

# EmbodiSteer: Steering Embodiment-Agnostic Visuomotor Policies with Joint-Space Guidance for Zero-Shot Cross-Embodiment Deployment

Shihfeng Wang<sup>\*,1,2,3</sup> Kangchen Lv<sup>\*,1,2,3</sup> Mingrui Yu<sup>†,1,2,3</sup> Xiang Li<sup>†,1,2,3</sup>

<sup>1</sup>Department of Automation, Tsinghua University, <sup>2</sup>Beijing Key Laboratory of Embodied Intelligence Systems

<sup>3</sup>Institute for Embodied Intelligence and Robotics, Tsinghua University

\*Equal contribution. †Corresponding author.

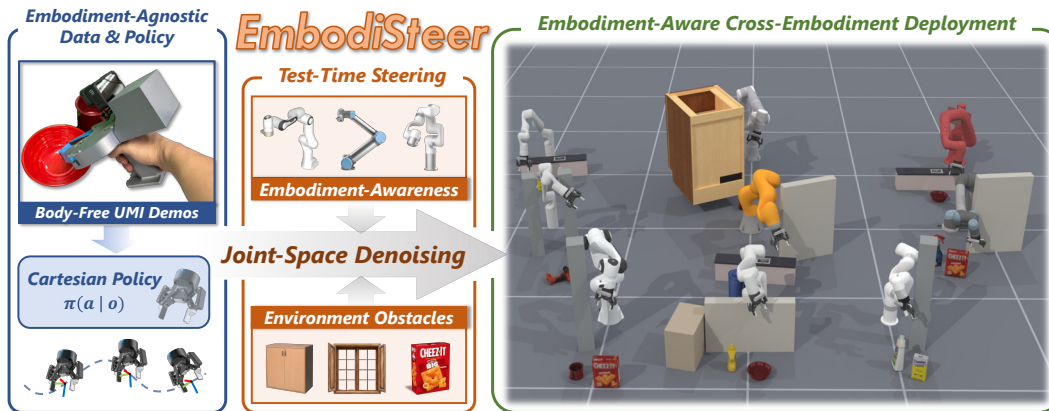


Figure 1: We present **EMBODISTEER**, an inference-time steering framework for embodiment-aware deployment of embodiment-agnostic visuomotor policies. Given a trained Cartesian policy (left), EmbodiSteer lifts the sampling process into the target robot’s joint space and incorporates robot embodiment and obstacle guidance during denoising (middle), enabling zero-shot whole-body collision-aware execution across diverse robots without finetuning (right).

**Abstract:** Scalable robot imitation learning relies on large-scale heterogeneous data from diverse robots or body-free data, making Cartesian end-effector actions a key interface for embodiment-agnostic policy learning. However, end-effector-only abstraction leaves Cartesian policies unaware of the deployed robot body, making them brittle under robot-specific constraints such as whole-body collision avoidance. To overcome this limitation, we present **EMBODISTEER**, a training-free framework that steers embodiment-agnostic visuomotor policies toward zero-shot, embodiment-aware deployment. **EMBODISTEER** keeps policy learning in Cartesian space while efficiently lifting inference-time diffusion sampling into the target robot’s joint space via forward kinematics and Jacobian-based updates. With whole-body collision-aware guidance over joint trajectories after each denoising step, the arm can be steered away from collisions while preserving learned end-effector behavior. Compared with Cartesian-only execution, **EMBODISTEER** reduces collision rate by 46.1% and improves task success rate by 28.5% across 9 simulated robots, and further achieves 90.0% collision rate reduction and 36.7% success rate increase on two physical robots in highly constrained scenarios. Our project page is at <https://frankwang67.github.io/EmbodiSteer-Page>.

**Keywords:** Cross-Embodiment Deployment, Whole-Body Collision Avoidance

## 1 Introduction

Recent progress in robot imitation learning has been driven by large-scale behavior cloning methods, from diffusion policies [1, 2] to vision-language-action models [3, 4, 5, 6]. One of the key factors

behind this scalability is the use of Cartesian end-effector poses as a unified action representation [7], whose dimensionality and semantics are independent of a robot’s degrees of freedom, joint ordering, or link geometry. Consequently, demonstrations from heterogeneous robots, simulators, and body-free data collection systems such as UMI [8] can be represented in a shared action space, providing an embodiment-agnostic foundation for cross-embodiment policy learning.

However, the embodiment-agnostic property of Cartesian policies comes with a deployment-time limitation of body unawareness, which becomes problematic when execution must satisfy embodiment-specific constraints. For example, the entire arm must avoid collisions in cluttered scenes, and predicted end-effector motions may become infeasible near workspace boundaries for a specific robot. Training directly in joint space could introduce embodiment awareness, but joint actions are morphology-specific and unavailable in UMI-style body-free demonstrations, making zero-shot deployment to diverse embodiments difficult without robot-specific finetuning data.

These observations motivate us to revisit the central challenge of **cross-embodiment deployment of embodiment-agnostic policies**. Existing research on cross-embodiment policy generalization largely treats embodiment differences as shifts in action and state distributions, addressing them through embodiment-specific action decoders [9, 10, 11], reusable low-level action primitives or skills [12, 13, 14], and embodiment-conditioned policy representations [5, 15, 16]. However, these works leave execution-time constraints of the target robot body largely unaddressed, so end-effector motion may transfer across robots while whole-body collisions still cause failures. In this work, we focus on the complementary problem of feasibly and safely executing an embodiment-agnostic policy on different robot arms at inference time. We argue that *true cross-embodiment deployment is more than reproducing end-effector behavior across robots*, since the policy output must be realized as embodiment-aware joint-space motion that respects the target robot’s whole-body constraints.

To address this problem, we propose **EMBODISTEER**, a training-free inference framework for embodiment-aware deployment of learned Cartesian policies. The policy remains being trained only with the end-effector actions, thus preserving its compatibility with embodiment-agnostic demonstrations. At inference, EMBODISTEER lifts Cartesian diffusion sampling process into the target robot’s joint space, where embodiment-specific constraints can steer the sampled motion toward collision-free, robot-feasible execution. Concretely, EMBODISTEER maintains a target-robot joint trajectory as the sampling state, but queries the frozen denoiser in its original Cartesian action space via forward kinematics. Each denoised Cartesian target is tracked in joint space with damped Jacobian updates, while CBF-inspired whole-body guidance applies joint corrections that move the robot body away from obstacles without largely disrupting the policy-predicted end-effector motion.

We validate EMBODISTEER by directly deploying trained Cartesian policies across robot embodiments in whole-body collision-aware manipulation scenarios, covering 9 simulated and 2 real robot embodiments. Compared with Cartesian-only execution, EMBODISTEER improves whole-body obstacle avoidance by 46.1% and task success rate by 28.5% in simulation, while consistently outperforming post-processing variants and cost-gradient-based steering baselines. On two physical robots, EMBODISTEER further reduces collisions by 90.0% and improves task success by 36.7% in highly constrained manipulation tasks, without robot-specific finetuning.

Overall, our contributions are summarized as follows:

1. We propose EMBODISTEER, a training-free inference framework that lifts inference-time sampling of embodiment-agnostic Cartesian policies into the target robot’s joint space, enabling zero-shot cross-embodiment deployment while retaining the scalability of Cartesian policy learning.
2. We introduce CBF-inspired whole-body collision-aware guidance for sampled joint trajectories, which uses small corrective joint updates to steer the robot body away from obstacles while leveraging a Jacobian-based task-space cost to preserve the learned end-effector behavior.
3. We design systematic simulation and real-world benchmarks for cross-embodiment deployment, demonstrating that EMBODISTEER achieves zero-shot whole-body collision-aware execution across diverse embodiments while outperforming Cartesian-only execution and other steering baselines.

## 2 Related Work

**Cross-Embodiment Policy Learning.** Large-scale cross-embodiment pretraining over diverse robot data is central to robot generalist policies, and a common strategy is to unify heterogeneous data in a shared action space. Cartesian end-effector poses are widely adopted for their embodiment-agnostic semantics, and also support body-free data collection systems [8, 17, 18]. Beyond directly mixing data from different platforms or aligning coordinate frames for training, prior works have explored several forms of embodiment alignment, such as latent action spaces with embodiment-specific decoders [9, 10, 19], reusable action primitives or skill abstractions [12, 13, 14], and embodiment-conditioned policy representations [20, 21, 15, 5]. More recently, ego-centric approaches use human hand motion as a shared intermediate space [22, 23, 24]. However, these works mainly address observation, state, and action distribution shifts across embodiments at the policy-learning level, whereas our work focuses on the inference-time problem of deploying an embodiment-agnostic policy under robot-specific constraints such as whole-body collision safety.

**Steering Diffusion Generation with Guidance.** Diffusion-based generation in robotics can be steered at inference time without modifying pretrained models, through mechanisms such as noise biasing [25], post-hoc filtering [26, 27], and guided sampling. While classifier-free guidance [28] requires guidance signals during training [29], classifier guidance [30] can leverage inference-time gradients without retraining, and has been widely applied to robotic diffusion generation [31, 32, 33, 34, 35, 36]. For visuomotor policies, DynaGuide [37] uses a learned dynamics model to steer base policies toward complex objectives, and ITPS [38] incorporates real-time user interactions through test-time guidance. Most related to our setting, EADP [39] steers embodiment-agnostic policies for embodiment-aware deployment, but focuses on low-level controller tracking costs for aerial manipulators. We instead target collision-aware whole-body execution for robot arms by lifting Cartesian policies into joint-space sampling and applying collision-aware guidance.

**Collision-Aware Visuomotor Policies.** Safety-critical robot manipulation in cluttered environments requires explicit collision awareness. Learning-based visuomotor policies can incorporate safety through post-processing mechanisms, such as control barrier functions [40, 41, 42], model predictive filters [43, 44], and risk estimators [45, 27]. For diffusion-based policies, obstacle-related costs can further steer the sampling process toward safer trajectories at inference time [46, 47, 48]. However, most existing safety mechanisms focus on end-effector-level collisions, leaving whole-arm constraints under-modeled. Recent works address robot-body safety by encoding whole-body geometry with point clouds, keypoints, or signed distance fields and learning from collision-free demonstrations [49, 50, 51, 52]. Such representations, however, are often coupled to robot morphologies and require robot-specific training data. In contrast, we use a pretrained embodiment-agnostic policy as the base and introduce whole-body collision constraints during zero-shot inference.

## 3 Preliminaries: Cartesian Diffusion Policy

Imitation learning aims to learn a visuomotor policy  $\pi$  from expert demonstrations, which maps observations such as images and proprioception to actions. Diffusion Policy [1] formulates action generation as a conditional sampling problem with diffusion models. During training, Gaussian noise is added to a clean action sample  $a_0$  from the dataset according to  $A_t = \sqrt{\bar{\alpha}_t}A_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon_t$  at diffusion timestep  $t$ , where  $\bar{\alpha}_t$  denotes noise schedule coefficient and  $\epsilon_t \sim \mathcal{N}(0, I)$ . The denoising model is trained to predict the added noise by minimizing  $\mathcal{L} = \text{MSELoss}(\epsilon_t, \epsilon_\theta(A_t, o, t))$ . At inference time, starting from  $A_T \sim \mathcal{N}(0, I)$ , the sample is iteratively denoised as

$$A_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( A_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \epsilon_\theta(A_t, o, t) \right) + \sigma_t z, \quad z \sim \mathcal{N}(0, I). \quad (1)$$

We use Cartesian end-effector poses as the action space for embodiment-agnostic policies, which predict an action chunk  $A = [a_1, \dots, a_H]$  over horizon  $H$ . Here, each action  $a_i = [\delta p_i, r_i] \in \mathbb{R}^9$  is represented relative to the chunk-start end-effector pose, where  $\delta p_i \in \mathbb{R}^3$  and  $r_i \in \mathbb{R}^6$  denote the

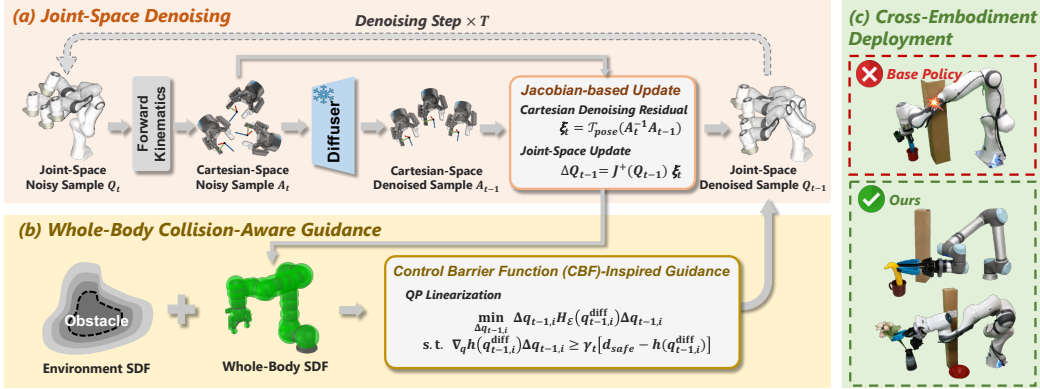


Figure 2: Overview of EMBODISTEER. (a) A trained Cartesian policy is lifted into the target-robot joint space for inference-time sampling. (b) CBF-inspired guidance steers sampled joint trajectories away from whole-body collisions while preserving end-effector behavior. (c) Visualization of real-world deployment across diverse robot embodiments on constrained manipulation tasks.

relative translation and 6D rotation representation [53], respectively. Notably, the gripper command is also included in the action but will be omitted for writing brevity in the following sections.

## 4 Method

As shown in Fig. 2, the Cartesian end-effector policy is trained on embodiment-agnostic demonstrations and maintained frozen during cross-embodiment inference-time steering. In this section, we first detail our problem formulation of whole-body collision-aware deployment, followed by our proposed joint-space denoising and CBF-inspired whole-body collision-aware guidance mechanism.

### 4.1 Problem Formulation

We consider deploying a frozen embodiment-agnostic Cartesian diffusion policy  $\pi$  on a target robot embodiment  $\mathcal{E}$  with obstacle information  $\mathcal{O}$ . The policy generates an end-effector action chunk  $A = [a_1, \dots, a_H]$  in Cartesian space, but has no access to the robot’s joint configuration and whole-body collision state during sampling. Therefore, the target-robot joint trajectory  $Q = [q_1, \dots, q_H]$ , where  $q_i \in \mathbb{R}^D$  for robot with  $D$  controllable arm joints, is required for reasoning about embodiment-specific constraints. The desired whole-body safety condition is

$$h(q_i) = \text{SDF}(\mathcal{E}(q_i), \mathcal{O}) \geq d_{\text{safe}}, \quad \forall i = 1, \dots, H, \quad (2)$$

where positive signed distance indicates clearance, negative values indicate penetration, and  $d_{\text{safe}} > 0$  is the clearance margin.

### 4.2 Joint-Space Denoising with Pretrained Cartesian Diffusion Policy

A straightforward way to lift Cartesian denoising into joint space is to solve inverse kinematics under the target robot  $\mathcal{E}$  after each denoising step, i.e.,  $q_{t,i} = \text{IK}_{\mathcal{E}}(a_{t,i})$  [36]. However, repeatedly solving IK inside the denoising loop is costly and hinders real-time deployment. Instead, we treat the Cartesian denoising update as a local task-space residual and map it to joint space using the target robot’s Jacobian  $J_{\mathcal{E}}(q) \in \mathbb{R}^{6 \times D}$ . Since adjacent denoising steps typically induce small Cartesian changes, the first-order kinematic relation provides a reliable joint-space update without IK solving at every step. Notably, the damped pseudoinverse Jacobian  $J_{\mathcal{E}}^+(q) = J_{\mathcal{E}}(q)^{\top} (J_{\mathcal{E}}(q) J_{\mathcal{E}}(q)^{\top} + \lambda_{\text{pinv}} I)^{-1}$  is used here to improve numerical stability and prevent large joint motions near singularities, where  $\lambda_{\text{pinv}}$  is the damping factor.

**Denoising Initialization.** We initialize joint-space sampling by drawing Cartesian Gaussian noise  $a_{T,i} \in \mathbb{R}^9$  and locally mapping it around the chunk-start configuration  $q^{\text{start}}$ :

$$q_{T,i} = q^{\text{start}} + \alpha \cdot \text{clip}(J_{\mathcal{E}}^+(q^{\text{start}}) \xi_{T,i}, -\Delta q_{\text{max}}, \Delta q_{\text{max}}), \quad \xi_{T,i} = \mathcal{T}_{\text{pose}}(a_{T,i}). \quad (3)$$

---

**Algorithm 1:** EMBODISTEER: Joint-Space Inference with Whole-Body Collision Guidance

---

**Input:** Observation  $o$ , obstacles  $\mathcal{O}$ , embodiment  $\mathcal{E}$

**Output:** Joint action chunk  $Q_0$

```
1 Sample Cartesian noise  $a_{T,i}$  and initialize  $Q_T = [q_{T,1}, \dots, q_{T,H}]$  using (3);
2 for  $t = T, \dots, 1$  do
  // Joint Space Denoising
3   Map  $Q_t$  to Cartesian action  $A_t$  using  $\text{FK}_{\mathcal{E}}$ ; // Kinematic mapping
4    $A_{t-1} \leftarrow \text{DDPMStep}(\epsilon_{\theta}(A_t, t, o), t, A_t)$ ; // Frozen Cartesian policy
5    $[\xi_{t,i}] \leftarrow \mathcal{T}_{\text{pose}}(A_t^{-1}A_{t-1})$ ; // Cartesian denoising residual
6   Update joint space sample  $Q_{t-1}^{\text{diff}}$  using (4); // Jacobian residual update
  // Whole-Body Collision-Aware Guidance
7   Solve (6) in batch to get solution  $\Delta Q_{t-1}^{\text{cbf}}$ ; // Solve QP
8   Apply the solution  $\Delta Q_{t-1}^{\text{cbf}}$  to  $Q_{t-1}^{\text{diff}}$  using (7) to get  $Q_{t-1}$ ; // Apply the guidance
9 end
10 return  $Q_0$ 
```

---

Here,  $\mathcal{T}_{\text{pose}}$  converts the Cartesian noise into a 6D pose twist relative to the chunk-start end-effector pose,  $\Delta q_{\text{max}}$  bounds the per-joint perturbation, and  $\alpha$  is a global scaling factor. This local kinematic mapping keeps joint-space samples consistent with the Cartesian diffusion initialization.

**Joint-Space Denoising Update.** At denoising step  $t$ , EMBODISTEER maintains a noisy joint action chunk  $Q_t = [q_{t,1}, \dots, q_{t,H}]$ . We first project  $Q_t$  through forward kinematics and convert the resulting end-effector poses into the relative Cartesian action chunk  $A_t$ . The frozen Cartesian denoising network is then queried in its original action space as in (1), yielding the Cartesian denoising target  $A_{t-1} = \text{DiffusionStep}(\epsilon_{\theta}(A_t, t, o), t, A_t)$ . Each denoising update is converted into a local 6D pose twist  $\xi_{t,i} = \mathcal{T}_{\text{pose}}(a_{t,i}^{-1}a_{t-1,i})$ , and mapped back to the joint space with the damped Jacobian:

$$q_{t-1,i}^{\text{diff}} = q_{t,i} + \text{clip}(J_{\mathcal{E}}^+(q_{t,i})\xi_{t,i}, -\Delta q_{\text{max}}, \Delta q_{\text{max}}). \quad (4)$$

### 4.3 CBF-Inspired Whole-Body Collision-Aware Guidance

Maintaining the denoising state in joint space allows EMBODISTEER to consider whole-body constraints during sampling. A direct approach is energy-based guidance with gradients over the joint trajectory[46, 54], which penalizes unsafe configurations at each denoising step:  $q_{t-1,i} = q_{t-1,i}^{\text{diff}} - \rho_t \nabla_q \mathcal{L}_{\text{coll}}(q_{t-1,i}^{\text{diff}})$ , where  $\mathcal{L}_{\text{coll}}(q)$  is the cost function punishing whole-body collision, and  $\rho_t$  is the guidance strength. However, because whole-body collision gradients often originate from non-end-effector links such as the elbow or forearm, pushing the robot body away from obstacles can also perturb the end-effector trajectory needed for task completion.

**CBF-Inspired Guidance Formulation.** EMBODISTEER adopts a CBF-inspired collision-aware guidance to encourage whole-body safety while preserving end-effector motion. For clarity, we describe the guidance for a single horizon step  $q_{t-1,i}^{\text{diff}}$ , and apply the same update independently to all steps in the horizon. We compute a guided joint configuration  $q_{t-1,i}$  by solving

$$\min_{q_{t-1,i}} \frac{1}{2} \left\| \text{FK}(q_{t-1,i}^{\text{diff}})^{-1} \text{FK}(q_{t-1,i}) \right\|_W^2 + \frac{1}{2} \lambda_{\text{cbf}} \|q_{t-1,i} - q_{t-1,i}^{\text{diff}}\|^2, \quad \text{s.t. } h(q_{t-1,i}) \geq d_{\text{safe}}, \quad (5)$$

where  $W$  is a diagonal weight matrix, and  $\lambda_{\text{cbf}}$  balances task-space preservation and joint-space regularization. The objective tracks the diffusion update in both task and joint space, while allowing the safety constraint to move the robot body away from obstacles when necessary.

**QP Linearization for Efficient Update.** To obtain a lightweight per-step update, we linearize the forward kinematics and safety function at  $q_{t-1,i}^{\text{diff}}$ . Let  $\Delta q_{t-1,i} = q_{t-1,i} - q_{t-1,i}^{\text{diff}}$ . Substituting  $\text{FK}(q_{t-1,i}^{\text{diff}})^{-1} \text{FK}(q_{t-1,i}) \approx J_{\mathcal{E}}(q_{t-1,i}^{\text{diff}}) \Delta q_i$  and  $h(q_{t-1,i}) \approx h(q_{t-1,i}^{\text{diff}}) + \nabla_q h(q_{t-1,i}^{\text{diff}})^{\top} \Delta q_i$

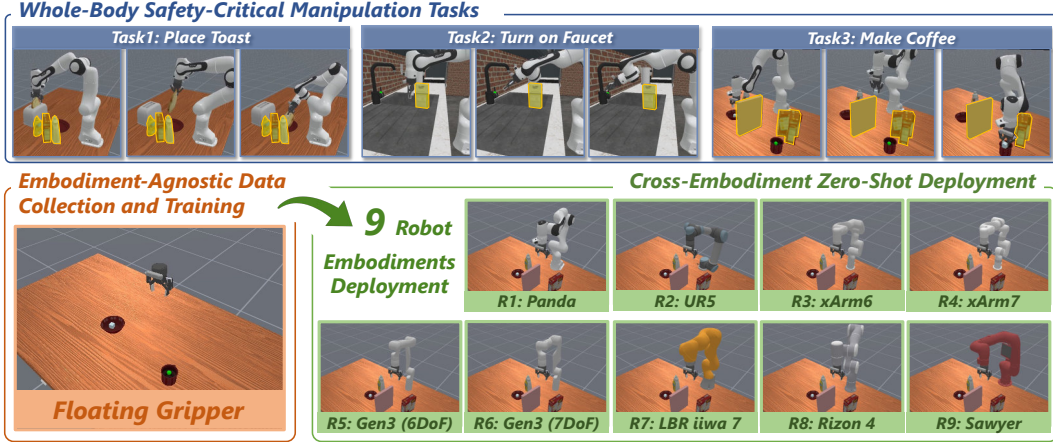


Figure 3: We evaluate EMBODISTEER on three manipulation tasks requiring both task completion and whole-body obstacle avoidance. Cartesian policies are trained from obstacle-free floating-gripper demonstrations and deployed zero-shot on 9 robot embodiments with test-time obstacles.

into (5) yields the single-constraint QP:

$$\min_{\Delta q_i} \frac{1}{2} \Delta q_{t-1,i}^\top H_{\mathcal{E}} (q_{t-1,i}^{\text{diff}}) \Delta q_{t-1,i}, \quad \text{s.t.} \quad \nabla_q h (q_{t-1,i}^{\text{diff}})^\top \Delta q_{t-1,i} \geq \gamma_t [d_{\text{safe}} - h (q_{t-1,i}^{\text{diff}})], \quad (6)$$

where  $H_{\mathcal{E}} (q_{t-1,i}^{\text{diff}}) = J_{\mathcal{E}} (q_{t-1,i}^{\text{diff}})^\top W J_{\mathcal{E}} (q_{t-1,i}^{\text{diff}}) + \lambda_{\text{cbf}} I$ , and  $\gamma_t$  controls the strength of the constraint. Since the QP has only one linear inequality, it can be solved efficiently in closed form or by a batched small-scale QP solver. Denoting the solution as  $\Delta q_{t-1,i}^{\text{cbf}}$ , the final guided sample is

$$q_{t-1,i} = q_{t-1,i}^{\text{diff}} + \text{clip}(\Delta q_{t-1,i}^{\text{cbf}}, -\Delta q_{\text{max}}^{\text{cbf}}, \Delta q_{\text{max}}^{\text{cbf}}). \quad (7)$$

## 5 Results

### 5.1 Simulation Results

**Experiment Setup.** We design three challenging manipulation tasks in ManiSkill [55] to evaluate body-aware cross-embodiment deployment, as shown in Fig. 3. These tasks require both accurate end-effector motion and collision avoidance between surrounding obstacles and the full robot body. For each task, we collect 200 diverse demonstrations with a floating gripper, recording only wrist-view RGB observations, end-effector states, and end-effector actions to keep the policy embodiment-agnostic. The trained Cartesian policy is deployed zero-shot on 9 robot arms with different morphologies, while sharing the same gripper and wrist-camera mounting.

**Baselines.** We consider four baseline groups: 1) **Cartesian denoising without guidance (EE)** directly executes the Cartesian end-effector actions predicted by the learned policy; 2) **Joint-space denoising without guidance (Joint)** lifts Cartesian sampling into the target robot’s joint space without collision guidance, isolating the effect of joint-space denoising alone; 3) **Cartesian denoising with post-hoc correction** leaves Cartesian sampling unchanged and applies whole-body collision avoidance only after trajectory generation, where *EE w/ Sampling* samples multiple actions and selects the candidate with the largest realized clearance, and *EE w/ CBF* solves IK followed by a one-shot CBF-QP correction; 4) **Joint-space denoising with guidance** adds collision-aware guidance during every joint-space denoising step, where *Joint w/ CG* uses cost-gradient guidance and EMBODISTEER uses our proposed task-aware CBF-QP guidance.

**Results and Analysis.** Tab. 1 reports quantitative results averaged over 9 robot embodiments. In obstacle-free settings, the Cartesian policy transfers directly across diverse arms with over 90% average task success. Joint-space denoising preserves task success and even yields a slight improvement,

Table 1: Quantitative results on manipulation tasks with obstacles, averaged over 9 robot embodiments. TSR is task success rate (%), RWD is reward per episode, and COR is collision occurrence rate (%). Best results among obstacle-present methods are shown in **bold**.

Obstacles	Method	PlaceToast			TurnOnFaucet			MakeCoffee			Average		
		TSR↑	RWD↑	COR↓	TSR↑	RWD↑	COR↓	TSR↑	RWD↑	COR↓	TSR↑	RWD↑	COR↓
w/o Obs.	EE	96.8	.980	–	83.6	.937	–	89.7	.932	–	90.0	.950	–
	Joint	94.4	.965	–	87.9	.957	–	89.7	.929	–	90.7	.950	–
w/ Obs.	EE	43.4	.634	52.6	41.6	.752	61.8	22.2	.456	58.3	35.7	.614	57.6
	EE w/ Sampling	47.1	.661	51.3	43.9	.767	53.2	24.1	.478	60.0	38.4	.635	54.8
	EE w/ CBF	65.8	.764	15.3	56.3	.797	29.9	48.9	.606	8.6	57.0	.722	17.9
	Joint w/ CG	57.7	.687	<b>0.2</b>	10.3	.403	52.0	22.4	.424	40.6	30.1	.505	30.9
	EMBODISTEER	<b>74.8</b>	<b>.829</b>	4.6	<b>60.6</b>	<b>.833</b>	<b>27.1</b>	<b>57.2</b>	<b>.670</b>	<b>2.8</b>	<b>64.2</b>	<b>.777</b>	<b>11.5</b>

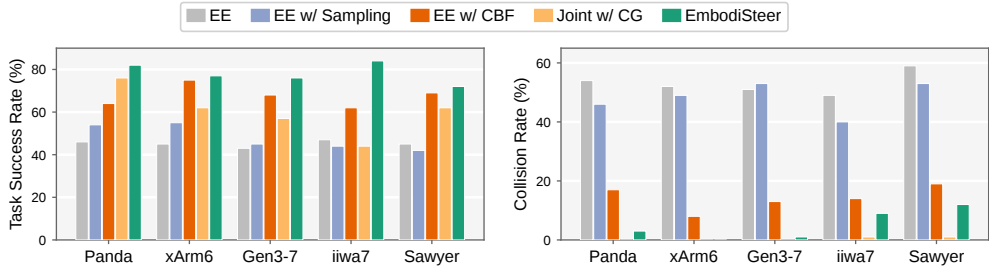


Figure 4: Per-embodiment comparison on the PLACETOAST task, where the task success rate (left) and collision rate (right) across five representative robot embodiments are reported.

showing that sampling in joint space maintains the learned end-effector behavior. With obstacles, however, Cartesian execution drops to 35.7% success and 0.614 reward, with 57.6% collision rate, revealing the limitation of end-effector-only deployment under whole-arm collision constraints.

Among obstacle-aware methods, EMBODISTEER achieves the best overall balance between task success and whole-body collision avoidance. Fig. 4 further compares success rate and collision rate across different embodiments, showing consistent gains over baselines. *EE w/ Sampling* only modestly improves success by 2.7% and reduces collision failures by 2.8%, while incurring much higher cost from repeated sampling and collision checking. *EE w/ CBF* provides a stronger post-hoc correction, improving success by over 20%, but applies safety only after denoising and cannot shape the generated trajectory. In contrast, EMBODISTEER injects guidance at every joint-space denoising step, producing safer and more feasible trajectories.

*Joint w/ CG* exhibits large task-dependent variation. It performs competitively on PLACETOAST, where coarse end-effector motion is often sufficient, but struggles on TURNONFAUCET and MAKECOFFEE, which require precise interactions. Although collision-cost gradients can push the arm away from obstacles, they can also significantly distort the end-effector trajectory. This highlights the benefit of EMBODISTEER’s CBF-QP guidance, which moves the robot body while minimizing deviation from the policy-predicted Cartesian motion. Finally, the constraint strength (the scale of  $\gamma$ ) ablation shows the trade-off between safety and task preservation. With  $\gamma = 0$ , EMBODISTEER reduces to joint-space denoising without collision guidance. Increasing  $\gamma$  initially reduces collisions and improves success, but overly large guidance eventually lowers task success by driving samples away from the learned policy distribution.

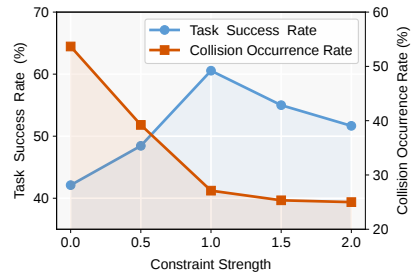


Figure 5: Ablation for the constraint strength on TURNONFAUCET.

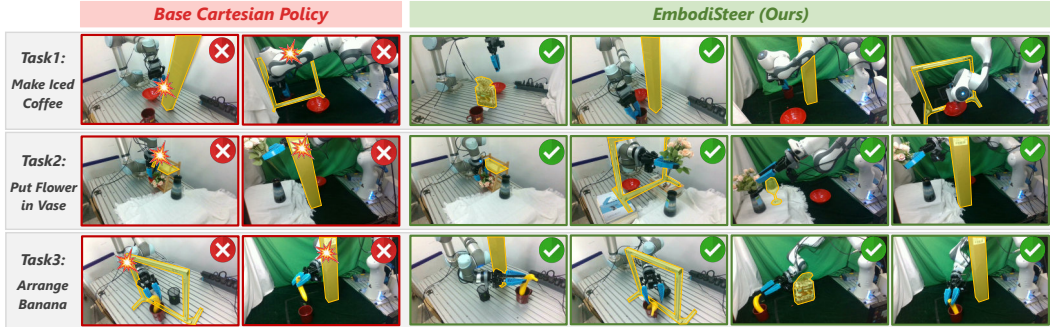


Figure 6: Real-world deployment of an arm-agnostic Cartesian policy on UR5 and Panda. Without guidance, the base policy often preserves task intent but collides with obstacles (yellow region), whereas EMBODISTEER steers joint trajectories to avoid arm-body collisions and complete tasks.

## 5.2 Real-World Experiments

**Experiment Setup.** We further evaluate EMBODISTEER on two physical embodiments, UR5 and Franka Panda, across three tabletop manipulation tasks, as shown in Fig. 6. For each task, we collect 200 demonstrations without extra obstacles, with the UMI handheld-gripper [8] and train a Cartesian diffusion policy without robot-specific demonstrations or finetuning. Obstacles are introduced only at deployment time, thus directly evaluating whether EMBODISTEER can introduce embodiment-aware collision avoidance to an embodiment-agnostic policy. EMBODISTEER also supports real-time physical deployment, achieving inference rates of 9.61 Hz on an RTX 4070 Ti SUPER.

**Results.** For each robot and task, we conduct 10 no-obstacle trials to evaluate base task competence and 10 obstacle trials to evaluate whole-body collision avoidance. The no-obstacle setting confirms that trained embodiment-agnostic Cartesian policies transfer to both physical embodiments, achieving 49/60 successes. With obstacles, however, the base policy drops to 25/60 successes and collides in 58/60 trials, showing that end-effector-only policies lack reliable embodiment-aware execution. In contrast, EMBODISTEER recovers performance to 47/60 successes while reducing collisions to 4/60. Qualitative comparisons in Fig. 6 further illustrate how EMBODISTEER improves collision-aware deployment while maintaining practical control rates across robots and tasks.

## 6 Limitations & Conclusion

**Limitations.** EMBODISTEER has several limitations. First, our guidance linearizes the whole-body constraint into a local QP and applies clipped joint corrections for stable execution; therefore, it provides an efficient feasible direction for collision avoidance but does not guarantee globally collision-free motion. Second, the guidance is activated only when the robot enters the SDF margin, making it a local correction mechanism rather than a global planner that can select the optimal collision-free motion modes far in advance. Finally, strong guidance may steer the robot into states outside the training distribution of the Cartesian policy, where task execution can fail, suggesting that broader and more diverse demonstrations remain important for robust deployment.

**Conclusions.** We propose EMBODISTEER for zero-shot embodiment-aware deployment of Cartesian diffusion policies by lifting inference-time sampling into the target robot’s joint space and applying task-preserving whole-body collision guidance. By keeping policy learning in an embodiment-agnostic Cartesian action space while enforcing robot-specific constraints only at deployment time, the method preserves the scalability of Cartesian imitation learning and adds the body awareness needed for safe execution. Across simulation and real-world experiments, EMBODISTEER improves task success and substantially reduces collisions over Cartesian execution and post-hoc guidance baselines, demonstrating a practical path toward reusable manipulation policies across diverse robot embodiments.

## Acknowledgments

If a paper is accepted, the final camera-ready version will (and probably should) include acknowledgments. All acknowledgments go at the end of the paper, including thanks to reviewers who gave useful comments, to colleagues who contributed to the ideas, and to funding agencies and corporate sponsors that provided financial support.

## References

- [1] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.
- [2] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.
- [3] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.
- [4] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- [5] J. Zheng, J. Li, Z. Wang, D. Liu, X. Kang, Y. Feng, Y. Zheng, J. Zou, Y. Chen, J. Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.
- [6] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. In *International Conference on Learning Representations*, volume 2025, pages 29982–30009, 2025.
- [7] Y. Wang, S. Zheng, H. Luo, W. Zhang, H. Yuan, C. Xu, H. Xu, Y. Feng, M. Yu, Z. Kang, et al. Rethinking visual-language-action model scaling: Alignment, mixture, and regularization. *arXiv preprint arXiv:2602.09722*, 2026.
- [8] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.
- [9] E. Bauer, E. Nava, and R. K. Katzschmann. Latent action diffusion for cross-embodiment manipulation. *arXiv preprint arXiv:2506.14608*, 2025.
- [10] Q. Bu, Y. Yang, J. Cai, S. Gao, G. Ren, M. Yao, P. Luo, and H. Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.
- [11] L. Wang, X. Chen, J. Zhao, and K. He. Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers. *Advances in neural information processing systems*, 37: 124420–124450, 2024.
- [12] M. Xu, Z. Xu, C. Chi, M. Veloso, and S. Song. Xskill: Cross embodiment skill discovery. In *Conference on robot learning*, pages 3536–3555. PMLR, 2023.
- [13] L. Zha, A. J. Hancock, M. Zhang, T. Yin, Y. Huang, D. Shah, A. Z. Ren, and A. Majumdar. Lap: Language-action pre-training enables zero-shot cross-embodiment transfer. *arXiv preprint arXiv:2602.10556*, 2026.

- [14] J. Zheng, J. Li, D. Liu, Y. Zheng, Z. Wang, Z. Ou, Y. Liu, J. Liu, Y.-Q. Zhang, and X. Zhan. Universal actions for enhanced embodied foundation models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22508–22519, 2025.
- [15] A. Patel and S. Song. Get-zero: Graph embodiment transformer for zero-shot embodiment generalization. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14262–14269. IEEE, 2025.
- [16] A. Gupta, L. Fan, S. Ganguli, and L. Fei-Fei. Metamorph: Learning universal controllers with transformers. *arXiv preprint arXiv:2203.11931*, 2022.
- [17] M. Xu, H. Zhang, Y. Hou, Z. Xu, L. Fan, M. Veloso, and S. Song. Dexumi: Using human hand as the universal manipulation interface for dexterous manipulation. *arXiv preprint arXiv:2505.21864*, 2025.
- [18] H. Ha, Y. Gao, Z. Fu, J. Tan, and S. Song. Umi on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers. *arXiv preprint arXiv:2407.10353*, 2024.
- [19] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.
- [20] C. Sferrazza, D.-M. Huang, F. Liu, J. Lee, and P. Abbeel. Body transformer: Leveraging robot embodiment for policy learning. *arXiv preprint arXiv:2408.06316*, 2024.
- [21] N. Bohlinger, G. Czechmanowski, M. Krupka, P. Kicki, K. Walas, J. Peters, and D. Tateo. One policy to run them all: an end-to-end learning approach to multi-embodiment locomotion. *arXiv preprint arXiv:2409.06366*, 2024.
- [22] H. Luo, W. Zhang, Y. Feng, S. Zheng, H. Xu, C. Xu, Z. Xi, Y. Fu, and Z. Lu. Being-h0. 7: A latent world-action model from egocentric videos. *arXiv preprint arXiv:2605.00078*, 2026.
- [23] R. Yang, Q. Yu, Y. Wu, R. Yan, B. Li, A.-C. Cheng, X. Zou, Y. Fang, X. Cheng, R.-Z. Qiu, et al. Egovla: Learning vision-language-action models from egocentric human videos. *arXiv preprint arXiv:2507.12440*, 2025.
- [24] R. Zheng, D. Niu, Y. Xie, J. Wang, M. Xu, Y. Jiang, F. Castañeda, F. Hu, Y. L. Tan, L. Fu, et al. Egoscale: Scaling dexterous manipulation with diverse egocentric human data. *arXiv preprint arXiv:2602.16710*, 2026.
- [25] A. Wagenmaker, M. Nakamoto, Y. Zhang, S. Park, W. Yagoub, A. Nagabandi, A. Gupta, and S. Levine. Steering your diffusion policy with latent space reinforcement learning. *arXiv preprint arXiv:2506.15799*, 2025.
- [26] Y. Liu, J. Hamid, A. Xie, Y. Lee, M. Du, and C. Finn. Bidirectional decoding: Improving action chunking via guided test-time sampling. In *International Conference on Learning Representations*, volume 2025, pages 4594–4627, 2025.
- [27] R. Römer, J. Balletshofer, J. Thumm, M. Pavone, A. P. Schoellig, and M. Althoff. From demonstrations to safe deployment: Path-consistent safety filtering for diffusion policies. *arXiv preprint arXiv:2511.06385*, 2025.
- [28] J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.
- [29] M. Reuss, M. Li, X. Jia, and R. Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023.
- [30] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

- [31] K. M. Lee, S. Ye, Q. Xiao, Z. Wu, Z. Zaidi, D. B. D’Ambrosio, P. R. Sanketi, and M. C. Gombolay. Learning diverse robot striking motions with diffusion models and kinematically constrained gradient guidance. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12017–12024, 2025. doi:10.1109/ICRA55743.2025.11127310.
- [32] W. Xiao, T.-H. Wang, C. Gan, R. Hasani, M. Lechner, and D. Rus. Safediffuser: Safe planning with diffusion probabilistic models. In *International Conference on Learning Representations*, 2025.
- [33] J. Zhang, L. Zhao, A. Papachristodoulou, and J. Umenberger. Constrained diffusers for safe planning and control. *Advances in Neural Information Processing Systems*, 38:34965–34998, 2026.
- [34] Y. Zhong, Q. Jiang, J. Yu, and Y. Ma. Dexgrasp anything: Towards universal robotic dexterous grasping with physics awareness. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22584–22594, 2025.
- [35] Z. Weng, H. Lu, D. Kragic, and J. Lundell. Dexdiffuser: Generating dexterous grasps with diffusion models. *IEEE Robotics and Automation Letters*, 9(12):11834–11840, 2024.
- [36] Y. Jia, Y. Jiang, K. Lv, Y. Ren, and X. Li. Arm-aware guided dexterous grasp generation with arm-agnostic grasp models. *IEEE Robotics and Automation Letters*, 11(5):5875–5882, 2026. doi:10.1109/LRA.2026.3674025.
- [37] M. Du and S. Song. Dynaguide: Steering diffusion policies with active dynamic guidance. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS)*, 2025.
- [38] Y. Wang, L. Wang, Y. Du, B. Sundaralingam, X. Yang, Y.-W. Chao, C. Pérez-D’Arpino, D. Fox, and J. Shah. Inference-time policy steering through human interactions. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15626–15633. IEEE, 2025.
- [39] H. Gupta, X. Guo, H. Ha, C. Pan, M. Cao, D. Lee, S. Scherer, S. Song, and G. Shi. Umi-on-air: Embodiment-aware guidance for embodiment-agnostic visuomotor policies. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*, 2026. URL <https://arxiv.org/abs/2510.02614>.
- [40] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada. Control barrier functions: Theory and applications. In *2019 18th European control conference (ECC)*, pages 3420–3431. Ieee, 2019.
- [41] S. Hu, Z. Liu, S. Liu, J. Cen, Z. Meng, and X. He. Vlsa: Vision-language-action models with plug-and-play safety constraint layer. *arXiv preprint arXiv:2512.11891*, 2025.
- [42] L. Brunke, Y. Zhang, R. Römer, J. Naimer, N. Staykov, S. Zhou, and A. P. Schoellig. Semantically safe robot manipulation: From semantic scene understanding to motion safeguards. *IEEE Robotics and Automation Letters*, 2025.
- [43] K. P. Wabersich and M. N. Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. *Automatica*, 129:109597, 2021.
- [44] S. Gros, M. Zanon, and A. Bemporad. Safe reinforcement learning via projection on a safe set: How to achieve optimality? *IFAC-PapersOnLine*, 53(2):8076–8081, 2020.
- [45] X. Zhai, B. Ou, Y. Wang, H. Y. Leong, Q. Yu, C. Hao, and Y. Liu. Cofreevla: Collision-free dual-arm manipulation via vision-language-action model and risk estimation. *arXiv preprint arXiv:2601.21712*, 2026.

- [46] H. Li, Q. Feng, Z. Zheng, J. Feng, Z. Chen, and A. Knoll. Language-guided object-centric diffusion policy for generalizable and collision-aware manipulation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12834–12841. IEEE, 2025.
- [47] H. Deng, W. Guo, Q. Wang, Z. Wu, and Z. Wang. Safebimanual: Diffusion-based trajectory optimization for safe bimanual manipulation. *arXiv preprint arXiv:2508.18268*, 2025.
- [48] A. Dastider, H. Fang, and M. Lin. Apex: Ambidextrous dual-arm robotic manipulation using collision-free generative diffusion models. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9526–9533. IEEE, 2024.
- [49] K. Lv, M. Yu, Y. Jia, C. Zhang, and X. Li. Kinematics-aware diffusion policy with consistent 3d observation and action space for whole-arm robotic manipulation. *IEEE Robotics and Automation Letters*, 2026. doi:10.1109/LRA.2026.3685437.
- [50] Q. Lv, H. Li, X. Deng, R. Shao, Y. Li, J. Hao, L. Gao, M. Y. Wang, and L. Nie. Spatial-temporal graph diffusion policy with kinematic modeling for bimanual robotic manipulation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 17394–17404, 2025.
- [51] K. Chen, Z. Bi, G. Zhao, C. Zheng, Y. Li, H. Zhao, and J. Ma. Samp: Spatial anchor-based motion policy for collision-aware robotic manipulators. *arXiv preprint arXiv:2509.11185*, 2025.
- [52] A. Fishman, A. Walsman, M. Bhardwaj, W. Yuan, B. Sundaralingam, B. Boots, and D. Fox. Avoid everything: Model-free collision avoidance with expert-guided fine-tuning. In *CoRL Workshop on Safe and Robust Robot Learning for Operation in the Real World*, 2024.
- [53] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019.
- [54] H. Ma, S. Bodmer, A. Carron, M. Zeilinger, and M. Muehlebach. Constraint-aware diffusion guidance for robotics: Real-time obstacle avoidance for autonomous racing. *arXiv preprint arXiv:2505.13131*, 2025.
- [55] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, X. Yuan, C. Bao, X. Lin, Y. Liu, T. kai Chan, Y. Gao, X. Li, T. Mu, N. Xiao, A. Gurha, V. N. Rajesh, Y. W. Choi, Y.-R. Chen, Z. Huang, R. Calandra, R. Chen, S. Luo, and H. Su. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *Robotics: Science and Systems*, 2025.

## A Method Details

### A.1 Cartesian Action Conversion and Pose-Twist Mapping

The main text writes the Cartesian denoising residual compactly as  $\mathcal{T}_{\text{pose}}(a_{t,i}^{-1}a_{t-1,i})$ . Here we make this notation explicit for the 9D pose component of the Cartesian action. Let the chunk-start end-effector pose be  $T_0 = (p_0, R_0)$  and let  $a_i = [\delta p_i, r_i]$ , where  $r_i \in \mathbb{R}^6$  is the continuous 6D rotation representation. We convert  $a_i$  into a world-frame pose through

$$\Phi(a_i; T_0) = (p_i, R_i), \quad p_i = p_0 + R_0 \delta p_i, \quad R_i = R_0 \phi_{6D}(r_i), \quad (8)$$

where  $\phi_{6D}(\cdot)$  maps the 6D rotation representation to  $SO(3)$  [53]. The shorthand  $a_{t,i}^{-1}a_{t-1,i}$  in the main text therefore denotes the relative transform

$$\Phi(a_{t,i}; T_0)^{-1} \Phi(a_{t-1,i}; T_0) = (\Delta p_{t,i}, \Delta R_{t,i}). \quad (9)$$

The pose-twist operator then returns

$$\xi_{t,i} = \mathcal{T}_{\text{pose}} \left( \Phi(a_{t,i}; T_0)^{-1} \Phi(a_{t-1,i}; T_0) \right) = \begin{bmatrix} \Delta p_{t,i} \\ \text{Log}(\Delta R_{t,i}) \end{bmatrix}, \quad (10)$$

where  $\text{Log} : SO(3) \rightarrow \mathbb{R}^3$  is the rotation-vector map.

The joint-space initialization follows (3). In experiments,  $q^{\text{start}}$  is the current robot joint configuration at the beginning of the action chunk. Cartesian Gaussian noise is converted relative to  $T_0$ , represented as a pose twist, and locally mapped to joint perturbations with the damped Jacobian pseudo-inverse. During denoising, the local kinematic approximation  $\xi_{t,i} \approx \mathcal{J}_{\mathcal{E}}(q_{t,i}) \Delta q_{t,i}$  yields (4). The per-joint clipping in Eqs. (3) and (4) is used to avoid large local linearization errors and unstable motions near singularities.

### A.2 Whole-Body SDF Representation

The whole-body safety function  $h(q)$  is computed through cuRobo’s collision geometry and signed-distance interfaces. The robot body is approximated by a set of collision spheres attached to the robot links, and we query the SDF value and gradient of these spheres with respect to known obstacle geometry. Let  $d_j(q)$  denote the signed distance returned for the  $j$ -th sphere at configuration  $q$ , where the convention is that positive values indicate penetration and negative values indicate clearance. Under this convention, the exact worst signed distance over all robot collision spheres is

$$d_{\max}(q) = \max_j d_j(q). \quad (11)$$

Equivalently, under the paper’s convention that positive  $h(q)$  indicates clearance and negative values indicate penetration, the exact whole-body clearance is  $h_{\text{exact}}(q) = -d_{\max}(q)$ .

Using the exact max gives the correct worst-sphere value but can introduce abrupt gradient switches when different collision spheres become the maximum. For smoother guidance gradients, we use a normalized smooth maximum over only the top- $k$  largest cuRobo signed distances. Let  $\mathcal{K}(q)$  be the indices of the  $k$  largest values in  $\{d_j(q)\}_j$ , and let  $m(q) = \max_{j \in \mathcal{K}(q)} d_j(q)$ . We compute

$$\tilde{d}_{\max}(q) = m(q) + \frac{1}{\tau} \log \left( \frac{1}{k} \sum_{j \in \mathcal{K}(q)} \exp(\tau [d_j(q) - m(q)]) \right), \quad (12)$$

where  $\tau$  is the smoothmax temperature. The CBF safety value used by the QP is

$$h(q) = -\tilde{d}_{\max}(q). \quad (13)$$

Because (12) is a smooth approximation to the exact max over spheres, it can introduce a small bias in the estimated worst distance. We use it because the smoother gradient is more stable for local guidance than the discontinuous gradient of the exact max. If the opposite SDF sign convention is used, the same idea becomes a top- $k$  smooth minimum over clearances.

In the experiments, obstacles are modeled as known cuboids in both simulation and real-world evaluation. We therefore assume access to obstacle pose and shape during deployment; perception of unknown obstacles is outside the scope of this work. Fig. 7 illustrates the representation: the physical obstacle layout is converted into known cuboids, while the robot body is approximated by link-attached collision spheres for whole-body SDF queries.

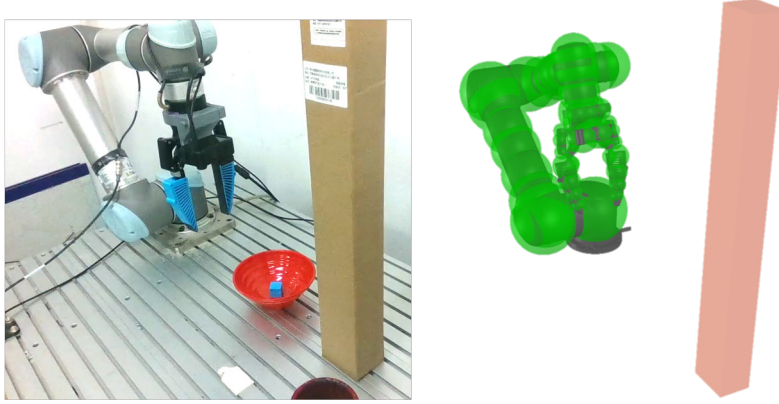


Figure 7: Whole-body SDF representation. The left image shows the real scene. The right image shows the corresponding cuRobo scene used for SDF queries, where the robot is represented by collision spheres and the obstacle is represented as a cuboid.

### A.3 CBF-Inspired QP Details

The main text derives the CBF-inspired nonlinear guidance objective and its linearized QP form. Here we provide additional details omitted for space. For each horizon step, (6) can be written as

$$\min_{\Delta q} \frac{1}{2} \Delta q^\top H \Delta q, \quad \text{s.t.} \quad a^\top \Delta q \geq b, \quad (14)$$

where

$$a = \nabla_q h(q_{t-1,i}^{\text{diff}}), \quad b = \gamma_t [d_{\text{safe}} - h(q_{t-1,i}^{\text{diff}})], \quad H = J_{\mathcal{E}}^\top W J_{\mathcal{E}} + \lambda_{\text{cbf}} I. \quad (15)$$

The scalar  $\gamma_t$  is the scheduled constraint strength at diffusion timestep  $t$ . Consistent with the reverse denoising update in the main text,  $t$  decreases from  $T - 1$  to 0 during inference. Following [54], we use

$$\gamma_t = \gamma \cdot \sigma\left(\beta \cdot \left(c - \frac{t}{T-1}\right)\right), \quad \sigma(z) = \frac{1}{1 + \exp(-z)}, \quad (16)$$

where  $\gamma$  is the base constraint strength. We set the slope  $\beta$  to 50 and the transition point  $c$  to 0.7 in all experiments. This schedule gives smaller constraint strength in early denoising steps and approaches  $\gamma$  in later steps, when samples are closer to the data manifold and less likely to be washed out by subsequent denoising updates. Fig. 8 visualizes this schedule for the default  $T = 16$  denoising steps and  $\gamma = 1.0$ . The QP uses the nonnegative right-hand side  $\max(b, 0)$ , so configurations already satisfying the safety margin receive zero correction from the constraint.

The closed-form solution follows from the KKT conditions of the single-constraint QP. With the clipped right-hand side  $\bar{b} = \max(b, 0)$ , the Lagrangian is

$$\mathcal{L}(\Delta q, \mu) = \frac{1}{2} \Delta q^\top H \Delta q + \mu(\bar{b} - a^\top \Delta q), \quad \mu \geq 0. \quad (17)$$

Stationarity gives  $H \Delta q - \mu a = 0$ , hence  $\Delta q = \mu H^{-1} a$ . If  $\bar{b} = 0$ , the unconstrained minimizer  $\Delta q = 0$  is feasible. Otherwise, complementary slackness makes the constraint active, so  $a^\top \Delta q =$

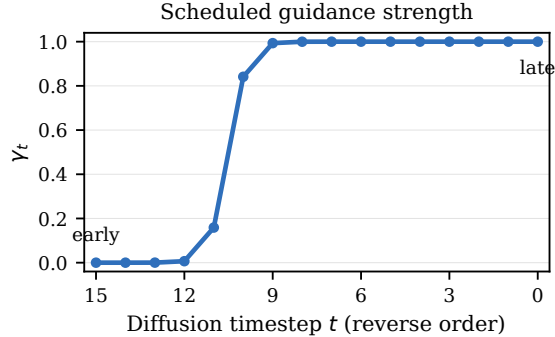


Figure 8: Scheduled constraint strength  $\gamma_t$  over reverse denoising steps. The guidance is weak in early noisy steps and approaches the base scale  $\gamma$  in later steps.

$\mu a^\top H^{-1} a = \bar{b}$  and  $\mu = \bar{b} / (a^\top H^{-1} a)$ . The resulting solution is

$$\Delta q^* = \begin{cases} 0, & b \leq 0, \\ \frac{b}{a^\top H^{-1} a + \varepsilon} H^{-1} a, & b > 0, \end{cases} \quad (18)$$

where  $\varepsilon$  is a small numerical stabilizer. The case  $b \leq 0$  corresponds to a locally satisfied safety constraint, for which the unconstrained minimizer  $\Delta q = 0$  is feasible. The update is then clipped as in (7). The QP is solved over the arm joints only; the gripper command is not modified by the collision-avoidance correction. The task-preservation metric uses the end-effector geometric Jacobian with separate position and rotation weights. Because the method uses a local linearization and clipped updates, the guidance should be interpreted as an efficient feasible correction direction rather than a formal global safety certificate.

## B Implementation Details

### B.1 Diffusion Policy Architecture

We use the same Cartesian diffusion-policy architecture across all embodiments. In both simulation and real-world experiments, RGB observations are encoded with the pretrained CLIP ViT-B/16 backbone (`vit_base_patch16_clip_224_openai`) loaded through `timm` and finetuned during policy training. The resulting observation feature is used as a global condition for a conditional 1D U-Net denoiser over the action chunk. The denoiser takes a noisy Cartesian action sequence and a diffusion timestep as input, and predicts the noise  $\epsilon_\theta$  used by the DDPM reverse step. The action sequence has horizon  $H$  and action dimension 10, consisting of relative end-effector translation, a continuous 6D rotation representation, and one gripper command. During deployment, EMBODISTEER keeps this trained denoiser unchanged and only changes the sampling variable from Cartesian actions to target-robot joint trajectories.

### B.2 Hyperparameters

Table 2 lists the hyperparameters used for training and deployment.

### B.3 Runtime Breakdown

Fig. 9 reports the measured runtime breakdown of one guided joint-space inference call on an RTX 4070 Ti SUPER, including all 16 reverse-diffusion steps. The full EMBODISTEER inference time is 103 ms, corresponding to 9.61 Hz deployment. The base Cartesian policy already requires observation encoding and U-Net denoising, which account for 46.8 ms. The remaining measured time

Table 2: Training and deployment hyperparameters.

Quantity	Value	Notes
Observation horizon	2	Diffusion policy input setting.
Action horizon $H$	16	Predicted action chunk length.
Execution horizon	16 in simulation, 6 in real world.	Number of actions executed before replanning.
Visual backbone	CLIP ViT-B/16	<code>vit_base_patch16_clip_224_openai</code> loaded through <code>timm</code> , pretrained and finetuned.
U-Net channels	[256, 512, 1024]	Conditional 1D U-Net downsampling dimensions.
Diffusion step embedding	128	Timestep embedding dimension.
Diffusion training timesteps	50	DDIM scheduler with squared-cosine beta schedule and epsilon prediction.
Inference denoising steps	16	Reverse diffusion steps at deployment.
Control frequency	10	Simulation and real-world control rate.
Training epochs	120	Used for both simulation and real-world policies.
Batch size	128 in simulation, 64 in real world.	Training batch size for simulation and real-world checkpoints.
Optimizer	AdamW	Learning rate $3 \times 10^{-4}$ , betas (0.95, 0.999), weight decay $10^{-6}$ .
Learning-rate schedule	Cosine	2000 warmup steps.
EMA	0.9999 max decay	EMA enabled during training.
Image augmentation	Crop + color jitter	Random crop ratio 0.95; brightness 0.3, contrast 0.4, saturation 0.5, hue 0.08.
Noise initialization scale $\alpha$	0.1	Scale for Jacobian-projected noise.
Jacobian damping $\lambda_{pinv}$	0.001	Used in $J_{\mathcal{E}}^+$ .
Joint update clip $\Delta q_{max}$	0.5	Used in joint denoising update.
Safety margin $d_{safe}$	0.05~0.10	Task-dependent; set to 0.05, 0.07, or 0.10 by obstacle layout.
SDF aggregation top- $k$	4	Number of critical collision spheres.
Smoothmax temperature $\tau$	20.0	Temperature in (12).
Base constraint strength $\gamma$	1.0	Base multiplier in (16).
QP regularization $\lambda_{cbf}$	0.01	Joint-space regularization.
Task position weight	1.0	Position rows in $J_{\mathcal{E}}^T W J_{\mathcal{E}}$ .
Task rotation weight	0.1	Rotation rows in $J_{\mathcal{E}}^T W J_{\mathcal{E}}$ .
CBF update clip $\Delta q_{max}^{cbf}$	0.1	Clipping applied to the guidance update.

is introduced by joint-space lifting, whole-body SDF queries, CBF-QP guidance, and associated runtime overhead.

#### B.4 Baseline Implementation Details

Table 3 summarizes how each baseline is implemented. All baselines use the same trained Cartesian policy checkpoint for a given task.

## C Simulation Results

### C.1 Task Definitions and Evaluation Protocol

The simulation benchmark contains three manipulation tasks: PLACE\_TOAST, TURN\_ON\_FAUCET, and MAKE\_COFFEE. Cartesian policies are trained from obstacle-free floating-gripper demonstrations generated by task-specific motion planning, and are evaluated zero-shot on robot embodiments with different kinematics and arm geometries. In obstacle-present settings, obstacles are introduced only at test time. Success rate (TSR), reward per episode (RWD), and collision occurrence rate (COR) are reported as in the main text.

Component	Included operations	ms	%
<i>Base Cartesian policy components</i>			
Diffusion U-Net	U-Net denoising, 16 steps	41.2	39.9
Observation Encoder	Image/state observation encoding	5.6	5.4
<b>Base policy subtotal</b>	Measured base-policy components	<b>46.8</b>	<b>45.4</b>
<i>Additional EMBODISTEER components</i>			
Kinematic Mapping	FK, Jacobian, DLS, twist	13.5	13.1
CBF Guidance	CBF autograd, QP, linearization setup	11.5	11.2
Collision SDF Query	Sphere FK, SDF query, world setup	2.8	2.7
Pose / Norm. / Scheduler	Pose conversion, normalizer, scheduler step	11.1	10.8
I/O and Action Formatting	Obs. preprocessing, joint tensor, action conversion	1.0	1.0
Other Runtime Overhead	Uninstrumented loop/PyTorch overhead	16.3	15.8
<b>Added subtotal</b>	Joint-space lifting and guidance overhead	<b>56.2</b>	<b>54.6</b>
<b>Full total</b>	Guided joint-space inference	<b>103.0</b>	<b>100.0</b>

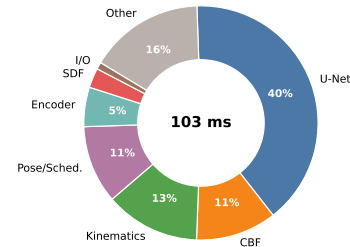


Figure 9: Runtime breakdown for one guided joint-space inference call on an RTX 4070 Ti SUPER.

Table 3: Baseline implementation details.

Method	Implementation
<i>EE</i>	Runs the Cartesian diffusion policy in its original action space and executes the predicted end-effector actions without collision-aware guidance.
<i>Joint</i>	Uses the same joint-space denoising procedure as EMBODISTEER, including FK queries to the frozen Cartesian denoiser and damped Jacobian updates, but does not apply collision guidance.
<i>EE w/ Sampling</i>	Keeps Cartesian denoising unchanged, samples multiple candidate action chunks, realizes them through the target robot, and selects the candidate with the largest clearance. Number of candidates: 16.
<i>EE w/ CBF</i>	Keeps Cartesian denoising unchanged, solves IK for the generated action chunk, and applies a one-shot CBF-QP correction after generation.
<i>Joint w/ CG</i>	Adds collision-cost gradient guidance during joint-space denoising. Unlike EMBODISTEER, the update is not explicitly regularized by the task-space QP objective, so whole-body collision gradients can perturb the end-effector trajectory.

As illustrated in Fig. 3, the benchmark uses the same three tasks and obstacle layouts across 9 robot embodiments. We therefore avoid duplicating the task and robot lists here; the appendix focuses on the complete per-robot quantitative results and guidance ablations.

## C.2 Full Quantitative Results

Table 4 reports the per-robot simulation results behind Table 1. For EMBODISTEER, the listed guidance strengths match the main-paper setting for each task.

Table 4: Full per-robot simulation results. Obs. indicates whether obstacles are present during evaluation; TSR is success-once rate, RWD is maximum reward per episode, and COR is collision occurrence rate. COR is reported only for obstacle-present settings.

Obs.	Method	Robot	TSR $\uparrow$	RWD $\uparrow$	COR $\downarrow$	Robot	TSR $\uparrow$	RWD $\uparrow$	COR $\downarrow$	Robot	TSR $\uparrow$	RWD $\uparrow$	COR $\downarrow$
<b>PLACE/TOAST</b>													
w/o Obs.	EE	UR5	100.0%	1.000	–	Panda	100.0%	1.000	–	xArm6	100.0%	1.000	–
		xArm7	98.0%	0.989	–	iiwa7	98.0%	0.989	–	Gen3-6D	81.0%	0.879	–
		Gen3-7D	98.0%	0.989	–	Rizon4	99.0%	0.991	–	Sawyer	97.0%	0.984	–
		<b>Avg.</b>	<b>96.8%</b>	<b>0.980</b>	–								

Continued on next page

Obs.	Method	Robot	TSR↑	RWD↑	COR↓	Robot	TSR↑	RWD↑	COR↓	Robot	TSR↑	RWD↑	COR↓
	Joint	UR5	100.0%	1.000	-	Panda	100.0%	1.000	-	xArm6	98.0%	0.988	-
		xArm7	99.0%	0.995	-	iiwa7	98.0%	0.989	-	Gen3-6D	73.0%	0.827	-
		Gen3-7D	85.0%	0.907	-	Rizon4	98.0%	0.983	-	Sawyer	99.0%	0.995	-
		<b>Avg.</b>	<b>94.4%</b>	<b>0.965</b>	<b>-</b>								
	EE	UR5	39.0%	0.560	50.0%	Panda	46.0%	0.585	54.0%	xArm6	45.0%	0.676	52.0%
		xArm7	48.0%	0.695	56.0%	iiwa7	47.0%	0.689	49.0%	Gen3-6D	36.0%	0.611	51.0%
		Gen3-7D	43.0%	0.665	51.0%	Rizon4	42.0%	0.552	51.0%	Sawyer	45.0%	0.676	59.0%
		<b>Avg.</b>	<b>43.4%</b>	<b>0.634</b>	<b>52.6%</b>								
w/ Obs.	EE w/ Samp.	UR5	48.0%	0.626	62.0%	Panda	54.0%	0.651	46.0%	xArm6	55.0%	0.736	49.0%
		xArm7	52.0%	0.717	50.0%	iiwa7	44.0%	0.663	40.0%	Gen3-6D	36.0%	0.613	47.0%
		Gen3-7D	45.0%	0.677	53.0%	Rizon4	48.0%	0.601	62.0%	Sawyer	42.0%	0.661	53.0%
		<b>Avg.</b>	<b>47.1%</b>	<b>0.661</b>	<b>51.3%</b>								
	EE w/ CBF	UR5	76.0%	0.797	11.0%	Panda	64.0%	0.692	17.0%	xArm6	75.0%	0.849	8.0%
		xArm7	66.0%	0.796	14.0%	iiwa7	62.0%	0.768	14.0%	Gen3-6D	62.0%	0.766	5.0%
		Gen3-7D	68.0%	0.809	13.0%	Rizon4	50.0%	0.580	37.0%	Sawyer	69.0%	0.816	19.0%
		<b>Avg.</b>	<b>65.8%</b>	<b>0.764</b>	<b>15.3%</b>								
	Joint w/ CG	UR5	35.0%	0.386	0.0%	Panda	76.0%	0.789	0.0%	xArm6	62.0%	0.752	0.0%
		xArm7	67.0%	0.786	0.0%	iiwa7	44.0%	0.628	1.0%	Gen3-6D	54.0%	0.699	0.0%
		Gen3-7D	57.0%	0.721	0.0%	Rizon4	62.0%	0.665	0.0%	Sawyer	62.0%	0.753	1.0%
		<b>Avg.</b>	<b>57.7%</b>	<b>0.687</b>	<b>0.2%</b>								
	Ours	UR5	78.0%	0.807	0.0%	Panda	82.0%	0.842	3.0%	xArm6	77.0%	0.859	0.0%
		xArm7	68.0%	0.805	0.0%	iiwa7	84.0%	0.905	9.0%	Gen3-6D	54.0%	0.712	0.0%
		Gen3-7D	76.0%	0.853	1.0%	Rizon4	82.0%	0.849	16.0%	Sawyer	72.0%	0.831	12.0%
		<b>Avg.</b>	<b>74.8%</b>	<b>0.829</b>	<b>4.6%</b>								

**TURNONFAUCET**

w/o Obs.	EE	Panda	81.0%	0.943	-	xArm6	99.0%	0.998	-	xArm7	78.0%	0.937	-
		UR5	90.0%	0.918	-	iiwa7	87.0%	0.964	-	Gen3-6D	96.0%	0.988	-
		Gen3-7D	53.0%	0.880	-	Rizon4	73.0%	0.822	-	Sawyer	95.0%	0.987	-
		<b>Avg.</b>	<b>83.6%</b>	<b>0.937</b>	<b>-</b>								
	Joint	UR5	88.0%	0.912	-	Panda	90.0%	0.974	-	xArm6	100.0%	1.000	-
		xArm7	90.0%	0.971	-	iiwa7	91.0%	0.975	-	Gen3-6D	95.0%	0.988	-
		Gen3-7D	63.0%	0.909	-	Rizon4	87.0%	0.952	-	Sawyer	87.0%	0.936	-
		<b>Avg.</b>	<b>87.9%</b>	<b>0.957</b>	<b>-</b>								
	EE	UR5	11.0%	0.417	86.0%	Panda	23.0%	0.471	94.0%	xArm6	26.0%	0.791	89.0%
		xArm7	41.0%	0.823	61.0%	iiwa7	66.0%	0.915	28.0%	Gen3-6D	73.0%	0.931	41.0%
		Gen3-7D	42.0%	0.860	40.0%	Rizon4	12.0%	0.657	88.0%	Sawyer	80.0%	0.904	29.0%
		<b>Avg.</b>	<b>41.6%</b>	<b>0.752</b>	<b>61.8%</b>								
w/ Obs.	EE w/ Samp.	UR5	6.0%	0.453	86.0%	Panda	35.0%	0.554	87.0%	xArm6	31.0%	0.811	84.0%
		xArm7	32.0%	0.789	47.0%	iiwa7	69.0%	0.911	18.0%	Gen3-6D	74.0%	0.935	28.0%
		Gen3-7D	50.0%	0.874	21.0%	Rizon4	13.0%	0.665	90.0%	Sawyer	85.0%	0.911	18.0%
		<b>Avg.</b>	<b>43.9%</b>	<b>0.767</b>	<b>53.2%</b>								
	EE w/ CBF	UR5	33.0%	0.506	29.0%	Panda	48.0%	0.661	46.0%	xArm6	60.0%	0.857	36.0%
		xArm7	55.0%	0.833	26.0%	iiwa7	78.0%	0.945	9.0%	Gen3-6D	87.0%	0.942	12.0%
		Gen3-7D	48.0%	0.851	1.0%	Rizon4	8.0%	0.632	97.0%	Sawyer	90.0%	0.949	13.0%
		<b>Avg.</b>	<b>56.3%</b>	<b>0.797</b>	<b>29.9%</b>								
	Joint w/ CG	UR5	8.0%	0.448	88.0%	Panda	7.0%	0.339	82.0%	xArm6	6.0%	0.432	75.0%
		xArm7	0.0%	0.239	65.0%	iiwa7	12.0%	0.501	42.0%	Gen3-6D	33.0%	0.667	30.0%
		Gen3-7D	11.0%	0.481	23.0%	Rizon4	2.0%	0.173	70.0%	Sawyer	14.0%	0.350	61.0%
		<b>Avg.</b>	<b>10.3%</b>	<b>0.403</b>	<b>59.6%</b>								
	Ours	UR5	32.0%	0.502	23.0%	Panda	57.0%	0.802	42.0%	xArm6	75.0%	0.900	30.0%
		xArm7	52.0%	0.842	28.0%	iiwa7	76.0%	0.926	17.0%	Gen3-6D	90.0%	0.971	7.0%
		Gen3-7D	61.0%	0.897	2.0%	Rizon4	15.0%	0.744	88.0%	Sawyer	87.0%	0.916	7.0%
		<b>Avg.</b>	<b>60.6%</b>	<b>0.833</b>	<b>27.1%</b>								

**MAKECOFFEE**

Continued on next page

Obs.	Method	Robot	TSR $\uparrow$	RWD $\uparrow$	COR $\downarrow$	Robot	TSR $\uparrow$	RWD $\uparrow$	COR $\downarrow$	Robot	TSR $\uparrow$	RWD $\uparrow$	COR $\downarrow$
w/o Obs.	EE	UR5	94.0%	0.952	-	Panda	94.0%	0.951	-	xArm6	87.0%	0.917	-
		xArm7	90.0%	0.943	-	iiwa7	85.0%	0.913	-	Gen3-6D	93.0%	0.960	-
		Gen3-7D	94.0%	0.965	-	Rizon4	88.0%	0.904	-	Sawyer	82.0%	0.881	-
							<b>Avg.</b>	<b>89.7%</b>	<b>0.932</b>	-			
Joint	UR5	94.0%	0.951	-	Panda	93.0%	0.943	-	xArm6	91.0%	0.937	-	
	xArm7	90.0%	0.936	-	iiwa7	92.0%	0.959	-	Gen3-6D	95.0%	0.975	-	
	Gen3-7D	88.0%	0.927	-	Rizon4	80.0%	0.833	-	Sawyer	84.0%	0.900	-	
						<b>Avg.</b>	<b>89.7%</b>	<b>0.929</b>	-				
w/ Obs.	EE	UR5	8.0%	0.277	85.0%	Panda	24.0%	0.408	60.0%	xArm6	27.0%	0.536	70.0%
		xArm7	28.0%	0.540	69.0%	iiwa7	26.0%	0.548	59.0%	Gen3-6D	25.0%	0.497	45.0%
		Gen3-7D	24.0%	0.513	47.0%	Rizon4	24.0%	0.385	44.0%	Sawyer	14.0%	0.401	46.0%
							<b>Avg.</b>	<b>22.2%</b>	<b>0.456</b>	<b>58.3%</b>			
	EE w/ Samp.	UR5	8.0%	0.243	87.0%	Panda	24.0%	0.393	63.0%	xArm6	24.0%	0.539	71.0%
		xArm7	30.0%	0.577	64.0%	iiwa7	39.0%	0.635	56.0%	Gen3-6D	20.0%	0.490	56.0%
		Gen3-7D	24.0%	0.523	52.0%	Rizon4	23.0%	0.384	56.0%	Sawyer	25.0%	0.516	35.0%
							<b>Avg.</b>	<b>24.1%</b>	<b>0.478</b>	<b>60.0%</b>			
	EE w/ CBF	UR5	19.0%	0.314	8.0%	Panda	62.0%	0.683	11.0%	xArm6	58.0%	0.699	14.0%
		xArm7	48.0%	0.615	2.0%	iiwa7	66.0%	0.762	4.0%	Gen3-6D	67.0%	0.769	6.0%
Gen3-7D		43.0%	0.574	4.0%	Rizon4	39.0%	0.494	21.0%	Sawyer	38.0%	0.543	7.0%	
						<b>Avg.</b>	<b>48.9%</b>	<b>0.606</b>	<b>8.6%</b>				
Joint w/ CG	UR5	13.0%	0.287	79.0%	Panda	18.0%	0.333	51.0%	xArm6	29.0%	0.528	56.0%	
	xArm7	31.0%	0.519	36.0%	iiwa7	17.0%	0.385	28.0%	Gen3-6D	31.0%	0.541	39.0%	
	Gen3-7D	33.0%	0.544	36.0%	Rizon4	5.0%	0.183	13.0%	Sawyer	25.0%	0.493	27.0%	
						<b>Avg.</b>	<b>22.4%</b>	<b>0.424</b>	<b>40.6%</b>				
Ours	UR5	28.0%	0.387	3.0%	Panda	75.0%	0.798	4.0%	xArm6	68.0%	0.757	3.0%	
	xArm7	64.0%	0.756	2.0%	iiwa7	73.0%	0.815	0.0%	Gen3-6D	69.0%	0.795	2.0%	
	Gen3-7D	56.0%	0.662	1.0%	Rizon4	38.0%	0.481	9.0%	Sawyer	44.0%	0.574	1.0%	
						<b>Avg.</b>	<b>57.2%</b>	<b>0.670</b>	<b>2.8%</b>				

### C.3 Additional Ablations and Sensitivity Analyses

We first analyze the main design choices and hyperparameters of EMBODISTEER. The constraint strength  $\gamma$  controls the strength of the CBF-QP constraint relative to the diffusion update: too small a value may leave collisions unresolved, while too large a value can over-steer the sample and hurt task execution. Following the linearized QP derivation, we use  $\gamma = 1.0$  as the default scale and sweep around this value. Fig. 10 therefore visualizes a guidance-strength sensitivity analysis across tasks.

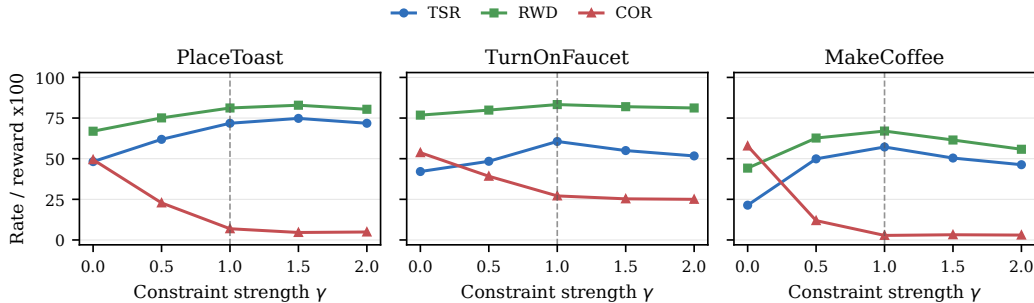


Figure 10: CBF-QP guidance-strength sensitivity analysis. The default  $\gamma = 1.0$  follows the linearized QP derivation and provides a robust task-collision tradeoff across the three tasks.

The regularization weight  $\lambda_{cbf}$  controls how much the QP penalizes joint-space deviation from the denoised sample. This term is included to prevent collision avoidance from being absorbed by unnecessarily large joint motions. Table 5 shows that performance is stable over a broad range, with the default  $\lambda_{cbf} = 0.01$  giving a balanced setting.

Table 5: CBF-QP regularization sensitivity analysis averaged over the 9 robot embodiments used in the paper. The  $\lambda_{\text{cbf}} = 0.01$  row corresponds to the default setting.

Task	$\lambda_{\text{cbf}}$	TSR $\uparrow$	RWD $\uparrow$	COR $\downarrow$
PLACETOAST	0.000	70.8%	0.800	5.9%
	0.001	70.0%	0.797	6.1%
	0.005	<b>72.4%</b>	0.809	<b>5.0%</b>
	0.010	71.8%	<b>0.812</b>	6.9%
	0.050	70.9%	0.797	5.3%
	0.100	70.1%	0.797	5.7%
	0.500	69.9%	0.797	5.9%
	1.000	70.9%	0.803	7.1%
TURNONFAUCET	0.000	57.2%	0.829	26.3%
	0.001	58.9%	<b>0.842</b>	25.7%
	0.005	58.6%	0.833	24.9%
	0.010	<b>60.6%</b>	0.833	27.1%
	0.050	57.0%	0.817	26.6%
	0.100	59.6%	0.831	25.9%
	0.500	59.7%	0.838	25.7%
	1.000	60.2%	0.835	<b>24.6%</b>
MAKECOFFEE	0.000	46.3%	0.603	3.6%
	0.001	42.4%	0.574	3.2%
	0.005	45.3%	0.591	2.6%
	0.010	<b>57.2%</b>	<b>0.670</b>	2.8%
	0.050	47.2%	0.606	<b>2.0%</b>
	0.100	42.7%	0.570	3.4%
	0.500	44.7%	0.590	3.6%
	1.000	46.4%	0.603	3.6%

We also ablate the guidance schedule by comparing the scheduled strength in (16) with a constant scale  $\gamma_t = \gamma = 1.0$  throughout denoising. The schedule is intended to avoid strong corrections in early noisy denoising steps, where samples are still far from the policy distribution. Table 6 shows that scheduling helps preserve task performance while maintaining similar collision rates.

Table 6: Guidance schedule ablation averaged over the 9 robot embodiments used in the paper. Scheduled guidance uses Eq. (16); constant guidance keeps  $\gamma_t = \gamma = 1.0$  for all denoising steps.

Task	Guidance scale	TSR $\uparrow$	RWD $\uparrow$	COR $\downarrow$
PLACETOAST	Scheduled $\gamma_t$	<b>71.8%</b>	<b>0.812</b>	6.9%
	Constant $\gamma_t$	70.2%	0.796	<b>5.6%</b>
TURNONFAUCET	Scheduled $\gamma_t$	<b>60.6%</b>	<b>0.833</b>	27.1%
	Constant $\gamma_t$	57.8%	0.827	<b>26.1%</b>
MAKECOFFEE	Scheduled $\gamma_t$	<b>57.2%</b>	<b>0.670</b>	<b>2.8%</b>
	Constant $\gamma_t$	46.2%	0.593	<b>2.8%</b>

Finally, we sweep the cost-gradient guidance scale  $\rho$  for the gradient-based baseline (*Joint w/ CG*). This verifies that the baseline is not disadvantaged by a single poorly chosen scale. As shown in Fig. 11, tuning  $\rho$  can reduce collisions, but the unconstrained gradient update often degrades task performance, especially in tasks requiring precise end-effector motion.

#### C.4 Additional Qualitative Results

Fig. 12 shows representative simulation rollouts across the three benchmark tasks. The base Cartesian policy often produces task-directed end-effector motion but cannot reason about out-of-distribution obstacles, which leads to end-effector collisions near the task object. It also ignores

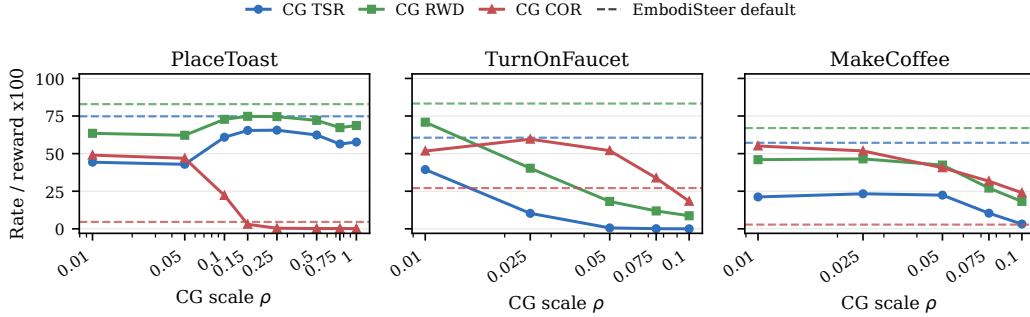


Figure 11: Cost-gradient guidance-strength sensitivity analysis. Solid curves show the CG baseline under different  $\rho$ , while dashed horizontal lines show EMBODISTEER with its default setting. Tuning  $\rho$  can reduce collisions, but CG consistently gives a worse task–collision tradeoff.

the target robot’s arm geometry, leading to arm-body obstacle contact. In contrast, EMBODISTEER preserves the same task intent while steering both the end-effector and the full robot body around the obstacle.

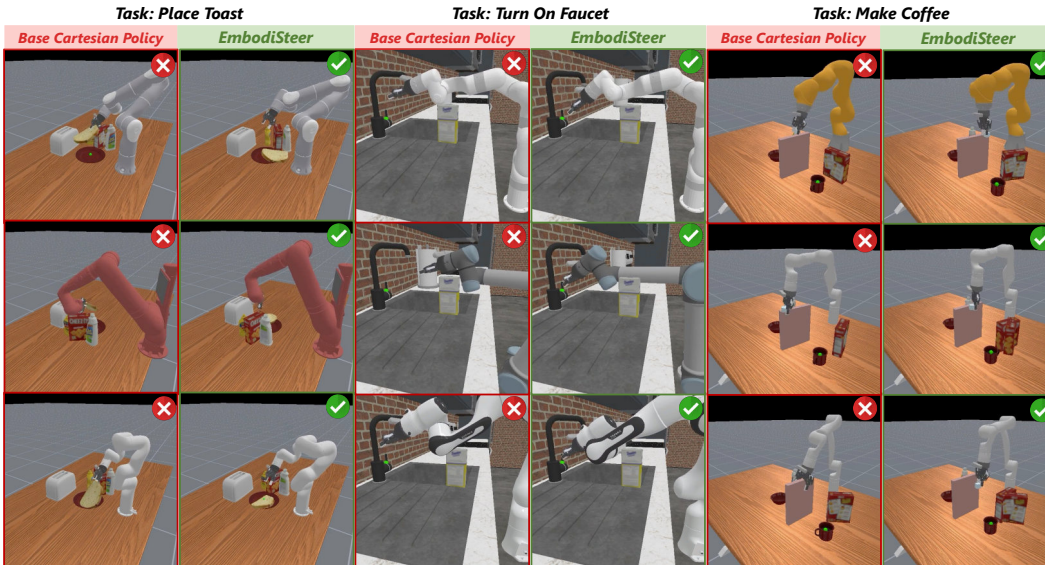


Figure 12: Additional simulation qualitative results. The base Cartesian policy can collide at the end-effector or along the arm body when obstacles are added at test time, while EMBODISTEER generates collision-aware whole-body motion and completes the task.

## D Real-World Results

### D.1 Setup and Annotation Protocol

Each real-world task uses 200 UMI handheld-gripper demonstrations collected without extra obstacles. Because these demonstrations are collected with a handheld gripper rather than a particular robot arm, the training data are embodiment-agnostic. For each task, a single Cartesian diffusion policy checkpoint is trained and shared by UR5 and Franka Panda without robot-specific finetuning. The three tasks are illustrated in Fig. 13: MAKEICEDCOFFEE requires grasping blocks from a bowl, which represent ice cubes, and placing them into a cup; PUTFLOWERINVASE requires grasping the bouquet and inserting it into the vase; and ARRANGEBANANA requires grasping a banana from a pen holder and placing it into a cup. For each robot-task pair, we evaluate 10 no-obstacle

trials and 10 obstacle-present trials distributed across the obstacle types in Fig. 14. A trial is marked successful if the task object reaches the intended final state without timeout, object drop, or human intervention.

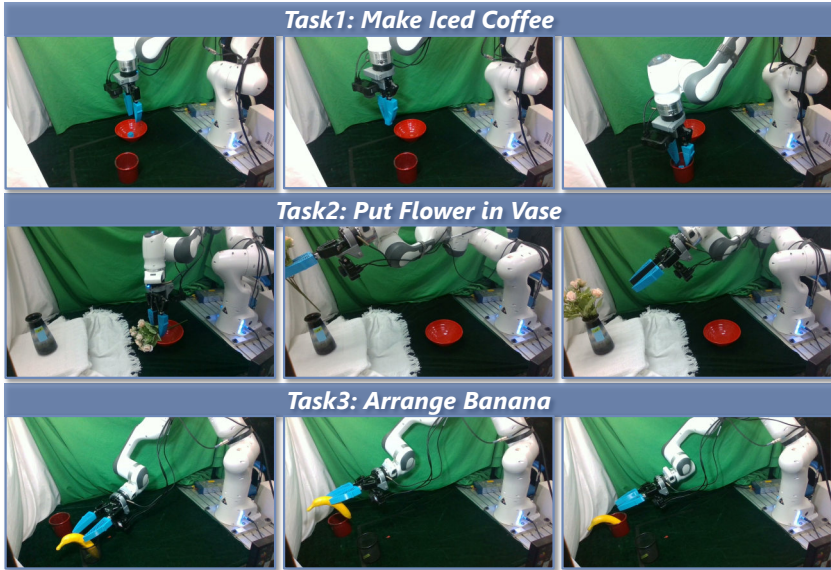


Figure 13: Real-world task protocols. Each row shows representative stages of one UMI-trained task: MAKEICEDCOFFEE, PUTFLOWERINVASE, and ARRANGEBANANA.

Task success and collision are annotated independently. For safety in the real-world setup, obstacles are movable: after contact, they can be knocked over or pushed aside, unlike the fixed obstacles in simulation that often stop the robot motion. As a result, a rollout can still finish the task after a collision. COR therefore measures whether any end-effector or arm-body obstacle contact occurs during the rollout, separately from task success.

## D.2 Full Real-World Results

Table 7 reports the real-world results by robot and task. Each task checkpoint is shared by both robots.

Table 7: Real-world results by robot and task. TSR and COR are reported as success/collision counts over 10 trials for each robot-task pair; the total row aggregates over 60 trials.

Robot	Task	No Obstacle	Obstacle, Base Policy		Obstacle, EMBODISTEER	
		TSR $\uparrow$	TSR $\uparrow$	COR $\downarrow$	TSR $\uparrow$	COR $\downarrow$
UR5	MAKEICEDCOFFEE	9/10	4/10	10/10	<b>8/10</b>	<b>1/10</b>
	PUTFLOWERINVASE	7/10	4/10	9/10	<b>6/10</b>	<b>1/10</b>
	ARRANGEBANANA	8/10	4/10	9/10	<b>9/10</b>	<b>0/10</b>
Panda	MAKEICEDCOFFEE	9/10	6/10	10/10	<b>8/10</b>	<b>1/10</b>
	PUTFLOWERINVASE	6/10	1/10	10/10	<b>6/10</b>	<b>0/10</b>
	ARRANGEBANANA	10/10	6/10	10/10	<b>10/10</b>	<b>1/10</b>
Total	All tasks	49/60	25/60	58/60	<b>47/60</b>	<b>4/60</b>

### D.3 Obstacle Layouts and Qualitative Results

The real-world obstacle layouts are shown in Fig. 14. The obstacle type and pose are varied across episodes. These layouts require the robot to avoid collisions with both the arm links and the end effector.

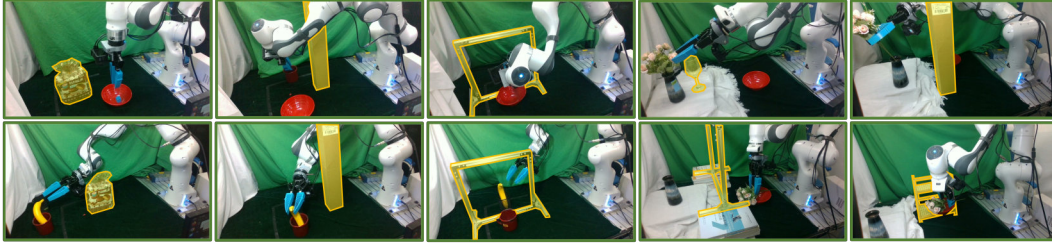


Figure 14: Real-world obstacle layouts. The obstacles are highlighted with yellow masks. The obstacle type and pose are varied across episodes.

Fig. 16 shows representative real-world rollouts for all three tasks. Each row compares the base Cartesian policy and EMBODISTEER on both UR5 and Franka Panda. The qualitative results mirror the quantitative trend in Table 7: the base policy often reaches toward the task object while contacting the obstacle, whereas EMBODISTEER modifies the whole-body motion to avoid end-effector and arm-body obstacles.

### E Failure Mode Analysis

Fig. 15 summarizes three representative failure modes observed in our experiments. First, although guidance improves collision avoidance on average, a large correction can steer the robot into joint states that are rarely covered by the demonstrations. The frozen Cartesian policy may then receive observations outside its training distribution and fail to recover task progress. Second, the CBF-QP update is based on a local linearization and clipped joint correction, so it provides a practical avoidance direction rather than a formal collision-free certificate; collisions can still occur in highly constrained layouts or when the required correction is too large. Third, some task phases are more sensitive to small end-effector deviations than others. In pick-and-place tasks, for example, guidance applied near grasping or placement can perturb precision-critical motion and cause grasp misses, drops, or misplacement, even when the resulting motion avoids obstacles.

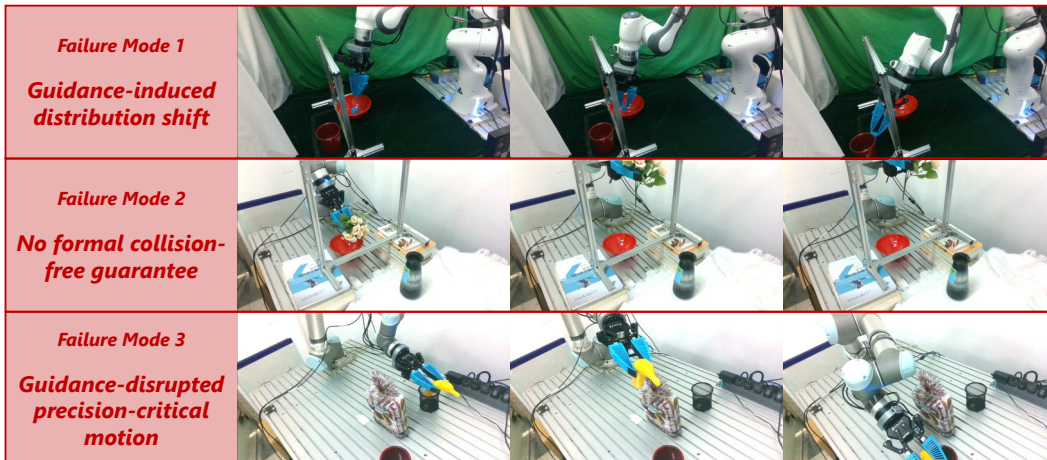


Figure 15: Representative failure modes. Guidance may move the robot into out-of-distribution states, fail to fully prevent collision under local linearization and clipping, or disrupt precision-critical pick/place motions.

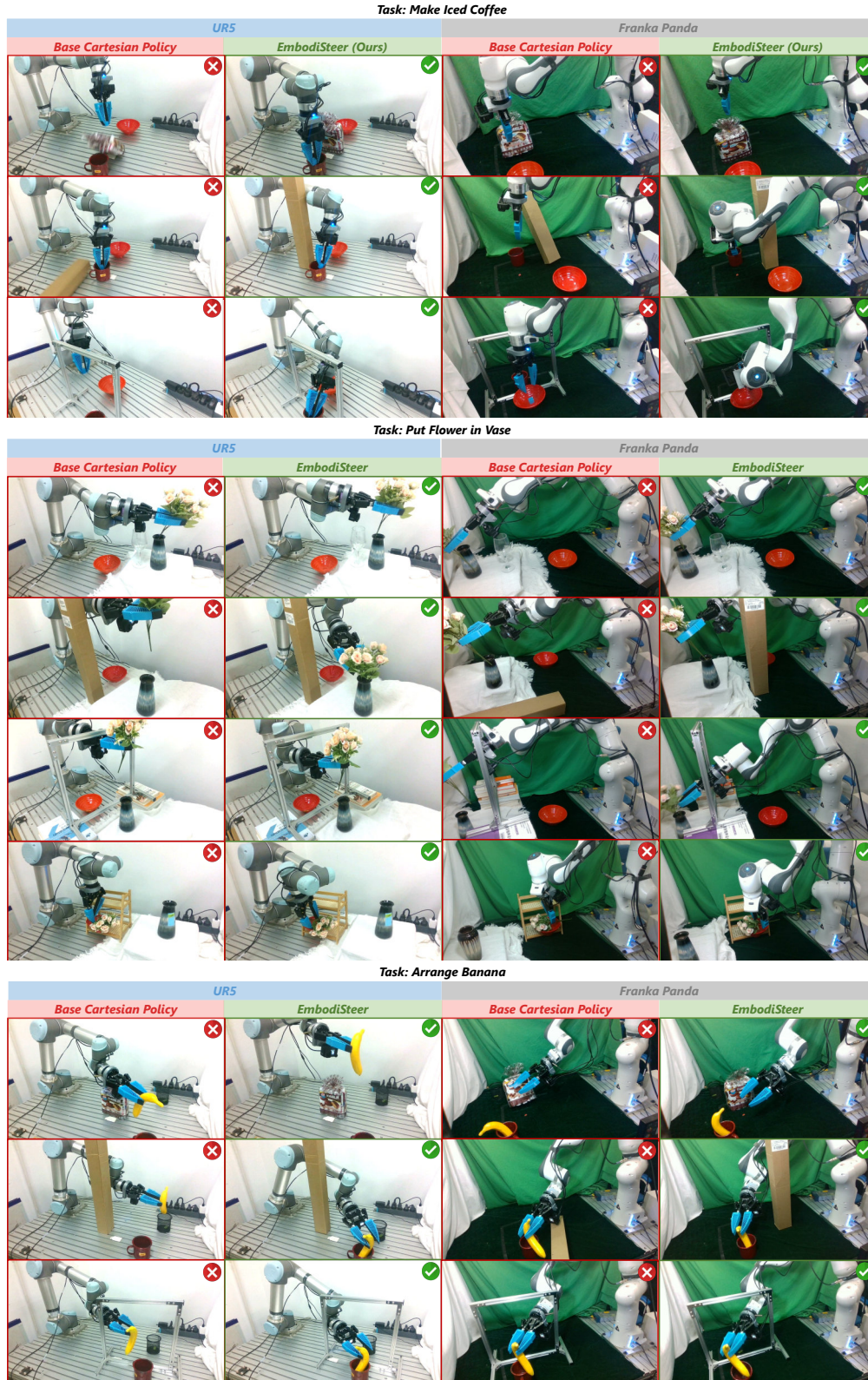


Figure 16: Real-world qualitative results. From top to bottom: MAKEICEDCOFFEE, PUTFLOWER-INVASE, and ARRANGEBANANA. Each row compares the base Cartesian policy and EMBODISTEER on UR5 and Franka Panda under obstacle layouts that test end-effector and arm-body avoidance.