BEYOND POSITIVE BIAS: CUSTOMIZING AND CORRECTING GENERATIVE ROBOT POLICIES VIA NEGATIVE GUIDANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion policies can represent multimodal robot manipulation behaviors, but deployment may introduce constraints that were not present during policy training. Extending negative guidance to robot control is nontrivial: guided action chunks are repeatedly executed in a closed loop, where excessive repulsion can compound over time, reduce task completion, or collapse the remaining valid behaviors. We extend Dynamic Negative Guidance to diffusion robot policies and develop a multi-target formulation for simultaneously avoiding multiple undesirable outcome modes. We instantiate the framework through two complementary applications. First, language-conditioned negative guidance provides a semantic interface for specifying single or multiple outcomes to avoid. Second, when undesirable behavior is identified through deployment feedback rather than language, an auxiliary diffusion policy trained on a small set of failure rollouts provides the negative guidance signal while the base-policy parameters remain fixed. We evaluate the framework across object picking, target placement, planar pushing, and precise charger insertion, including a real-robot manipulation experiment. As an additional generalization study, we further evaluate language-conditioned negative guidance with π_0 flow-matching policies on LIBERO. Results show that dynamic negative guidance suppresses specified undesirable outcomes while better preserving task completion and alternative valid behaviors than static negative guidance, and that failure-policy guidance improves task success across the evaluated deployment-time environment changes.

1 INTRODUCTION

Diffusion policies have emerged as a powerful paradigm for robot manipulation Chi et al. (2025); Qi et al. (2025b;a); Li et al. (2024); Cao et al. (2025). By modeling action generation through iterative denoising, they can capture complex, multimodal behavior distributions with multiple valid ways to complete a task. Deployment, however, often introduces constraints absent during training. A robot may need to avoid a newly placed obstacle, bypass an unavailable insertion port, or suppress behaviors that fail in a modified environment.

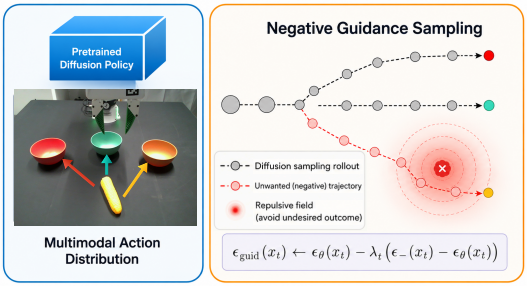


Figure 1: Steering diffusion policies away from undesirable outcomes.

Negative guidance provides a natural mechanism for expressing such constraints, but extending it to closed-loop robot control is nontrivial. Unlike one-shot generation, diffusion policies repeatedly generate and execute action chunks, so guided actions alter the observations and decisions encountered later in the rollout. Excessive repulsion can therefore produce infeasible actions, compound errors, or increase task failures. Moreover, suppressing an undesirable outcome is not enough: the policy should still complete the task and preserve useful alternatives. These challenges motivate our study of Dynamic Negative Guidance (DNG) Koulischer et al. (2025) for diffusion robot policies.

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.

Existing approaches to deployment-time constraints often update the policy using newly collected interaction data Sun et al. (2020); Nagabandi et al. (2018); Hansen et al. (2021). Instead, we ask whether undesirable outcomes can be incorporated directly into the sampling process while keeping the base policy fixed. Building on DNG, we develop a framework for steering diffusion policies away from undesirable behaviors and extend it to multiple simultaneous forbidden outcomes. Our goal is *empirical behavioral avoidance* rather than formal safety: the method does not provide hard safety guarantees and should be viewed as an inference-time behavioral steering mechanism rather than a strict safety filter. We instantiate the framework in two complementary settings:

(1) Language-Conditioned Avoidance. When undesirable outcomes can be specified semantically, language provides a direct interface for describing what the robot should avoid. We train with both broad and target-specific instructions, such as "plug the charger into any receptacle" and "plug the charger into the first receptacle," but no exclusion instructions. At inference time, target-specific conditions are repurposed as negative guidance directions, allowing the policy to suppress one or multiple outcomes. For example, the same learned conditions can steer the policy toward any receptacle except the first, second, and fourth. We further evaluate this principle with π_0 flow-matching policies on LIBERO.

(2) Failure-Driven Avoidance. When undesirable behavior is discovered through deployment rather than language, we collect a small set of failure rollouts and train an auxiliary diffusion policy to model the failure distribution. During inference, this auxiliary policy provides the negative guidance signal used to steer the frozen base policy away from observed failure modes, without modifying its parameters.

We establish that DNG can be made effective in receding-horizon closed-loop robot control, derive its joint-complement form for mutually exclusive behavioral modes, and show that the negative score can be instantiated either semantically or from deployment failures. We evaluate the framework across multi-object picking, target placement, planar pushing with trajectory constraints, and precise charger insertion, including real-robot manipulation. The evaluation covers single- and multi-target language constraints, obstacle-induced failures, and changes in receptacle availability. To characterize language-conditioned avoidance, we introduce the *Avoidance-Uniformity Score* (AUS), which prioritizes successful avoidance while secondarily rewarding diversity among the remaining valid outcomes.

Across these settings, dynamic negative guidance suppresses specified outcomes while better preserving task completion and alternative behaviors than static negative guidance. Static guidance can achieve strong nominal suppression only at the cost of substantially higher failure rates, whereas dynamic guidance provides a more favorable avoidance–completion trade-off. Failure-policy guidance further improves task performance under unseen deployment constraints. Overall, our results establish negative guidance as a practical mechanism for inference-time steering of fixed diffusion robot policies, including simultaneous avoidance of multiple outcome modes through either semantic language constraints or observed deployment failures.

2 Related Work

2.1 Diffusion Policies and Inference-Time Guidance

Diffusion policies model multimodal robot behaviors through iterative denoising Chi et al. (2025); Qi et al. (2025c;b), while language conditioning enables flexible behavior specification Brohan et al. (2022); Huang et al. (2025); Li et al. (2025); Qi et al. (2025a). Recent work steers generative robot policies at inference time using learned objectives, predictive models, or guidance mechanisms Wagenmaker et al. (2025); Du & Song (2026); Wang et al. (2026); Singhal et al. (2025). Diffuser Janner et al. (2022) similarly uses guided diffusion for trajectory optimization, generating state-action trajectories toward high-reward behavior.

Related work in image generation suppresses undesirable concepts through inference-time safety guidance Schramowski et al. (2023) or parameter-level concept erasure Gandikota et al. (2023). Our work instead studies *exclusion* in closed-loop control, where guided actions affect future observations and avoidance must preserve task completion and alternative behaviors. DynaGuide Du & Song (2026) uses reference observations and latent dynamics, while PPGuide Wang et al. (2026)

uses an auxiliary performance predictor. Rejection-sampling approaches provide another inference-time alternative by generating multiple candidate action chunks and selecting among them using a learned value or outcome model Park et al. (2026). We build on Dynamic Negative Guidance Koulischer et al. (2025), adapting posterior-weighted repulsion to closed-loop manipulation and multiple forbidden outcomes. We instantiate this guidance using semantic language conditions or an auxiliary diffusion model learned from failure rollouts, and additionally evaluate it with π_0 flow-matching policies on LIBERO.

2.2 Deployment-Time Adaptation and Constraint Avoidance

Deployment shifts can introduce constraints absent during training Zhang et al. (2026); Shi et al. (2026); Wei et al. (2025). Test-time training and online adaptation address such changes by updating policy parameters Sun et al. (2020); Hansen et al. (2020), whereas inference-time guidance modifies action generation while keeping the base policy fixed.

Constraint avoidance has also been studied through safe reinforcement learning, control barrier functions, planning, negative demonstrations, and learned rewards Garcıa & Fernández (2015); Dalal et al. (2018); Cheng et al. (2019); Ebert et al. (2018); Brown et al. (2019). These methods often require explicit constraints, reward design, negative data, or additional optimization. In contrast, our framework expresses exclusions directly during sampling, using either semantic language conditions or an auxiliary model of observed failures. The method provides empirical avoidance rather than formal safety guarantees.

3 Method

We develop an inference-time framework for steering a fixed diffusion policy away from undesirable outcomes. We first review single-target Dynamic Negative Guidance (DNG) Koulischer et al. (2025), then extend it to multiple simultaneous avoidance constraints. We instantiate the framework using language-specified outcomes or an auxiliary diffusion policy trained on deployment failures, and additionally extend the language-conditioned formulation to flow-matching policies.

3.1 Dynamic Negative Guidance

Let $s_\theta(x_t)$ denote the base diffusion score and $s_{\theta,c}(x_t)$ the score conditioned on c. Standard Classifier-Free Guidance (CFG) Ho & Salimans (2022) steers toward a desired condition c_+ using

$$s_{\mathrm{CFG}}(x_t) = s_\theta(x_t) + \lambda \left(s_{\theta,c_+}(x_t) - s_\theta(x_t) \right), \tag{1}$$

where λ is a fixed guidance scale. Negative guidance reverses this direction to suppress an undesirable condition c_-. A fixed repulsive scale, however, can unnecessarily perturb samples that have already moved away from the undesirable mode.

DNG instead targets the complement distribution

$$p_t(x_t \mid \neg c_-) \propto p_t(x_t) \left(1 - p_t(c_- \mid x_t)\right), \tag{2}$$

which gives

$$s_{\mathrm{guid}}(x_t) = s_\theta(x_t) - \lambda_0 \frac{p_t(c_- \mid x_t)}{1 - p_t(c_- \mid x_t)} \left(s_{\theta,c_-}(x_t) - s_\theta(x_t) \right). \tag{3}$$

Thus, repulsion is stronger when the current sample is likely to correspond to the undesirable condition and weakens as it moves away. We estimate $p_t(c_- \mid x_t)$ iteratively during reverse diffusion following Koulischer et al. (2025).

3.2 Multi-Target Negative Guidance

Deployment may require avoiding multiple mutually exclusive outcomes $\mathcal{C} = \{c_1, \ldots, c_K\}$. Their joint complement distribution is

$$p_t(x_t \mid \neg \mathcal{C}) \propto p_t(x_t) \left(1 - \sum_{j=1}^{K} p_t(c_j \mid x_t) \right). \tag{4}$$

Algorithm 1 Multi-Target Dynamic Negative Guidance

Require: ϵ_θ, $\{\epsilon_{\theta,c_i}\}_{i=1}^K$, $\{\lambda_0^{(i)}, p_0^{(i)}\}_{i=1}^K$, τ, δ, ε_Z

Sample $x_T \sim \mathcal{N}(0, I)$
$p_T^{(i)} \leftarrow p_0^{(i)}$ for all i
$Z_T \leftarrow 1 - \sum_j p_T^{(j)}, \quad \lambda_T^{(i)} \leftarrow \lambda_0^{(i)} p_T^{(i)} / Z_T$ for all i
for $t = T, \dots, 1$ **do**
  $\epsilon_{\text{guid}} \leftarrow \epsilon_\theta - \sum_i \lambda_t^{(i)} (\epsilon_{\theta,c_i} - \epsilon_\theta)$
  $x_{t-1} \leftarrow \text{DDPMStep}(x_t, \epsilon_{\text{guid}})$
  $\beta_t \leftarrow 1 - \alpha_t, \quad \mu_t \leftarrow \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t) \right)$
  for $i = 1, \dots, K$ **do**
    $\mu_{c_i} \leftarrow \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_{\theta,c_i}(x_t) \right)$
    $E_i \leftarrow -\frac{\tau}{2\beta_t} \left(\|x_{t-1} - \mu_{c_i}\|^2 - \|x_{t-1} - \mu_t\|^2 \right) + \frac{\delta}{2\beta_t}$
    $\widetilde{p}_{t-1}^{(i)} \leftarrow \text{Clip}\left(p_t^{(i)} e^{E_i}, p_{\min}, p_{\max} \right)$
  end for
  $\rho \leftarrow \min\left(1, \frac{1-\varepsilon_Z}{\sum_j \widetilde{p}_{t-1}^{(j)}} \right)$
  $p_{t-1}^{(i)} \leftarrow \rho \widetilde{p}_{t-1}^{(i)}$ for all i
  $Z_{t-1} \leftarrow 1 - \sum_j p_{t-1}^{(j)}, \quad \lambda_{t-1}^{(i)} \leftarrow \lambda_0^{(i)} \frac{p_{t-1}^{(i)}}{Z_{t-1}}$ for all i
end for
return x_0

Taking its score yields

$$s_{\text{guid}}(x_t) = s_\theta(x_t) - \sum_{i=1}^{K} \lambda_t^{(i)}(x_t) \left(s_{\theta,c_i}(x_t) - s_\theta(x_t) \right), \tag{5}$$

where

$$\lambda_t^{(i)}(x_t) = \lambda_0^{(i)} \frac{p_t(c_i \mid x_t)}{1 - \sum_{j=1}^{K} p_t(c_j \mid x_t)}. \tag{6}$$

The shared denominator accounts for the total forbidden posterior mass; setting $\lambda_0^{(i)} = 1$ for all i recovers the joint complement score.

Because independently updated posterior estimates may sum to more than one, we jointly normalize the provisional estimates $\{\widetilde{p}_t^{(i)}\}_{i=1}^K$ as

$$\rho_t = \min\left(1, \frac{1 - \varepsilon_Z}{\sum_{j=1}^{K} \widetilde{p}_t^{(j)}} \right), \qquad p_t(c_i \mid x_t) = \rho_t \widetilde{p}_t^{(i)}, \tag{7}$$

where $\varepsilon_Z > 0$ ensures that the denominator in Eq. 6 remains positive.

Here, $\beta_t = 1 - \alpha_t$, $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$, σ_t^2 is the reverse variance used by the DDPM sampler, $\tau > 0$ scales the likelihood-ratio update, δ is a calibration offset, and ε_Z lower-bounds the remaining posterior mass.

3.3 Negative Guidance for Robot Manipulation

We instantiate the framework using two complementary representations of undesirable behavior. In both cases, the deployed base policy remains fixed.

Language-Conditioned Avoidance. Language provides a direct interface for specifying outcomes to avoid. The policy is trained with broad instructions, such as "pick any object," and target-specific instructions, such as "pick the apple," but does not require exclusion instructions during training.

At inference time, target-specific conditions are repurposed as negative directions. For diffusion policies,

$$s_{\theta,c_i}(x_t) \propto -\epsilon_\theta(x_t, c_i). \tag{8}$$

A single target uses Eq. 3, while multiple targets are combined using Eq. 5. Thus, behaviors learned from positive language instructions can be suppressed at inference time without negative demonstrations or exclusion training.

Flow-Matching Extension. We additionally apply language-conditioned negative guidance to the flow-matching formulation used by π_0. Let p_t^b denote the path marginal under a broad instruction and suppose it decomposes into undesirable and remaining behaviors. When the conditional distributions use the same flow interpolant, their probability fluxes satisfy

$$p_t^b v_b = \pi_- p_t^- v_{c_-} + (1 - \pi_-) p_t^{\neg -} v_{\neg -}. \tag{9}$$

Defining

$$r_t(x_t) = p_t(c_- \mid x_t) = \frac{\pi_- p_t^-(x_t)}{p_t^b(x_t)}, \tag{10}$$

the exact complement velocity is obtained by weighting $v_{c_-} - v_b$ by $r_t/(1 - r_t)$. Because this posterior is not directly available from a pretrained flow policy, our implementation replaces it with a bounded adaptive weight ρ_t and uses

$$v_{\text{guid}}(x_t, t) = v_b(x_t, t) - \lambda_0 \frac{\rho_t}{\max(1 - \rho_t, Z_{\min})} \left(v_{c_-}(x_t, t) - v_b(x_t, t) \right). \tag{11}$$

After each Euler step, ρ_t is updated using a local compatibility ratio between the guided proposal and the broad and negative flow proposals. This produces a self-attenuating guidance schedule without interpreting the deterministic flow step as a DDPM reverse transition. Multiple forbidden outcomes share the remaining-weight denominator $1 - \sum_j \rho_t^{(j)}$; details are provided in Appendix 7.1.

Failure-Driven Avoidance. When undesirable behavior is identified through deployment rather than language, we collect failure rollouts and train an auxiliary diffusion policy to model them. Its prediction $\epsilon_{\text{fail}}(x_t)$ defines the negative score

$$s_{\text{fail}}(x_t) \propto -\epsilon_{\text{fail}}(x_t). \tag{12}$$

We substitute this score for the language-conditioned negative branch in the same dynamic guidance update. The auxiliary policy thereby converts observed failures into a repulsive sampling signal without modifying the base policy or requiring a semantic description of the failure.

4 Results

4.1 Language-Conditioned Negative Guidance

Tasks. We evaluate language-conditioned negative guidance across three simulated task categories: multi-object picking, Push-T with trajectory constraints, and charger insertion. We additionally evaluate the approach on a real-world bowl placement task. As a generalization study beyond the diffusion-policy architecture used in our main experiments, we further evaluate language-conditioned negative guidance with π_0 on LIBERO-Goal.

For each task, we train a language-conditioned base policy using both broad instructions (e.g., *pick any object*) and specific instructions (e.g., *pick the cube*). At inference time, we apply negative guidance to suppress one or multiple specified outcomes, such as *pick any object except the cube* or *plug the charger into any receptacle except 1, 2, and 4*. The avoidance constraints are incorporated through the sampling process while the base-policy parameters remain fixed.

Why language-conditioned negative guidance. Language provides a direct interface for specifying deployment-time exclusion constraints. In many settings, a user can identify an outcome that should be avoided without enumerating all acceptable alternatives. Negative guidance uses this information to modify the policy distribution through target-specific conditions already represented by the base policy.

We compare dynamic negative guidance with a static Classifier-Free-Guidance-style baseline (CFG) Ho & Salimans (2022), which applies a fixed repulsive weight throughout denoising. We also include rejection sampling as an alternative that filters sampled candidates rather than modifying the denoising update. The evaluation therefore considers three aspects jointly: suppression of the specified outcomes, successful task completion, and diversity among the remaining valid behaviors.

Evaluation. We evaluate whether guidance satisfies specified avoidance constraints while preserving task performance and, when multiple valid outcomes remain, their diversity. For four-object picking and charger insertion, we conduct 200 trials per constraint and report the frequency of each outcome and task failure. We additionally report the *Avoidance-Uniformity Score* (AUS), which primarily measures successful task completion under the avoidance constraint and secondarily rewards balanced selection among the remaining valid outcomes; its formal definition is provided in Appendix 7.5. For Push-T, we block one of three possible trajectories with an obstacle, specify the trajectory to avoid through language, and report the maximum task score based on the overlap between the final T-block configuration and the target. For the π_0 LIBERO-Goal experiments, we report the number of rollouts corresponding to the specified avoidance category together with the number of task failures, and compare dynamic guidance with the unguided general-language policy and static CFG across guidance scales.

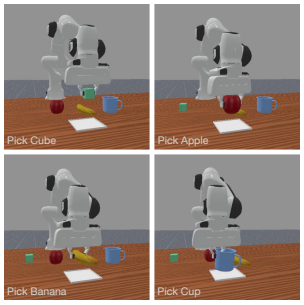


Figure 2: Four-object picking with language-conditioned avoidance of specified target objects.

Method	Avoid	Cube	Apple	Banana	Cup	Fail	AUS
Baseline	None	28.5%	20.5%	16.5%	22.5%	12.0%	–
Ours	Cube	4.0%	11.0%	13.0%	62.0%	10.0%	**0.710**
CFG		2.5%	5.5%	1.5%	83.0%	7.5%	0.662
Rejection Sampling		25.5%	5.0%	19.0%	27.5%	23.0%	0.460
Ours	Apple	9.5%	8.5%	10.5%	61.0%	10.5%	**0.657**
CFG		5.0%	4.5%	1.5%	81.5%	7.5%	0.645
Rejection Sampling		7.0%	22.0%	19.5%	28.5%	23.0%	0.499
Ours	Banana	37.0%	20.0%	1.5%	32.5%	9.0%	**0.851**
CFG		60.0%	6.0%	1.0%	27.0%	6.0%	0.800
Rejection Sampling		27.0%	23.5%	3.0%	27.5%	19.0%	0.769
Ours	Cup	50.5%	29.0%	3.5%	4.5%	12.5%	**0.721**
CFG		78.0%	10.5%	1.0%	3.5%	7.0%	0.678
Rejection Sampling		30.5%	21.0%	19.5%	4.0%	25.0%	0.679

Table 1: Four-object picking under target-avoidance constraints. Columns report outcome frequencies over 200 trials. Bold denotes the highest AUS for each avoidance setting.

Four-object picking. As shown in Fig. 2 and Table 1, our method achieves the highest AUS for all four avoidance targets. Static CFG strongly suppresses the forbidden object but often concentrates the remaining probability mass on a single alternative. For example, when avoiding the cube, CFG selects the cup in 83.0% of all trials. Dynamic negative guidance instead retains probability mass across multiple valid objects while maintaining low forbidden-target and failure rates. Rejection sampling preserves a relatively balanced distribution in some settings, but its higher failure and residual forbidden-target rates reduce its AUS.

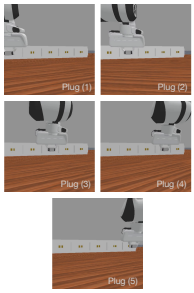


Figure 3: Charger insertion with simultaneous avoidance constraints.

Method	Avoid	(1)	(2)	(3)	(4)	(5)	Fail	AUS
Baseline	None	21.0%	13.5%	19.0%	14.5%	15.5%	16.5%	–
Ours	(1, 2, 3)	5.0%	0.0%	4.5%	17.5%	53.5%	19.5%	**0.602**
CFG		4.5%	1.0%	0.0%	2.5%	68.0%	24.0%	0.509
Rejection Sampling		3.0%	2.5%	7.0%	23.0%	18.5%	46.0%	0.402
Ours	(1, 4, 5)	17.0%	20.5%	35.0%	3.0%	6.5%	18.0%	**0.512**
CFG		46.5%	5.5%	18.0%	5.0%	8.5%	16.5%	0.198
Rejection Sampling		5.0%	16.5%	23.5%	5.5%	8.5%	41.0%	0.379
Ours	(2, 3, 5)	30.5%	2.5%	7.5%	25.0%	15.5%	19.0%	**0.539**
CFG		31.5%	1.0%	9.0%	7.5%	26.5%	24.5%	0.318
Rejection Sampling		16.5%	7.5%	8.0%	15.5%	7.0%	45.5%	0.317

Table 2: Charger-insertion results for representative multi-target avoidance constraints. Bold denotes the highest AUS for each constraint. Results for all ten constraint combinations are reported in Appendix 7.11.

Charger insertion. Figure 3 and Table 2 evaluate the simultaneous avoidance of three receptacles. This setting tests whether guidance can suppress several forbidden modes without discarding the two remaining valid alternatives. Our method achieves the highest AUS in eight of the ten configurations and ties CFG in one additional configuration.

Rejection sampling frequently suppresses the specified receptacles, but its failure rate ranges from 41.0% to 54.5%. These failures do not constitute successful avoidance and therefore substantially reduce its AUS. Overall, dynamic negative guidance suppresses multiple specified outcomes simultaneously while retaining a larger fraction of successful behavior from the base policy.

Avoid	Baseline	Rejection Sampling	Scale 10		Scale 20		Scale 30		Scale 40		Scale 100	
			Ours	CFG	Ours	CFG	Ours	CFG	Ours	CFG	Ours	CFG
Middle	0.5776	0.7000	0.8433	0.7666	0.8029	0.7303	**0.8475**	0.6488	0.7543	0.5768	0.6090	0.2817
Left	0.7327	0.6150	0.9040	0.8735	**0.9257**	0.8063	0.8631	0.7045	0.8873	0.6181	0.8023	0.3666

Table 3: Push-T performance across avoidance targets and guidance scales. Higher is better; bold indicates the best score per target.

Push-T. During training, the T-block can reach the target through left, middle, or right trajectories. At evaluation time, an obstacle blocks one trajectory, and contact with the obstacle causes the task to fail. We specify the blocked trajectory through language, such as *avoid pushing from the left*.

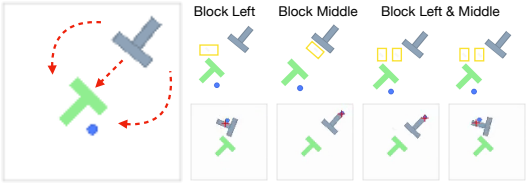


Figure 4: Push-T manipulation with obstacle-aware trajectory avoidance.

As shown in Fig. 4 and Table 3, dynamic negative guidance improves over the unguided base policy under both tested constraints. Moderate guidance scales provide the strongest performance, whereas overly large scales degrade task completion. Dynamic guidance achieves the highest score for both constraints, reaching 0.8475 when avoiding the middle trajectory and 0.9257 when avoiding the left trajectory.

Generalization to π_0 on LIBERO-Goal. We further evaluate whether language-conditioned negative guidance transfers to π_0, which uses flow matching rather than diffusion-based action generation. On LIBERO-Goal, we use *"operate on the table"* as the broad instruction and task descriptions as language-specified outcomes to avoid. We evaluate both single- and multi-target avoidance against the unguided general-language policy and static CFG, with 100 evaluation trials for each group of avoidance target experiment.



Figure 6: Real-world bowl-placement setup.

Table 4 shows that static CFG often nearly eliminates the specified outcome, but at the cost of a large increase in task failures. Dynamic guidance provides a substantially better avoidance–completion trade-off at moderate scales. For example, when avoiding *put the wine bottle on top of the cabinet* at scale 3, dynamic guidance reduces the specified-outcome count from 10 to 3 while maintaining the baseline failure count of 18; CFG reduces the outcome count to 0 but increases failures to 98.

The same trend holds for multi-target avoidance. At scale 3, dynamic guidance reduces the combined specified-outcome count from 25 to 6 while retaining the baseline failure count of 18, whereas CFG reduces the

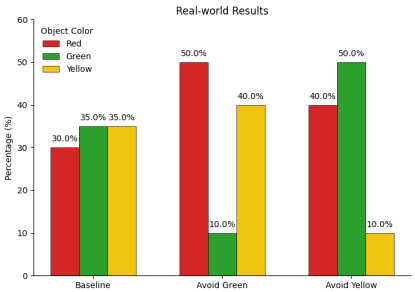


Figure 5: Outcome frequencies under language-conditioned avoidance.

Avoidance target	Count	General	Scale 1		Scale 1.5		Scale 3		Scale 5		Scale 10		Scale 20	
		Baseline	Ours	CFG	Ours	CFG	Ours	CFG	Ours	CFG	Ours	CFG	Ours	CFG
Single-target avoidance														
A1	Avoided	9	6	0	3	0	2	2	2	0	1	0	0	0
	Failure	18	17	89	24	95	18	94	20	96	24	98	40	97
A2	Avoided	10	7	1	9	0	3	0	7	0	1	0	3	0
	Failure	18	16	84	16	90	18	98	18	98	19	97	41	90
A3	Avoided	27	19	2	19	3	11	0	14	0	6	1	5	0
	Failure	18	22	95	22	96	26	99	18	100	33	99	68	100
Multi-target avoidance														
A4 + A2 + A5	Avoided	25	14	0	14	0	6	0	3	0	0	0	0	0
	Failure	18	19	96	17	92	18	96	26	92	49	98	82	94
A6 + A2 + A7	Avoided	24	14	0	14	0	7	0	2	0	1	0	1	0
	Failure	18	15	100	17	99	22	99	41	98	79	95	90	86

Table 4: Avoided-category and failure counts on LIBERO-Goal for the general-language baseline, dynamic negative guidance (Ours), and static CFG. Actions are abbreviated as: A1 = put the bowl on the stove; A2 = put the wine bottle on top of the cabinet; A3 = turn on the stove; A4 = open the middle drawer; A5 = open the top drawer and put the bowl inside; A6 = put the bowl on top of the cabinet; and A7 = put the wine bottle on the rack. For multi-target settings, *Avoided* sums all specified categories. Lower is better.

specified outcomes to 0 but increases failures to 96. These results indicate that the language-conditioned negative-guidance principle also transfers to π_0 flow matching while better preserving task completion than fixed negative guidance.

Real-world experiments. We evaluate language-conditioned negative guidance on a 7-DoF Franka Emika Panda equipped with an Intel RealSense RGB camera. The robot places an object into one of three colored bowls (Fig. 6). We collect 50 teleoperated demonstrations per task mode and train the same language-conditioned diffusion-policy architecture used in simulation.

We conduct 20 trials for each of three instructions: place into any bowl (Baseline), avoid the green bowl, and avoid the yellow bowl. All 60 real-world trials successfully placed the object into one of the bowls. As shown in Fig. 5, guidance reduces selection of either forbidden bowl to 10.0% and shifts successful outcomes toward the allowed bowls. Thus, the same sampling-time mechanism steers real-robot target selection without modifying the base policy.

4.2 Failure-Driven Negative Guidance

Protocol. We study deployment changes that make previously valid behaviors unsuccessful. The base policy remains frozen while failure rollouts collected in the modified environment are used to train an auxiliary diffusion policy. During inference, the auxiliary policy provides the negative score that steers the base policy away from the observed failure distribution. We compare against the unguided base policy, static negative CFG, and PPGuide Wang et al. (2026). PPGuide learns a discriminative outcome predictor and differentiates through it during sampling, whereas our method directly models the distribution of failure trajectories.

Push-T: obstacle-induced failures. We first introduce obstacles unseen during base-policy training that block the middle trajectory, the left trajectory, or both. We collect failure rollouts under each configuration and use the resulting auxiliary policy to guide the frozen base policy.

As shown in Table 5, our method improves over the unguided policy and PPGuide in all three configurations. Relative to the unguided baseline, the best score increases from 0.4957 to 0.7745 when the middle trajectory is blocked, from 0.6821 to 0.8442 when the left trajectory is blocked, and from 0.3083 to 0.6472 when both are blocked. The best scale is 5 for the single-obstacle settings and 10 for the two-obstacle setting. Static CFG degrades rapidly as its guidance scale increases, whereas dynamic guidance remains effective across the evaluated range.

Avoid	Baseline	PPGuide	Scale 2		Scale 5		Scale 10		Scale 20	
			Ours	CFG	Ours	CFG	Ours	CFG	Ours	CFG
Middle	0.4957	0.5450	0.6461	0.4187	**0.7745**	0.1975	0.6977	0.1648	0.7356	0.0627
Left	0.6821	0.7250	0.7774	0.5521	**0.8442**	0.4521	0.7746	0.4499	0.8132	0.1807
Middle & Left	0.3083	0.3900	0.4709	0.3871	0.6208	0.1937	**0.6472**	0.1443	0.6311	0.1417

Table 5: Push-T task scores under obstacle-induced deployment failures. Bold denotes the highest score for each configuration.

Out-of-power	Baseline	PPGuide	Scale 2		Scale 5		Scale 10		Scale 20	
			Ours	CFG	Ours	CFG	Ours	CFG	Ours	CFG
Receptacle 1	62.5%	61.0%	69.5%	36.5%	72.5%	7.0%	72.5%	0.0%	**74.0%**	0.5%
Receptacle 2	70.0%	56.0%	74.0%	39.5%	73.0%	8.0%	73.0%	0.0%	**75.5%**	0.0%
Receptacles 2 & 3	51.0%	39.0%	63.0%	25.0%	58.0%	8.0%	64.5%	1.0%	**76.0%**	0.0%

Table 6: Charger-insertion success rates when the indicated receptacles are unavailable. Bold denotes the highest success rate for each condition.

Charger insertion: unavailable receptacles. We next consider changes in outcome validity rather than geometric obstructions. All receptacles are powered during base-policy training, but receptacle 1, receptacle 2, or both receptacles 2 and 3 become unavailable during deployment. We train the auxiliary policy from 20 observed failure rollouts, and define success as insertion into any receptacle that remains powered.

As shown in Table 6, our method improves over the unguided policy in all three conditions. Success increases from 62.5% to 74.0% when receptacle 1 is unavailable, from 70.0% to 75.5% when receptacle 2 is unavailable, and from 51.0% to 76.0% when receptacles 2 and 3 are simultaneously unavailable. PPGuide obtains 61.0%, 56.0%, and 39.0%, respectively, and does not improve over the unguided policy in these conditions. Dynamic guidance performs best at scale 20, while static CFG approaches zero success as the guidance scale increases.

Summary. Across both geometric obstacles and changes in outcome validity, failure-driven negative guidance improves the frozen base policy using only a small set of deployment failures.

5 Conclusion

We presented negative guidance as an inference-time mechanism for steering fixed diffusion robot policies away from undesirable outcomes. We extended Dynamic Negative Guidance to multiple simultaneous avoidance targets and studied two complementary settings: language-conditioned avoidance and failure-driven avoidance from observed deployment failures. Across simulated and real-robot manipulation tasks, dynamic guidance suppresses targeted outcomes while better preserving task completion and alternative behaviors than static guidance. We also show that the same language-conditioned principle transfers to PI0 flow-matching policies on LIBERO-Goal. Overall, negative guidance provides a practical approach to deployment-time customization without modifying the base policy.

6 Limitations

Language-conditioned avoidance is limited by the semantic grounding already present in the policy, while failure-driven guidance requires representative failure rollouts and an auxiliary model. Negative guidance reduces, but does not guarantee, undesirable outcomes and therefore does not provide formal safety guarantees. Our evaluation also focuses primarily on task-specific diffusion policies and a limited set of manipulation environments. Although the PI0 experiment provides initial evidence of transfer beyond diffusion policies, broader evaluation on generalist policies and real-world tasks remains future work.

References

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics

transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Daniel Brown, Wonjoon Goo, Prabhat Nagarajan, and Scott Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pp. 783–792. PMLR, 2019.

Yuhong Cao, Jeric Lew, Jingsong Liang, Jin Cheng, and Guillaume Sartoretti. Dare: Diffusion policy for autonomous robot exploration. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11987–11993. IEEE, 2025.

Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 3387–3395, 2019.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

Maximilian Du and Shuran Song. Dynaguide: Steering diffusion polices with active dynamic guidance. *Advances in Neural Information Processing Systems*, 38:44192–44221, 2026.

Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568*, 2018.

Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2426–2436, 2023.

Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

Nicklas Hansen, Rishabh Jangir, Yu Sun, Guillem Alenyà, Pieter Abbeel, Alexei A Efros, Lerrel Pinto, and Xiaolong Wang. Self-supervised policy adaptation during deployment. *arXiv preprint arXiv:2007.04309*, 2020.

Nicklas Hansen, Rishabh Jangir, Yu Sun, Guillem Alenyà, Pieter Abbeel, Alexei A. Efros, Lerrel Pinto, and Xiaolong Wang. Self-supervised policy adaptation during deployment. In *International Conference on Learning Representations*, 2021.

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

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

Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

Felix Koulischer, Johannes Deleu, Gabriel Raya, Thomas Demeester, and Luca Ambrogioni. Dynamic negative guidance of diffusion models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 34892–34921, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/5647763d4245b23e6a1cb0a8947b38c9-Paper-Conference.pdf.

Hang Li, Qian Feng, Zhi Zheng, Jianxiang Feng, Zhaopeng Chen, and Alois Knoll. Language-guided object-centric diffusion policy for generalizable and collision-aware manipulation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12834–12841. IEEE, 2025.

Yishu Li, Wen Hui Leng, Yiming Fang, Ben Eisner, and David Held. Flowbothd: History-aware diffuser handling ambiguities in articulated objects manipulation. *arXiv preprint arXiv:2410.07078*, 2024.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

Kwanyoung Park, Seohong Park, Youngwoon Lee, and Sergey Levine. Scalable offline model-based rl with action chunks. In *International Conference on Learning Representations*, volume 2026, pp. 9599–9623, 2026.

Han Qi, Changhe Chen, and Heng Yang. Compose by focus: Scene graph-based atomic skills. *arXiv preprint arXiv:2509.16053*, 2025a.

Han Qi, Haocheng Yin, Yilun Du, and Heng Yang. Strengthening generative robot policies through predictive world modeling. *arXiv e-prints*, pp. arXiv–2502, 2025b.

Han Qi, Haocheng Yin, and Heng Yang. Control-oriented clustering of visual latent representation. In *International Conference on Learning Representations*, volume 2025, pp. 45264–45296, 2025c.

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

Liangzhi Shi, Shuaihang Chen, Feng Gao, Yinuo Chen, Kang Chen, Tonghe Zhang, Hongzhi Zang, Weinan Zhang, Chao Yu, and Yu Wang. Beyond imitation: Reinforcement learning-based sim-real co-training for vla models. *arXiv preprint arXiv:2602.12628*, 2026.

Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. *arXiv preprint arXiv:2501.06848*, 2025.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. *arXiv preprint arXiv:2506.15799*, 2025.

Zixing Wang, Devesh K Jha, Ahmed H Qureshi, and Diego Romeres. Ppguide: Steering diffusion policies with performance predictive guidance. *arXiv preprint arXiv:2603.10980*, 2026.

Adam Wei, Abhinav Agarwal, Boyuan Chen, Rohan Bosworth, Nicholas Pfaff, and Russ Tedrake. Empirical analysis of sim-and-real cotraining of diffusion policies for planar pushing from pixels. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5625–5632. IEEE, 2025.

Antong Zhang, Han Qi, and Heng Yang. Beacon: Cross-domain co-training of generative robot policies via best-effort adaptation. *arXiv preprint arXiv:2605.08571*, 2026.

7 APPENDIX

7.1 FLOW-MATCHING EXTENSION WITH π_0

We evaluate language-conditioned negative guidance with the flow-matching formulation used by π_0 on LIBERO-Goal. We obtain the broad velocity from a π_0 checkpoint trained with the general instruction "operate on the table," and obtain task-specific negative velocities from a checkpoint trained with the ordinary LIBERO-Goal task descriptions. Both models use the same action representation and flow interpolation. No exclusion instructions or negative demonstrations are used during policy training.

Complement-flow motivation. We make the dependence on the observation o implicit. Let $p_t^b(x)$ and $v_b(x,t)$ denote the path marginal and velocity field associated with the broad instruction. For a single forbidden outcome, suppose that the broad path marginal admits the mixture decomposition

$$p_t^b(x) = \pi_- p_t^-(x) + (1-\pi_-) p_t^{\neg -}(x). \tag{13}$$

When all components use the same flow interpolant, linearity of probability flux gives

$$p_t^b(x) v_b(x,t) = \pi_- p_t^-(x) v_{c_-}(x,t) + (1-\pi_-) p_t^{\neg -}(x) v_{\neg -}(x,t). \tag{14}$$

With

$$r_t(x) := p_t(c_- \mid x) = \frac{\pi_- p_t^-(x)}{p_t^b(x)}, \tag{15}$$

division by $p_t^b(x)$ gives

$$v_b(x,t) = r_t(x) v_{c_-}(x,t) + (1 - r_t(x))\, v_{\neg -}(x,t). \tag{16}$$

Consequently, the ideal complement velocity is

$$v_{\neg -}(x,t) = v_b(x,t) - \frac{r_t(x)}{1-r_t(x)} \left(v_{c_-}(x,t) - v_b(x,t) \right). \tag{17}$$

This identity motivates the form of our guidance rule. It is exact when the mixture assumption holds and the component velocity fields and posterior r_t are exact. In practice, the pretrained flow policies provide the conditional velocities but not the posterior r_t. We therefore use the adaptive surrogate described below.

Adaptive flow-guidance weight. Let the Euler solver use step size $h > 0$, so that $t_{n+1} = t_n - h$. We maintain a bounded weight $\rho_n \in [p_{\min}, p_{\max}]$, initialized to a fixed prior ρ_0. Given the broad and negative velocities at (x_n, t_n), we first compute

$$\gamma_n = \lambda_0 \frac{\rho_n}{\max(1-\rho_n, Z_{\min})}, \qquad v_{\text{guid},n} = v_b - \gamma_n (v_{c_-} - v_b). \tag{18}$$

The broad, negative, and guided Euler proposals are

$$x_{n+1}^b = x_n - h v_b, \qquad x_{n+1}^- = x_n - h v_{c_-}, \qquad x_{n+1}^g = x_n - h v_{\text{guid},n}. \tag{19}$$

We measure the relative compatibility of the guided proposal with the negative and broad proposals using

$$\Delta_n = \left\| x_{n+1}^g - x_{n+1}^- \right\|_2^2 - \left\| x_{n+1}^g - x_{n+1}^b \right\|_2^2. \tag{20}$$

The adaptive weight is then updated as

$$\rho_{n+1} = \text{clip}\left(\rho_n \exp\left[\frac{-\tau \Delta_n + \delta}{2h^2} \right], p_{\min}, p_{\max} \right), \tag{21}$$

where τ controls the sensitivity to proposal separation and δ is an optional calibration offset.

Equation 21 can be interpreted through the local isotropic kernel

$$K_h(y;\mu) \propto \exp\left(-\frac{\tau}{2h^2} \|y - \mu\|_2^2 \right). \tag{22}$$

For $\delta = 0$, the multiplicative update is the compatibility ratio

$$\frac{K_h(x^g_{n+1}; x^-_{n+1})}{K_h(x^g_{n+1}; x^b_{n+1})}. \tag{23}$$

The kernels are used only to define a local comparison between deterministic Euler proposals; no stochastic transition is added to the flow sampler. Accordingly, ρ_n should be interpreted as an adaptive guidance weight, rather than as a calibrated estimate of $p_t(c_- \mid x_t, o)$.

For a single negative branch with $\delta = 0$, let $d_n = v_{c_-} - v_b$. Substitution of the guided velocity gives

$$\frac{\Delta_n}{h^2} = (1 + 2\gamma_n) \|d_n\|_2^2 \geq 0. \tag{24}$$

Thus, the weight is non-increasing whenever the broad and negative velocity fields differ. The resulting guidance is deliberately self-attenuating: the negative branch produces an initially strong correction, after which its influence decreases according to the local disagreement between the two velocity fields. This update is tied to the chosen Euler discretization and is not intended to recover an exact continuous-time posterior.

Multiple forbidden outcomes. For conditions $\mathcal{C} = \{c_1, \ldots, c_K\}$, we maintain one adaptive weight $\rho_n^{(i)}$ per condition and define

$$Z_n = \max\left(1 - \sum_{j=1}^{K} \rho_n^{(j)}, Z_{\min}\right). \tag{25}$$

The guided velocity is

$$v_{\text{guid},n} = v_b - \sum_{i=1}^{K} \lambda_0^{(i)} \frac{\rho_n^{(i)}}{Z_n} \left(v_{c_i} - v_b\right). \tag{26}$$

After constructing the shared guided proposal $x^g_{n+1} = x_n - h v_{\text{guid},n}$, each weight is updated independently:

$$\Delta_n^{(i)} = \left\| x^g_{n+1} - x^{(i)}_{n+1} \right\|_2^2 - \left\| x^g_{n+1} - x^b_{n+1} \right\|_2^2, \tag{27}$$

$$\rho_{n+1}^{(i)} = \text{clip}\left(\rho_n^{(i)} \exp\left[\frac{-\tau \Delta_n^{(i)} + \delta}{2h^2}\right], p_{\min}, p_{\max}\right), \tag{28}$$

where $x^{(i)}_{n+1} = x_n - h v_{c_i}$. The shared denominator accounts for the combined negative weight, while $Z_{\min} > 0$ and weight clipping prevent numerical divergence when that weight becomes large.

We compare this adaptive rule with a static CFG-style baseline that replaces the state-dependent coefficients in Eq. 26 with fixed guidance scales.

Numerical settings. The π_0 sampler uses $N = 10$ Euler steps from $t = 1$ to $t = 0$, corresponding to a step size $h = 0.1$. For single-target guidance, we initialize the adaptive weight to $\rho_0 = 0.25$. For multi-target guidance, we initialize each weight to $\rho_0^{(i)} = 0.20$ and use the same guidance scale for all negative branches, $\lambda_0^{(i)} = \lambda_0$. We sweep the guidance scale over

$$\lambda_0 \in \{1,\ 1.5,\ 3,\ 5,\ 10,\ 20\}, \tag{29}$$

covering weak to strong negative guidance. In all settings, we use $\tau = 0.2$, $\delta = 0$, $p_{\min} = 10^{-6}$, $p_{\max} = 0.8$, and $Z_{\min} = 10^{-6}$. With $h = 0.1$, $\tau = 0.2$, and $\delta = 0$, the adaptive-weight update simplifies to

$$\rho_{n+1}^{(i)} = \text{clip}\left(\rho_n^{(i)} \exp\left[-10\Delta_n^{(i)}\right], 10^{-6}, 0.8\right), \tag{30}$$

where $\Delta_n^{(i)}$ is the difference between the squared distances to the negative and broad Euler proposals, summed over the complete predicted action chunk.

7.2 Implementation Details

7.2.1 Shared policy architecture and training setup

We use the same diffusion-policy template with task-specific datasets and normalization statistics. Observations consist of RGB observations chunk in horizon 2, resized to 96×96, low-dimensional robot state, and a language embedding. The policy predicts a horizon of 16 future actions and executes the next 8 actions before replanning.

The visual encoder is a ResNet backbone whose BatchNorm layers are replaced with GroupNorm. Language inputs are encoded with CLIP ViT-B/32 and then projected through a two-hidden-layer MLP with hidden sizes $[512, 512]$. The denoising backbone is a `ConditionalUnet1D` conditioned on the flattened observation history. The observation feature at each step is the concatenation of a 512-dimensional visual feature, a 9-dimensional robot state, and a 512-dimensional projected language feature. The action dimension is 8.

Unless noted otherwise, models are trained with AdamW using learning rate 10^{-4}, weight decay 10^{-6}, batch size 64, EMA with power 0.75, cosine learning-rate decay, 100 diffusion iterations, and 500 warmup steps. Checkpoints are saved at epochs 400 except charger insertion loads epoch 1000 due to the larger datasets.

7.2.2 Dynamic negative guidance at inference time

Our inference-time guidance follows the dynamic negative guidance implementation used across the evaluation scripts. For a single negative condition, the guided score is

$$\hat{\epsilon}_{\text{guided}} = \hat{\epsilon} - \lambda_t \left(\hat{\epsilon}^- - \hat{\epsilon} \right), \qquad \lambda_t = \lambda \frac{p_t}{1 - p_t}, \tag{31}$$

where $\hat{\epsilon}$ is the prediction under the broad instruction, $\hat{\epsilon}^-$ is the prediction under the negative instruction, and p_t is the posterior updated during reverse diffusion. The posterior is updated from the relative distance of the sampled state to the conditional and negative conditional reverse means, with clamping parameters $\texttt{p_min} = 10^{-6}$ and $\texttt{p_max} = 0.8$.

For most language-conditioned ManiSkill experiments, we use prior probability 0.25, base guidance scale 3.0, temperature 0.2, and offset 0.0. For multi-target charger insertion, the implementation keeps an independent posterior per forbidden target, uses `prior_list = [0.20] * K`, `guidance_scale_list = [3.0] * K`, and combines the negative corrections additively with denominator $1 - \sum_j p_j$.

7.2.3 CFG baseline

We also compare against a classifier-free-guidance-style baseline Ho & Salimans (2022) implemented with the same pretrained policy, the same negative prompts, and the same frozen "forget" model used for negative guidance. The key difference is that CFG does not use the posterior-weighted dynamic scale when forming the reverse-process update. Instead, it applies a fixed guidance coefficient to each negative branch.

For example, the charger-insertion CFG uses three simultaneous forbidden prompts, "Place charger into first one (1).", "Place charger into second one (2).", and "Place charger into fifth one (5).", and updates the denoising prediction as

$$\hat{\epsilon}_{\text{CFG}} = \hat{\epsilon} - 3 \sum_{j=1}^{K} \left(\hat{\epsilon}_j^- - \hat{\epsilon} \right), \tag{32}$$

where the coefficient 3 is hard-coded in the sampling loop. Analogous CFG scripts in the four-object and can-placement tasks also use static scale-3 settings in the released evaluations, while Push-T evaluates CFG over the same scale sweeps used for DNG. The charger CFG script still computes per-target posterior values p_j, but unlike DNG these posterior terms are not used to scale the guidance update. In this sense, the CFG baseline matches the same conditional inputs and

evaluation protocol as DNG while removing the adaptive, state-dependent weighting that is central to our method.

7.3 Task Setup

7.3.1 Four-object picking

The training data consist of demonstrations for picking each of four object categories. Each dataset is labeled with both generic instruction, e.g., “Place any object into the bin.”, and specific instruction, e.g., “Place the cube into the bin”. Training is using 100 demonstrations from each dataset source.

Evaluation is conducted in a simulated manipulation environment containing four candidate objects and a target bin. Each condition is evaluated over random seeds, with a maximum rollout length of 450 environment steps.

7.3.2 Can placement

The training data consist of demonstrations for placing a can onto each of three target locations. Each dataset is labeled with both generic instructions, e.g., “Place the can on any place.”, and target-specific instructions, e.g., “Place the can on the blue spot.” Training uses the same optimization settings as the four-object task.

Evaluation is performed in a simulated placement environment containing three candidate target locations. Each condition is evaluated over 200 episodes with a maximum rollout length of 450 environment steps. During evaluation, the broad instruction is “Place the can on any place.”, while negative instructions specify locations that should be avoided. Outcomes are categorized according to the final placement location, producing an empirical distribution over target selections.

7.3.3 Charger insertion with multiple forbidden targets

The training data consist of demonstrations for inserting a charger into multiple receptacles together with additional demonstrations labeled by generic instructions. Each dataset is annotated with both general instructions, e.g., “Place charger into any receptacle.”, and receptacle-specific instructions, e.g., “Place charger into the third receptacle.” Owing to the larger training set, the policy is trained for 1000 epochs.

Evaluation is conducted in a simulated charger-insertion environment containing multiple receptacles. Each condition is evaluated over 200 episodes with a maximum rollout length of 450 environment steps. During evaluation, the broad instruction is “Place charger into any receptacle.”, while negative instructions specify one or more receptacles that should be avoided. Outcomes are categorized according to the receptacle selected by the policy, producing an empirical distribution over insertion targets.

7.3.4 Push-T trajectory avoidance

Language-conditioned The Push-T experiments evaluate a diffusion policy trained to solve a planar pushing task through multiple feasible trajectories. Training data include demonstrations corresponding to different successful pushing strategies together with language annotations describing trajectory preferences.

Evaluation is conducted in a planar pushing environment with obstacle-induced trajectory constraints. Each condition is evaluated over 200 test episodes using random seeds disjoint from those used during training, with a maximum horizon of 300 interaction steps. The broad instruction is ‘any way”, while negative instructions specify trajectories that should be avoided, such as ‘left”. Performance is measured by the maximum task reward achieved during each episode, and reported results correspond to the mean score across all evaluation episodes.

Failure Policy guided Similarly, we train a general policy with push-T demonstrations consisting of pushing from left, from middle, and from right, in a clean background. Then we deploy this general policy in environment with obstacles, but in this setting we don’t explicitly specify where the obstacle is and this information is revealed by letting the policy interact with the environment.

During this deployment, we collect a small number of failure data (e.g., 20 demonstrations) and utilize this set of failure data to train a failure policy. With the small tryout, the failure policy can guide the general policy to avoid obstacles in future deployment.

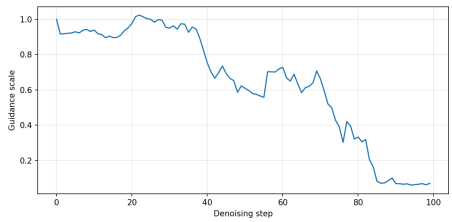
7.4 Evolution of Dynamic Guidance Scale During Denoising

To better understand the behavior of Dynamic Negative Guidance (DNG), we record the effective guidance scale

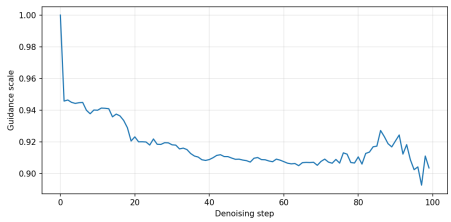
$$\lambda_t = \lambda \frac{p_t}{1 - p_t} \tag{33}$$

throughout a representative reverse-diffusion trajectory. Here, p_t denotes the estimated posterior probability of the forbidden condition at denoising step t, and λ is the base guidance scale.

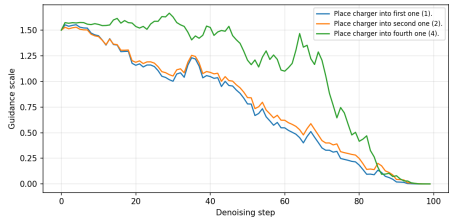
Figure 7 shows the evolution of the effective guidance scale across four representative tasks. In all cases, the guidance strength tends to decrease over the denoising process. This behavior indicates that the generated trajectory progressively moves away from the forbidden condition, causing the estimated posterior probability of the undesired outcome to decrease. As a result, the repulsive guidance term becomes less active as sampling proceeds.



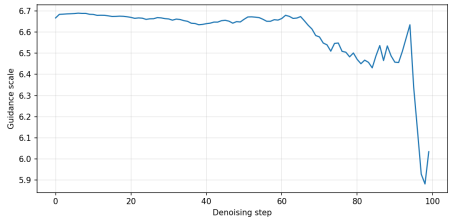
(a) Four-object picking



(b) Can placement



(c) Charger insertion



(d) Push-T trajectory avoidance

Figure 7: Evolution of the effective dynamic guidance scale during representative reverse-diffusion trajectories.

7.5 Avoidance-Uniformity Score

The Avoidance-Uniformity Score (AUS) summarizes three objectives: suppressing forbidden outcomes, avoiding task failure, and preserving diversity among the remaining valid outcomes. Let $\mathcal{A}$ denote the set of forbidden outcomes, $\mathcal{R}$ the set of remaining valid outcomes, p_i the empirical probability of outcome i, and p_{fail} the empirical task-failure probability.

We first define the probability of successful, constraint-satisfying behavior:

$$p_{\text{valid}} = 1 - \sum_{i \in \mathcal{A}} p_i - p_{\text{fail}} = \sum_{j \in \mathcal{R}} p_j .$$

Thus, probability mass assigned to either a forbidden outcome or task failure directly reduces p_{valid}.

Conditioned on valid behavior, we measure how evenly the probability mass is distributed across the remaining feasible outcomes:

Task	Method	$\lambda = 0.1$	$\lambda = 0.2$	$\lambda = 0.3$	$\lambda = 0.4$
Object Picking	Ours	0.811	0.773	0.735	0.697
	CFG	0.833	0.765	0.696	0.628
Can Placement	Ours	0.906	0.863	0.820	0.777
	CFG	0.906	0.833	0.760	0.687
Charger Insertion	Ours	0.561	0.535	0.510	0.485
	CFG	0.475	0.434	0.394	0.353

Table 7: Sensitivity of AUS to the uniformity weight λ. Each entry reports the mean AUS across all avoidance configurations within the corresponding task. Increasing λ places greater emphasis on preserving a balanced distribution over the remaining valid outcomes.

$$U = 1 - \frac{\frac{1}{2}\sum_{j\in\mathcal{R}} \left| \frac{p_j}{\sum_{k\in\mathcal{R}} p_k} - \frac{1}{|\mathcal{R}|} \right|}{1 - \frac{1}{|\mathcal{R}|}}.$$

Here, $U \in [0, 1]$ is a normalized uniformity score. A value of $U = 1$ corresponds to uniform selection among the remaining valid outcomes, whereas $U = 0$ corresponds to maximal concentration on a single valid outcome. When $p_{\text{valid}} = 0$, we set $U = 0$ by convention.

We define AUS as

$$\text{AUS} = p_{\text{valid}} \left[1 - \lambda(1 - U)\right], \qquad \lambda = 0.3.$$

The primary factor, p_{valid}, measures the probability of simultaneously completing the task and satisfying the avoidance constraint. The uniformity term serves as a secondary criterion for distinguishing methods that retain multiple valid modes from those that collapse onto a single alternative. With $\lambda = 0.3$, distributional imbalance can reduce the valid-success score by at most 30%, ensuring that successful constraint satisfaction remains the dominant objective.

7.6 Sensitivity to the AUS uniformity weight

The Avoidance-Uniformity Score (AUS) is designed to primarily measure successful constraint satisfaction while assigning a secondary preference to preserving diversity among the remaining valid outcomes. Recall that

$$\text{AUS} = p_{\text{valid}} \left[1 - \lambda(1 - U)\right],$$

where

$$p_{\text{valid}} = 1 - p_{\text{avoid}} - p_{\text{fail}},$$

and $U \in [0, 1]$ measures the uniformity of the conditional distribution over the remaining valid outcomes. The hyperparameter λ therefore controls only the maximum penalty associated with imbalance among valid outcomes: $\lambda = 0$ ignores uniformity entirely, while larger values place increasing emphasis on preserving multimodality.

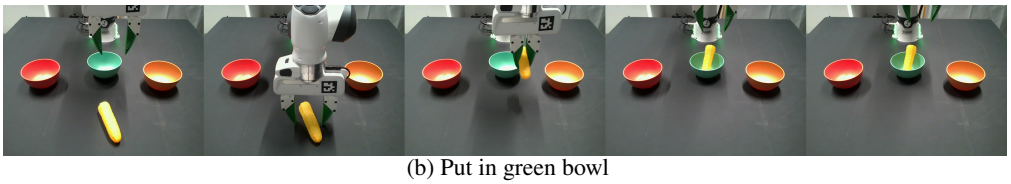
In the main experiments, we use $\lambda = 0.3$. To evaluate whether our conclusions depend strongly on this choice, we recompute AUS for $\lambda \in \{0.1, 0.2, 0.3, 0.4\}$ without changing any policy outputs or evaluation trajectories. Table 7 reports the mean AUS across avoidance configurations for each task family.

The qualitative comparison is stable over a broad range of λ. For charger insertion, our method obtains the highest mean AUS for all tested values, indicating that its advantage does not depend on strong weighting of the uniformity term. For can placement, our method ties CFG at $\lambda = 0.1$ and achieves higher mean AUS for $\lambda \geq 0.2$. For object picking, CFG has a higher mean AUS at $\lambda = 0.1$, where the metric places only a small penalty on mode collapse. Once preservation of the remaining multimodal behavior receives a modest weight ($\lambda \geq 0.2$), our method achieves the highest mean AUS.

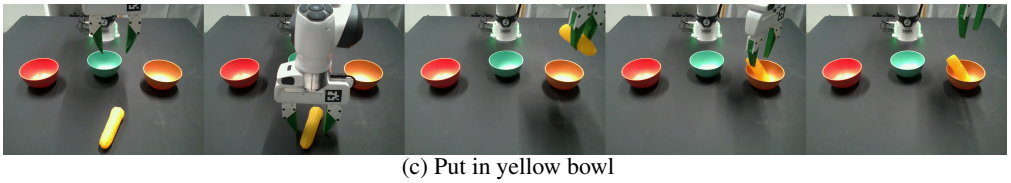
These results also illustrate the intended role of λ. Small values make AUS approach the valid-success probability and therefore favor methods that aggressively suppress the forbidden outcome



(a) Put in red bowl



(b) Put in green bowl



(c) Put in yellow bowl

Figure 8: Representative real-world trajectories for placing the corn into a bowl.

even if the remaining distribution collapses onto a single mode. Increasing λ distinguishes such behavior from policies that both satisfy the constraint and preserve multiple feasible alternatives. We use $\lambda = 0.3$ in the main paper as a middle setting: constraint-satisfying task success remains the dominant term, while maximal mode imbalance can reduce the score by at most 30%. Importantly, the main comparison is not specific to this exact value, as the same overall trend is observed for neighboring values $\lambda \in [0.2, 0.4]$.

7.7 Real-World Trajectories

To provide additional qualitative insight into the behavior induced by negative guidance, Figure 8 visualizes representative real-world executions using temporally ordered snapshots from single rollouts. The trained diffusion policy is multimodal, with probability to place the corn in red/green/yellow bowl.

7.8 PPGuide Baseline Details

We adapt PPGuide (Wang et al., 2026) to the same frozen base policies and deployment-time failure settings used by our failure-policy guidance experiments. PPGuide uses rollout-level success/failure feedback to train a discriminative guidance model. Each rollout is treated as a bag of replanning-step instances

$$x_t = [\mathrm{obs_cond}_t,\ a^k_{t:t+H_p-1}],$$

where $\mathrm{obs_cond}_t$ is the flattened policy conditioning vector and $a^k_{t:t+H_p-1}$ is the normalized sampled action chunk. We train an attention MIL model from trajectory-level outcomes, use the attention z-score threshold τ to pseudo-label instances as success-relevant (SR), failure-relevant (FR), or irrelevant (IR), and then train a three-way MLP classifier f_ψ on these labels.

At inference time, the base diffusion policy remains fixed and PPGuide modifies the predicted noise using gradients of the classifier:

$$\hat{\epsilon}_{\mathrm{PPGuide}} = \hat{\epsilon}_\theta + w_{\mathrm{SR}} \nabla_{a^k} \log p_\psi(\mathrm{SR} \mid x_t) - w_{\mathrm{FR}} \nabla_{a^k} \log p_\psi(\mathrm{FR} \mid x_t).$$

We use asymmetric guidance strengths $w_{\mathrm{FR}} = 0.30$ and $w_{\mathrm{SR}} = 0.03$, which provide an appropriate PPGuide setting with stronger failure repulsion than success attraction. Guidance is applied every other denoising step, following PPGuide's alternating guidance schedule. The MIL encoder and guidance classifier are both lightweight MLPs with hidden sizes 512 and 256; we train the MIL stage for 300 epochs and the guidance classifier for 200 epochs using AdamW with learning rate 10^{-4}.

Charger deployment failures. For charger insertion, we use the frozen epoch-1000 general charger policy and the broad instruction "Place charger into any of the receptacle." We use $H_o = 2$, $H_p = 16$, $H_a = 8$, and action dimension 8, giving PPGuide input dimension 2194. Rollout bags are built from the existing charger demonstrations with the same instance format as online collection. For a single unavailable receptacle, we use 20 failure demonstrations from that receptacle and 20 success demonstrations from each remaining receptacle. For the two-receptacle failure condition, we use 20 failure demonstrations from each unavailable receptacle and 20 success demonstrations from each available receptacle. We evaluate on 200 seeds and define success as insertion into any powered receptacle.

Push-T obstacle-induced failures. For Push-T, PPGuide is trained independently for each obstacle setting using the same frozen checkpoint as our failure-policy experiments. Rollout bags are collected by executing the unguided base policy in the obstacle-modified environment until 20 failed rollouts are observed, while retaining successful rollouts encountered during collection. Success is defined by maximum episode reward at least 0.9. We use $H_o = 2$, $H_p = 16$, action dimension 2, PPGuide input dimension 164, $\tau = 2.0$, and the same $w_{\mathrm{FR}} = 0.30, w_{\mathrm{SR}} = 0.03$ guidance setting. Evaluation uses 200 seeds.

7.9 Rejection Sampling Baseline Details

We implement rejection sampling as an inference-time baseline that leaves the pretrained diffusion policy unchanged. Instead of modifying the denoising update, the baseline samples multiple candidate action chunks from the broad-instruction policy and uses a separately trained outcome predictor to filter candidates that are likely to produce a forbidden outcome.

For each task, we train an outcome predictor g_ϕ that maps the current observation history and a candidate normalized action chunk to a distribution over terminal outcomes,

$$g_\phi(o_{t-H_o+1:t}, a_{t:t+H_p-1}) \in \Delta^{M-1}.$$

For four-object picking, the $M = 4$ classes are $\{\texttt{cube}, \texttt{apple}, \texttt{banana}, \texttt{cup}\}$. For charger insertion, the $M = 5$ classes correspond to the five receptacles. The predictor is trained on task-specific demonstration datasets with unambiguous episode-level labels; broad-instruction demonstrations are excluded when training the predictor so that each action window has a unique target label. We use the same normalized state/action statistics as the diffusion policy, observation horizon $H_o = 2$, prediction horizon $H_p = 16$, action horizon $H_a = 8$, and 96×96 image inputs. The policy vision encoder is kept frozen, and the predictor combines frozen visual features, normalized robot state, and the normalized action chunk using an MLP observation encoder, a small temporal convolutional action encoder, and a linear classification head. We train for 100 epochs with AdamW, learning rate 10^{-4}, weight decay 10^{-6}, batch size 64, dropout 0.2, and seed 0. The episode split uses the first 80 demonstrations per target for training and the next 20 for validation.

At evaluation time, the frozen base policy is queried with the same broad instruction used by the unguided baseline, e.g., "Place any object into the bin." or "Place charger into any receptacle." At every replanning step, we independently sample N candidate chunks from the base diffusion policy using the standard 100-step DDPM sampler. For a forbidden set A, each candidate is assigned risk

$$r(a) = \sum_{i \in A} p_\phi(i \mid o_{t-H_o+1:t}, a).$$

For single-target four-object experiments, A contains one object; for charger insertion, A may contain multiple receptacles. Candidates with $r(a) < \tau$ are accepted, with $\tau = 0.2$ in the reported runs. If at least one candidate is accepted, we execute one accepted candidate uniformly at random. If all candidates are rejected, we fall back to the candidate with the smallest predicted forbidden risk. We then unnormalize the selected chunk and execute the action slice 1:9, matching the action-horizon convention used by the base policy.

Rejection-sampling evaluations use $N = 8$ candidates, 200 test episodes and same settings as our pipeline when applicable.

Table 8: Ablation on the negative guidance scale for four-object picking with *banana* as the avoidance target. Results are reported over 200 rollouts. Higher AUS is better.

Method	Scale	Cube	Apple	Banana	Cup	Fail	AUS
CFG baseline	1	48.0%	11.5%	3.0%	32.0%	5.5%	0.830
CFG baseline	3	60.0%	6.0%	1.0%	27.0%	6.0%	0.800
CFG baseline	5	65.0%	6.5%	1.5%	22.0%	5.0%	0.783
CFG baseline	10	69.5%	4.5%	1.5%	17.0%	7.5%	0.734
NG	1	31.0%	23.5%	7.0%	28.5%	10.0%	0.811
NG	3	37.0%	20.0%	1.5%	32.5%	9.0%	**0.851**
NG	5	38.5%	21.5%	2.5%	28.5%	9.0%	0.844
NG	10	53.5%	15.0%	1.5%	24.5%	5.5%	0.829

7.10 Ablation on Negative Guidance Scale

We study the sensitivity of language-conditioned negative guidance to the guidance scale using the four-object picking task with *banana* as the avoidance target. We evaluate both dynamic negative guidance (NG) and the static CFG baseline over scales $\{1, 3, 5, 10\}$, with 200 rollouts per setting. Table 8 reports the outcome distribution and Avoidance-Uniformity Score (AUS).

Dynamic negative guidance is relatively robust across the tested scales and achieves its highest AUS of 0.851 at scale 3. Increasing the scale beyond this point continues to suppress the avoided outcome, but increasingly shifts probability mass toward a smaller subset of the remaining valid behaviors. In contrast, the static CFG baseline becomes progressively less balanced as the scale increases, with its AUS decreasing from 0.830 at scale 1 to 0.734 at scale 10. These results support the use of a moderate guidance scale and show that dynamic guidance is less sensitive to the choice of scale than fixed negative guidance.

7.11 Full table for charger in language-guided negative guidance

Method	Avoid	(1)	(2)	(3)	(4)	(5)	Fail	AUS
Baseline	None	21.0%	13.5%	19.0%	14.5%	15.5%	16.5%	–
Ours	$(1, 2, 3)$	5.0%	0.0%	4.5%	17.5%	53.5%	19.5%	**0.602**
CFG		4.5%	1.0%	0.0%	2.5%	68.0%	24.0%	0.509
Rejection Sampling		3.0%	2.5%	7.0%	23.0%	18.5%	46.0%	0.402
Ours	$(1, 2, 4)$	6.5%	2.5%	13.0%	7.5%	51.0%	19.5%	**0.526**
CFG		7.5%	0.0%	6.0%	2.5%	64.0%	20.0%	**0.526**
Rejection Sampling		4.0%	1.0%	18.0%	9.0%	20.5%	47.5%	0.378
Ours	$(1, 2, 5)$	7.5%	2.5%	21.0%	33.0%	18.0%	18.0%	**0.504**
CFG		11.5%	2.5%	13.5%	19.5%	29.5%	23.5%	0.312
Rejection Sampling		4.5%	5.0%	22.5%	19.0%	5.5%	43.5%	0.405
Ours	$(1, 3, 4)$	16.5%	8.0%	4.0%	4.0%	45.5%	22.0%	**0.423**
CFG		21.0%	0.0%	2.0%	1.0%	51.0%	25.0%	0.357
Rejection Sampling		5.0%	15.5%	6.0%	9.5%	18.5%	45.5%	0.331
Ours	$(1, 3, 5)$	18.5%	13.5%	9.5%	25.5%	15.0%	18.0%	**0.354**
CFG		27.5%	1.0%	6.5%	16.5%	27.5%	21.0%	0.129
Rejection Sampling		4.5%	15.5%	11.0%	17.5%	6.5%	45.0%	0.324
Ours	$(1, 4, 5)$	17.0%	20.5%	35.0%	3.0%	6.5%	18.0%	**0.512**
CFG		46.5%	5.5%	18.0%	5.0%	8.5%	16.5%	0.198
Rejection Sampling		5.0%	16.5%	23.5%	5.5%	8.5%	41.0%	0.379
Ours	$(2, 3, 4)$	28.5%	3.0%	2.5%	3.0%	41.0%	22.0%	**0.658**
CFG		16.5%	0.0%	0.5%	0.5%	55.5%	27.0%	0.603
Rejection Sampling		18.0%	2.0%	3.0%	6.5%	16.0%	54.5%	0.334
Ours	$(2, 3, 5)$	30.5%	2.5%	7.5%	25.0%	15.5%	19.0%	**0.539**
CFG		31.5%	1.0%	9.0%	7.5%	26.5%	24.5%	0.318
Rejection Sampling		16.5%	7.5%	8.0%	15.5%	7.0%	45.5%	0.317
Ours	$(2, 4, 5)$	56.0%	11.0%	4.5%	1.5%	9.5%	17.5%	0.451
CFG		69.5%	0.0%	1.0%	0.5%	8.0%	21.0%	**0.500**
Rejection Sampling		15.0%	6.0%	25.0%	3.5%	5.0%	45.5%	0.370
Ours	$(3, 4, 5)$	48.5%	15.0%	7.0%	0.5%	11.5%	17.5%	**0.535**
CFG		68.5%	0.5%	0.5%	1.5%	9.0%	20.0%	0.486
Rejection Sampling		18.0%	19.0%	8.5%	4.0%	5.5%	45.0%	0.367

Table 9: Charger-insertion results under multi-target avoidance constraints. Bold denotes the highest AUS for each avoidance setting.