LOOK BEFORE YOU FLOW: INFERENCE-TIME RISK STEERING FOR FROZEN FLOW-MATCHING VLAS

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing vision-language-action (VLA) policies execute natural-language instructions in physical environments but lack mechanisms to avoid scene hazards during inference, including fragile objects, bystanders, and obstacles. Recent safety methods for VLAs either require retraining the policy, depend on known object geometry and dynamics models, or specify constraints in natural language without modifying the continuous action output. In this work, we introduce **MARS** (*Multi-risk Action Repulsion and Steering*), a training-free inference-time framework that steers frozen flow-matching VLAs away from scene risks by translating semantic risk knowledge from a vision-language model into action-space corrections. MARS applies *Grounded Risk Verbalization* (GRV) to identify scene risks via a frozen VLM and express each as a counterfactual action prompt in the VLA's native instruction format, and *Trajectory-Anchored Flow Repulsion* (TAFR) to steer the flow-matching ODE by accumulating clipped velocity residuals in a correction state decoupled from the nominal trajectory, preventing guidance perturbations from compounding across integration steps. These designs enable MARS to bridge semantic risk recognition and action-space intervention without retraining, geometric models, or learned critics. In comprehensive evaluations across three safety-critical simulation benchmarks and real-world manipulation tasks on two robot platforms, MARS improves safety across four VLA backbones ranging from 0.45B to 3B parameters. It reduces safety cost by approximately 40%, improves task success rate by up to 9.9%, and increases real-world safe success rate by up to 20% all without modifying any model parameters. Anonymous project page: `https://mars-iclr-2026.github.io/`

1 INTRODUCTION

Recent advances in robot learning have produced vision-language-action (VLA) policies that map visual observations and natural-language instructions to continuous action sequences, enabling generalist robots to operate across tasks and environments. State-of-the-art systems pair pretrained visual-linguistic representations with flow-matching decoders: $\pi_{0.5}$ (Physical Intelligence et al., 2025) combines high-level semantic prediction with a flow-matching controller, while GR00T N1 couples a vision-language model (VLM) reasoning module with a flow-matching Diffusion Transformer to produce action chunks across robot embodiments (NVIDIA et al., 2025; Lipman et al., 2023). As these policies extend to more varied physical environments, their actions must account for both the requested task and the surrounding scene.

Completing the task does not necessarily mean the robot acts safely: it may follow an instruction while colliding with nearby geometry, disturbing a protected object, or endangering nearby agents. We use *safety* to denote task execution that avoids such physical violations involving the robot, protected objects, and the surrounding scene. Recent benchmarks evaluate these violations separately from nominal task success (Cui et al., 2026). The objects requiring protection can change even when the task stays the same: a wine bottle may be present on a countertop during one execution and absent during the next. Safety therefore requires adapting action generation to the risks in the current scene, and retraining the policy whenever these risks change is impractical.

Recent work has begun to address VLA safety, but each direction leaves a gap at the intersection of these needs. **(1) Training dependence on discrete actions.** SafeVLA formulates safety as a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

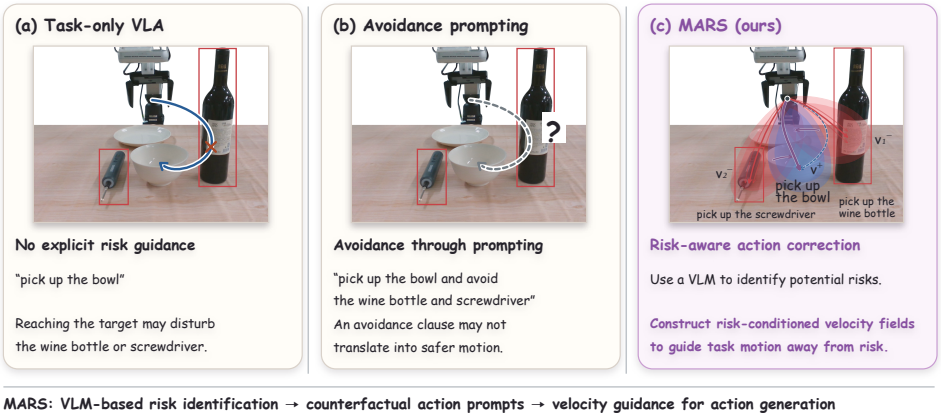


Figure 1: Motivation for MARS. (a) A task-only policy disturbs a non-target object. (b) Appending avoidance clauses pushes the prompt outside the VLA's training distribution, eliciting unexpected actions. (c) MARS uses counterfactual action prompts and velocity-field residuals to steer a frozen VLA at inference time.

constrained Markov decision process and applies safe reinforcement learning, but its implementation fine-tunes a causal transformer over a discrete action space (Zhang et al., 2025a). Extending this pipeline to the continuous action chunks of flow-matching VLAs would require redesigning cost modeling, critic learning, and policy optimization. **(2) Geometric-only constraints.** VLSA introduces AEGIS, a safety layer that converts visual and depth observations into geometric obstacle models and filters nominal actions through control barrier functions (Hu et al., 2025). AEGIS provides geometric constraints but depends on known object geometry and robot dynamics models, and cannot express semantic risks: a wine bottle is not merely an obstacle to circumnavigate but a fragile object that should not be disturbed. **(3) No action-space mechanism.** VLM-based methods can identify relevant scene entities and represent safety constraints as natural-language specifications (Santos et al., 2024; Feng et al., 2025), but they operate at the level of constraint description. They define no mechanism for modifying the continuous action output of a flow-matching policy. No existing method bridges semantic risk recognition to action-space correction for frozen flow-matching VLAs without additional training.

At the same time, vision-language models have achieved strong open-vocabulary recognition, spatial grounding, and instruction-following capabilities without task-specific fine-tuning (Bai et al., 2025a;b; Liu et al., 2023b), and recent benchmarks have begun evaluating whether these capabilities extend to safety hazard assessment (Yu et al., 2026). A VLM can inspect a scene and name whatever risk is present, from a wine bottle to a sharp tool that was not there an hour ago, without retraining. This is exactly the scene-level adaptability that approaches (1)–(3) lack, but leveraging it requires a mechanism that translates VLM risk recognition into corrections of the VLA's continuous action output.

Figure 1 illustrates the challenge. A flow-matching VLA executing "pick up the bowl" may sweep through a nearby wine bottle (panel a). Appending avoidance clauses to the instruction pushes compound commands outside the VLA's training distribution and degrades performance (panel b), an effect that worsens with multiple simultaneous risks. Neither approach connects VLM risk recognition to action-space corrections.

To address limitations (1)–(3), we propose **Multi-risk Action Repulsion and Steering (MARS)**, an inference-time framework that connects VLM risk recognition to the continuous action space of a frozen flow-matching VLA (panel c of Figure 1). To bridge the gap between semantic risk recognition and action-space intervention (limitation 3), MARS applies **Grounded Risk Verbalization (GRV)**: a VLM inspects the scene, identifies risks, and produces structured records each containing a counterfactual action prompt expressing the undesired interaction (e.g., "pick up the bottle" rather than "avoid the bottle"). Because these are ordinary action commands, they remain within the VLA's training distribution. From shared initial noise, the frozen VLA evaluates its velocity field under both the task instruction and each counterfactual risk prompt at every Ordinary Differential

Equation (ODE) step. To provide action-space corrections without retraining or geometric models (limitations 1 and 2), **Trajectory-Anchored Flow Repulsion (TAFR)** guides the flow-matching ODE integration: at each step, clipped velocity residuals between the nominal and counterfactual prompts accumulate in a correction state decoupled from the model's evaluation trajectory, preventing guidance perturbations from compounding across integration steps. Because both the VLM and VLA remain frozen, MARS requires no additional training data, reward modeling, or policy optimization.

Contributions. Our contributions are as follows:

- We propose **MARS**, a training-free inference-time framework that improves the safety of frozen flow-matching VLA policies without updating VLM or VLA parameters. MARS is compatible with any flow-matching-based VLA.
- We introduce **GRV**, which uses a VLM to identify scene risks and produce counterfactual action prompts, and **TAFR**, which guides the flow-matching ODE via trajectory-anchored velocity residuals to produce action-space corrections. Together, they translate semantic risk knowledge into action-space interventions.
- We evaluate MARS on three simulation benchmarks, **VLA-Arena**, **LIBERO-Safety**, and **SafeLIBERO**, against four VLA baselines (0.45B–3B parameters) and validate on two real-world platforms (Dobot CR-10 and Galaxea A1X). MARS reduces safety cost by approximately 40% while improving success rate by up to 9.9 percentage points across benchmarks. In real-world experiments, MARS raises average safe success rate from 55% to 75% (Dobot CR-10) and from 56.7% to 73.3% (Galaxea A1X) while preserving success rate.

2 Related Work

Vision-language-action policies. Foundation-scale VLAs connect visual observations, language instructions, and robot actions via autoregressive action tokenization (Brohan et al., 2023; Kim et al., 2024), diffusion-based denoising (Chi et al., 2023; Ho et al., 2020), and flow matching, which learns a velocity field whose ODE transports noise to action samples (Lipman et al., 2023; Liu et al., 2022b). Recent flow-matching VLAs, including π_0 (Black et al., 2024), $\pi_{0.5}$ (Physical Intelligence et al., 2025), GR00T N1 (NVIDIA et al., 2025), and X-VLA (Zheng et al., 2025), generate continuous action chunks over multiple integration steps, exposing intermediate velocity evaluations that external modules can intercept.

Inference-time guidance for flow-matching policies. Classifier-free approaches mix conditioned and unconditioned VLA outputs to improve spatial grounding (Fang et al., 2026b; Zhou et al., 2026). Critic-based approaches inject learned value-function gradients into the velocity field to steer action selection (Zhao et al., 2026; Yang et al., 2026). Other methods operate on intermediate flow states by projecting guidance into trust regions, shaping flows via energy functions, or reversing integration to recover in-distribution modes (Fang et al., 2026a; Song et al., 2026; Tang et al., 2026). Across these lines, none translates open-vocabulary semantic constraints into velocity-field corrections without auxiliary training.

Safety for embodied agents. SafeVLA applies constrained reinforcement learning to align VLA behavior (Zhang et al., 2025a), but operates on discrete action tokens and does not extend to continuous flow-matching action chunks. Geometric methods filter or constrain flow-matching outputs using control-barrier functions (Hu et al., 2025), trajectory-level barrier constraints (English et al., 2026; Sinaei et al., 2026), configuration-space guards (Beaudin et al., 2026), or internal object probes (Zhang et al., 2026). These methods enforce spatial safety but require known object geometry or robot dynamics. Language-based methods ground constraints in 3D maps or reachability representations (Santos et al., 2024; Feng et al., 2025) but do not modify continuous flow-matching outputs (Appendix A).

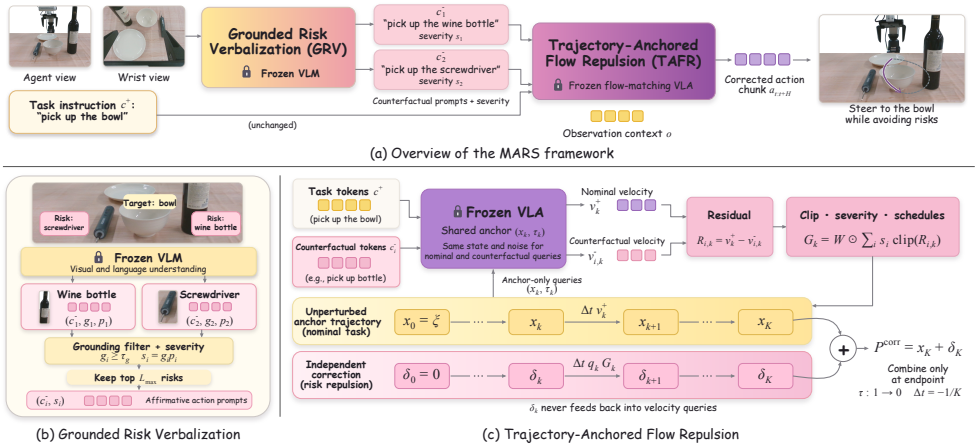


Figure 2: Overview of MARS. **GRV** (§3.1) uses a frozen VLM to produce counterfactual action prompts from scene risks. The frozen VLA evaluates nominal and counterfactual prompts from shared noise $\boldsymbol{\xi}$. **TAFR** (§3.2) accumulates trajectory-anchored velocity residuals to produce a corrected action chunk.

3 MARS: Multi-risk Action Repulsion and Steering

Figure 2 gives an overview. A frozen VLM identifies scene risks and GRV converts each into a counterfactual action prompt. The nominal instruction and all counterfactual prompts enter the frozen VLA as a single batch from shared initial noise. Different language conditions yield distinct velocity fields at the action head. TAFR compares these fields at each ODE step and accumulates clipped residuals into a correction state decoupled from the nominal trajectory. We formalize the setup below, then detail **GRV** (§3.1) and **TAFR** (§3.2).

Let $o_t = [I_t^1, \ldots, I_t^n]$ denote multi-view observations at replanning step t and c_t^+ the natural-language task instruction. A flow-matching VLA F_θ maps initial noise $\boldsymbol{\xi}$ to an action chunk $P = (P_h)_{h=1}^H$, where each $P_h \in \mathbb{R}^d$ encodes a single-step action and H is the chunk horizon. This mapping is deterministic for a given condition: $P = F_\theta(o_t, c_t^+; \boldsymbol{\xi})$. Internally, F_θ integrates a conditional velocity field $v_\theta(\mathbf{x}, \tau; o_t, c)$, where $\mathbf{x} \in \mathbb{R}^{H \times d}$ is the flow state in the backbone's normalized action coordinates. We adopt the convention $\tau = 1$ for noise and $\tau = 0$ for actions, with $K \geq 1$ Euler steps at $\tau_k = 1 - k/K$ and step size $\Delta t = -1/K$. MARS produces a corrected chunk $\{P_h^{\mathrm{corr}}\}_{h=1}^H$ that follows c_t^+ by intervening on v_θ rather than modifying θ.

3.1 Grounded Risk Verbalization

GRV uses a frozen VLM to inspect the current scene and convert each identified risk into a counterfactual action prompt for downstream velocity-field steering. GRV keeps the task instruction unmodified and delegates scene interpretation to the VLM. It operates as two coupled components: a *VLM interface* that governs prompt construction and risk-record generation, and an *invocation scheduler* that determines when the VLM is called based on the current risk state.

VLM interface. GRV instantiates a frozen VLM $\mathcal{V}_\phi$ that receives agent-view and wrist-view images at 256×256 resolution and produces structured risk records via JSON schema enforcement. Given observation o_t and the task instruction c_t^+, the VLM identifies *candidate risk objects*, i.e. non-target entities whose disturbance would violate the task specification. For each candidate i, it returns a record

$$\mathcal{V}_\phi(o_t, c_t^+) \ni \left(e_{t,i},\ \beta_{t,i},\ c_{t,i}^-,\ g_{t,i},\ p_{t,i}\right), \tag{1}$$

where $e_{t,i}$ names the entity, $\beta_{t,i}$ is its image region, $c_{t,i}^-$ is a counterfactual action prompt, and $(g_{t,i}, p_{t,i}) \in [0,1]^2$ score grounding confidence and proximity. A candidate is retained when $g_{t,i} \geq \tau_g$ and carries severity $s_{t,i}$:

$$\mathcal{I}_t = \{i : g_{t,i} \geq \tau_g\}, \qquad s_{t,i} = g_{t,i}\, p_{t,i}. \tag{2}$$

When more than $L_{\max}$ candidates pass both thresholds, GRV keeps those with the largest severity, so the retained count $L = |\mathcal{I}_t|$ never exceeds $L_{\max}$. Each counterfactual prompt $c^-_{t,i}$ frames the undesired interaction as an affirmative action command (e.g., "pick up the wine bottle"), keeping it within the VLA's training distribution rather than introducing negation or compound avoidance clauses. TAFR does not execute these prompts. It uses them only as reference directions for suppressing the corresponding behavior. The retained records $\{(c^-_{t,i},\ s_{t,i})\}_{i\in\mathcal{I}_t}$ and their severity weights constitute the interface to TAFR. When $\mathcal{I}_t = \emptyset$, MARS returns the unmodified VLA output. The complete prompt template and parsing details are provided in Appendix B.

Invocation scheduler. The VLM is not invoked at every replanning step. Under normal conditions, GRV calls the VLM once every N replanning cycles (default $N{=}4$) and reuses the most recent risk records for the intervening cycles. To hide VLM latency, GRV pipelines the VLM forward pass with TAFR's VLA forward passes: at each replanning step, GRV launches the VLM assessment on the latest observation concurrently with TAFR's nominal and counterfactual velocity-field evaluations. TAFR uses the most recent cached risk records for the current step, while GRV's updated records become available for subsequent steps. The critical-path latency is therefore dominated by the slower of the two branches rather than their sum, at the cost of risk records lagging by at most one invocation cycle. When GRV detects potentially risky moving objects, it reduces the invocation interval to $N{=}1$, querying the VLM at every replanning cycle. This trades additional VLM inference cost for reduced response latency in dynamic scenes. Appendix C defines the request, motion, and failure rules.

3.2 Trajectory-Anchored Flow Repulsion

Given the retained risk records from GRV, TAFR intervenes on the flow-matching ODE (Zheng et al., 2023) to steer the action chunk away from each identified risk. The nominal task instruction c^+_t and all counterfactual prompts $\{c^-_{t,i}\}_{i\in\mathcal{I}_t}$ are evaluated by the frozen VLA in a single batch from shared initial noise $\boldsymbol{\xi}$, so the velocity fields at the action head differ only in their language conditioning. At each ODE integration step, TAFR computes the residual between the nominal and each counterfactual velocity field. This residual defines a local repulsion direction away from the undesired interaction (Ho & Salimans, 2022; Schramowski et al., 2023; Koulischer et al., 2025). Clipped, severity-weighted residuals accumulate in a separate correction state, while an anchor state carries the unperturbed nominal trajectory forward. This decoupling bounds the total intervention while preserving the original task behavior.

Velocity residuals. At each ODE step k, TAFR evaluates the frozen VLA at the current anchor state $\mathbf{x}_k$ and flow time τ_k under both the nominal instruction c^+_t and each counterfactual instruction $c^-_{t,i}$:

$$\mathbf{v}^+_k = v_\theta(\mathbf{x}_k, \tau_k; o_t, c^+_t), \qquad \mathbf{v}^-_{i,k} = v_\theta(\mathbf{x}_k, \tau_k; o_t, c^-_{t,i}), \qquad \mathbf{R}_{i,k} = \mathbf{v}^+_k - \mathbf{v}^-_{i,k}. \tag{3}$$

All $L{+}1$ conditions share the observation and anchor state and are evaluated as a single batch, with per-risk residuals retained until severity-weighted aggregation (Liu et al., 2022a).

Guidance scheduling. TAFR controls the correction magnitude along three axes: per-dimension, along the action horizon, and across ODE integration steps. Raw residuals can be arbitrarily large where the two conditions disagree sharply, so TAFR first clips each component to a per-dimension cap $b_a \geq 0$ in normalized velocity units:

$$R^{\text{clip}}_{i,k,h,a} = \max\{-b_a, \min\{b_a, R_{i,k,h,a}\}\}. \tag{4}$$

A power-ramp weight matrix $\mathbf{W} \in \mathbb{R}^{H\times d}$ then modulates the clipped residuals along the action horizon:

$$W_{h,a} = w_a \cdot \left(\frac{h}{H}\right)^{\gamma_{\text{eff}}}, \qquad \gamma_{\text{eff}} = \gamma\,(1-r), \qquad r = 1 - \prod_{i\in\mathcal{I}_t}(1-s_i), \tag{5}$$

where w_a is the per-dimension maximum guidance weight, γ is the base exponent, $s_i \equiv s_{t,i}$ is the per-risk severity from equation 2 (records stay fixed within a replanning cycle, so we drop t), and

r is the noisy-OR aggregate risk. Increasing risk decreases γ_{eff} and flattens the ramp, allowing an urgent risk to affect earlier actions. Along the ODE integration axis, a temporal weight

$$q_k = \left(\frac{k+1}{K}\right)^{\alpha} \tag{6}$$

ramps guidance strength over the K Euler steps, allocating more of the intervention budget to lower-noise evaluations where the velocity field is closer to the action endpoint. Together with the clipping caps and the limit $L_{\max}$ on retained risks, these schedules bound the endpoint intervention in the backbone's normalized action coordinates (Appendix I). Their individual contributions are isolated in the scheduling ablations.

Trajectory-anchored integration. A direct guided sampler would evaluate the next velocity field at the corrected state. The correction would then shift all subsequent velocity evaluations, causing the generated action to drift away from the original task trajectory. TAFR instead maintains two decoupled states: an *anchor* $\mathbf{x}_k$ that follows the unperturbed nominal trajectory, and a *correction* $\boldsymbol{\delta}_k$ that accumulates the intervention independently. Given shared initial noise $\boldsymbol{\xi}$ and K Euler steps with step size $\Delta t = -1/K$:

$$\mathbf{x}_0 = \boldsymbol{\xi}, \qquad \boldsymbol{\delta}_0 = \mathbf{0}. \tag{7}$$

At each step $k = 0, \ldots, K-1$, the weighted guidance signal is

$$\mathbf{G}_k = \mathbf{W} \odot \sum_{i \in \mathcal{I}_t} s_i \, \mathbf{R}_{i,k}^{\text{clip}}, \tag{8}$$

and the two states update independently:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta t \; \mathbf{v}_k^{+}, \tag{9}$$

$$\boldsymbol{\delta}_{k+1} = \boldsymbol{\delta}_k + \Delta t \; q_k \; \mathbf{G}_k. \tag{10}$$

All velocities $\mathbf{v}_k^{+}$ and $\mathbf{v}_{i,k}^{-}$ are evaluated at $\mathbf{x}_k$, not at $\mathbf{x}_k + \boldsymbol{\delta}_k$. The nominal evaluation path is therefore unchanged, and the residual field does not depend on previous corrections. When $\mathcal{I}_t = \emptyset$, all severities are zero, $\boldsymbol{\delta}_K = \mathbf{0}$, and the output reduces exactly to the unguided trajectory. The corrected chunk $P^{\text{corr}} = \mathbf{x}_K + \boldsymbol{\delta}_K$ is decoded with the backbone's usual action postprocessor, and its first $1 \le H_{\text{exec}} \le H$ actions are executed before replanning. Appendix J gives the corrected-state variant, whose feedback loop removes the fixed-field property and invalidates the bound in Appendix I. The two variants are compared in the anchoring ablation in §4.3.

4 Experiments

We evaluate MARS on complementary simulation and real-world benchmarks that test task completion and safety constraint satisfaction, comparing against four VLA baselines that span different architectural designs.

Benchmarks. For simulation, we evaluate on three safety-critical benchmarks: VLA-Arena (Zhang et al., 2025b) (five safety suites, three difficulty levels), LIBERO-Safety (Cui et al., 2026) (four safety task suites, three difficulty levels), and SafeLIBERO (Hu et al., 2025) (obstacle-augmented LIBERO tasks at two proximity levels). For real-world evaluation, we design four table-top task families on two robot platforms; details are in Section 4.2. Full benchmark descriptions, metrics, and evaluation protocols are provided in Appendix D.

Baselines. We compare against four VLA models: $\pi_{0.5}$ (Physical Intelligence et al., 2025), StarVLA-PI and StarVLA-GR00T (StarVLA Community, 2026), and SmolVLA (Shukor et al., 2025), using officially released checkpoints with recommended fine-tuning. MARS wraps each backbone without modifying weights or architecture. Full descriptions are in Appendix D.

Metrics. We report four metrics throughout: *success rate* (SR; goal reached regardless of safety violations), *safe success rate* (SSR; goal reached without any violation), *cumulative cost* (CC; benchmark-native safety cost), and *collision rate* (CR; fraction of episodes with any obstacle contact). Each benchmark uses a subset of these metrics; see Appendix D for benchmark-specific details.

Table 1: Aggregate simulation results. VLA-Arena: SR and CC over five suites and three levels. LIBERO-Safety: SSR over four suites and three levels. SafeLIBERO: SR/SSR/CR over all 32 scenarios (1,600 episodes). †MARS as plug-in wrapper.

	VLA-Arena		LIBERO-Safety	SafeLIBERO		
Method	SR ↑	CC ↓	SSR ↑	SR ↑	SSR ↑	CR ↓
$\pi_{0.5}$	<u>39.3</u>	12.6	<u>56.4</u>	<u>59.6</u>	<u>18.7</u>	79.4
$\pi_{0.5}$ + MARS†	**49.2**	**7.5**	**60.8**	**61.4**	**21.3**	**47.6**
StarVLA-PI	34.8	16.4	38.8	28.8	8.8	83.3
StarVLA-PI + MARS†	36.1	12.7	40.2	28.6	8.6	83.7
StarVLA-GR00T	25.0	16.9	36.1	24.4	8.6	82.1
StarVLA-GR00T + MARS†	26.9	17.3	37.0	24.6	8.8	82.4
SmolVLA	18.0	<u>8.4</u>	29.2	7.1	2.1	<u>53.9</u>
SmolVLA + MARS†	21.7	8.6	28.8	6.6	2.1	54.3

4.1 Simulation Results

Table 1 summarizes aggregate performance across the three safety benchmarks. On $\pi_{0.5}$, MARS improves both task success and safety across all three benchmarks (SR $+9.9$ pp on VLA-Arena, SSR $+4.4$ pp on LIBERO-Safety, SR $+1.8$ pp on SafeLIBERO), while safety cost drops by roughly 40% (CC on VLA-Arena, CR on SafeLIBERO). On medium-strength backbones, gains are smaller and task success is mixed.

Table 2 reports the VLA-Arena SR broken down by difficulty level. At L0, MARS improves SR for every backbone, with the largest gains on SmolVLA (+16.4 pp) and $\pi_{0.5}$ (+10.8 pp). At L1/L2, only $\pi_{0.5}$ sustains gains (+12.0, +6.8 pp). Weaker backbones degrade toward zero SR, and MARS occasionally causes slight declines, as observed for StarVLA-GR00T at L2 ($5.7\% \rightarrow 4.0\%$) and for SmolVLA on SafeLIBERO, where both SR and SSR decrease. We analyze the mechanism behind this backbone dependence in §4.3. Per-suite breakdowns are in Tables 4–6 (Appendix).

Table 2: VLA-Arena SR (%) by difficulty level, averaged over five safety suites.

Method	**L0**	**L1**	**L2**
$\pi_{0.5}$	56.0	36.0	26.0
$\pi_{0.5}$ + MARS†	66.8	48.0	32.8
StarVLA-PI	67.6	20.0	10.4
StarVLA-PI + MARS†	79.2	19.6	9.6
StarVLA-GR00T	56.0	13.2	5.7
StarVLA-GR00T + MARS†	59.2	17.6	4.0
SmolVLA	38.8	7.2	1.6
SmolVLA + MARS†	55.2	9.6	0.4

4.2 Real-World Experiments

We evaluate on two robot platforms, Dobot CR-10 and Galaxea A1X, across four task families with 20 trials conducted for each task and method. Full task descriptions are provided in Appendix D. As shown in Figure 3, MARS raises average SSR from 55% to 75% (+20 pp) on Dobot CR-10 and from 56.67% to 73.33% (+16.7 pp) on Galaxea A1X, with gains driven by obstacle and hazard avoidance. Figure 4 shows a representative rollout, and complete rollouts are in Figure 8 (Appendix G).

4.3 Ablation and Component Analysis

Trajectory anchoring. Figure 5b compares four guidance strategies on $\pi_{0.5}$. Prompt injection reduces SR to $\sim 40\%$ of the baseline on all benchmarks, confirming that avoidance clauses push compound instructions far out of distribution. Corrected-state steering (Appendix J) improves over the baseline, but its feedback loop limits further gains. TAFR achieves the best results ($+9.9$ pp SR on VLA-Arena, $+5.4$ pp over corrected-state), confirming that decoupling corrections from the evaluation trajectory is critical. Additional ablations are in Appendix H.

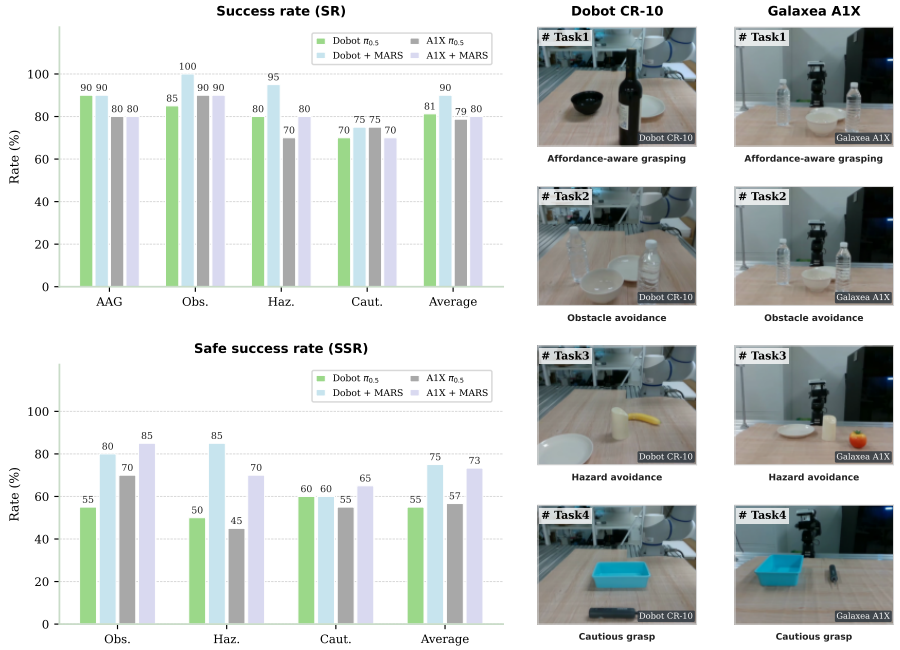


Figure 3: Real-world results and setups. *Left*: SR and SSR (%) on four tasks across Dobot CR-10 and Galaxea A1X. *Right*: workspace configurations for each task.

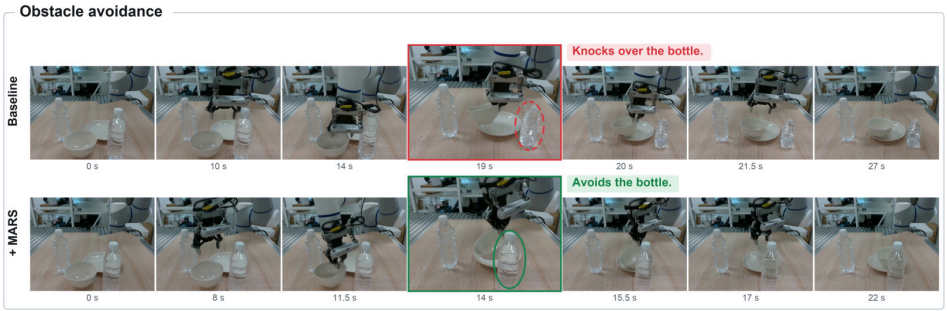
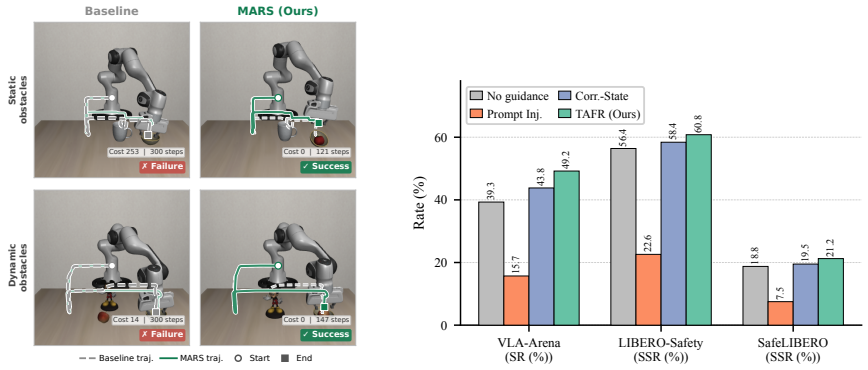


Figure 4: Obstacle-avoidance rollout (Dobot CR-10). *Top*: baseline topples the bottles. *Bottom*: MARS navigates around them from the same initial state.

Sensitivity to recognition and repulsion quality. MARS depends on two sequential capabilities: the VLM must correctly identify risks (*risk detection*), and the VLA must ground the counterfactual prompt into a velocity field distinguishable from the nominal one (*repulsion completion*). Formal definitions of both metrics are in Appendix E. Figure 6(a) visualizes how these factors co-vary across backbone$\times$VLM configurations on VLA-Arena. Repulsion completion rate is the dominant factor: $\pi_{0.5}$ (73%) achieves 50.5% task SR with the best VLM, while SmolVLA (18%) reaches only 22.0%, a 28.5 pp gap driven primarily by VLA capability. When repulsion completion is low, the backbone produces nearly identical velocity fields for nominal and counterfactual prompts, so the residuals that TAFR accumulates carry little directional signal. In this regime, clipping and scheduling noise can dominate the correction, steering the action chunk in arbitrary directions rather than away from the risk. This explains the slight SR declines observed on weaker backbones in Table 2 and Table 1: MARS is not merely ineffective but can be counterproductive when the backbone lacks sufficient language-conditioned action discrimination. On the recognition side, all three evaluated VLMs (Qwen2.5-VL-7B-Instruct (Bai et al., 2025a), Qwen3-VL-8B-Instruct (Bai et al., 2025b), and Qwen3.5-9B (Qwen Team, 2026)) achieve $\geq$85% category accuracy with preserved spatial ordering. Upgrading the VLM improves SR by 5.5 pp on $\pi_{0.5}$ but only 1.5 pp on SmolVLA, confirming an



(a) Qualitative rollouts on VLA-Arena. (b) Trajectory anchoring ablation on $\pi_{0.5}$.

Figure 5: Case study and trajectory anchoring analysis.

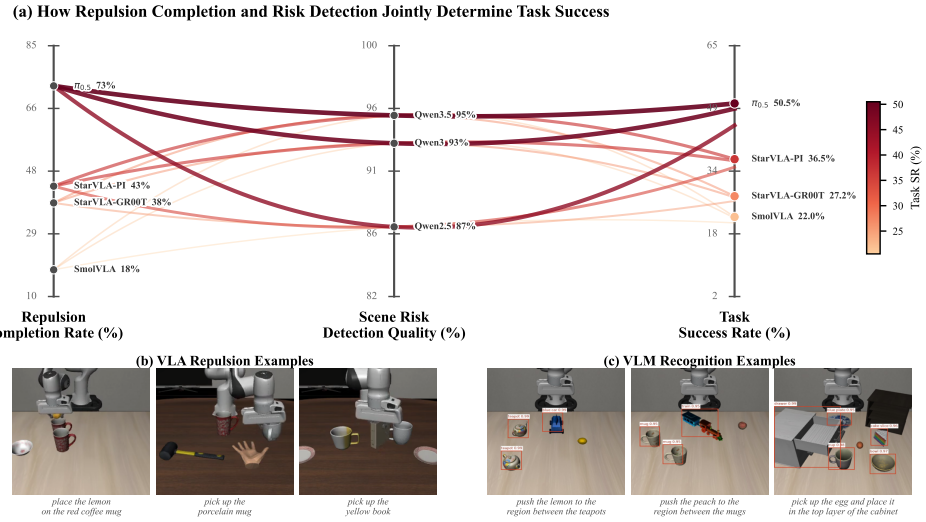


Figure 6: Component-level analysis of MARS. **(a)** Parallel coordinates of repulsion completion rate (VLA), risk detection quality (VLM), and task SR across backbone×VLM configurations. **(b)** VLA repulsion examples confirming the counterfactual prompt is within the backbone's executable distribution. **(c)** VLM recognition examples with bounding boxes and confidence scores.

asymmetric bottleneck: a stronger VLM cannot compensate for weak backbone repulsion, whereas a capable backbone benefits measurably from better risk recognition.

5 Conclusion

In this paper, we introduce MARS, a training-free inference-time framework that steers frozen flow-matching VLA policies away from scene risks. GRV uses a frozen VLM to identify protected entities and express undesired interactions as affirmative counterfactual commands. TAFR evaluates these alongside the nominal instruction at a shared anchor state, accumulating clipped velocity residuals in a correction decoupled from the model's evaluation trajectory. The nominal sampling path remains unchanged, and componentwise clipping gives an explicit bound on the endpoint intervention. Across three simulation benchmarks and real-world experiments on two robot platforms, MARS reduces safety cost by approximately 40% while improving task success rate, and raises real-world safe success rate by up to 20 percentage points while preserving success rate, without modifying any model parameters.

REPRODUCIBILITY STATEMENT

All hyperparameters, prompt templates, parsing contracts, and algorithmic details needed to reimplement MARS are specified in Appendices B–I. The VLMs and all VLA backbones use publicly available checkpoints. Code and evaluation scripts will be released upon acceptance.

AI USE STATEMENT

In this work, we used generative AI tools for language editing (improving readability and grammar) and LaTeX/template maintenance. We have not used generative AI tools to generate or analyze data, formulate hypotheses, design experiments, implement methods, interpret results, or draft scientific content. The authors reviewed all AI-assisted edits and take responsibility for the final content of this work.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 22–31, 2017. URL https://proceedings.mlr.press/v70/achiam17a.html.

Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022. URL https://arxiv.org/abs/2209.15571.

Aaron D. Ames, Xiangru Xu, Jessy W. Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8): 3861–3876, 2017. doi: 10.1109/TAC.2016.2638961.

Shuai Bai et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025a. URL https://arxiv.org/abs/2502.13923.

Shuai Bai et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025b. URL https://arxiv.org/abs/2511.21631.

Alex Beaudin, Hanna Krasowski, Kartik Nagpal, Sanjit A. Seshia, Murat Arcak, and Negar Mehr. Any-body guard: Universal safeguarding for manipulation policies via action masking. *arXiv preprint arXiv:2606.22278*, 2026.

Kevin Black et al. π_0: A Vision-Language-Action Flow Model for General Robot Control. *arXiv preprint arXiv:2410.24164*, 2024.

Anthony Brohan et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

Rongxu Cui, Zongzheng Zhang, Jingrui Pang, Haohan Chi, Jinbang Guo, Saining Zhang, Shaoxuan Xie, Xin Jin, Yao Mu, Jiaolong Yang, Guocai Yao, Xianyuan Zhan, Ya-Qin Zhang, and Hao Zhao. Libero-safety: A comprehensive benchmark for physical and semantic safety in vision-language-action models. *arXiv preprint arXiv:2606.23686*, 2026.

William English, Hao Zheng, and Rickard Ewetz. Neuro-symbolic safety guidance for vision-language-action models via constrained flow matching. *arXiv preprint arXiv:2607.01378*, 2026.

Kai Fang, Hailong Pei, and Xuemin Chi. Smoother action chunking flow policy via prior-corrected orthogonal trust-region guidance. *arXiv preprint arXiv:2605.24433*, 2026a.

Yu Fang, Yuchun Feng, Dong Jing, Jiaqi Liu, Yue Yang, Zhenyu Wei, Daniel Szafir, and Mingyu Ding. When vision overrides language: Evaluating and mitigating counterfactual failures in vlas. *arXiv preprint arXiv:2602.17659*, 2026b.

Zeyuan Feng, Haimingyue Zhang, and Somil Bansal. From words to safety: Language-conditioned safety filtering for robot navigation. *arXiv preprint arXiv:2511.05889*, 2025.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Songqiao Hu, Zeyi Liu, Shuang Liu, Jun Cen, Zihan Meng, Shihefeng Wang, Xiang Li, and Xiao He. Vlsa: Vision-language-action models with plug-and-play safety constraint layer. *arXiv preprint arXiv:2512.11891*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Felix Koulischer, Johannes Deleu, Gabriel Raya, Thomas Demeester, and Luca Ambrogioni. Dynamic negative guidance of diffusion models. In *International Conference on Learning Representations*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, 2023a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, volume 36, 2023b. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6dcf277ea32ce3288914faf369fe6de0-Abstract-Conference.html.

Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B. Tenenbaum. Compositional visual generation with composable diffusion models. In *European Conference on Computer Vision*. Springer, 2022a.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022b. URL https://arxiv.org/abs/2209.03003.

NVIDIA. GR00T N1.5. https://research.nvidia.com/labs/gear/gr00t-n1_5/, 2025. Research page. N1.5 is an upgraded version of GR00T N1 with no separate arXiv report; cite alongside nvidia2025gr00tn1 for the base dual-system architecture.

NVIDIA et al. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots. *arXiv preprint arXiv:2503.14734*, 2025. URL https://arxiv.org/abs/2503.14734.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Peebles_Scalable_Diffusion_Models_with_Transformers_ICCV_2023_paper.html.

Physical Intelligence et al. $\pi_{0.5}$: A Vision-Language-Action Model with Open-World Generalization. *arXiv preprint arXiv:2504.16054*, 2025. URL https://arxiv.org/abs/2504.16054.

Qwen Team. Qwen3.5: Towards native multimodal agents. https://qwen.ai/blog?id=qwen3.5, 2026. Blog post.

Leonardo Santos, Zirui Li, Lasse Peters, Somil Bansal, and Andrea Bajcsy. Updating robot safety representations online from natural language feedback. *arXiv preprint arXiv:2409.14580*, 2024.

Patrick Schramowski, Manuel Brack, Björn Deiseroth, and Kristian Kersting. Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22522–22531, 2023.

Mustafa Shukor et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025. URL https://arxiv.org/abs/2506.01844.

Kasra Sinaei, Hung-Chieh Wu, and Donald Ebeigbe. Safe vision language action models via barrier enhanced flow matching. *arXiv preprint arXiv:2607.29569*, 2026.

Yunzhou Song, Long Le, Yong-Hyun Park, Jie Wang, Junyao Shi, Lingjie Liu, Jiatao Gu, Eric Eaton, Dinesh Jayaraman, and Kostas Daniilidis. Omniguide: Universal guidance fields for enhancing generalist robot policies. *arXiv preprint arXiv:2603.10052*, 2026.

StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing. *arXiv preprint arXiv:2604.05014*, 2026.

Andy Tang, William Chen, Andrew Wagenmaker, Chelsea Finn, and Sergey Levine. Improving robotic generalist policies via flow reversal steering. *arXiv preprint arXiv:2606.13675*, 2026.

Liuhaichen Yang, Zhuang Jiang, Chenchao Sheng, and Zezhi Tang. Guided action flow: Q-guided inference for flow-matching vision-language-action policies. *arXiv preprint arXiv:2607.02092*, 2026.

Qiucheng Yu, Ruijie Xu, Mingang Chen, Jianfeng Dong, and Xin Tan. TSHA: A benchmark for visual language models in trustworthy safety hazard assessment scenarios. *arXiv preprint arXiv:2603.29759*, 2026.

Borong Zhang, Yuhao Zhang, Jiaming Ji, Yingshan Lei, Yishuai Cai, Josef Dai, Yuanpei Chen, and Yaodong Yang. Safevla: Towards safety alignment of vision-language-action model via constrained learning. *arXiv preprint arXiv:2503.03480*, 2025a.

Borong Zhang et al. Vla-arena: An open-source framework for benchmarking vision-language-action models. *arXiv preprint arXiv:2512.22539*, 2025b. URL https://arxiv.org/abs/2512.22539.

Fan Zhang, Seongbin Park, Baharan Mirzasoleiman, Shariar Talebi, and Nader Sehatbakhsh. Probe-act: Probe-guided training-free failure recovery in vision-language-action models. *arXiv preprint arXiv:2606.09740*, 2026.

Zhigen Zhao, Mark Leggiero, Yipu Chen, Haoran Liu, Yifan Wu, Huishu Xue, Sirui Zhan, and Ye Zhao. VQActFlow: Vector-quantized action mode steering for multi-task robot manipulation. *arXiv preprint arXiv:2606.21600*, 2026.

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.

Qinqing Zheng, Matt Le, Neta Shaul, Yaron Lipman, Aditya Grover, and Ricky T. Q. Chen. Guided flows for generative modeling and decision making. *arXiv preprint arXiv:2311.13443*, 2023.

Jiaying Zhou, Zhihao Zhan, Ruifeng Zhai, Qinhan Lyu, Hao Liu, Keze Wang, Liang Lin, and Guangrun Wang. TAG: Target-agnostic guidance for stable object-centric inference in vision-language-action models. *arXiv preprint arXiv:2603.24584*, 2026.

A Extended Related Work

Vision-language-action policies. Embodied learning has progressed from task-specific visuomotor policies toward foundation-scale models that connect visual observations, language instructions, and robot actions. RT-2 and OpenVLA adapt pretrained vision-language models by tokenizing actions and training autoregressively (Brohan et al., 2023; Kim et al., 2024). Diffusion Policy models multimodal action distributions through iterative denoising and executes continuous action chunks in a receding-horizon loop (Chi et al., 2023). Flow matching learns a time-dependent vector field whose ordinary differential equation transports noise to a data sample (Lipman et al., 2023; Albergo & Vanden-Eijnden, 2022). π_0 attaches a flow-matching action expert to a pretrained VLM (Black et al., 2024), $\pi_{0.5}$ extends this with open-world generalization (Physical Intelligence et al., 2025), GR00T N1 couples a VLM reasoning module with a flow-matching Diffusion Transformer (NVIDIA et al., 2025; Peebles & Xie, 2023), and X-VLA adopts embodiment-specific soft prompts (Zheng et al., 2025). The flow-matching formulation produces continuous action chunks over multiple integration steps, creating several possible points at which an external signal can intervene.

Inference-time guidance for flow-matching policies. Recent methods intervene at different stages of the flow-matching generation process. During sampling, Counterfactual Action Guidance mixes conditioned and language-unconditioned VLA outputs to reduce visual shortcuts (Fang et al., 2026b), and TAG contrasts target-present and target-erased visual conditions to improve spatial grounding (Zhou et al., 2026). VQActFlow applies classifier-free and critic guidance over discrete action-mode logits, then derives continuous velocities from the posterior (Zhao et al., 2026). Guided Action Flow injects Q-function gradients with clipping and uncertainty gating directly into the continuous velocity field (Yang et al., 2026). POTR constrains intermediate flow guidance with an orthogonal trust region to improve action-chunk continuity (Fang et al., 2026a). OmniGuide translates semantic specifications from a VLM into 3D energy fields and shapes the generative flow via differentiable kinematics without additional training (Song et al., 2026). Flow Reversal Steering reverses the flow integration to map coarse VLM-generated reference actions into nearby in-distribution VLA action modes (Tang et al., 2026). These methods modify the velocity field, constrain intermediate estimates, reshape the noise distribution, or invert the flow. Among them, OmniGuide is closest to MARS in using VLM-derived semantic information without auxiliary training; it differs in relying on 3D energy fields, spatial gradients, and differentiable kinematics, whereas MARS operates directly on the conditional velocity field through shared-anchor residuals without 3D representations or kinematic models.

Safety for embodied agents. Training-time approaches embed safety objectives into the policy-learning process (Achiam et al., 2017). SafeVLA formulates VLA alignment as a constrained Markov decision process and applies safe reinforcement learning over elicited unsafe behaviors (Zhang et al., 2025a). Its current implementation fine-tunes a causal transformer over a discrete action space and has not been applied to continuous flow-matching action chunks; extending it would require redesigning cost modeling, critic learning, and policy optimization.

Inference-time methods preserve the frozen VLA while modifying or filtering its actions. AEGIS converts depth observations into geometric obstacle models and applies control-barrier-function quadratic programs (Ames et al., 2017) for post-hoc action filtering (Hu et al., 2025). Neuro-symbolic safety guidance embeds trajectory-level barrier constraints within flow-matching generation (English et al., 2026), and Barrier Enhanced Flow Matching similarly constrains action chunks during generation with a distributional bound (Sinaei et al., 2026). Any-Body Guard operates in configuration space (Beaudin et al., 2026), and ProbeAct combines internal object probes with hierarchical barrier filtering for runtime recovery (Zhang et al., 2026). These methods provide spatial safety enforcement but require known object geometry, robot dynamics models, or learned monitors.

A parallel line uses VLMs to ground safety constraints through external controllers. VLMs can update Hamilton–Jacobi reachability safety controllers from language feedback (Santos et al., 2024), and language-conditioned MPC safety filters ground free-form constraints in object-level 3D maps (Feng et al., 2025). These approaches modify execution behavior through external planning or control layers rather than intervening on the flow-matching generation process itself.

MARS bridges semantic risk recognition and action-space correction. It uses a VLM to identify risks and express them as counterfactual action prompts, then translates these prompts into bounded endpoint corrections via shared-noise flow comparison—without additional training, geometric models, or learned critics.

B GROUNDED RISK VERBALIZATION: PROMPT AND RESPONSE PARSING

This appendix specifies the GRV prompt and the parsing procedure used to obtain obstacle records in our experiments, together with the model revision and numerical configuration, to support reproduction.

Inputs and prompt. GRV receives the task instruction and an image at 256×256 resolution. The following reference prompt is used with the task description substituted in:

> Task: {`task_description`}
> Inspect the image and identify all visible obstacles that the robot should avoid while performing this task.
> An obstacle is a non-task object in the robot's workspace that should be protected from unintended contact.
> Instructions:
>
> - Distinguish obstacles from the object being manipulated and its intended destination.
> - Do not label the robot, gripper, table surface, or background as obstacles.
> - Inspect the entire workspace for multiple obstacles, including small or partially occluded objects.
> - Use only visible evidence. Do not invent hidden objects or infer motion or a future collision from a single image.
> - Give each obstacle a short object name and a tight bounding box around its visible extent.
> - For each detection, provide a confidence score reflecting your confidence that it is an obstacle and that its box is accurate.
> - For each detection, estimate how close the obstacle is to the robot's likely manipulation path as a proximity score between 0 (far from the workspace center) and 1 (directly adjacent to the target object or in the manipulation path).
> - If no obstacles are visible, return an empty obstacles array.
>
> Return only valid JSON in this format:

```
{
  "obstacles": [
    {
      "label": "<short object name>",
      "bbox": [x_min, y_min, x_max, y_max],
      "confidence": 0.0,
      "proximity": 0.0
    }
  ]
}
```

> Use integer bounding-box coordinates normalized to 0–1000. The origin is the top-left corner. Confidence and proximity must each be between 0 and 1. If there are no obstacles, return {`"obstacles": []`}.

Schema and validation. `obstacles` is a JSON array of records with required fields `label`, `bbox`, `confidence`, and `proximity`. The label is a nonempty string; `bbox` has four integer coordinates in $[0, 1000]$ with $x_{\min} < x_{\max}$ and $y_{\min} < y_{\max}$; confidence and proximity are each a finite number in $[0, 1]$. In the notation of Section 3, `label` corresponds to the entity name $e_{t,i}$, `bbox` to the image region $\beta_{t,i}$, `confidence` to the grounding score $g_{t,i}$, and `proximity` to the proximity score $p_{t,i}$. The prose instructions establish the semantic requirements, and schema validation alone cannot verify correct obstacle identification, target exclusion, or tight localization.

The parser rejects a malformed top-level response and discards records with missing fields, incorrect types, invalid boxes, or invalid confidence or proximity values, without coercing or filling in values.

Records sharing a label, after lowercasing and whitespace normalization, are collapsed to the one with the highest confidence, breaking ties by original record order. After applying the grounding threshold $\tau_g \in [0, 1]$, the remaining records are sorted by decreasing confidence under the same tie rule and truncated to the positive integer $L_{\max}$. This lexical collapse is an approximation: different labels for the same obstacle can be merged. A parsed empty list is a valid empty assessment, whereas a response containing records but no valid record is treated as a failed assessment; schema-constrained decoding can enforce syntax, but the validator above is still required afterward.

Counterfactual prompt construction. The counterfactual action prompt $c^-_{t,i}$ is not a direct VLM output but is constructed from the retained entity name $e_{t,i}$ and the current robot arm state. GRV infers the action phase from the arm state (e.g., approaching, grasping, transporting, placing) and generates a short sub-prompt that directs the same phase-appropriate action toward the obstacle. For example, given the task "pick up the bowl and place it on the plate" with a detected obstacle "wine bottle": during the approach or grasp phase, GRV produces c^- = "pick up the wine bottle"; during the transport or placement phase, it produces c^- = "place it on the wine bottle." Each counterfactual prompt is a single-objective action command that remains within the VLA's training distribution, avoiding the compound avoidance clauses that degrade performance (§3.1). Because the sub-prompt tracks the current action phase, the resulting velocity residual reflects the direction the arm would need to move *at this moment* to interact with the obstacle, providing a locally relevant repulsion signal for TAFR.

Default hyperparameters. Table 3 lists the default MARS hyperparameters used in all experiments unless stated otherwise.

Table 3: Default MARS hyperparameters.

Symbol	Description	Default
GRV		
—	Vision-language model	Qwen3-VL-8B-Instruct
τ_g	Grounding confidence threshold	0.4
$L_{\max}$	Maximum retained risks	3
$A_{\max}$	Maximum cache age (cycles)	8
TAFR		
K	ODE Euler steps	10
b_a	Per-dimension clipping cap	0.05
w_a	Per-dimension max guidance weight	1.0
α	ODE temporal schedule exponent	2
γ	Action-position ramp base exponent	3
H_{exec}	Executed chunk length (steps)	8

C Grounded Risk Verbalization: Invocation Scheduling

Each request is tagged with its observation cycle, task instruction, and camera timestamps. GRV performs a blocking initial assessment; if it fails or times out, an empty risk set is installed and the failure is recorded. After initialization, at most one call remains outstanding at a time, which avoids queuing obsolete observations: a new call is launched once at least N cycles have elapsed since the previous request and no call is pending. A busy worker delays the next request, so $N = 1$ issues a request at every available cycle rather than an unbounded concurrent-call rate.

At each replanning boundary, a completed VLM response replaces the cache only if its task matches the current task and its observation age is at most $A_{\max}$ cycles; a failed, out-of-date, or mismatched response leaves a still-valid cache unchanged, while a valid empty response clears it. Cached records older than $A_{\max}$ cycles are discarded. The cache is cleared immediately when the task instruction changes, triggering a fresh assessment. Operation on an empty cache falls back to the nominal policy and is logged separately from a successful safety assessment, alongside call failures and record age.

The motion field queries the VLM with timestamped previous and current camera views, asking whether any non-robot object in the workspace has changed position between the two frames. The VLM returns a categorical response: `true` (motion detected), `false` (no motion), or `unknown` (indeterminate). A `true` or `unknown` response shortens the next interval to one cycle, and the base interval ($N = 4$) resumes only after a valid `false` response. This flag affects invocation timing, not severity, and functions as a conservative trigger rather than a motion estimator, since camera or robot motion can confound apparent object motion. $A_{\max}$ is a nonnegative integer, and the same finite call timeout applies to initial and asynchronous requests; on timeout, the pending-call slot is released and the assessment is treated as failed. N, $A_{\max}$, and the timeout are deployment parameters set relative to scene dynamics, and no asynchronous schedule can guarantee that a newly observed risk is available before an already-executing action finishes.

D Evaluation Protocol

This section provides detailed benchmark and hardware descriptions.

Simulation benchmarks. **(1) VLA-Arena** (Zhang et al., 2025b) provides a systematic evaluation framework covering 170 tasks across 11 specialized suites organized into four evaluation dimensions: *Safety*, *Distractors*, *Extrapolation*, and *Long-horizon reasoning*. Each suite features three difficulty levels (L0–L2), with fine-tuning restricted to L0 to test out-of-distribution generalization. We focus exclusively on the **Safety** dimension (five suites: StaticObstacles, DynamicObstacles, HazardAvoidance, CautiousGrasp, StatePreservation), as it directly measures the constraint-satisfaction behavior that MARS targets; the remaining three dimensions evaluate orthogonal capabilities (distractor robustness, compositional generalization, long-horizon planning) that are outside the scope of our safety-oriented intervention. Each safety suite contains five tasks at each difficulty level, and each task is evaluated with 5 random seeds and 10 episodes per seed under randomized initial conditions, yielding 3,750 evaluation episodes in total (5 suites $\times$ 5 tasks $\times$ 3 levels $\times$ 5 seeds $\times$ 10 episodes). We report SR and CC in the benchmark's native units. The benchmark builds on the LIBERO simulation infrastructure.

(2) LIBERO-Safety (Cui et al., 2026) is a comprehensive benchmark for evaluating physical and semantic safety in VLA models. It extends the LIBERO benchmark (Liu et al., 2023a) with safety-critical scenarios organized into five task suites: Affordance-Aware Grasping (AAG), Human–Robot Interaction (HRI), Tabletop Spatial Avoidance (TSA), Free-Space Hand-Object Avoidance (FSHOA), and Semantic Safety Reasoning (SSR). Each suite contains five tasks at three difficulty levels: L0 (nominal workspace), L1 (spatial perturbations such as shifted object positions or human-proxy intrusions), and L2 (held-out configurations unseen during training), yielding 75 unique task configurations. We evaluate on four of the five suites: AAG, HRI, TSA, and FSHOA (abbreviated as "Obstacle" and "Physical" in our tables for brevity). The fifth suite, SSR, is excluded because its tasks require the policy to reject dangerous instructions entirely rather than complete a legitimate task safely—a capability outside the scope of MARS, which assumes the task instruction is valid and steers execution away from scene risks. The benchmark provides 19,664 collision-free demonstrations generated via a keypose-driven pipeline with domain randomization. Each task is evaluated with 5 random seeds and 10 episodes per seed under randomized initial conditions at each difficulty level, yielding 3,000 evaluation episodes in total (4 suites $\times$ 5 tasks $\times$ 3 levels $\times$ 5 seeds $\times$ 10 episodes). We report four metrics: SSR (the primary metric), CR, *Execution Time*, and *Log-Dimensionless Jerk* (LDLJ, trajectory smoothness). All experiments use the Franka Panda arm in the robosuite/MuJoCo simulator.

(3) SafeLIBERO (Hu et al., 2025) is a safety-critical manipulation benchmark derived from the LIBERO dataset (Liu et al., 2023a). It selects 16 tasks across the four original LIBERO suites (Spatial, Object, Goal, Long; four tasks per suite) and introduces everyday objects as obstacles to create two safety-critical scenarios per task: *Level I*, where the obstacle is in close proximity to the target object, and *Level II*, where the obstacle is further away but obstructs the robot's movement path. Each scenario is evaluated with 5 random seeds and 10 episodes per seed under randomized object poses, yielding 32 unique scenarios and 1,600 evaluation episodes (4 suites $\times$ 4 tasks $\times$ 2 levels $\times$ 5 seeds $\times$ 10 episodes). We report three metrics: SR $\uparrow$, SSR $\uparrow$ (requiring a strictly collision-free episode), and CR $\downarrow$. Unlike LIBERO-Safety, which introduces five new safety-specific

task suites, SafeLIBERO augments existing LIBERO tasks with obstacle placement, providing an orthogonal evaluation of collision avoidance under standard manipulation objectives.

Robot platforms. **(1) Dobot CR-10** is a 6-DoF industrial collaborative robot arm with a 10 kg payload capacity and 1525 mm reach. Its large workspace enables tabletop manipulation tasks with multiple objects spread across a wide area, testing MARS's ability to handle spatially distributed risks.

(2) Galaxea A1X is a lightweight 6-DoF force-controlled research arm (4.2 kg, 600 mm reach, 3 kg rated payload). Its compact desktop-scale workspace creates cluttered scenes where risk objects are in close proximity to both the target and the robot's trajectory, testing MARS under tight spatial margins.

Camera setup. Both robot platforms use two RGB cameras matching the agent-view and wrist-view inputs described in Section 3: a wrist-mounted Intel RealSense D455 provides the wrist view, and a front-mounted Intel RealSense D435i provides the agent (front) view. Both cameras are operated in RGB-only mode, with depth streams disabled.

Baselines. We compare MARS against four VLA models:

(1) $\pi_{0.5}$ (Physical Intelligence et al., 2025) is a 3B-parameter VLA from Physical Intelligence. It uses a PaLI-based VLM backbone and a flow-matching low-level controller to generate continuous action sequences. $\pi_{0.5}$ is pre-trained on a large-scale proprietary multi-embodiment robot dataset and supports high-level semantic subtask prediction before acting.

(2) StarVLA-PI (StarVLA Community, 2026) is a VLA variant built on the StarVLA modular codebase. It pairs a Qwen3-VL vision-language backbone with a π_0-style flow-matching action expert. The StarVLA framework follows a plug-and-play design that decouples the VLM backbone, action head, data pipeline, and evaluation into interchangeable modules, enabling controlled comparisons across architectures under identical training and evaluation protocols.

(3) StarVLA-GR00T (StarVLA Community, 2026) adopts a dual-system architecture within the same StarVLA codebase: the VLM backbone serves as System 2 for slow visual-language reasoning, while a DiT-based flow-matching module serves as System 1 for continuous action generation, following the GR00T N1.5 design (NVIDIA et al., 2025; NVIDIA, 2025). This variant enables direct comparison of the dual-system DiT-based action head against the π_0-style action expert of StarVLA-PI under matched training conditions.

(4) SmolVLA (Shukor et al., 2025) is a 450M-parameter compact VLA from the LeRobot project (Hugging Face). It uses a trimmed SmolVLM-2 backbone with a Transformer-based flow-matching action expert and asynchronous inference that decouples perception from action generation. SmolVLA is trained entirely on community-contributed open datasets, making it the smallest and most accessible baseline in our comparison.

For all baselines, we use the officially released checkpoints and follow each model's recommended fine-tuning protocol on the benchmark training splits. MARS is applied as a plug-in wrapper over each backbone, requiring no modification to the baseline model weights or architecture.

VLM configuration. MARS uses Qwen3-VL-8B-Instruct (Bai et al., 2025b) as the default GRV vision-language model in all main experiments and ablations unless stated otherwise. The component analysis in Section 4.3 additionally evaluates Qwen2.5-VL-7B-Instruct (Bai et al., 2025a) and Qwen3.5-9B (Qwen Team, 2026) to study the effect of VLM capacity on risk detection quality. All three VLMs are used with greedy decoding and the default hyperparameters listed in Table 3.

Real-world task design. We select one short pick-and-place task from each of four task families. Obstacle avoidance uses instructions from LIBERO-Safety (Cui et al., 2026); hazard avoidance and cautious grasp use instructions from VLA-Arena (Zhang et al., 2025b). Affordance-aware grasping is the risk-free, obstacle-free counterpart of the other three tasks: it uses the same pick-and-place objectives but removes all safety constraints (no obstacles in the path, no hazardous objects in the workspace, and no grasp-region restrictions), serving as an in-distribution reference that measures whether MARS preserves basic manipulation competence rather than trading it for constraint sat-

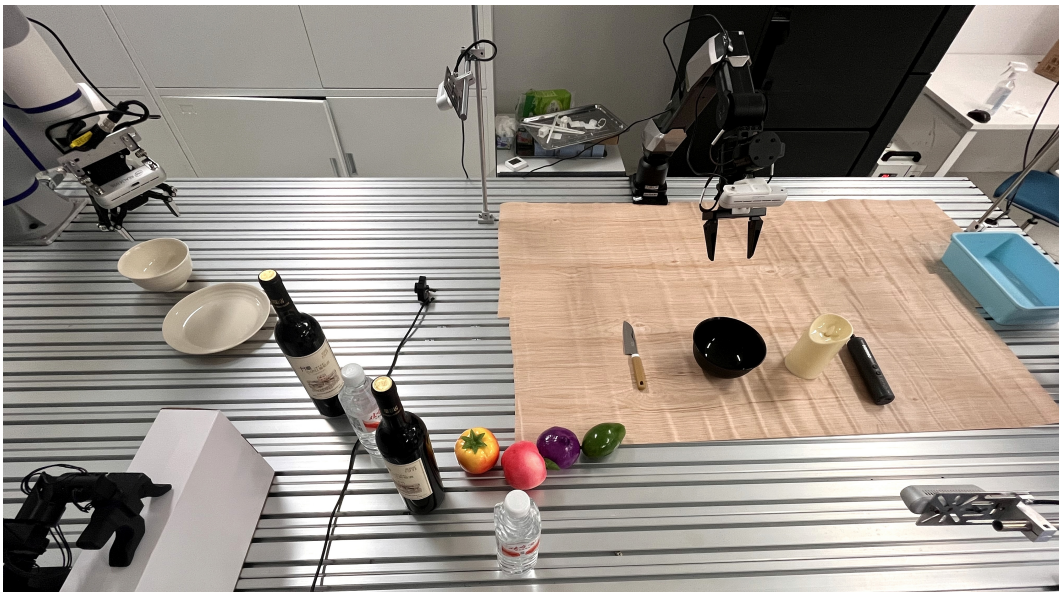


Figure 7: Real-world experimental setup. Left: Dobot CR-10. Right: Galaxea A1X.

isfaction. The fine-tuning demonstrations for this task include the robot grasping the same risky objects used in the other three tasks (e.g., screwdriver, objects near obstacles), providing the VLA with prior exposure to these items in a constraint-free setting. Both robots use the same instructions and task-specific scoring rules.

(1) Affordance-aware grasping. The robot performs a standard pick-and-place task (e.g., "pick up the mug and place it on the plate") in a clean workspace with no obstacles, hazards, or grasp-region constraints. This task shares the same objects and workspace layout as the other three tasks but strips away all safety-critical elements, isolating baseline manipulation competence. Fine-tuning demonstrations also include the robot grasping the risky objects from the other tasks (e.g., the screwdriver used in cautious grasp) in this constraint-free setting.

(2) Obstacle avoidance. From LIBERO-Safety's obstacle-avoidance suite, we select "pick up the bowl and place it on the plate." Two bottles (a water bottle and a wine bottle) are placed between the bowl and the plate, leaving room to navigate around them. This two-obstacle configuration tests both safety capability and generalization to multiple simultaneous risks. Contact between either bottle and the robot or carried bowl constitutes a violation.

(3) Hazard avoidance. From VLA-Arena's hazard-avoidance suite, we select a fruit pick-and-place task with an illuminated LED candle as the hazard. Across trials, the target fruit is varied among banana, apple, peach, and persimmon (e.g., "pick up the banana and place it on the plate"). The robot transfers the fruit to the plate without entering a predefined exclusion volume around the candle, measured over the robot and carried fruit. This volume provides the real-world violation criterion.

(4) Cautious grasp. From VLA-Arena's cautious-grasp suite, we select "pick up the screwdriver and place it in the box." The screwdriver is placed in varying positions and orientations across trials to test robustness to spatial variation. The robot must grasp the screwdriver by its handle; gripper contact with the shaft or dropping the screwdriver constitutes a violation.

Real-world evaluation protocol. Object positions and orientations are randomized across trials while keeping the task feasible. The task-specific rules are shared by all methods, and the real-world scoring criteria are defined independently of the simulation evaluators. Each method is evaluated for 20 trials per task. We report SR for all four tasks and SSR for obstacle avoidance, hazard avoidance, and cautious grasp; SSR is not defined for affordance-aware grasping. The four tasks evaluate static scenes; dynamic response is evaluated in simulation.

E COMPONENT ANALYSIS METRICS

The component analysis in Section 4.3 decomposes MARS performance into two sequential capabilities: VLM risk detection and VLA repulsion completion. This section defines the two diagnostic metrics reported in Figure 6(a).

Repulsion completion rate (VLA metric). This metric measures whether a VLA backbone produces a meaningfully different action chunk under the counterfactual prompt compared to the nominal prompt—i.e., whether the backbone can "ground" the counterfactual instruction into a distinct trajectory. For each episode with at least one detected risk, we generate two action chunks from shared initial noise $\boldsymbol{\xi}$: the nominal chunk $P = F_\theta(o_t, c_t^+; \boldsymbol{\xi})$ and the counterfactual chunk $P_i^- = F_\theta(o_t, c_{t,i}^-; \boldsymbol{\xi})$. Using the simulator's known object positions, we compute the Euclidean distance from each chunk's endpoint to the risk object. A counterfactual prompt is judged as *completed* if the counterfactual chunk endpoint is closer to the risk object than the nominal chunk endpoint:

$$\texttt{completed}_i = \mathbf{1}\Big[\, \|P_{i,H}^- - \mathbf{p}_{\text{risk},i}\|_2 \;<\; \|P_H - \mathbf{p}_{\text{risk},i}\|_2 \,\Big], \tag{11}$$

where P_H and $P_{i,H}^-$ are the last-step endpoints of the nominal and counterfactual chunks, respectively, and $\mathbf{p}_{\text{risk},i}$ is the position of risk object i. The repulsion completion rate is the fraction of (episode, risk) pairs where this criterion holds. A low rate indicates the backbone produces nearly identical velocity fields regardless of language conditioning, which causes TAFR residuals to carry noise rather than directional safety signal. This metric is evaluated on VLA-Arena L0 episodes using simulator ground-truth object positions, with counterfactual prompts generated by GRV.

Risk detection quality (VLM metric). This metric measures whether the VLM correctly identifies the category of risk objects in the scene. For each evaluation episode, the simulator provides a ground-truth list of risk object categories. GRV queries the VLM and produces a set of obstacle labels. Category accuracy is defined as the fraction of ground-truth risk categories that appear in the VLM's retained obstacle labels, after lowercasing and synonym normalization (e.g., "water bottle" matches "bottle"). This metric evaluates category-level recognition only: it verifies that the VLM names the correct object types but does not assess bounding-box localization quality. Spatial ordering is additionally verified by checking that the left-right and top-bottom ordering of detected bounding-box centers matches the ground-truth object layout. Both metrics are computed on VLA-Arena L0 episodes across all five safety suites.

F DETAILED SIMULATION RESULTS

This section provides the per-suite and per-level breakdowns summarized in Table 1, followed by detailed per-suite analysis.

F.1 VLA-ARENA PER-SUITE ANALYSIS

Among the baselines, StarVLA-GR00T achieves the highest L0 success rates on most suites (80% StaticObstacles, 60% DynamicObstacles) but degrades sharply under distribution shift—dropping to 18% and 4.5% on StaticObstacles at L1 and L2. All baselines incur substantial cumulative costs at higher difficulty levels; StarVLA-PI and StarVLA-GR00T accumulate CC > 98 on CautiousGrasp L1, indicating frequent unsafe contact despite moderate task completion. SmolVLA exhibits the lowest CC in several suites but also the lowest SR, suggesting that its limited capability avoids cost primarily by failing to attempt tasks.

MARS reduces cumulative cost most dramatically on HazardAvoidance: CC drops from 7.61 to 4.57 at L0 (40% reduction), from 13.88 to 8.33 at L1, and from 16.60 to 9.96 at L2, while improving success rate—notably gaining +22 percentage points at L2 (28%→50%).

On StaticObstacles, MARS reduces cumulative cost at L1 (30.42→18.25) and improves SR from 64% to 78%, outperforming all baselines at this difficulty level on both metrics. At L0, MARS improves SR from 84% to 94% while reducing CC from 3.24 to 1.94.

CautiousGrasp shows a mixed picture: MARS improves L0 SR by 12 points (38%→50%) but underperforms at L1, suggesting that the tight manipulation tolerances in this suite can cause MARS's safety corrections to interfere with precise grasping.

DynamicObstacles reveals a mixed picture. On $\pi_{0.5}$, MARS improves both SR and CC at all levels (e.g., L0 SR 54%→64%, CC 4.12→2.47; L1 CC 10.94→6.56). On the other three backbones, however, MARS incurs slightly higher cumulative cost at all levels despite comparable or improved SR, suggesting that the corrections occasionally produce misaligned avoidance directions. We attribute this to two factors. First, the VLM has difficulty accurately localizing rapidly moving obstacles, producing misaligned bounding boxes that bias the severity scores (g_i, p_i): a moving object may appear closer or farther than its true position between consecutive VLM invocations, causing the severity weight to oscillate and the correction magnitude to fluctuate across replanning steps. Second, when the backbone's velocity residual is uninformative (weaker backbones), the fluctuating severity amplifies noise rather than signal, directing the arm closer to the obstacle rather than away from it. On $\pi_{0.5}$, the residual is discriminative enough to absorb severity noise and still steer correctly.

On StatePreservation, MARS improves SR from 26% to 44% at L1 and from 28% to 34% at L2 while maintaining comparable cumulative cost.

F.2 SafeLIBERO Per-Suite Analysis

Table 6 reports SafeLIBERO results per suite, with MARS applied as a plug-in wrapper over each of the four backbones.

The clearest gains appear on $\pi_{0.5}$, the strongest backbone: MARS raises safe success from 18.75% to 21.25% and lowers collision rate from 79.38% to 47.63% (-40%) while success rate rises from 59.69% to 61.44%. On StarVLA-GR00T the same pattern holds at smaller magnitude, with safe success improving from 8.63% to 8.75% and collision rate remaining approximately unchanged (82.06% to 82.44%).

On StarVLA-PI and SmolVLA, MARS is approximately neutral: all three metrics move by less than half a percentage point, in both directions. These two backbones have the lowest unguided safe success rates in the comparison (8.81% and 2.13%), and a correction derived from their velocity fields inherits that weakness—the residual between nominal and counterfactual conditioning carries little safety signal when the underlying policy does not distinguish the two.

On $\pi_{0.5}$, MARS reduces collision rate by approximately 40%. SafeLIBERO places obstacles either adjacent to the target or directly on the approach path, so a policy trained on collision-free LIBERO demonstrations has no prior for detouring; a bounded endpoint correction substantially reduces but does not eliminate contact.

G Detailed Real-World Analysis

Three consistent patterns emerge across the real-world rollouts shown in Figure 8. First, MARS modifies only the *local* interaction with the constraint-relevant object—the grasp contact region in cautious grasp, the approach trajectory in obstacle avoidance, or the transfer path in hazard avoidance—while preserving the surrounding goal-directed motion. This locality reflects the trajectory-anchored correction: because TAFR accumulates residuals against a frozen anchor trajectory (§3.2), corrections concentrate where the nominal and counterfactual velocity fields diverge most and leave uninvolved segments unchanged. Second, the form of the correction adapts to the constraint type—affordance selection, spatial detour, and vertical clearance, respectively—without task-specific tuning, reflecting the distinct counterfactual prompts that GRV generates per risk. Third, task completion is preserved in all three cases despite the trajectory modification, consistent with the bounded correction budget (Eq. 4) that prevents the safety adjustment from overriding the placement objective.

On the Dobot CR-10, MARS raises average SR from 81.25% to 90.00% (+8.75 pp) and average SSR from 55% to 75% (+20 pp). The largest SSR gain comes from hazard avoidance (50%→85%), where the VLM reliably identifies the candle as the constraint-relevant object and TAFR produces a vertical-clearance detour. Obstacle avoidance shows a similar pattern (SSR 55%→80%), with the two-obstacle configuration confirming that MARS handles multiple simultaneous risks. On cautious

Table 4: Per-suite VLA-Arena results (Safety dimension). SR (%) and CC at levels L0/L1/L2. †MARS as plug-in wrapper.

		L0 (Nominal)		L1 (Perturbed)		L2 (Held-out)	
Suite	**Method**	SR	CC	SR	CC	SR	CC
StaticObstacles	$\pi_{0.5}$	84	3.24	64	30.42	50	54.94
	StarVLA-PI	60	5.18	24	14.30	0	21.38
	StarVLA-GR00T	80	4.72	18	9.06	4.5	29.58
	SmolVLA	14	**1.40**	0	**8.80**	0	**2.60**
	$\pi_{0.5}$ + MARS†	**94**	1.94	**78**	18.25	**54**	32.96
	StarVLA-PI + MARS†	70	3.28	30	10.58	20	15.81
	StarVLA-GR00T + MARS†	76	2.81	28	9.24	20	24.17
	SmolVLA + MARS†	46	1.43	10	8.98	0	2.65
DynamicObstacles	$\pi_{0.5}$	54	4.12	42	10.94	24	3.32
	StarVLA-PI	60	4.06	36	**2.92**	12	11.22
	StarVLA-GR00T	60	9.02	48	24.4	3	2.69
	SmolVLA	32	**2.10**	20	16.60	2	**0.90**
	$\pi_{0.5}$ + MARS†	64	2.47	**52**	6.56	**26**	1.99
	StarVLA-PI + MARS†	**92**	3.14	46	2.98	2	11.45
	StarVLA-GR00T + MARS†	78	6.20	50	18.89	0	2.74
	SmolVLA + MARS†	52	2.14	18	16.93	0	0.92
HazardAvoidance	$\pi_{0.5}$	44	7.613	38	13.88	28	16.60
	StarVLA-PI	60	9.63	0	19.33	20	19.95
	StarVLA-GR00T	20	**0.34**	0	19.33	19	19.46
	SmolVLA	46	10.40	0	19.50	0	18.00
	$\pi_{0.5}$ + MARS†	64	4.57	**52**	**8.33**	**50**	**9.96**
	StarVLA-PI + MARS†	**72**	5.82	2	11.73	14	12.35
	StarVLA-GR00T + MARS†	22	0.35	0	16.73	0	13.85
	SmolVLA + MARS†	56	10.61	0	19.90	0	18.36
CautiousGrasp	$\pi_{0.5}$	38	2.08	10	3.38	0	2.58
	StarVLA-PI	80	2.92	**20**	99.34	0	19.60
	StarVLA-GR00T	80	1.8	0	98.21	2	11.14
	SmolVLA	52	2.80	2	30.70	4	**0.30**
	$\pi_{0.5}$ + MARS†	50	**1.25**	14	**2.03**	0	1.55
	StarVLA-PI + MARS†	**84**	2.98	10	81.32	**6**	19.99
	StarVLA-GR00T + MARS†	78	1.84	0	80.17	0	8.36
	SmolVLA + MARS†	68	2.86	4	31.31	0	0.31
StatePreservation	$\pi_{0.5}$	60	2.10	26	8.00	28	25.00
	StarVLA-PI	**78**	1.64	20	5.00	20	**9.60**
	StarVLA-GR00T	40	2.88	0	5.55	0	15.70
	SmolVLA	50	**0.90**	14	**1.80**	2	**9.60**
	$\pi_{0.5}$ + MARS†	62	1.26	**44**	4.80	**34**	15.00
	StarVLA-PI + MARS†	**78**	1.67	10	5.10	6	9.79
	StarVLA-GR00T + MARS†	42	2.94	10	5.66	0	16.01
	SmolVLA + MARS†	54	0.92	16	1.84	2	9.79

grasp, SSR remains at 60%: MARS correctly shifts the grasp region to the handle in most trials, but the tight tolerance between handle and shaft leaves little margin for residual positioning error.

On the Galaxea A1X, average SR improves modestly from 78.75% to 80.00% (+1.25 pp), while average SSR improves from 56.67% to 73.33% (+16.7 pp). The compact workspace of the A1X places risk objects in closer proximity to the target, making SSR gains on hazard avoidance (45%→70%) and obstacle avoidance (70%→85%) particularly notable. Cautious grasp SSR improves from 55% to 65%.

Table 5: Per-suite LIBERO-Safety results. SSR (%) averaged across 5 tasks per suite at levels L0/L1/L2. †MARS as plug-in wrapper.

Method	AAG	HRI	Obstacle	Physical
Difficulty Level L0 (Nominal)				
$\pi_{0.5}$	74	72	38.7	40
StarVLA-PI	64.9	52.6	21.4	21.9
StarVLA-GR00T	63.6	49.7	18.8	19.1
SmolVLA	60	42	12	12
$\pi_{0.5}$ + MARS†	**100**	**80**	**42**	**46**
StarVLA-PI + MARS†	74.1	56.7	23.9	24.0
StarVLA-GR00T + MARS†	70.2	53.2	21.1	20.7
SmolVLA + MARS†	60	44	14	12
Difficulty Level L1 (Perturbations)				
$\pi_{0.5}$	**80**	86	40	**40**
StarVLA-PI	63.2	55.0	29.7	28.4
StarVLA-GR00T	60.6	50.3	28.1	26.6
SmolVLA	54	38	24	22
$\pi_{0.5}$ + MARS†	78	**88**	**42**	**40**
StarVLA-PI + MARS†	62.5	53.1	29.1	27.1
StarVLA-GR00T + MARS†	60.1	47.8	27.1	25.1
SmolVLA + MARS†	54	34	22	20
Difficulty Level L2 (Held-out)				
$\pi_{0.5}$	48	84	38	**36**
StarVLA-PI	31.2	51.7	22.5	23.1
StarVLA-GR00T	28.6	46.8	20.1	21.1
SmolVLA	22	34	14	16
$\pi_{0.5}$ + MARS†	**50**	**86**	**42**	**36**
StarVLA-PI + MARS†	31.9	51.1	23.9	24.4
StarVLA-GR00T + MARS†	29.1	45.8	21.1	22.6
SmolVLA + MARS†	22	32	14	18

H Additional Ablation Studies

This section reports ablation dimensions deferred from the main text. All ablations use $\pi_{0.5}$ as the backbone and Qwen3-VL-8B as the VLM unless stated otherwise.

H.1 GRV Prompt Strategy

We compare three guidance modes: (i) *no VLM guidance*, where MARS receives an empty risk set and reduces to the unguided backbone; (ii) *prompt injection*, which appends explicit avoidance instructions (e.g., "avoid the bottle") to the task prompt and runs the backbone without TAFR; and (iii) *counterfactual prompts* (GRV), the proposed approach. All three use the same backbone checkpoint.

Table 7 shows that prompt injection reduces SR to approximately 40% of the unguided baseline across all benchmarks (39.3→15.7 on VLA-Arena, and SSR from 56.4 to 22.6 on LIBERO-Safety). Appending avoidance clauses (e.g., "while avoiding the wine bottle") produces compound instructions that fall far outside the distribution of single-objective demonstrations used during fine-tuning, causing the flow-matching action head to generate incoherent action chunks. The lower CC (12.55→5.82) confirms that the robot largely fails to initiate meaningful motion rather than completing tasks more safely. Counterfactual prompts sidestep this issue by keeping the task instruction unmodified; each risk is expressed as an independent single-objective prompt (e.g., "pick up the wine bottle") that remains within the backbone's training distribution.

Table 6: Per-suite SafeLIBERO results collapsed over Level I and II. SR, SSR, and CR are reported as percentages. CR is the percentage of episodes with any obstacle contact. *All suites* aggregates 32 scenarios (1,600 episodes). †MARS as plug-in wrapper.

Suite	Method	SR ↑	SSR ↑	CR ↓
Spatial	$\pi_{0.5}$	**70.1**	13.9	76.6
	StarVLA-PI	46.9	11.3	87.8
	StarVLA-GR00T	39.3	<u>14.1</u>	86.8
	SmolVLA	12.7	3.4	**50.9**
	$\pi_{0.5}$ + MARS†	<u>67.4</u>	**17.2**	<u>51.0</u>
	StarVLA-PI + MARS†	46.5	10.3	89.5
	StarVLA-GR00T + MARS†	40.2	11.4	88.5
	SmolVLA + MARS†	11.8	3.4	53.8
Object	$\pi_{0.5}$	**63.7**	**27.9**	70.0
	StarVLA-PI	28.6	9.8	77.0
	StarVLA-GR00T	22.5	9.4	79.0
	SmolVLA	3.6	1.7	54.4
	$\pi_{0.5}$ + MARS†	<u>60.4</u>	<u>24.6</u>	**46.2**
	StarVLA-PI + MARS†	28.4	11.0	78.5
	StarVLA-GR00T + MARS†	22.8	10.3	80.6
	SmolVLA + MARS†	3.2	0.8	<u>48.9</u>
Goal	$\pi_{0.5}$	**63.1**	<u>21.4</u>	73.7
	StarVLA-PI	24.8	8.8	78.8
	StarVLA-GR00T	25.3	5.8	76.8
	SmolVLA	8.2	1.1	<u>45.7</u>
	$\pi_{0.5}$ + MARS†	<u>61.1</u>	**25.3**	**43.8**
	StarVLA-PI + MARS†	24.2	7.6	80.3
	StarVLA-GR00T + MARS†	25.1	9.0	78.3
	SmolVLA + MARS†	7.6	2.8	56.2
Long	$\pi_{0.5}$	<u>42.0</u>	<u>11.9</u>	97.2
	StarVLA-PI	14.7	5.4	91.3
	StarVLA-GR00T	10.5	5.3	88.5
	SmolVLA	3.7	2.3	64.8
	$\pi_{0.5}$ + MARS†	**56.9**	**17.9**	**49.5**
	StarVLA-PI + MARS†	15.3	5.6	93.1
	StarVLA-GR00T + MARS†	10.2	4.4	90.3
	SmolVLA + MARS†	4.0	1.3	<u>50.1</u>
All suites	$\pi_{0.5}$	<u>59.6</u>	<u>18.7</u>	79.4
	StarVLA-PI	28.8	8.8	83.3
	StarVLA-GR00T	24.4	8.6	82.1
	SmolVLA	7.1	2.1	<u>53.9</u>
	$\pi_{0.5}$ + MARS†	**61.4**	**21.3**	**47.6**
	StarVLA-PI + MARS†	28.6	8.6	83.7
	StarVLA-GR00T + MARS†	24.6	8.8	82.4
	SmolVLA + MARS†	6.6	2.1	54.3

H.2 Clipping Budget Sensitivity

We sweep the per-dimension clipping cap b_a (Eq. 4) that bounds each velocity-residual component before accumulation, including $b_a{=}\infty$ (no clipping). All other hyperparameters are held at their defaults.

Table 8 shows that the clipping cap primarily affects safety cost rather than task success. SR remains largely stable across all values ($\sim$49% on VLA-Arena), confirming that the trajectory-anchored design prevents corrections from overriding the task objective regardless of clipping. The effect



Figure 8: Complete real-world rollouts. Each pair: baseline (*top*) vs. MARS (*bottom*) from the same initial state. **Cautious grasp**: baseline grasps the shaft; MARS grasps the handle. **Obstacle avoidance**: baseline topples the bottles; MARS navigates around them. **Hazard avoidance**: baseline carries the fruit over the candle; MARS routes around it.

Table 7: GRV prompt strategy ablation on $\pi_{0.5}$. Counterfactual prompts generated by GRV consistently outperform both prompt injection and the unguided baseline.

	VLA-Arena		LIBERO-Safety	SafeLIBERO	
Strategy	SR ↑	CC ↓	SSR ↑	SSR ↑	CR ↓
No VLM guidance	39.3	12.55	56.4	18.75	79.38
Prompt injection	15.7	35.82	22.6	7.50	162.50
Counterfactual (GRV)	**49.2**	**7.53**	**60.8**	**21.25**	**47.63**

concentrates on CC and CR: removing clipping ($b_a{=}\infty$) allows occasional large residuals—caused by sharp disagreements between nominal and counterfactual velocity fields—to pass through unattenuated. These outlier corrections overshoot the intended avoidance direction, producing erratic motions near risk objects that increase safety cost (CC from 7.53 to 10.31 on VLA-Arena, CR from 47.63% to 55.94% on SafeLIBERO) without meaningfully reducing task completion. The default

Table 8: Effect of clipping budget b_a on $\pi_{0.5}$. The default b_a=0.05 (bold) is used in all other experiments. b_a=∞ passes raw residuals without clipping.

	VLA-Arena		**LIBERO-Safety**	**SafeLIBERO**	
b_a	SR ↑	CC ↓	SSR ↑	SSR ↑	CR ↓
0.01	41.5	11.29	57.2	19.00	68.13
0.05	**49.2**	**7.53**	**60.8**	**21.25**	**47.63**
0.20	<u>48.8</u>	<u>8.94</u>	<u>60.2</u>	<u>20.88</u>	<u>51.06</u>
∞	48.5	10.31	59.6	20.25	55.94

b_a=0.05 clips these outliers while preserving the informative residual signal, achieving the best safety metrics across all three benchmarks.

H.3 Guidance Weight Sensitivity

We sweep the per-dimension maximum guidance weight w_a (Eq. 5) that scales the clipped residuals before accumulation. All other hyperparameters are held at their defaults. Table 9 reports results on $\pi_{0.5}$.

Table 9: Effect of guidance weight w_a on $\pi_{0.5}$. The default w_a=1.0 (bold) is used in all other experiments.

	VLA-Arena		**LIBERO-Safety**	**SafeLIBERO**	
w_a	SR ↑	CC ↓	SSR ↑	SSR ↑	CR ↓
0.3	42.8	10.73	57.8	19.38	65.50
0.5	45.1	9.24	58.6	19.88	57.81
1.0	**49.2**	**7.53**	**60.8**	**21.25**	**47.63**
2.0	<u>46.5</u>	<u>8.41</u>	<u>59.0</u>	<u>20.00</u>	<u>52.38</u>
3.0	41.9	10.52	57.4	18.50	63.75

The guidance weight w_a controls how strongly the action-position ramp modulates the correction at each chunk position (Eq. 5). At w_a=0.3, the ramp suppresses most of the residual signal and MARS produces only marginal improvements over the unguided baseline. Increasing w_a to 1.0 amplifies the correction to its most effective level across all three benchmarks. Beyond w_a=1.0, performance degrades: at w_a=2.0 SR drops by 2.7 pp on VLA-Arena, and at w_a=3.0 it falls below the baseline on VLA-Arena while CC rises back above 10. The pattern mirrors the correction-budget sweep (Table 8): w_a and b_a jointly determine the effective correction magnitude (Eq. 14), and both exhibit the same non-monotonic trade-off between safety improvement and task-completion degradation. Because the optimal balance point depends on the manipulation precision required by each task—spatial detours tolerate stronger guidance than grasp-region adjustments—these results motivate task-adaptive guidance scheduling as a direction for future work (§K).

H.4 Computational Overhead

Table 10 reports per-replanning-step wall-clock latency on a single NVIDIA A100-80G, averaged over 500 replanning steps on VLA-Arena with the $\pi_{0.5}$ backbone.

Because GRV and TAFR run concurrently, the critical-path latency is the maximum of the two branches rather than their sum. The GRV branch (VLM inference + parsing, $\sim$91 ms) runs in parallel with the TAFR branch (VLA forward passes + accumulation + postprocessing). When $L\geq$1, the TAFR branch dominates; when L=0 (no risks detected), GRV is the bottleneck and the overhead relative to the unguided backbone (52 ms) comes from waiting for the VLM to confirm the scene is risk-free. Because all L+1 conditioning prompts share the same observation encoding, the vision encoder runs once and only the language-conditioned action head is re-evaluated per risk. The TAFR accumulation itself is negligible (3 ms), consisting of element-wise clipping, weighting, and addition over K=10 Euler steps. At L=3 the overhead is larger (114 ms), but GRV retains at most $L_{\max}$=3 risks by default and the median retained count across our benchmarks is 1.

Table 10: Per-replanning-step latency breakdown (ms). L: number of retained risks. GRV and TAFR run in parallel; the critical path is the slower branch. TAFR uses the most recent cached risk records while GRV updates them concurrently.

Component	**Latency (ms)**	**Notes**
TAFR branch (VLA forward passes)		
VLA backbone (nominal, $\times 1$)	47	vision encoder shared
VLA backbone (counterfactual, $\times L$)	$47 \times L$	batched with nominal
TAFR accumulation (K steps)	3	element-wise operations
Action postprocessing	5	decode + denormalize
GRV branch (VLM inference, runs in parallel)		
VLM inference	89	concurrent with TAFR
JSON parsing + validation	2	runs after VLM returns
Critical path = $\max$*(TAFR branch, GRV branch)*		
Critical path (L=0)	**91**	GRV-bound (89 + 2 $>$ 52)
Critical path (L=1)	**102**	TAFR-bound (102 $>$ 91)
Critical path (L=3)	**114**	TAFR-bound (114 $>$ 91)

This parallelization introduces a one-cycle staleness in risk records: TAFR steers based on cached records from the previous GRV invocation while the current VLM assessment completes concurrently. In practice, this causes a slight reduction in detection accuracy for rapidly changing scenes but keeps the critical-path latency close to that of the unguided backbone.

H.5 VLM Invocation Interval

We ablate the VLM invocation interval N (§3.1), which controls how frequently GRV queries the VLM for updated risk records. Table 11 reports both safety performance and per-replanning-step critical-path latency on VLA-Arena with the $\pi_{0.5}$ backbone (L=1).

Table 11: Effect of VLM invocation interval N on $\pi_{0.5}$. Latency is per-replanning-step critical-path time. The default N=4 (bold) balances latency and safety.

N	SR ↑	CC ↓	Latency (ms) ↓
1	**49.8**	**7.31**	193
2	<u>49.5</u>	<u>7.40</u>	148
4	49.2	7.53	<u>114</u>
8	48.9	7.68	**108**

Reducing N from 4 to 1 (querying the VLM at every replanning step) improves SR by only 0.6 pp and CC by 0.22, because all risk objects in our benchmarks are visible from the first observation—more frequent VLM calls do not change *which* risks are detected but only update severity scores (g_i, p_i) as the arm moves. By contrast, latency increases sharply: at N=1 the VLM branch (91 ms) runs at every step rather than being amortized, raising the critical-path latency from 114 ms to 193 ms—a 69% increase. At N=8 latency drops further but safety degrades slightly as cached severity scores become stale. The default N=4 offers the best trade-off, paying $<$0.6 pp in safety for a $\sim$41% latency reduction relative to N=1.

H.6 Behavior Under Low Baseline Success Rates

Tables 1 and 6 show that MARS can produce small performance degradations when the backbone's unguided success rate is low. On SafeLIBERO, StarVLA-PI + MARS slightly underperforms the StarVLA-PI baseline (SR 28.63% vs. 28.75%, SSR 8.63% vs. 8.81%, CR 83.69% vs. 83.31%), and SmolVLA + MARS similarly trails SmolVLA (SR 6.63% vs. 7.06%, SSR 2.06% vs. 2.13%, CR 54.25% vs. 53.94%). On VLA-Arena, StarVLA-GR00T + MARS raises aggregate SR from 25.0% to 26.9% but incurs a marginal CC increase (16.93→17.26), showing that a net task-success gain can still carry a small safety-cost penalty on this backbone.

We attribute this pattern to three interacting factors. First, TAFR derives its correction from the velocity residual between nominal and counterfactual conditioning (Eq. 3). When the backbone cannot reliably ground even single-objective prompts—as evidenced by low task completion—its velocity fields under nominal and counterfactual instructions are nearly indistinguishable. The resulting residual carries minimal safety-relevant signal and instead reflects stochastic noise in the flow-matching integration. Second, the bounded correction (Eq. 14) still displaces the final action chunk endpoint, even when the displacement direction is uninformative. For a backbone already operating near chance level, any perturbation to the action trajectory is more likely to disrupt the few successful trajectories than to convert failures into successes. Third, the additional computational overhead of counterfactual forward passes slightly increases replanning latency, which can cause marginal timing misalignment in time-sensitive manipulation sequences.

These observations suggest a practical guideline: MARS is most effective when the backbone achieves sufficient unguided task completion to produce discriminative velocity fields under different language conditioning. Below a backbone-dependent competence threshold, the counterfactual residual degenerates into noise, and the net effect of TAFR shifts from corrective to perturbative. In deployment, this can be addressed by monitoring the magnitude of the velocity residual and disabling TAFR when the residual norm falls below a confidence threshold, effectively falling back to the unguided backbone.

I TAFR Correction Bound

We derive a worst-case bound on the per-coordinate correction magnitude $|\delta_{K,h,a}|$ from the TAFR update rules (Eqs. 7–10). Telescoping Eq. 10 with $\boldsymbol{\delta}_0 = \mathbf{0}$ gives

$$\delta_{K,h,a} = -\frac{1}{K} \sum_{k=0}^{K-1} q_k \, G_{k,h,a}. \tag{12}$$

From Eq. 8, each component satisfies $|G_{k,h,a}| \leq W_{h,a} \sum_{i \in \mathcal{I}_t} s_i \, |R^{\text{clip}}_{i,k,h,a}|$. Clipping (Eq. 4) gives $|R^{\text{clip}}_{i,k,h,a}| \leq b_a$, and each severity satisfies $s_i \leq 1$ with at most $L_{\max}$ retained risks, so

$$|G_{k,h,a}| \leq W_{h,a} \cdot L_{\max} \cdot b_a. \tag{13}$$

Since $W_{h,a} \leq w_a$ and $q_k \leq 1$, the per-coordinate correction is bounded by

$$|\delta_{K,h,a}| \;\leq\; \bar{q} \cdot w_a \cdot L_{\max} \cdot b_a \;\leq\; w_a \cdot L_{\max} \cdot b_a, \tag{14}$$

where $\bar{q} = \frac{1}{K} \sum_{k=0}^{K-1} \left(\frac{k+1}{K}\right)^{\alpha} \to \frac{1}{\alpha+1}$ as $K \to \infty$. For the default α=2, the temporal schedule alone reduces the effective bound by a factor of $\frac{1}{3}$; the action-position ramp $(h/H)^{\gamma_{\text{eff}}}$ further attenuates early chunk positions.

The bound depends only on user-specified hyperparameters (w_a, b_a, $L_{\max}$, α) and is independent of the VLA's velocity field magnitude or the number of ODE steps. It bounds deviation in the backbone's normalized action coordinates; the physical displacement after denormalization depends on the backbone's action scale.

J Corrected-State Integration Variant

TAFR evaluates all velocities at the anchor state $\mathbf{x}_k$ (Eq. 3). A natural alternative evaluates at the corrected state $\mathbf{x}_k + \boldsymbol{\delta}_k$:

$$\hat{\mathbf{v}}^{+}_{k} = v_\theta(\mathbf{x}_k + \boldsymbol{\delta}_k,\ \tau_k;\ o_t, c^{+}_t), \tag{15}$$

$$\hat{\mathbf{v}}^{-}_{i,k} = v_\theta(\mathbf{x}_k + \boldsymbol{\delta}_k,\ \tau_k;\ o_t, c^{-}_{t,i}), \tag{16}$$

with residuals $\hat{\mathbf{R}}_{i,k} = \hat{\mathbf{v}}^{+}_{k} - \hat{\mathbf{v}}^{-}_{i,k}$ and updates

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta t \; \hat{\mathbf{v}}^{+}_{k}, \tag{17}$$

$$\boldsymbol{\delta}_{k+1} = \boldsymbol{\delta}_k + \Delta t \; q_k \; \hat{\mathbf{G}}_k. \tag{18}$$

Here the accumulated correction $\boldsymbol{\delta}_k$ shifts the evaluation point at step $k{+}1$, which changes $\hat{\mathbf{v}}^+_{k+1}$ and all $\hat{\mathbf{v}}^-_{i,k+1}$. This creates a feedback loop: a guidance perturbation at step k alters every subsequent velocity evaluation, and the resulting errors compound across integration steps. The per-coordinate bound of Eq. 14 no longer holds because the residuals themselves depend on prior corrections. MARS uses the anchored variant (Eqs. 9–10) to eliminate this feedback; the empirical comparison in §4.3 confirms that anchoring improves both safety and task completion.

K Limitations and Future Work

We identify two principal limitations of the current MARS framework and outline corresponding directions for future work.

Fixed guidance schedule across heterogeneous tasks. MARS uses a single set of guidance hyperparameters (b_a, w_a, γ, α) across all tasks and safety constraint types. The w_a ablation (Table 9) and b_a ablation (Table 8) both show a non-monotonic trade-off: tasks requiring spatial detours (e.g., obstacle avoidance) tolerate stronger guidance, while tasks requiring precise grasp-region adjustments (e.g., cautious grasp) degrade under the same settings. Because w_a and b_a jointly determine the effective correction magnitude (Eq. 14), a fixed global configuration cannot simultaneously satisfy both regimes. Future work could introduce *adaptive guidance scheduling*, where the guidance weight is modulated online based on the constraint type inferred by GRV (e.g., spatial detour vs. affordance selection), the magnitude of the velocity residual at each ODE step, or the proximity and severity of the detected risk. One concrete instantiation would learn a lightweight mapping from the GRV risk record $(g_i, p_i, \text{type}_i)$ to per-risk guidance parameters, using the existing velocity-residual magnitude as a self-supervised signal—large residuals indicate that the backbone distinguishes the two prompts and can tolerate stronger guidance, while small residuals suggest the correction is uninformative and should be attenuated.

Ineffectiveness on low-capability backbones. MARS's correction signal depends entirely on the VLA backbone's ability to produce distinguishable velocity fields under different language conditioning. As shown in the component analysis (§4.3) and the low-SR analysis (§H.6), backbones with low repulsion completion rates (e.g., SmolVLA at 18%) generate nearly identical velocity fields for nominal and counterfactual prompts. In this regime, the residuals carry noise rather than directional safety signal, and the bounded correction perturbs the trajectory in arbitrary directions—making MARS not merely ineffective but counterproductive. This is a fundamental limitation of the conditional-residual approach: it cannot create language-conditioned action discrimination that the backbone does not already possess. Practical deployment should monitor the velocity-residual norm and disable TAFR when it falls below a confidence threshold, falling back to the unguided backbone. A more ambitious direction would combine MARS with lightweight backbone adaptation (e.g., LoRA fine-tuning on a small set of safety-relevant demonstrations) to improve counterfactual grounding before applying inference-time steering, or explore alternative repulsion signals that do not require the backbone to distinguish prompts, such as visual feature differences between nominal and risk-object-masked observations.