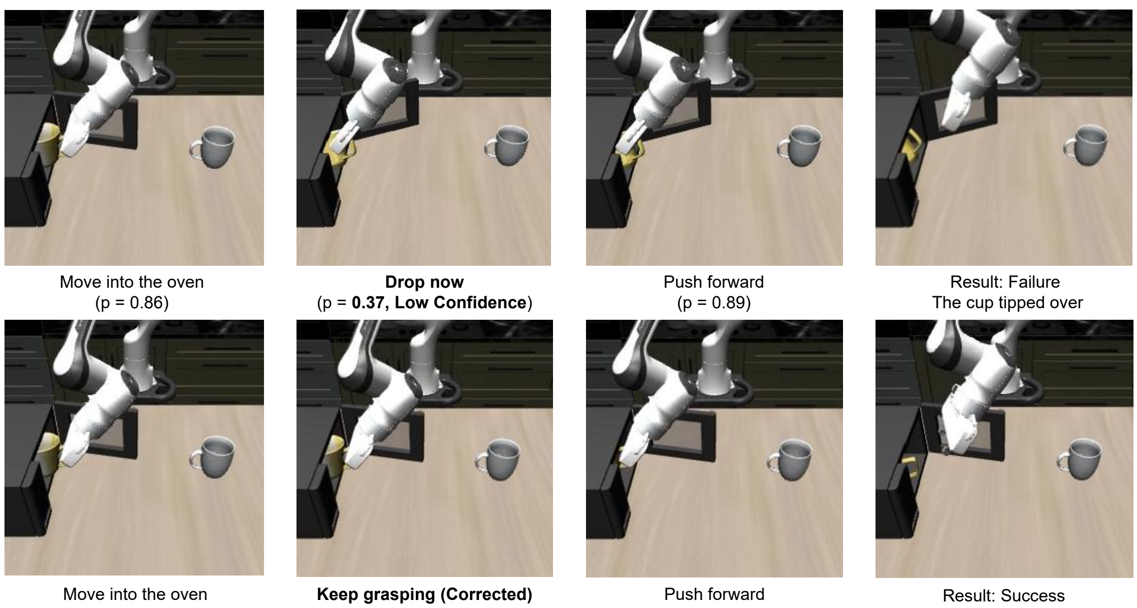

Figure 3. Illustration of self-correction ability in AsyncVLA on the LIBERO-Long task suite. The top row shows the first-round actions generated by SFM, and the bottom row shows the second-round actions regenerated by the following AFM.

"push forward" action, the correct action should be "keep grasping". After AFM's regeneration, the action with low confidence is corrected, and the task is successfully completed. It demonstrates that AFM with self-correction enhances AsyncVLA's robustness to perturbations from first-round generated erroneous actions with large deviation.

WidowX Robot Benchmark We evaluate AsyncVLA and baselines on the WidowX Robot benchmark after further finetuning on Bridge-V2 dataset [66]. We test across 4 generalization categories with environmental variations, and show the quantitative results in Tab. 2. AsyncVLA achieves the best performance in general, with the highest success rates in "put carrot on plate" and "stack cubes" tasks. Due to its much larger amount of training data, π_0 performs better in "put spoon on towel" and "put eggplant in basket" task. Compared to Magma, AsyncVLA shows a slightly lower success rate in "put eggplant in basket" task, with a small gap of 4.2%. However, AsyncVLA still demonstrates competitive performance across all 4 generalization categories, achieving the highest average success

Model	Put Spoon on Towel	Put Carrot on Plate	Stack Cubes	Put Eggplant in Basket	Avg.
OpenVLA [30]	0	0	0	4.1	1.0
SpatialVLA [55]	20.8	20.8	25.0	70.8	34.4
Magma [73]	37.5	29.2	20.8	**91.7**	44.8
π_0-FAST [53]	29.1	21.9	10.8	66.6	32.1
Octo-Base [19]	12.5	8.3	0	43.1	16.0
Octo-Small [19]	47.2	9.7	4.2	56.9	29.5
RoboVLM [39]	45.8	20.8	4.2	79.2	37.5
π_0 [3]	**83.8**	52.5	52.5	87.9	69.2
ThinkAct [25]	58.3	37.5	8.7	70.8	43.8
Discrete Diffusion VLA [42]	37.5	29.2	20.8	–	–
UD-VLA [9]	58.3	62.5	54.1	75.0	62.5
AsyncVLA (Ours)	70.8	**66.7**	**58.3**	87.5	**70.8**

Table 2. Comparison of different VLA models on the WidowX Robot benchmark. Evaluation is conducted in SimplerEnv [41].

Model	Pick Coke		Move Near		O/C Drawer		Put in Drawer		Average	
	M	A	M	A	M	A	M	A	M	A
OpenVLA [30]	16.3	54.5	46.2	47.7	35.6	17.7	0.0	0.0	24.5	30.0
TraceVLA [81]	28.0	60.0	53.7	56.4	57.0	31.0	0.0	0.0	34.7	36.9
SpatialVLA [55]	86.0	88.0	77.9	72.7	57.4	41.8	0.0	6.3	55.3	52.2
Magma [73]	75.0	68.6	53.0	78.5	58.9	59.0	8.3	24.0	48.8	57.5
π_0-FAST [53]	75.3	77.6	67.5	68.2	42.9	31.3	0.0	0.0	46.4	44.3
RT-1-X [5]	56.7	49.0	31.7	32.3	59.7	29.4	40.7	10.1	47.2	30.2
RT-2-X [6]	78.7	82.3	77.9	79.2	25.0	35.3	7.4	20.6	47.3	54.4
Octo-Base [19]	17.0	0.6	4.2	3.1	22.7	1.1	0.0	0.0	11.0	1.2
RoboVLM [39]	77.3	75.6	61.7	60.0	43.5	10.6	24.1	0.0	51.7	36.6
π_0 [3]	**97.9**	**90.1**	78.7	80.7	62.3	27.6	46.6	20.5	71.4	54.7
ThinkAct [25]	92.0	84.0	72.4	63.8	50.0	47.6	–	–	–	–
MolmoAct [32]	77.7	76.1	77.1	61.3	60.0	**78.8**	–	–	–	–
Discrete Diffusion VLA [42]	85.4	82.5	67.5	64.6	60.6	23.6	–	–	–	–
AsyncVLA (Ours)	96.2	89.6	**82.3**	**81.7**	**70.5**	56.0	**50.4**	**26.0**	**74.9**	**63.3**

Table 3. Comparison on the Google Robot benchmark under both visual matching (M) and variant aggregation (A) settings. The task "O/C Drawer" is short for "open or close the (top/middle/bottom) drawer". Evaluation is conducted in SimplerEnv [41].

rate among all VLA models.

Google Robot Benchmark We evaluate AsyncVLA and baselines on the Google Robot benchmark after further fine-tuning on the Fractal dataset [5]. We test across 4 task categories under 2 protocols: visual matching (M) that mirrors the real-world setup by varying object positions and variant aggregation (A) that introduces substantial perturbations in the environment. As shown in Tab. 3, AsyncVLA achieves the best performance in general, with the highest success rates in "move near" and "put in drawer" tasks. Due to its larger amount of training data, π_0 performs slightly better than our model in "pick coke can" task, with a margin of only 1.7% in success rate. AsyncVLA achieves lower success rate on variant aggregation suite of "put eggplant in basket" compared to MolmoAct and Magma. However, AsyncVLA maintains competitive performance across all 4 task categories overall, achieving the highest average success rate among all models.

4.3. Ablation Study

We conduct ablation studies on 4 tasks in the WidowX Robot benchmark. We evaluate 4 model variants: "w/o Unified Training" means without the unified training stage in Algorithm 2 and training the unified AFM and SFM model in the same way as vanilla SFM models; "w/o AFM Inference" means without the self-correction stage of AFM and generating the predicted actions with only SFM; "w/o Confidence Rater" means without the confidence rater and randomly generating the mask for AFM, where the probability of being masked is equal to 0.5 for each action token; "AsyncVLA (Ours)" means our complete method.

As shown in Tab. 4, all 4 tasks demonstrate that AsyncVLA consistently outperforms the models without the ablated components. Without the unified training stage

Model	Put Spoon on Towel	Put Carrot on Plate	Stack Cubes	Put Eggplant in Basket	Avg.
w/o Unified Training	4.1	8.3	0.0	16.7	7.3
w/o AFM Inference	58.3	54.2	33.3	45.8	47.9
w/o Confidence Rater	66.7	54.2	54.2	75.0	62.5
AsyncVLA (Ours)	**70.8**	**66.7**	**58.3**	**87.5**	**70.8**

Table 4. Ablation study on AsyncVLA's components in the WidowX Robot benchmark. Evaluation is conducted in SimplerEnv [41].

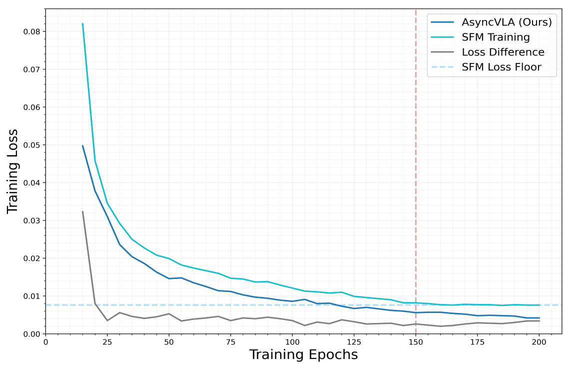

Figure 4. Training loss curve comparison when only part of the LIBERO-Spatial dataset is used for training.

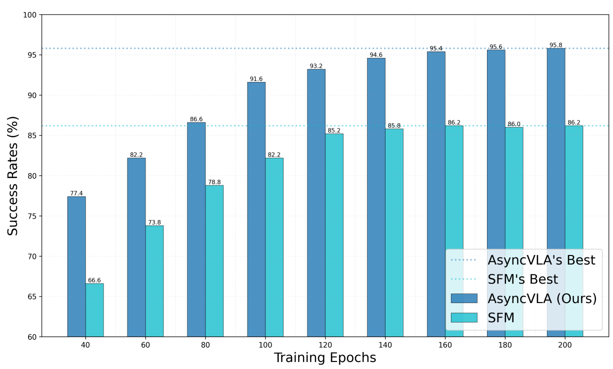

Figure 5. Success rate comparison in the training process. Evaluation is conducted on LIBERO-Spatial test suite.

proposed in Algorithm 2, the model performs miserably, achieving only an average success rate of 7.3%. The reason is that the model generates even worse actions in the AFM inference stage, as the input in AFM inference is not aligned with the input in vanilla SFM training stage. With the proposed unified training plus the AFM inference, the average success rate of the model increases by 14.6%, from 47.9% to 62.5%. The addition of the confidence rater further improves the average success rate to 70.8%. Our AsyncVLA achieves the best results, validating the effectiveness of the unified training for SFM and AFM, the AFM's regeneration that enables self-correction, and the confidence rater that determines the positions of masked action tokens.

4.4. Training Data Efficiency

To demonstrate the data efficiency of the proposed unified training method in data-constrained settings, we separately train the models using AsyncVLA's unified training method or vanilla SFM's training method. The models are trained for 200 epochs with constant learning rates using part of the LIBERO-Spatial dataset. The training loss curve comparison is shown in Fig. 4. It is seen that the training loss of AsyncVLA decreases faster and is remarkably and constantly lower than the training loss of SFM. Even when the training loss of SFM reaches a floor of 0.0076 and keeps almost unchanged after 150 epochs, the AsyncVLA's loss continues to decrease and ultimately reaches 0.0042 at the 200-th epoch.

We evaluate the two models' success rates on LIBERO-Spatial test suite every 20 epochs in Fig. 5. AsyncVLA constantly outperforms SFM by at least 7.8%. When the success rate of SFM actually stops increasing and fluctuates around 86.2% after 140 epochs, the success rate of AsyncVLA still stably improves and finally reaches 95.8%. This demonstrates that AsyncVLA better exploits the training data with longer training epochs in data-constrained settings. Such exploitation is attributed to the training data efficiency of the proposed unified training method that equivalently plays the role of data augmentation.

5. Conclusion

In this work, we introduce AsyncVLA, a novel framework that reframes action generation as a two-stage and confidence-aware process. Instead of using a fixed number of uniform denoising steps, AsyncVLA adaptively schedules the time steps in AFM. Moreover, we propose the confidence rater in AsyncVLA that estimates the relative confidence of each action token. Besides, we propose a unified training procedure for SFM and AFM, which endows a single model with both modes and improves KV-cache utilization. With the above improvement, AsyncVLA can dynamically reconsider its initially generated action tokens, focusing additional regeneration and asynchronous self-correction on the low-confidence components of each action chunk. Our extensive experiments demonstrate that AsyncVLA achieves state-of-the-art performance across general embodied evaluations.

AsyncVLA: Asynchronous Flow Matching for Vision-Language-Action Models

Supplementary Material

A. Implementation Details

A.1. Training

We select part of the Open X-Embodiment dataset as our robot demonstration pre-training data. We perform pre-training on 4 H200 GPU nodes (8 GPUs per node, 32 GPUs in total) under BF16 precision with gradient checkpointing enabled. We use ZeRO-2 optimizer sharding and flash-attention-2 for efficient memory usage. The global batch size is set to 2048. Pre-training takes roughly 2.5 days. We use cosine decay learning rate scheduler with the largest learning rate set as 1×10^{-4}. Further fine-tuning on LIBERO, Bridge-V2, and Fractal is performed on a single H200 node with 8 GPUs, requiring 15–32 hours depending on the dataset size. The chat template includes the prompt "You are a helpful physical assistant." at the beginning of each sample. Throughout all stages, we use AdamW optimizer with weight decay set to 0, β_1 = 0.9, and β_2 = 0.999. In the mask generation, we set the confidence threshold as $T = 0.5$. When training the confidence rater, we set $\alpha = 0.01$, $\beta = 0.98$, and $\epsilon = 1 \times 10^{-6}$.

Dataset	Weight
Bridge-V2	24.14%
RT-1	13.80%
TOTO	10.34%
VIOLA	10.34%
RoboTurk	10.34%
Jaco Play	10.34%
Berkeley Autolab UR5	10.34%
Berkeley Fanuc Manipulation	10.34%

Table 5. Dataset Weights

In Table 5, we summarize the detailed datasets and their weights during model pre-training stage.

A.2. Details of Flow Matching

For both synchronous flow matching (SFM) and asynchronous flow matching (AFM), we use a uniform schedule of 10 discretization steps. from noise at $t = 1$ toward its target action at $t = 0$. This choice provides a balanced trade-off between model's efficiency and performance.

A.3. Structure of Confidence Rater

The backbone of the confidence rater is a 4-layer transformer with a linear layer rate head, which sums up to 308M parameters and takes up 7.56% of the total 4.08B parameters of the overall VLA model. Each of the four transformer blocks contains multi-head self-attention with 32 attention heads and a feed-forward network width of 6144.

B. Percentage of Inference Time

Component	Percentage of Inference Time
SFM	86.8%
Confidence Rater	2.7%
AFM	10.5%

Table 6. Inference-time breakdown of SFM, confidence rater, and AFM.

During inference, the majority of computational cost is incurred by the flow matching process, especially in SFM, as shown in Tab. 6. In SFM, the model computes a full forward pass over all vision-language (VL) and action tokens for every diffusion step. This includes constructing the entire VL KV-cache from scratch, as the transformer must update all tokens uniformly at each of the 10 discretization steps. Consequently, SFM accounts for 86.8% of the total inference time. In contrast, AFM approach is substantially more efficient because it reuses the VL KV-cache produced in the very first pass. Specifically, the model processes the VL tokens only once at the beginning of inference; during this step, the transformer builds the KV-cache entries corresponding to vision tokens, language tokens, and instruction tokens. After this initialization, AFM does not recompute these caches in subsequent steps. Instead, it only updates the subset of action tokens whose confidence is below the threshold and are scheduled for refinement at the current timestep. The confidence rater introduces 2.7% overhead, since it is executed only once per action chunk, produces a lightweight scalar confidence value for each action token, and does not participate in iterative diffusion-style updates.

Overall, the reuse of VL KV-cache and the partial-token update mechanism explain why AFM inference is significantly faster than SFM, while still enabling dynamic trajectory refinement and higher success rates.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025. 5

[2] Johan Bjorck, Fernando Castañeda, Nikita Chentanez, Da Xinyue, Runyu Ding, Linxi Fan, Spencer Huang, Yifeng

Huang, Dieter Fox Fu, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.17434*, 2024. 6

[3] Kevin Black, Noah Brown, Danny Dries, Adnan Esmail, Michael Fiume, Chelsea Finn, Niccolo Fusi, Lachy Groom, Karol Hausman, Brian Ichter, and et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024. 2, 5, 6, 7

[4] Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. 2

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

[6] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. 1, 7

[7] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgeniy Chebotar, Xinghao Chen, Krzysztof Choromanski, Tianhe Ding, Danny Driess, Avinav Dubey, Chelsea Finn, et al. RT-X: Generalizable robot policy via large-scale multi-task learning. *arXiv preprint arXiv:2310.08864*, 2023. 1

[8] Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. WorldVLA: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025. 6

[9] Jiayi Chen, Wenxuan Song, Pengxiang Ding, Ziyang Zhou, Han Zhao, Feilong Tang, Donglin Wang, and Haoang Li. Unified diffusion VLA: Vision-language-action model via joint discrete denoising diffusion process. *arXiv preprint arXiv:2511.01718*, 2025. 2, 3, 6, 7

[10] Xinyi Chen, Yilun Chen, Yanwei Fu, Ning Gao, Jiaya Jia, Weiyang Jin, Hao Li, Yao Mu, Jiangmiao Pang, Yu Qiao, Yang Tian, Bin Wang, et al. InternVLA-M1: A spatially guided vision-language-action framework for generalist robot policy. *arXiv preprint arXiv:2510.13778*, 2025. 1

[11] Shuang Cheng, Yihan Bian, Dawei Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, and Bowen Zhou. SDAR: A synergistic diffusion-autoregression paradigm for scalable sequence generation. *arXiv preprint arXiv:2510.06303*, 2025. 2, 3

[12] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023. 1

[13] StarVLA Community. StarVLA: A lego-like codebase for vision-language-action model developing. *GitHub repository*, 2025. 2

[14] Shaoqi Dong, Chaoyou Fu, Haihan Gao, Yi-Fan Zhang, Chi Yan, Chu Wu, Xiaoyu Liu, Yunhang Shen, Jing Huo, Deqiang Jiang, Haoyu Cao, Yang Gao, Xing Sun, Ran He, and Caifeng Shan. VITA-VLA: Efficiently teaching vision-language models to act via action expert distillation. *arXiv preprint arXiv:2510.09607*, 2025. 2

[15] Danny Driess, Fei Xia, Mehdi S M Sajjadi, Corey Chen, Jonathan Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Vuong, Tianhe Yu, Wenhao D'Costa, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023. 1

[16] Yunhai Feng, Jiaming Han, Zhuoran Yang, Xiangyu Yue, Sergey Levine, and Jianlan Luo. Reflective planning: Vision-language models for multi-stage long-horizon robotic manipulation. *arXiv preprint arXiv:2502.16707*, 2025. 2

[17] Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv preprint arXiv:2401.02117*, 2024. 1

[18] Dechen Gao, Boqi Zhao, Andrew Lee, Ian Chuang, Hanchu Zhou, Hang Wang, Zhe Zhao, Junshan Zhang, and Iman Soltani. Vita: Vision-to-action flow matching policy. *arXiv preprint arXiv:2507.13231*, 2025. 2

[19] Divya Ghosh, Homer Rich Walk, Karl Pertsck, Kevin Black, Sudeep Mees, Tobias Hejna, Charles Xu Kreisman, Jianlan Liu, and Xi Li. Octo: An open-source generalist robot policy. *Robotics: Science and Systems*, 2024. 1, 2, 7

[20] Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Hao Peng, Jiawei Han, and Lingpeng Kong. Scaling diffusion language models via adaptation from autoregressive models. *arXiv preprint arXiv:2410.17891*, 2024. 2, 3

[21] Ankit Goyal, Hugo Hadfield, Xuning Yang, Valts Bulkis, and Fabio Ramos. VLA-0: Building state-of-the-art VLAs with zero modification. *arXiv preprint arXiv:2510.13054*, 2025. 1, 2

[22] Xiaoshuang Gu, Hongguang Liu, Yunhai Guo, Jun Li, Qingyong Yan, Hong Zhao, Shuai Liu, and Linqi Zeng. A survey on vision-language-action models for embodied ai. *arXiv preprint arXiv:2401.07172*, 2024. 1

[23] Junxian He et al. Diffusion-BERT: Generative masked language models. *arXiv preprint arXiv:2211.15029*, 2022. 2

[24] Zhi Hou, Tianyi Zhang, Yuwen Xiong, Haonan Duan, Hengjun Pu, Ronglei Tong, Chengyang Zhao, Xizhou Zhu, Yu Qiao, Jifeng Dai, and et al. Dita: Scaling diffusion transformer for generalist vision-language-action policy. *arXiv preprint arXiv:2503.19757*, 2025. 6

[25] Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. ThinkAct: Vision-language-action reasoning via reinforced visual latent planning. *arXiv preprint arXiv:2507.16815*, 2025. 7

[26] Jiannan Huang, Ding Ding, Zhixing Tang, Kai Liu, Yunhai Chen, Pengcheng He, and Bin Yang. A survey on integration of large language models with intelligent robots. *arXiv preprint arXiv:2404.09228*, 2024. 1

[27] Wenhui Huang, Changhe Chen, Han Qi, Chen Lv, Yilun Du, and Heng Yang. MoTVLA: A vision-language-action model with unified fast-slow reasoning. *arXiv preprint arXiv:2510.18337*, 2025. 1

[28] Zhiling Huang, Yuke Zhu, Fei Xia, and Manolis Savva. Open-ended language-guided planning for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18779–18790, 2023. 1

[29] Yuhua Jiang, Shuang Cheng, Yihao Liu, Ermo Hua, Che Jiang, Weigao Sun, Yu Cheng, Feifei Gao, Biqing Qi, and Bowen Zhou. Nirvana: A specialized generalist model with task-aware memory mechanism. *arXiv preprint arXiv:2510.26083*, 2025. 2

[30] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Rafael Rafailov, Ananya P. Foster, Pannag R. Sanketi, Quan Vuong, Sergey Levine, and et al. OpenVLA: An open-source vision-language-action model. In *8th Annual Conference on Robot Learning*, 2024. 1, 2, 6, 7

[31] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025. 1, 2, 6

[32] Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, Winson Han, Wilbert Pumacay, Angelica Wu, Rose Hendrix, Karen Farley, Eli VanderBilt, Ali Farhadi, Dieter Fox, and Ranjay Krishna. MolmoAct: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025. 7

[33] Baicheng Li, Dong Wu, Zike Yan, Xinchen Liu, Zecui Zeng, Lusong Li, and Hongbin Zha. Reflection-based task adaptation for self-improving VLA. *arXiv preprint arXiv:2510.12710*, 2025. 2, 3

[34] Chenxuan Li, Jiaming Liu, Guanqun Wang, Xiaoqi Li, Sixiang Chen, Liang Heng, Chuyan Xiong, Jiaxin Ge, Renrui Zhang, Kaichen Zhou, and Shanghang Zhang. A self-correcting vision-language-action model for fast and slow system manipulation. *arXiv preprint arXiv:2405.17418*, 2025. 2, 3

[35] Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, et al. SimpleVLA-RL: Scaling VLA training via reinforcement learning. *arXiv preprint arXiv:2509.09674*, 2025. 1

[36] Michael Li, Jianfong Li, Zhi-Qiang Yan, Jun Ma, Jian-Ping Zhang, Li-Ting Wang, Qing-Shan Zhou, and Hai-Ping Chen. Do as I can, not as I say: Grounding language in robotic affordances. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20281–20290, 2024. 1

[37] Peiyan Li, Yixiang Chen, Hongtao Wu, Xiao Ma, Xiangnan Wu, Yan Huang, Liang Wang, Tao Kong, and Tieniu Tan. BridgeVLA: Input-output alignment for efficient 3d manipulation learning with vision-language models. *arXiv preprint arXiv:2506.07961*, 2025. 1

[38] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Siheng Xu, Yizhong Zhang, and et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024. 1

[39] Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, Hanbo Zhang, and Huaping Liu. Towards generalist robot policies: What matters in building vision-language-action models. *arXiv preprint arXiv:2412.14058*, 2024. 7

[40] Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhiwei Li, Bao-Long Bi, Ling-Rui Mei, Junfeng Fang, Xiao Liang, Zhijiang Guo, Le Song, and Cheng-Lin Liu. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025. 2, 3

[41] Xuanlin Liang, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, Homer Rich Walk, Chuyuan Lunawat, Isabel Ishikaa, Sean Kimani, Sergey Levine, and et al. Evaluating real-world robot manipulation policies in simulation. In *Conference on Robot Learning*, pages 3705–3728, 2024. 7, 8

[42] Zhixuan Liang, Yizhuo Li, Tianshuo Yang, Chengyue Wu, Sitong Mao, Liuao Pei, Xiaokang Yang, Jiangmiao Pang, Yao Mu, and Ping Luo. Discrete diffusion VLA: Bringing discrete diffusion to action decoding in vision-language-action policies. *arXiv preprint arXiv:2508.20072*, 2025. 2, 3, 6, 7

[43] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 2

[44] Bo Liu, Yifeng Yuan, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Han, and Peter Stone. Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, pages 44776–44791, 2023. 1, 5

[45] Haoning Liu, Shuqiang Liu, Jun Song, Guozheng Zhang, Hong Liu, and Jianwen Zhang. A review of foundation models for vision, language and action in robotics. *arXiv preprint arXiv:2402.17643*, 2024. 1

[46] Jijia Liu, Feng Gao, Bingwen Wei, Xinlei Chen, Qingmin Liao, Yi Wu, Chao Yu, and Yu Wang. What can RL bring to VLA generalization? An empirical study. *arXiv preprint arXiv:2505.19789*, 2025. 1

[47] Jun Luo, Tong Zheng, Chueru Wu, Weiyu Wang, Xinyang Luo, Zhiao Zhou, and Shuran Song. Aloha: A low-cost hardware system for bimanual robotic manipulation. *arXiv preprint arXiv:2309.03055*, 2023. 1

[48] Qi Lv, Weijie Kong, Hao Li, Jia Zeng, Zherui Qiu, Delin Qu, Haoming Song, Qizhi Chen, Xiang Deng, and Jiangmiao Pang. F1: A vision-language-action model bridging understanding and generation to actions. *arXiv preprint arXiv:2509.06951*, 2025. 1

[49] Daniel J Mankowitz, Ilija Radosavovic, Xuanlin Xiao, Zhi-Qiang Zhou, Ziyuan Li, Haoyang Yu, Yujia Du, Yu-Liang Chen, Bo Song, Deepali Sunder, et al. Robocat: A self-improving robotic agent. *arXiv preprint arXiv:2306.00287*, 2023. 1

[50] Daniel J Mankowitz, Ilija Radosavovic, Xuanlin Xiao, Zhi-Qiang Zhou, Ziyuan Li, Haoyang Yu, Yujia Du, Yu-Liang Chen, Bo Song, Deepali Sunder, et al. Robocat: A self-improving robotic agent. *arXiv preprint arXiv:2306.00287*, 2023. 1

[51] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025. 2, 3

[52] Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Acorn Pooley, Arijit Gupta, Ajay Mandelkar, Ajinkya Jain, et al. Open X-Embodiment: Robotic learning datasets and RT-X models. *arXiv preprint arXiv:2310.08864*, 2023. 5

[53] 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, 6, 7

[54] Delin Qu, Haoming Song, Qizhi Chen, Zhaoqing Chen, Xianqiang Gao, Modi Shi, Guanghui Ren, Maoqing Yao, Bin Zhao, and Dong Wang. EmbodiedOneVision: Interleaved vision-text-action pretraining for general robot control. *arXiv preprint arXiv:2508.21112*, 2025. 2

[55] Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Jiayuan Wang, Bin Gu, and Zhiqiang Zhao. SpatialVLA: Exploring spatial representations for visual language-action model. *arXiv preprint arXiv:2501.15830*, 2025. 6, 7

[56] Scott Reed, Kory Zolna, Emilio Parisotto, Sergio Matthews, Melves Bartolo, Marcus Frean, Juhani Li, Lars Buesing, Wang Po-Wei, Deqing Niu, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022. 1

[57] Moritz Reuss, Ömer Erdinç Yağmurlu, Fabian Wenzel, Moritz Löwe, and Rudolf Lustig. Multimodal diffusion transformer: Learning versatile behavior from multimodal goals. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA*, 2024. 6

[58] Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *arXiv preprint arXiv:2403.01809*, 2024. 2, 3

[59] Ranjan Sapkota, Yang Cao, and Manoj Karkee. Vision-language-action models: Concepts, progress, applications and challenges. *arXiv preprint arXiv:2505.04769*, 2025. 1

[60] Ali Shafiullah, Shaurya Bahl, Stephen James, Deepak Pathak, and Pieter Abbeel. Language-driven generalization via CLIP for robot policy learning. *IEEE Robotics and Automation Letters (RA-L)*, 9(3):1885–1892, 2024. 1

[61] Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. MemoryVLA: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2508.19236*, 2025. 1

[62] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025. 1

[63] Nan Sun, Yongchang Li, Chenxu Wang, Huiying Li, and Huaping Liu. CollabVLA: Self-reflective vision-language-action model dreaming together with human. *arXiv preprint arXiv:2509.14889*, 2025. 1, 2

[64] Alexander Swerdlow, Mihir Prabhudesai, Siddharth Gandhi, Deepak Pathak, and Katerina Fragkiadaki. Unified multimodal discrete diffusion. *arXiv preprint arXiv:2503.20853*, 2025. 2, 3

[65] Yang Tian, Sizhe Yang, Jia Zeng, Ping Wang, Dahua Lin, Hao Dong, and Jiangmiao Pang. Predictive inverse dynamics models are scalable learners for robotic manipulation. In *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR)*, 2025. 1

[66] Homer Rich Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Maximilian Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Ho Vuong, Andre Wang He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A dataset for robot learning at scale. *arXiv preprint arXiv:2310.03816*, 2023. 1, 5, 6

[67] Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, Siteng Huang, Yifan Tang, Wenhui Wang, Ru Zhang, Jianyi Liu, and Donglin Wang. VLA-Adapter: An effective paradigm for tiny-scale vision-language-action model. *arXiv preprint arXiv:2509.09372*, 2025. 1

[68] Junjie Wen, Minjie Zhu, Jiaming Liu, Zhiyuan Liu, Yicun Yang, Linfeng Zhang, Shanghang Zhang, Yichen Zhu, and Yi Xu. dVLA: Diffusion vision-language-action model with multimodal chain-of-thought. *arXiv preprint arXiv:2509.25681*, 2025. 1, 2, 3, 6

[69] Junjie Wen, Minjie Zhu, Yichen Zhu, Zhibin Tang, Jinming Li, Zhongyi Zhou, Chengmeng Li, Xiaoyu Liu, Yaxin Peng, Chaomin Shen, and Feifei Feng. Diffusion-VLA: Generalizable and interpretable robot foundation model via self-generated reasoning. *arXiv preprint arXiv:2412.03293*, 2025. 1

[70] Yuqing Wen, Hebei Li, Kefan Gu, Yucheng Zhao, Tiancai Wang, and Xiaoyan Sun. LLaDA-VLA: Vision language diffusion action models. *arXiv preprint arXiv:2509.06932*, 2025. 2, 3

[71] Chengyue Wu, Hao Zhang, Shuchen Xue, Shizhe Diao, Yonggan Fu, Zhijian Liu, Pavlo Molchanov, Ping Luo, Song Han, and Enze Xie. Fast-dLLM v2: Efficient block-diffusion llm. *arXiv preprint arXiv:2509.26328*, 2025. 2, 3

[72] Zhenyu Wu, Yuheng Zhou, Xiuwei Xu, Ziwei Wang, and Haibin Yan. MoManipVLA: Transferring vision-language-action models for general mobile manipulation. *arXiv preprint arXiv:2503.13446*, 2025. 1

[73] Jianwei Yang, Reuben Tan, Qianhui Wu, Ruijie Zheng, Baolin Peng, Yongyuan Liang, Yu Gu, Mu Cai, Seonghyeon Ye, Joel Jang, Yuquan Deng, Lars Liden, and Jianfeng Gao. Magma: A foundation model for multimodal ai agents. *arXiv preprint arXiv:2502.13130*, 2025. 7

[74] Chao Yu, Yuanqing Wang, Zhen Guo, Hao Lin, Si Xu, Hongzhi Zang, Quanlu Zhang, Yongji Wu, Chunyang Zhu, Junhao Hu, Zixiao Huang, Mingjie Wei, Yuqing Xie, Ke Yang, Bo Dai, Zhexuan Xu, et al. RLinf: Flexible and efficient large-scale reinforcement learning via macro-to-micro

flow transformation. *arXiv preprint arXiv:2509.15965*, 2025. 1

[75] Hongzhi Zang, Mingjie Wei, Si Xu, Yongji Wu, Zhen Guo, Yuanqing Wang, Hao Lin, Liangzhi Shi, Yuqing Xie, Zhexuan Xu, Zhihao Liu, et al. RLinf-VLA: A unified and efficient framework for VLA+RL training. *arXiv preprint arXiv:2510.06710*, 2025. 1

[76] Andy Zhai, Brae Liu, Bruno Fang, Chalse Cai, Ellie Ma, Ethan Yin, Hao Wang, Hugo Zhou, James Wang, Lights Shi, Lucy Liang, Make Wang, Qian Wang, Roy Gan, Ryan Yu, Shalfun Li, Starrick Liu, Sylas Chen, Vincent Chen, and Zach Xu. Igniting vlms toward the embodied space. *arXiv preprint arXiv:2509.11766*, 2025. 2

[77] Qinglun Zhang, Zhen Liu, Haoqiang Fan, Guanghui Liu, Bing Zeng, and Shuaicheng Liu. Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation. *arXiv preprint arXiv:2412.04987*, 2024. 2

[78] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. In *CVPR*, 2024. 1, 2, 4

[79] 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. 1

[80] Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025. 2

[81] Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. TraceVLA: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *Proceedings of the Thirteenth International Conference on Learning Representations (ICLR)*, 2025. 6, 7

[82] Zhide Zhong, Haodong Yan, Junfeng Li, Xiangchen Liu, Xin Gong, Tianran Zhang, Wenxuan Song, Jiayi Chen, Xinhu Zheng, Hesheng Wang, and Haoang Li. FlowVLA: Visual chain of thought-based motion reasoning for vision-language-action models. *arXiv preprint arXiv:2508.18269*, 2025. 1, 2