ACG: Action Coherence Guidance for Flow-based VLA models

Anonymous Authors

***Abstract*— Diffusion and flow matching models have emerged as powerful robot policies, enabling Vision-Language-Action (VLA) models to generalize across diverse scenes and instructions. Yet, when trained via imitation learning, their high generative capacity makes them sensitive to noise in human demonstrations: jerks, pauses, and jitter which reduce action coherence. Reduced action coherence causes instability and trajectory drift during deployment, failures that are catastrophic in fine-grained manipulation where precision is crucial. In this paper, we present Action Coherence Guidance (ACG) for VLA models, a training-free test-time guidance algorithm that improves action coherence and thereby yields performance gains. Evaluated on RoboCasa, DexMimicGen, and real-world SO-101 tasks, ACG consistently improves action coherence and boosts success rates across diverse manipulation tasks.**

I. Introduction

Diffusion and flow matching models are reshaping how robots learn to manipulate objects [1]. These generative models act as robot policies that directly model complex action distributions from human demonstrations, enabling strong generalization across diverse manipulation tasks. This paradigm has been further extended to Vision-Language-Action (VLA) models, enabling generalization across a wide range of scenes and language instructions [2]–[5].

Despite these advances, diffusion and flow matching policies trained via imitation learning remain highly sensitive to noise in human demonstrations, such as pauses, jerks, or jitter [6]–[9]. Their large generative capacity often memorizes these imperfections, which degrades *action coherence* of the learned policies [10], [11]. Formally, *action coherence* denotes the smoothness and consistency of successive actions, which can be measured by variability or jerks [12], [13].

During deployment, the loss of action coherence leads to two key failures. First, unstable actions can cause instability at critical moments. For example, in a pick-and-place task, the robot may fumble near the target object or inadvertently push it away. Second, even minor action noise can accumulate over time, causing the robot's trajectory to drift from desired states. Therefore, enhancing action coherence between action sequences is essential for ensuring reliable and robust manipulation.

To address this challenge, we leverage guidance strategies from the flow matching literature, which improve prediction quality without additional training. A representative example is Classifier-Free Guidance (CFG) [14], which has been widely used in image and video generation. CFG strengthens conditioning signals by guiding the samples away from the unconditional generation direction. Recent studies extend the CFG by perturbing the predicted denoising vector and reversing it as guidance, thereby steering the sampling process away from undesirable characteristics and improving the fidelity of generated outputs [18]–[22].

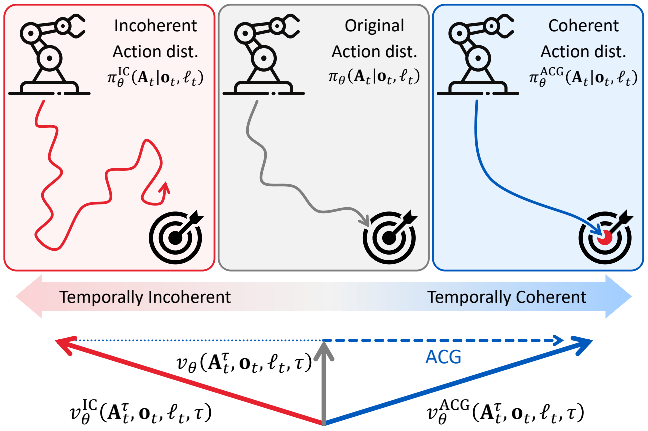
Incoherent
Action dist.
$\pi_\theta^{\mathrm{IC}}(\mathbf{A}_t|\mathbf{o}_t,\ell_t)$
Original
Action dist.
$\pi_\theta(\mathbf{A}_t|\mathbf{o}_t,\ell_t)$
Coherent
Action dist.
$\pi_\theta^{\mathrm{ACG}}(\mathbf{A}_t|\mathbf{o}_t,\ell_t)$
Temporally Incoherent
Temporally Coherent
$v_\theta(\mathbf{A}_t^\tau,\mathbf{o}_t,\ell_t,\tau)$
ACG
$v_\theta^{\mathrm{IC}}(\mathbf{A}_t^\tau,\mathbf{o}_t,\ell_t,\tau)$
$v_\theta^{\mathrm{ACG}}(\mathbf{A}_t^\tau,\mathbf{o}_t,\ell_t,\tau)$

Fig. 1. Conceptual illustration of ACG. ACG constructs an incoherent vector field v_θ^{IC} and combines it with the original vector field v_θ to extrapolate a guidance vector that steers sampling toward coherent action sequences.

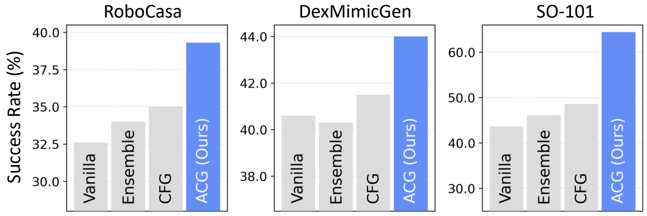

Fig. 2. Performance improvements of ACG over Vanilla GR00T-N1 [2], naïve multi-inference ensembling, and classifier-free guidance [14] on simulated manipulation benchmarks (RoboCasa [15], DexMG [16]) and real-world pick-and-place tasks (SO-101 [17]).

Building on these insights, we propose **A**ction **C**oherence **G**uidance (**ACG**), a simple yet effective test-time guidance strategy that enhances action coherence in flow-based policy. As illustrated in Fig. 1, we begin by constructing a denoising vector that drives the policy toward incoherent actions. Specifically, ACG disrupts temporal communication by replacing the attention map in the self-attention layers with an identity attention map. This disruption breaks coordination among tokens, yielding temporally incoherent action sequences. ACG then guides the sampling in the opposite direction of this incoherent denoising vector, encouraging temporal consistency. As a result, ACG generates action sequences with significantly enhanced coherence.

We evaluate ACG on standard multi-task manipulation benchmarks, RoboCasa [15] and DexMimicGen [16], as well as on real-world pick-and-place tasks using the SO-101 robot [17]. As shown in Fig. 2, across all settings, ACG delivers substantial performance gains, particularly on fine manipulation tasks such as pressing buttons (+23.1 pp), insertion (+11.8 pp), and real-world pick-and-place (+28.8 pp). These results demonstrate the effectiveness of ACG in generating coherent action sequences with flow-based VLA models, without requiring additional training.

II. Related Work

A. VLA Models with Flow Matching Policy

Manipulation has been approached through imitation learning from human demonstrations [1], [6], [10], [23]–[27], and the research has shifted toward building foundation models, known as Vision-Language-Action (VLA) models, to enable generalization across various tasks under a single framework. While the early VLA architectures relied on autoregressive LLM architecture [28]–[31], it has been shifted to employ diffusion and flow matching policy action heads [2]–[5]. VLA models take multimodal inputs, including visual observations, language instructions, states, and leverage them to predict corresponding actions. Their architecture typically consists of a vision-language backbone coupled with an action head, where the latter can be autoregressive or generative. In this work, we focus on improving generative flow-based VLA models (*e.g.*, GR00T-N1 [2], π_0 [3], SmolVLA [4]), which offer improved stability and expressiveness for continuous control in manipulation tasks.

B. Guidance for Flow Matching Policy

Guidance is a widely used technique in diffusion and flow-based generative models to improve sample quality [14], [32]. A prominent example is Classifier-Free Guidance (CFG) [14], which enhances text adherence by steering generation away from the unconditional vector field. Inspired by CFG, recent studies have explored guidance strategies in robot control, showing performance improvements in goal-conditioned imitation learning by generating negative guidance through removing the goal condition [33]–[35]. However, in VLA models, applying CFG by replacing the language condition often shows unstable behaviors, since action distribution can vary significantly with even subtle differences in language instructions [5], [36].

Guiding the model with language condition also poses challenges in visual generation. In particular, CFG often overly enforces the text condition, leading to the generation of unrealistic samples and reduced sample diversity. To address this, recent work has explored perturbation guidance, which steers pretrained diffusion models toward higher quality by leveraging an intentionally degraded version of the model rather than the unconditional model [18]. Such degradation can be conducted by dropping out units [18], [19], or perturbing the attention maps [20]–[22].

C. Action Coherent Policy

A straightforward remedy is to enforce temporal consistency in the generated action sequence. For instance, smoothing the action sequence with a Gaussian kernel can generate smoother actions but at the cost of distorting the pretrained action distribution.

a) Action Chunking for Coherent Generation: Prior work improves coherence through action chunking [1], [10], which generates multiple actions (typically k steps at a time) simultaneously. By shortening the effective task horizon by a factor of k, action chunking reduces compounding error and promotes smoother trajectories [11], [37]–[41]. ACT [10] further proposed temporal ensembling, which can be effectively used with autoregressive architectures, but applying it to flow matching policies incurs substantial inference overhead due to their non-autoregressive nature.

While chunking enhances coherence across timesteps, it does not eliminate the incoherence that remains within each chunk. Shaky or unstable motions can still occur within each action chunk, often leading to critical mistakes at key moments, such as grasping or picking up the objects. To address this limitation, we aim to improve action chunking by reducing incoherence *within each generated action chunk*.

b) Guidance for Action Coherence: More recently, researchers have explored guidance strategies to improve action coherence, motivated by the inference overhead of temporal ensembling [10] in flow-matching policies. These approaches guide the generation to be more consistent with the preceding action chunks [11], [39], [40]. Nevertheless, ensuring coherent actions within each chunk remains underexplored, despite its importance for fine-grained manipulation tasks. To the best of our knowledge, this work is the first to (i) introduce perturbation guidance into robot control and (ii) explicitly address action coherence within each action chunk.

III. Preliminaries

A. Flow Matching Policy

Diffusion and flow matching policies have recently emerged as a powerful framework for imitation learning, enabling robots to capture the stochastic and multi-modal characteristics of human demonstrations [1]–[4]. Here, we outline the training procedure and output representation of a rectified flow matching policy. Human demonstration dataset can be denoted as $\{D_i\}_{i=1}^N$, where N is the number of demonstrations and each demonstration D is given by

$$D = \{(\ell_1, \mathbf{o}_1, \mathbf{a}_1), (\ell_2, \mathbf{o}_2, \mathbf{a}_2), \ldots, (\ell_T, \mathbf{o}_T, \mathbf{a}_T)\}, \quad (1)$$

where ℓ_t denotes a language instruction and $\mathbf{o}_t$ represents the observation at time t. The observation consists of multiple RGB images and the robot's joint state. The goal is then to model the action distribution $p(\mathbf{A}_t \mid \mathbf{o}_t, \ell_t)$, where $\mathbf{A}_t = [\mathbf{a}_t, \mathbf{a}_{t+1}, \ldots, \mathbf{a}_{t+k-1}]$ denotes an *action chunk*.

During training, the noisy input action chunk $\mathbf{A}_t^\tau$ is constructed from a clean action chunk $\mathbf{A}_t$, a flow matching timestep $\tau \in [0, 1]$, and sampled noise $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, such that $\mathbf{A}_t^\tau = \tau \mathbf{A}_t + (1-\tau)\epsilon$. The conditional flow matching policy $v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau)$ is trained to match the conditional denoising vector field $u(\mathbf{A}_t^\tau \mid \mathbf{A}_t)$,

$$u(\mathbf{A}_t^\tau \mid \mathbf{A}_t) = \frac{d\mathbf{A}_t^\tau}{d\tau} = \mathbf{A}_t - \epsilon, \quad (2)$$

where subscripts denote robot timesteps and superscripts denote flow matching timesteps. The training objective is

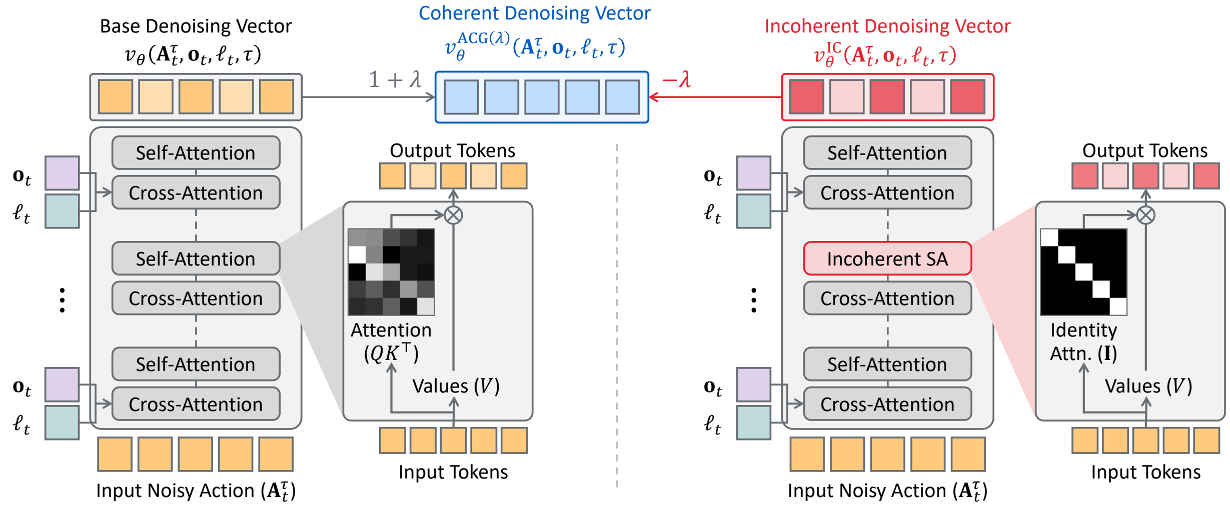
Base Denoising Vector
$v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau)$
Coherent Denoising Vector
$v_\theta^{\mathrm{ACG}(\lambda)}(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau)$
Incoherent Denoising Vector
$v_\theta^{\mathrm{IC}}(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau)$
$1+\lambda$
$-\lambda$
$\mathbf{o}_t$
ℓ_t
Self-Attention
Cross-Attention
Incoherent SA
Output Tokens
Attention ($QK^\top$)
Identity Attn. ($\mathbf{I}$)
Values (V)
Input Tokens
Input Noisy Action ($\mathbf{A}_t^\tau$)

Fig. 3. Illustration of ACG. Based on the original inference procedure (left), we modify the self-attention layer by replacing the attention map with the identity map to generate incoherent action sequence (right). Finally, we guide the denoising vector with the opposite direction of the incoherent denoising vector.

given by the following loss function [42], [43]:

$$\mathcal{L}(\theta) = \mathbb{E}_{(\ell_t, \mathbf{o}_t, \mathbf{A}_t), \tau}\left[\, \| v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau) - (\mathbf{A}_t - \epsilon) \|^2 \,\right]. \tag{3}$$

During inference, the flow matching policy generates an action chunk by integrating the learned vector field, starting from random noise $\mathbf{A}_t^0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The action chunk is generated via forward Euler integration as follows:

$$\mathbf{A}_t^{\tau+\delta} = \mathbf{A}_t^\tau + \delta\, v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau), \tag{4}$$

where δ is the integration step size, which is set to $1/16$ in our experiments (*i.e.*, 16 denoising timesteps) [2].

B. Classifier-Free Guidance for Flow Matching Policy

Classifier-Free Guidance (CFG) [14] has been widely used to improve conditioning in visual generation, and it has also been explored for policy networks [33]–[35]. CFG is applied at inference time and defines a guided distribution as

$$\pi^{\mathrm{CFG}(\lambda)}(\mathbf{A}_t | \mathbf{o}_t, \ell_t) \propto \pi_\theta(\mathbf{A}_t | \mathbf{o}_t, \ell_t) \left(\frac{\pi_\theta(\mathbf{A}_t | \mathbf{o}_t, \ell_t)}{\pi_\theta(\mathbf{A}_t | \mathbf{o}_t, \emptyset)} \right)^\lambda, \tag{5}$$

where $\emptyset$ denotes a null text, $\pi_\theta(\mathbf{A}_t | \mathbf{o}_t, \emptyset)$ indicates the unconditional distribution, and λ is the guidance scale. Here, following the Goal-Conditioned Behavioral Cloning (GCBC) setting commonly adopted in recent flow matching policies, only the language instruction ℓ_t is regarded as the CFG condition, while the observation $\mathbf{o}_t$ is not [33], [34]. This formulation shows that CFG amplifies the contribution of the conditional distribution while pushing the sampling process away from the unconditional distribution.

To achieve this, flow-based generative models utilize the following guided vector fields [44], [45]:

$$\begin{aligned} v_\theta^{\mathrm{CFG}(\lambda)}(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau) = {} & (1+\lambda)\, v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau) \\ & - \lambda\, v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \emptyset, \tau), \end{aligned} \tag{6}$$

where $v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau)$ and $v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \emptyset, \tau)$ are the conditional and unconditional vector fields, respectively. In summary, CFG strengthens conditional generation by pushing the denoising direction away from the unconditional vector field and toward the conditional vector field.

Algorithm 1 Action Coherence Guidance (ACG)

```
Input  : observation $\mathbf{o}_t$, instruction $\ell_t$, guidance scale $\lambda$,
         denoising step size $\delta$
Output: generated action chunk $\mathbf{A}_t^1$
$\mathbf{A}_t^0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$                    ▷ Initialize with Gaussian noise
for $\tau = 0$ to 1 with step size $\delta$ do
    $v^{\mathrm{original}} \leftarrow v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau)$          ▷ Infer original vector
    $v^{\mathrm{IC}} \leftarrow v_\theta^{\mathrm{IC}}(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau)$          ▷ Infer incoherent vector
    $v_\theta^{\mathrm{ACG}} \leftarrow (1+\lambda)\, v^{\mathrm{original}} - \lambda\, v^{\mathrm{IC}}$   ▷ Guide toward opposite
    $\mathbf{A}_t^{\tau+\delta} \leftarrow \mathbf{A}_t^\tau + \delta\, v_\theta^{\mathrm{ACG}}$              ▷ Denoise action chunk
end
return $\mathbf{A}_t^1$
```

IV. Method

A. Guiding Policy with Incoherent Action Generation

Noisy patterns such as jerks, hesitations, and overshooting in human demonstrations introduce incoherence in flow-based imitation learning, which is a critical issue for robotic manipulation tasks that demand precise object interactions. To address this, we propose a training-free *coherent action generation* method that steers the policy away from an intentionally constructed *incoherent vector field*. In particular, we contrast the original policy with its incoherent variant, guiding the model away from the unstable and temporally incoherent behaviors and toward stable and coherent actions, as illustrated in Fig. 1.

To achieve this, we define the guided distribution of coherent action generation as

$$\pi^{\mathrm{ACG}(\lambda)}(\mathbf{A}_t | \mathbf{o}_t, \ell_t) \propto \pi_\theta(\mathbf{A}_t | \mathbf{o}_t, \ell_t) \left(\frac{\pi_\theta(\mathbf{A}_t | \mathbf{o}_t, \ell_t)}{\pi_\theta^{\mathrm{IC}}(\mathbf{A}_t | \mathbf{o}_t, \ell_t)} \right)^\lambda, \tag{7}$$

where $\pi_\theta^{\mathrm{IC}}(\mathbf{A}_t | \mathbf{o}_t, \ell_t)$ denotes the distribution induced by the incoherent denoising vector. Correspondingly, the vector field

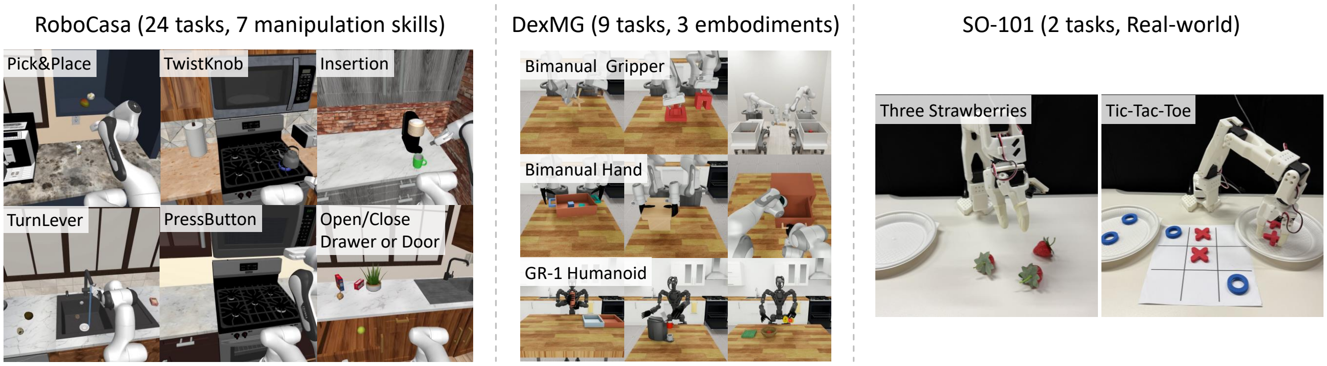
RoboCasa (24 tasks, 7 manipulation skills)
Pick&Place
TwistKnob
Insertion
TurnLever
PressButton
Open/Close Drawer or Door
DexMG (9 tasks, 3 embodiments)
Bimanual Gripper
Bimanual Hand
GR-1 Humanoid
SO-101 (2 tasks, Real-world)
Three Strawberries
Tic-Tac-Toe

Fig. 4. Visual demonstration of the simulation and real-world benchmarks used in our experiments. RoboCasa [15] contains 24 atomic manipulation tasks in kitchen environments, grouped into seven skill domains. DexMG [16] provides three bimanual embodiments for dexterous manipulation tasks. SO-101 [17] features real-world pick-and-place tasks, including strawberries and tic-tac-toe.

for coherent action generation is formulated as:

$$v_\theta^{\mathrm{ACG}(\lambda)}(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau) = (1+\lambda)\, v_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau) \\ - \lambda\, v_\theta^{\mathrm{IC}}(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau), \tag{8}$$

where λ is the guidance scale and $v_\theta^{\mathrm{IC}}(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau)$ denotes the incoherent vector field. The specific design of the vector field is described in the following subsection. The action chunk is then generated through forward Euler integration, in the same way as in CFG:

$$\mathbf{A}_t^{\tau+\delta} = \mathbf{A}_t^\tau + \delta\, v_\theta^{\mathrm{ACG}(\lambda)}(\mathbf{A}_t^\tau, \mathbf{o}_t, \ell_t, \tau). \tag{9}$$

The overall procedure is summarized in Algorithm 1.

B. Constructing Incoherent Action Generation Vector

To construct the incoherent action vector field, we examine the architecture of the flow matching policy. The flow matching policy adopts a fully transformer-based architecture [46], which employs self-attention to ensure coherence across generated tokens by allowing tokens to communicate with one another. In a flow matching policy, each token represents an action at a specific timestep along the temporal horizon. The attention operation is expressed as:

$$\mathrm{Attn}(Q,K,V) = \underbrace{\mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)}_{\text{Attention Map}} V, \tag{10}$$

where the attention map controls temporal coherence, how strongly each action attends to others, thereby promoting temporal coherence among the value tokens V.

Based on the attention mechanism, we construct v_θ^{IC} as an incoherent action generation vector field by replacing the attention operation with *incoherent self-attention*. Concretely, the attention map is replaced with an identity attention map, forcing each action token to attend only to itself, which can be written as:

$$\mathrm{Attn}_{\mathrm{IC}}(Q,K,V) = \underbrace{\mathbf{I}}_{\text{Identity Attention Map}} V = V, \tag{11}$$

where $\mathbf{I}$ indicates the identity matrix, enforcing temporal disconnection across the value tokens. As a result, v_θ^{IC} produces an action chunk with reduced temporal coherence and serves as a useful reference for the opposite direction of coherent action generation, as illustrated in Fig. 3.

V. Experiments

We now present experiments that evaluate the effectiveness of ACG for coherent action generation and, consequently, manipulation performance in VLA models. Section V-A describes the experimental setup and baselines. The subsequent sections address the following research questions:

- How does ACG improve the performance of VLA models in manipulation tasks? (Section V-B)
- Does ACG generate coherent actions? (Section V-C)
- Which self-attention layers are most effective to perturb for guidance? (Section V-D)
- How well does ACG perform across various flow-based VLA models? (Section V-D)

A. Experimental Setup

a) Benchmarks: We evaluate our method on two simulation benchmarks and one real-world benchmark. The simulation benchmarks are drawn from open-source suites for tabletop manipulation, as illustrated in Fig. 4. RoboCasa [15] spans seven manipulation skill domains with a total of 24 tasks, while DexMimicGen [16] provides diverse bimanual manipulation tasks across multiple embodiments.

For real-world evaluation, we conducted two pick-and-place tasks with SO-101 [17]: Three Strawberries and Tic-Tac-Toe. In Three Strawberries, performance is scored by the number of strawberries placed, one (33.3%), two (66.7%), or three (100%). In Tic-Tac-Toe, performance is divided equally between picking (50%) and placing (50%) the pieces, following previous literature [47]. Every experiment is conducted three times to report standard deviation, with 24 trials for simulation and 10 trials for real-world experiments (*i.e.*, 72 and 30 trials in total).

b) Implementation Details: We primarily use GR00T-N1 [2] for most of our experiments. For simulation, we perform multi-task training on RoboCasa [15] and cross-embodiment training on DexMimicGen [16], using 100

TABLE I

QUANTITATIVE COMPARISON ACROSS SIMULATION AND REAL-WORLD BENCHMARKS.

Method	Simulation		Real-world		Average
	RoboCasa	DexMG[1†]	Three Strawberries	Tic-Tac-Toe	
Vanilla GR00T-N1	32.6% (±2.07%)	40.6% (±3.08%)	43.6% (±5.29%)	38.3% (±2.89%)	38.8% (±2.34%)
Action Smoothing Methods					
Ensemble ($n = 2$)	34.0% (±0.62%)	40.3% (±4.42%)	56.7% (±6.67%)	45.0% (±5.00%)	44.0% (±2.56%)
Ensemble ($n = 5$)	33.9% (±0.71%)	40.0% (±5.79%)	54.4% (±1.92%)	43.3% (±5.77%)	42.9% (±2.93%)
Action Smoothing	34.0% (±1.40%)	41.2% (±3.04%)	47.8% (±1.92%)	36.7% (±2.89%)	39.9% (±1.55%)
Feature Smoothing	34.4% (±1.15%)	42.4% (±4.75%)	57.8% (±3.85%)	45.0% (±5.00%)	44.9% (±1.07%)
Guidance-based Action Generation Methods					
CFG	35.0% (±1.35%)	41.5% (±3.08%)	50.0% (±3.33%)	43.3% (±2.89%)	42.5% (±1.34%)
WNG	35.0% (±0.47%)	42.0% (±3.54%)	65.6% (±5.09%)	48.3% (±5.77%)	47.7% (±3.34%)
ACG (Ours)	**39.3%** (±3.02%)	**44.0%** (±2.41%)	**74.4%** (±3.85%)	**56.7%** (±2.89%)	**53.6%** (±0.73%)

demonstrations per task for both benchmarks. For the real-world evaluation, we conduct single-task training on SO-101 for the Three Strawberries and Tic-Tac-Toe tasks with 50 and 40 human demonstrations, respectively. To improve grasp reliability given the hardware limitations, we additionally covered the SO-101 gripper with a rubber thimble. All post-training of GR00T-N1 is performed with a batch size of 128, 60k iterations, and a peak learning rate of 0.0001, following the original setup [2]. For ACG, we replace the 4th–6th self-attention layers with incoherent ones out of the eight total layers, using a guidance scale of 3.0. All source code and datasets will be released publicly.

c) Baselines: We compare our method with several practical baselines designed to enhance action generation. First, we evaluate the vanilla VLA model, which does not employ any inference-time algorithm.

(i) *Vanilla GR00T-N1 [2]*: This baseline simply samples actions from the base flow-matching policy without any additional mechanism.

Next, we consider simple yet representative action smoothing methods that can enforce temporal coherence.

(ii) *Ensemble*: We generate multiple action trajectories from the policy with different initial noise and average them. We report results with ensemble sizes of 2 and 5.

(iii) *Smoothing*: We apply temporal smoothing with a Gaussian filter ($\sigma = 0.1$) to either the final action predictions or the intermediate action features before the self-attention layers.

Lastly, we include other guidance-based action generation methods for comparison.

(iv) *Classifier-Free Guidance (CFG) [14]*: We finetune the vanilla model with a null text (dropout rate 0.1) to obtain an unconditional model from the conditional one, and generate actions according to (6).

(v) *White Noise Guidance (WNG)*: As an alternative to construct an incoherent denoising vector field (v_θ^{IC}), we inject white noise ($\sigma = 1.0$) into the intermediate action features before the self-attention layers, disrupting temporal consistency across timesteps.

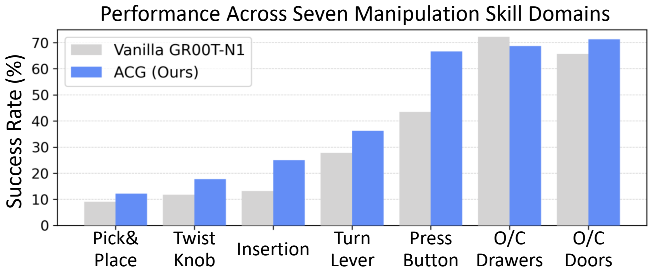

Fig. 5. Performance across the seven manipulation skill domains in RoboCasa [15]. ACG shows strong improvements on fine-grained manipulation tasks including insertion and button pressing.

B. Benchmark Results

As shown in Tab. I, ACG consistently outperforms Vanilla GR00T-N1 [2], a state-of-the-art VLA model, achieving gains of 6.7 pp on RoboCasa, 3.4 pp on DexMG, 30.8 pp on the Three Strawberries task, and 9.6 pp on average. These consistent improvements across diverse benchmarks demonstrate that ACG generalizes well in a plug-and-play manner without requiring additional training.

Next, we analyze other baseline approaches. Action smoothing methods yield modest improvements over the vanilla baseline, suggesting that action smoothness is indeed important for manipulation. However, because these methods directly smooth model features or output, they can blur fine-grained action details, leading to only marginal gains.

Guidance-based methods achieve greater improvements. Due to the iterative nature of flow matching models, guidance can steer the model more effectively toward the desired distribution without additional tuning. While CFG [14] strengthens text conditioning, it performs worse than other smoothing-based approaches on manipulation tasks, where coherent action generation is crucial.

WNG achieves the second-best performance after ACG. This supports our intuition that steering a flow matching model using a perturbed variant of the original denoising vector field can be effective. However, it often exhibits a trade-off between temporal coherence and accuracy: small

[1†] GR00T-N1 was finetuned on RoboCasa and DexMG with reported hyperparameters. RoboCasa performance matched the original report, while DexMG performance was slightly lower.

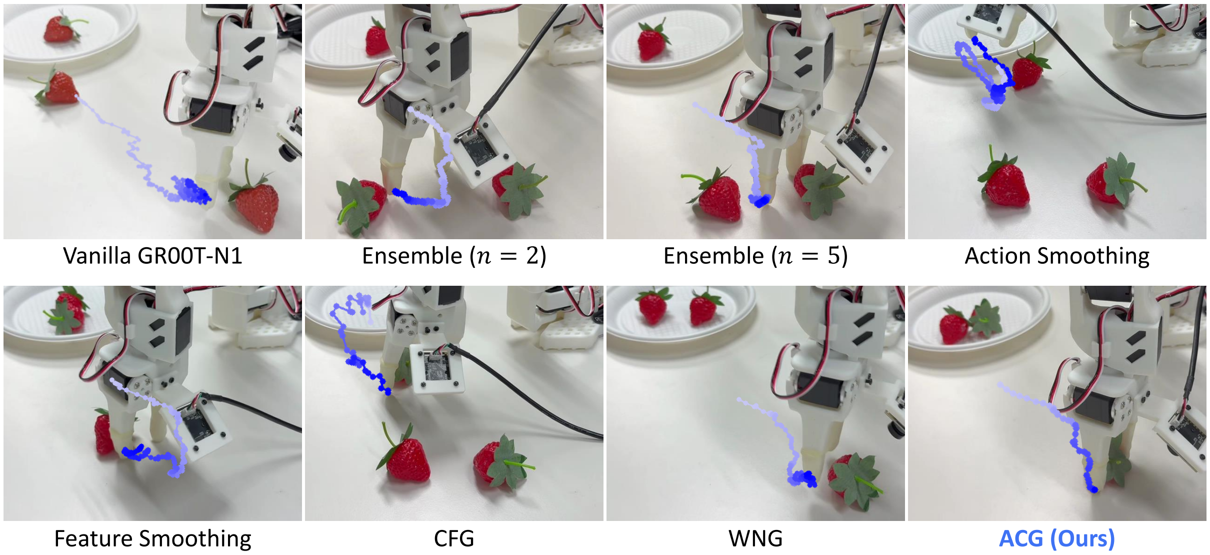

Fig. 6. Qualitative comparison on grasping strawberries. We visualize the trajectory of the endpoint of the gripper to show stability and accuracy of the actions. The color gradient indicates temporal progression, with the trajectory becoming darker over time. While the baseline methods often generate temporally incoherent or inaccurate actions, ACG generates coherent and successful trajectory.

TABLE II
QUANTITATIVE ACTION COHERENCE COMPARISON.

	ATV (rad/s, ↓)	JerkRMS ($\times 10^3$ rad/s^3, ↓)
Vanilla GR00T-N1	1.314 (±0.037)	1.353 (±0.115)
Action Smoothing Methods		
Ensemble ($n = 2$)	1.145 (±0.040)	1.340 (±0.143)
Ensemble ($n = 5$)	**0.984** (±0.048)	<u>1.172</u> (±0.128)
Action Smoothing	1.291 (±0.016)	1.277 (±0.145)
Feature Smoothing	1.287 (±0.072)	1.233 (±0.125)
Guidance-based Action Generation Methods		
CFG	1.332 (±0.047)	1.317 (±0.083)
WNG	1.274 (±0.061)	1.265 (±0.134)
Incoherent (v_θ^{IC})	4.509 (±0.061)	1.993 (±0.403)
ACG (Ours)	<u>1.130</u> (±0.139)	**1.156** (±0.148)

noise injection fails to sufficiently disrupt temporal structure, whereas large noise injection erodes pretrained knowledge. In contrast, ACG effectively disrupts temporal coherence without degrading task-relevant knowledge, achieving 4.3 pp improvement over WNG on RoboCasa and 8.8 pp improvement on Three Strawberries.

Task-wise performance analysis (Fig. 5) further highlights that ACG is especially effective on fine-grained manipulation skills, such as button pressing (+23.1 pp) and insertion (+11.8 pp). These results indicate that action coherence is particularly crucial for fine-grained manipulation tasks.

C. Action Coherence Analysis

We now examine whether ACG indeed improves the coherence of action sequences, quantitatively and qualitatively.

To quantitatively evaluate the action stability of the proposed method, we measure the Action Total Variation (ATV, rad/s) [12] and JerkRMS (rad/s^3) [13]. ATV quantifies the temporal coherence of the action sequences predicted by the flow matching policy. Jerk, defined as the third derivative of the motor angle $\mathbf{s}$, represents the rate of change in acceleration. JerkRMS measures the root mean square of this jerk to assess the smoothness of the resulting movements. Formally, the metrics are defined as follows:

$$\mathrm{ATV} = \frac{1}{M(T-1)} \sum_{t=1}^{T-1} \sum_{j=1}^{M} |a_{t+1}^{j} - a_{t}^{j}|, \tag{12}$$

$$\mathrm{JerkRMS} = \sqrt{\frac{1}{T-3} \sum_{t=1}^{T-3} \| \dddot{\mathbf{s}}_t \|_2^2}, \tag{13}$$

where a_t^j denotes the action of the j-th motor at the t-th timestep, and $\dddot{\mathbf{s}}_t$ denotes the jerk of the motors. In our experiments, we report the average ATV and JerkRMS across the six motors of SO-101. For a fair comparison, we compute the metrics only over the approach phase (*i.e.*, the first 64 timesteps) toward a single strawberry, since later trajectories diverge depending on task success or failure.

As shown in Tab. II and Fig. 6, the Vanilla GR00T-N1 [2] model exhibits severe jerking and jittering, often knocking the strawberry away. Ensemble methods significantly improve temporal coherence; however, as seen in our qualitative results, they often hesitate between strawberries because the averaging process blurs distinct action paths. Similarly, while action and feature smoothing methods reduce action variation, they frequently produce inaccurate actions since they modify the sequence without any task-specific prior.

For guidance-based action generation methods, CFG does not improve action coherence compared to Vanilla GR00T-N1. These results highlight the orthogonality between improving action coherence and strengthening the goal condition in manipulation tasks. Based on our benchmark results (Tab. I), we argue that action coherence is more crucial than goal conditioning for manipulation performance. Lastly, while WNG can generate coherent actions similar to smooth-

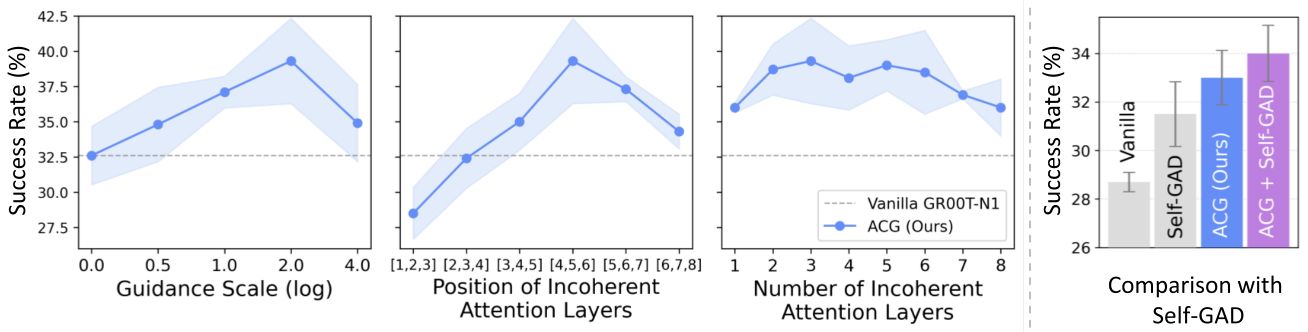

Fig. 7. (Left) Hyperparameter analysis of ACG. Appropriate guidance scales and the number and positions of incoherent attention layers substantially improve performance. (Right) Comparison with Self-GAD [20] on the RoboCasa [15] benchmark. While Self-GAD enhances action coherence between action chunks, ACG focuses on coherence within each action chunk. The results suggest that intra-chunk coherence plays a more crucial role than inter-chunk coherence, and that combining both methods can yield further improvements.

ing methods, it often sacrifices accuracy, sometimes pushing the strawberry away.

As expected, the incoherent variant (v_θ^{IC}) exhibits even worse action coherence than Vanilla GR00T-N1. By extrapolating this incoherent variant using (8), ACG achieves the most coherent actions, as indicated by JerkRMS, while maintaining an ATV score comparable to ensemble methods. Unlike ensemble methods, which often produce inaccurate action sequences, ACG generates accurate and temporally consistent actions (Fig. 6), leading to higher success rates. In summary, ACG produces highly coherent action sequences without sacrificing accuracy, thereby substantially improving manipulation performance.

D. Ablation Study

We conduct an ablation study to gain deeper insights into the effectiveness and robustness of ACG.

a) Hyperparameter Impact: We analyze three hyperparameters of ACG, the guidance scale (default: 3.0), the number of incoherent self-attention layers, and their positions (default: 3 layers at positions 4, 5, and 6). We vary each hyperparameter individually on the RoboCasa benchmark [15] while keeping the others fixed. As shown in Fig. 7, increasing the guidance scale improves performance, consistent with prior findings in the classifier-free guidance literature [14], [34]. However, excessively large scales degrade performance, likely due to increased deviation from the base pretrained distribution. Using 2–6 incoherent attention layers consistently improves performance, suggesting that ACG is robust to this hyperparameter. For their positions, while early layers occasionally reduce performance, the middle and later layers generally yield better results.

b) Comparison with Self-GAD [39]: Self-GAD is a concurrent work that improves cross-chunk coherence through test-time guidance. It is applicable only under the receding horizon setting, where the policy predicts 16 actions but executes only the first 8. While the receding horizon improves reactiveness, it effectively doubles the decision horizon, which amplifies compounding errors and lowers the baseline performance. Although their receding horizon setting lowers the baseline performance, we used the setup when comparing with Self-GAD for fair comparison.

TABLE III

Generalization of ACG Across VLA Models

	Three Strawberries	Tic-Tac-Toe
π_0 [3]	41.1% (±5.09%)	33.3% (±2.89%)
w/ ACG	**53.3%** (±3.33%)	**46.7%** (±2.89%)
SmolVLA [4]	16.7% (±3.33%)	23.3% (±2.89%)
w/ ACG	**22.2%** (±1.92%)	**30.0%** (±5.00%)

While both ACG and Self-GAD outperform the baseline, ACG achieves higher success rates. This indicates that, although mitigating inter-chunk errors and enhancing intra-chunk coherence both improve performance, the latter is more critical for manipulation tasks. Furthermore, combining ACG with Self-GAD achieves the best results, showing that our method is complementary to inter-chunk approaches and can be integrated for additional gains.

c) Generalization to Flow-based Action Generation Models: We also evaluate our approach on various VLA models. Specifically, we test it with standard flow-based VLA models, including π_0 [3] and SmolVLA [4]. Thanks to its simplicity, the proposed method can be readily applied to any model that incorporates self-attention layers over action tokens. As shown in Tab. III, ACG outperforms the vanilla model with a notable margin, demonstrating its generalization capability across the model variations.

VI. Conclusion

We introduced ACG, a test-time guidance method that steers flow policies toward generating coherent action trajectories by leveraging the negative predictions of an intentionally constructed incoherent variant of the base model. ACG achieves notable gains in success rates across both simulation benchmarks (RoboCasa and DexMimicGen) and real-world tasks with SO-101, and further demonstrates robustness when applied to different DiT-based flow models.

However, these gains come with computational costs. Naïvely generating the guidance vector requires a second forward pass through the incoherent vector field. Fortunately, this cost can be reduced below 2× by reusing the intermediate features. As shown in Section V-C, later attention layers contribute most to incoherence, enabling us to cache the outputs of earlier layers. In our base setting, we shared the first half of the layers, which reduces the computational overhead to about 1.5×. Still, it remains an open question whether perturbing only a small fraction of the latter layers suffices for deeper networks.

We hope that ACG inspires the research community to view test-time guidance not only as a tool for image and video generation, but also as a principle for generating coherent and reliable actions in robotics. Accordingly, exploring different guidance strategies to mitigate other drawbacks of flow matching could be a promising research direction.

References

[1] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *IJRR*, 2023.

[2] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, *et al.*, "Gr00t n1: An open foundation model for generalist humanoid robots," *arXiv preprint arXiv:2503.14734*, 2025.

[3] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, *et al.*, "π_0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.

[4] M. Shukor, D. Aubakirova, F. Capuano, P. Kooijmans, S. Palma, A. Zouitine, M. Aractingi, C. Pascal, M. Russi, A. Marafioti, *et al.*, "Smolvla: A vision-language-action model for affordable and efficient robotics," *arXiv preprint arXiv:2506.01844*, 2025.

[5] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu, "Rdt-1b: a diffusion foundation model for bimanual manipulation," in *ICLR*, 2025.

[6] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *CoRL*, 2021.

[7] S. Fang, H. Yu, Q. Fang, R. M. Aronson, and E. S. Short, "Demonstration sidetracks: Categorizing systematic non-optimality in human demonstrations," *arXiv preprint arXiv:2506.11262*, 2025.

[8] Y. Yuan, X. Li, Y. Heng, L. Zhang, and M. Wang, "Good better best: Self-motivated imitation learning for noisy demonstrations," *arXiv preprint arXiv:2310.15815*, 2023.

[9] V. Tangkaratt, N. Charoenphakdee, and M. Sugiyama, "Robust imitation learning from noisy demonstrations," in *AISTATS*, 2021.

[10] T. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," *RSS*, 2023.

[11] Y. Liu, J. I. Hamid, A. Xie, Y. Lee, M. Du, and C. Finn, "Bidirectional decoding: Improving action chunking via guided test-time sampling," in *ICLR*, 2025.

[12] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, 1992.

[13] T. Flash and N. Hogan, "The coordination of arm movements: an experimentally confirmed mathematical model," *Neuroscience*, 1985.

[14] J. Ho and T. Salimans, "Classifier-free diffusion guidance," *arXiv preprint arXiv:2207.12598*, 2022.

[15] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu, "Robocasa: Large-scale simulation of everyday tasks for generalist robots," in *RSS*, 2024.

[16] Z. Jiang, Y. Xie, K. Lin, Z. Xu, W. Wan, A. Mandlekar, L. J. Fan, and Y. Zhu, "Dexmimicgen: Automated data generation for bimanual dexterous manipulation via imitation learning," in *ICRA*, 2025.

[17] TheRobotStudio and H. Face, "Standard open so-100 & so-101 arms," "https://github.com/TheRobotStudio/SO-ARM100", 2024.

[18] T. Karras, M. Aittala, T. Kynkäänniemi, J. Lehtinen, T. Aila, and S. Laine, "Guiding a diffusion model with a bad version of itself," *NeurIPS*, 2024.

[19] J. Hyung, K. Kim, S. Hong, M.-J. Kim, and J. Choo, "Spatiotemporal skip guidance for enhanced video diffusion sampling," in *CVPR*, 2025.

[20] D. Ahn, H. Cho, J. Min, W. Jang, J. Kim, S. Kim, H. H. Park, K. H. Jin, and S. Kim, "Self-rectifying diffusion sampling with perturbed-attention guidance," in *ECCV*, 2024.

[21] S. Hong, G. Lee, W. Jang, and S. Kim, "Improving sample quality of diffusion models using self-attention guidance," in *ICCV*, 2023.

[22] S. Hong, "Smoothed energy guidance: Guiding diffusion models with reduced energy curvature of attention," *NeurIPS*, 2024.

[23] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *NeurIPS*, 1988.

[24] N. M. Shafiullah, Z. Cui, A. A. Altanzaya, and L. Pinto, "Behavior transformers: Cloning k modes with one stone," *NeurIPS*, 2022.

[25] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," in *CoRL*, 2022.

[26] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *ICRA*, 2018.

[27] P. Florence, L. Manuelli, and R. Tedrake, "Self-supervised correspondence in visuomotor policy learning," *RA-L*, 2019.

[28] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. Sanketi, Q. Vuong, *et al.*, "Openvla: An open-source vision-language-action model," in *CoRL*, 2025.

[29] D. Ghosh, H. R. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo, *et al.*, "Octo: An open-source generalist robot policy," in *RSS*, 2024.

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

[31] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in *CoRL*, 2023.

[32] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *NeurIPS*, 2021.

[33] M. Reuss, M. Li, X. Jia, and R. Lioutikov, "Goal-conditioned imitation learning using score-based diffusion policies," in *RSS*, 2023.

[34] K. Frans, S. Park, P. Abbeel, and S. Levine, "Diffusion guidance is a controllable policy improvement operator," *arXiv preprint arXiv:2505.23458*, 2025.

[35] T. Pearce, T. Rashid, A. Kanervisto, D. Bignell, M. Sun, R. Georgescu, S. V. Macua, S. Z. Tan, I. Momennejad, K. Hofmann, *et al.*, "Imitating human behaviour with diffusion models," in *ICLR*, 2023.

[36] J. Liu, H. Chen, P. An, Z. Liu, R. Zhang, C. Gu, X. Li, Z. Guo, S. Chen, M. Liu, *et al.*, "Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model," *arXiv preprint arXiv:2503.10631*, 2025.

[37] L. X. Shi, A. Sharma, T. Z. Zhao, and C. Finn, "Waypoint-based imitation learning for robotic manipulation," in *CoRL*, 2023.

[38] S. Belkhale, Y. Cui, and D. Sadigh, "Hydra: Hybrid robot actions for imitation learning," in *CoRL*, 2023.

[39] R. Malhotra, Y. Liu, and C. Finn, "Self-guided action diffusion," *arXiv preprint arXiv:2508.12189*, 2025.

[40] K. Black, M. Y. Galliker, and S. Levine, "Real-time execution of action chunking flow policies," *arXiv preprint arXiv:2506.07339*, 2025.

[41] D. Son and S. Park, "Lipo: A lightweight post-optimization framework for smoothing action chunks generated by learned policies," *arXiv preprint arXiv:2506.05165*, 2025.

[42] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *ICLR*, 2023.

[43] X. Liu, C. Gong, and Q. Liu, "Flow straight and fast: Learning to generate and transfer data with rectified flow," *arXiv preprint arXiv:2209.03003*, 2022.

[44] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *ICLR*, 2021.

[45] Q. Zheng, M. Le, N. Shaul, Y. Lipman, A. Grover, and R. T. Chen, "Guided flows for generative modeling and decision making," *arXiv preprint arXiv:2311.13443*, 2023.

[46] W. Peebles and S. Xie, "Scalable diffusion models with transformers," in *ICCV*, 2023.

[47] J. Jang, S. Ye, Z. Lin, J. Xiang, J. Bjorck, Y. Fang, F. Hu, S. Huang, K. Kundalia, Y.-C. Lin, *et al.*, "Dreamgen: Unlocking generalization in robot learning through video world models," *arXiv preprint arXiv:2505.12705*, 2025.