SafeBimanual: Diffusion-based Trajectory Optimization for Safe Bimanual Manipulation

Haoyuan Deng[1], **Wenkai Guo**[1], **Qianzhun Wang**[1], **Zhenyu Wu**[2], **Ziwei Wang**[1]*
[1] Nanyang Technological University, [2] Beijing University of Posts and Telecommunications
haoyuan.deng@ntu.edu.sg, ziwei.wang@ntu.edu.sg

Abstract: Bimanual manipulation has been widely applied in household services and manufacturing, which enables the complex task completion with coordination requirements. Recent diffusion-based policy learning approaches have achieved promising performance in modeling action distributions for bimanual manipulation. However, they ignored the physical safety constraints of bimanual manipulation, which leads to the dangerous behaviors with damage to robots and objects. To this end, we propose a test-time trajectory optimization framework named **SafeBimanual** for any pre-trained diffusion-based bimanual manipulation policies, which imposes the safety constraints on bimanual actions to avoid dangerous robot behaviors with improved success rate. Specifically, we design diverse cost functions for safety constraints in different dual-arm cooperation patterns including avoidance of tearing objects and collision between arms and objects, which optimizes the manipulator trajectories with guided sampling of diffusion denoising process. Moreover, we employ a vision-language model (VLM) to schedule the cost functions by specifying keypoints and pairwise relationship, so that the optimal safety constraint is dynamically generated in the entire bimanual manipulation process. SafeBimanual demonstrates superiority on 8 simulated tasks in RoboTwin with a 13.7% increase in success rate and a 18.8% reduction in unsafe interactions over state-of-the-art diffusion-based methods. Extensive experiments on 4 real-world tasks further verify its practical value by improving the success rate by 32.5%. https://denghaoyuan123.github.io/SafeBimanip/

Keywords: Bimanual Manipulation, Diffusion Policy, Trajectory Optimization

1 Introduction

Bimanual manipulation [1, 2, 3, 4, 5, 6] is a fundamental capability for robots to perform real-world tasks in household service [2], surgical operations [7] and manufacturing [8]. Compared with the unimanual manipulation system, bimanual manipulators are able to handle more complex tasks such as cooking and sewing, where one arm stabilizes the target object and the other arm performs the demanded action. Therefore, learning generalizable and robust bimanual manipulation policy is desired in real-world robot manipulator deployment. However, the extremely high degrees of freedoms (DOFs) in bimanual manipulators poses challenges in policy learning due to the large policy space.

To address this, hierarchical policy learning [9, 10, 11] has been widely studied where the original large space is decomposed with reduced complexity. To further decrease the complexity caused by action discretization, diffusion policy [12, 13, 14, 15] models the action distribution via continuous denoising process for policy generation, which shows excellent scalability to high-dimensional output with the score-based learning. However, spatial and temporal coordination between arms are required in bimanual manipulation tasks, and the arm misalignment usually leads to dangerous

*Corresponding author.

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

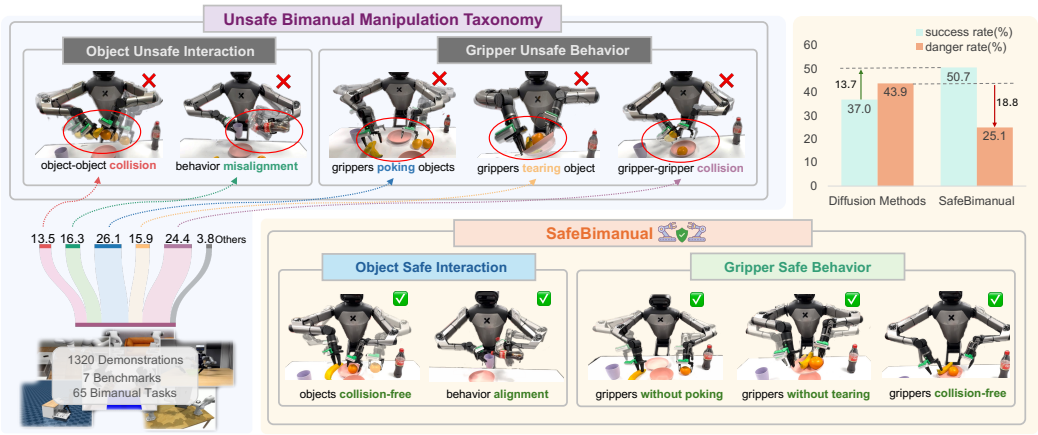

Figure 1: **SafeBimanual** imposes safety constraints into policy optimization to enable physically safe and task-effective bimanual manipulation across diverse scenarios.

behaviors including trajectory intersection of arms and inconsistent end-effector actions on the same objects because existing methods fail to consider diverse physical safety constraints in real-world deployment, which causes severe consequences such as collision and object tearing.

In this paper, we present a SafeBimanual method to impose the safety constraints of the real-world bimanual manipulation on diffusion policy to avoid dangerous robot behaviors. Different from existing diffusion policy which directly samples actions from denoising distribution [12, 13, 14, 15], our method acquires safe trajectory in test time by guided action sampling of pre-trained diffusion policies, which is efficiently adapted to different safety constraints across real-world deployment scenarios. More specifically, we design diverse cost functions for safety constraint in different bimanual cooperation patterns such as gripper movement consistency to avoid object tearing in symmetric cooperation and object collision avoidance in asymmetric cooperation. As illustrated in Figure 1, these cost functions are designed based on the dominant unsafe interaction patterns revealed by an unsafe bimanual manipulation taxonomy, which systematically analyzes 65 bimanual manipulation tasks across 7 benchmarks. The gradient of cost functions is applied to generate the guidance of diffusion denoising process. As the optimal cost functions should be dynamically generated in different stages of the entire bimanual manipulation process, we employ a vision-language model (VLM) to schedule the cost functions, where keypoints and corresponding pairwise relationship in the cooperation patterns are specified. Extensive experiments show that SafeBimanual increases the success rate in the RoboTwin simulator and the real world by 13.7% and 32.5% respectively compared with the state-of-the-art bimanual manipulation methods, as the unsafe interactions are reduced by 18.8% and 30.0% during the arm cooperation. Our contributions can be summarized as follows:

- We propose a test-time trajectory optimization framework for safe bimanual manipulation, which plays as a plug-and-play module for diffusion policy to efficiently adapt to diverse safety constraints across deployment scenarios.
- We present a VLM-based safety constraint scheduler by dynamically considering cooperation patterns, where keypoints and corresponding relationship are specified to generate optimal cost functions.
- We conducted extensive experiments in both simulators and real robots on a wide range of household tasks. The higher success rate and fewer unsafe interactions compared existing methods demonstrate the effectiveness of safety awareness in bimanual manipulation.

2 Related Work

Bimanual Manipulation. Dual-arm systems [16, 17] can complete more complex manipulation tasks than single-arm manipulators [18, 19] because of the higher flexibility and efficiency, which have been widely deployed in household service [20], surgical operations [21] and industrial assem-

bly [22]. Early attempts applied reinforcement learning and imitation learning to generate bimanual manipulation policy with the guidance of designed reward and human expert demonstrations respectively. To deal with the large policy space caused by the high DOFs of dual-arm systems, the policy was learned hierarchically [23] where the bimanual tasks was decomposed into different single-arm manipulation tasks such as stabilizing and acting. However, learning the bimanual manipulation policy in the discrete action space still causes extremely high complexity.

Diffusion Models in Robotics. Inspired by the great success in image generation, diffusion models [12, 14, 15, 24] have aroused extensive interests in policy learning because of the expressive ability of multimodal action distributions and the scalibility to high-dimensional space, which improves the stability, efficiency and success rate of bimanual manipulation policy learning. To leverage richer information such as geometry, temporal relation and robot states in manipulation tasks, compact representations of 3D point cloud [14, 15], videos [25] and proprioception [26] are learned as conditions for diffusion policy generation. Meanwhile, diffusion sampling strategy is improved with higher efficiency to adapt to the limited computational resources in robots, where faster sampling paths [27, 28] or even one-step diffusion [29] are explored for real-time applications. As spatial and temporal coordination between arms is required in bimanual manipulation, existing diffusion policy that ignores the safety constraints usually causes arm misalignment with dangerous behaviors such as collision and object tearing.

Trajectory Optimization. Trajectory optimization plays a pivotal role in robotic manipulation, enabling robots to generate feasible and efficient motions toward task goals [30, 31]. Traditional methods typically impose geometric or contact constraints to ensure collision-free trajectories [32, 33], often rely on hand-crafted formulations for specific tasks and struggle to scale in high-DOF systems like bimanual manipulation. To address this, data-driven methods have emerged that learn task objectives and constraints directly from demonstrations or interaction data [34, 35, 12], improving adaptability in unstructured scenes. Benefiting from the rise of foundation models, recent works such as VoxPoser [36] and ReKeP [37] demonstrate that leveraging VLM to infer affordances and constraints can further enhance generalization and planning efficiency [38, 39, 40]. In this work, SafeBimanual is inspired by these advances and tackles unsafe interactions in bimanual manipulation. We impose safety constraints into the diffusion denoising process via a VLM scheduler to avoid dangerous robot behaviors.

3 Methodology

In this section, we first introduce problem statement (3.1). Then, we formulate imposing safety constraints on trajectories as guided sampling in denoising process of pre-trained diffusion policies (3.2). To specify, we design cost functions based on the dominant unsafe interaction patterns revealed by an unsafe bimanual manipulation taxonomy to guide the denoising process (3.3). We further propose an adaptive scheduler to dynamically select optimal cost functions in different bimnaual task stages based on the specified cooperation patterns (3.4). Our overall pipeline can be seen in Figure 2.

3.1 Problem Statement

Bimanual manipulation involves coordinating two robotic arms to jointly manipulate objects, enabling the execution of complex tasks such as folding, assembly, or collaborative transport. Given the observation $\mathcal{O}_t$ consisting of visual, proprioceptive, and optional instruction inputs at time t, diffusion-based bimanual manipulation policies model the denoising distribution $p(A_t \mid \mathcal{O}_t)$ to interactively predict optimal coordinated action sequences $A_t = (a_{t+1}, \ldots, a_{t+n})$, where each $a_i \in \mathbb{R}^d = (q_i^\ell, q_i^r, g_i^\ell, g_i^r)$ encodes the joint positions (q_i^ℓ, q_i^r) and gripper states (g_i^ℓ, g_i^r) of the two arms. However, directly sampling actions from the denoising distribution without considering physical safety constraints may result in coordination misalignment, inter-arm collisions, or object tearing during bimanual coordination, especially in real-world where precise spatial and temporal synchronization between arms is critical. In light of this, we focus on imposing safety-aware guid-

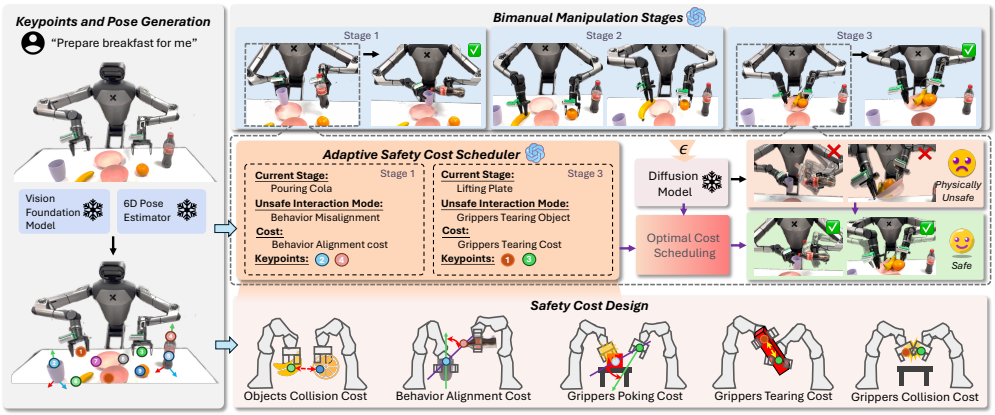

Figure 2: **SafeBimanual**. The framework integrates a Vision-Language Model (VLM)-based Adaptive Safety Cost Scheduler with stage-appropriate safety constraints. These constraints guide the diffusion denoising process to optimize dual-arm trajectories for safe and coordinated manipulation during deployment.

ance during the diffusion denoising process to promote physically feasible bimanual coordination. We consider bimanual manipulation tasks decomposed into K sequential stages $\{s_t\}_{t=1}^{K}$. For each stage s_t, the objective is to infer the potential unsafe interaction pattern and to quantify it using a set of differentiable safety costs $\mathcal{C}(A_t, \mathcal{P}, s_t)$, where $\mathcal{P}$ denotes the relational keypoints between the two arms and the manipulated objects [37], thereby steering the trajectory toward a physically safer version A_t^{safe}.

3.2 Guided Sampling in Denoising Process.

Following the Denoising Diffusion Probabilistic Model (DDPM) framework [41], the policy generates actions by iteratively denoising from Gaussian noise:

$$p\big(A_t^{k-1} \mid A_t^k, O_t\big) = \mathcal{N}\big(A_t^{k-1};\, \mu(A_t^k, O_t, k),\, \Sigma\big)\,, \tag{1}$$

where $\mu(A_t^k, O_t, k)$ is the predicted mean, Σ is determined by the noise schedule, and k indexes the denoising timestep. To impose safety during generation, we extend classifier-guided diffusion to incorporate task-aware constraints through energy-based guidance. Since directly evaluating costs on noisy samples A_t^k is ineffective, we estimate the corresponding clean action chunk $A_{0|k}$ using the noise prediction network ε_θ [42]. We then compute a cost scheduled by the Adaptive Safety Cost Scheduler, denoted as $\mathcal{C}_{\text{sched}}(A_{0|k}, \mathcal{P}, s_t)$, and inject its gradient into the denoising update:

$$A_t^{k-1} = \mu(A_t^k, O_t, k) - \rho_k \nabla_{A_k} \mathcal{C}_{\text{sched}}(A_{0|k}, \mathcal{P}, s_t) + \sigma_k \varepsilon, \quad \varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2}$$

where ρ_k controls the strength of safety guidance and σ_k is the noise schedule. This formulation steers trajectory A_t^{safe} generation toward physically feasible and safe bimanual behaviors, guided by both task semantics and safety priors.

3.3 Safety Cost Formulation

Unsafe Bimanual Manipulation Taxonomy. To systematically characterize unsafe bimanual manipulation, we analysis 1,320 demonstrations from 65 tasks across 7 bimanual manipulation benchmarks [43, 44, 45, 46, 47], focusing exclusively on unsafe modes caused by dual-arm behaviors. As illustrated in Figure 1, our study shows that over 96.2% of such unsafe interactions fall into five representative categories, which we group into two physically grounded classes: 1) *object unsafe interaction*: object-object collision and behavior misalignment; 2) *gripper unsafe behavior*: gripper poking, tearing, and gripper-gripper collision. This taxonomy emerges naturally from the physical characteristics of unsafe interactions: object-level unsafety typically involves misaligned object motion or unintended collisions, while gripper-level unsafety results from improper end-effector behaviors such as poking, tearing, or self-collision. The taxonomy forms a clear basis for defining modular safety constraints.

Safety Cost. Given keypoints $\mathcal{P} = \{k_t^\ell,\, k_t^r\}$ identifying high-risk interaction regions of the left and right arm, we formulate five cost terms based on their spatial relationships, each addressing one of the five dominant unsafe interaction patterns. We express these constraints as differentiable functions of the estimated clean trajectory $A_{0|k} = \big(q^\ell, q^r\big)$, so that gradients can be directly backpropagated into the action space. Following [48, 37], we assume rigidity between the end-effector and the grasped keypoints within an action chunk. The grasped keypoints include both the gripper tip and object-side points, which are computed as functions of the joint state q_t^i at each timestep:

$$k_t^i = \mathcal{F}(q_t^i)\, T_i^{(0)}\, \hat{k}^i, \quad p_i^{\text{tip}} = \mathcal{F}(q_t^i)\, T_i^{\text{tip}}, \quad i \in \{\ell, r\} \tag{3}$$

where $\mathcal{F}(q_t^i)$ denotes the differentiable forward kinematics of arm i, $T_i^{(0)}\, \hat{k}^i$ is the static offset from the object keypoint to the end-effector, computed from the first action of action chunk, and T_i^{tip} is the fixed transform from the gripper tip to the end-effector. Then each safety cost $\mathcal{C}_i(A_{0|k}, \mathcal{P})$ is defined through differentiable metrics quantifying geometric relationships between keypoint:

1) Objects Collision Cost. To mitigate collision risks between manipulated objects, we impose a distance-based safety constraint $\mathcal{C}_1$ over the object keypoints $\mathcal{P}$ with the highest collision likelihood:

$$\mathcal{C}_1(A_{0|k}) = -\left\| k_t^\ell - k_t^r \right\|_2, \tag{4}$$

where k_t^ℓ, k_t^r are the transformed object keypoints of the left and right arms at timestep t, computed in Eq (3). By enforcing this constraint, we maintain a safe distance between the manipulated objects, effectively reducing the risk of unintended collisions or contact.

2) Behavior Alignment Cost. Take the example of a pouring water task where the left arm holds the bottle and the right arm holds the cup. To enforce safe task-level alignment, which needs (1) to enforce axis alignment between the bottle keypoint k_t^ℓ and the cup axis, and (2) to adjust the vertical offset concerning the cup position k_t^r. The cost $\mathcal{C}_2$ can be denoted as:

$$\mathcal{C}_2(A_{0|k}) = \left\| (I - z\, z^\top)\, l_A \right\|^2 \;+\; \lambda\, (z^\top l_A - h_0)^2, \quad l_A = k_t^\ell - k_t^r \tag{5}$$

where $z \in \mathbb{R}^3$ is the unit vector along the cup axis, and h_0 specifies the bottle-to-cup kepoints vertical offset. Eq. (5) generalizes to bimanual tasks requiring spatial behavior alignment, such as insertion and stacking.

3) Grippers Poking Cost. To avoid unintended surface contact such as poking or scratching, we penalize deviations between the gripper's approach direction and the keypoint–tip vector:

$$\mathcal{C}_3(A_{0|k}) = \left\| (I - aa^\top)\, (k_t^i - p_i^{\text{tip}}) \right\|_2, \tag{6}$$

where $a = Approach\big(\mathcal{F}(q_t^i)\big)$ is the gripper's approach axis and p_i^{tip} is the gripper tip position. $\mathcal{C}_3$ penalizes any component of the keypoint–tip vector that deviates from the intended approach axis.

4) Grippers Tearing Cost. To prevent tearing a rigidly grasped object, we penalize deviations from the initial grasp width d_0 between the two gripper tips:

$$\mathcal{C}_4(A_{0|k}) = \big(\|p_\ell^{\text{tip}} - p_r^{\text{tip}}\| - d_0\big)^2, \tag{7}$$

This constraint helps prevent the unintended stretch or shear forces on the manipulated object.

5) Grippers Collision Cost. To avoid end-effector collisions, we penalize proximity between the two gripper tips:

$$\mathcal{C}_5(A_{0|k}) = -\|p_\ell^{\text{tip}} - p_r^{\text{tip}}\|_2, \tag{8}$$

This constraint maintains a minimum clearance between the grippers, reducing the risk of opposing forces between arms that may lead to mechanical deadlock or system failure.

3.4 Adaptive Safety Cost Scheduler

Different stages of a bimanual task impose distinct safety priorities, such as collision avoidance during simultaneous moving of two objects and alignment during pouring. To dynamically schedule

safety constraints, we deploy a vision-language model (GPT-4o) in a structured chain-of-thought (CoT) process [49, 50], executing two sequential stages: **(1) Identify unsafe pattern.** Given the observation $\mathcal{O}_t$, relational keypoints $\mathcal{P}$, and current task stage s_t, the VLM infers the most likely unsafe interaction pattern from a predefined taxonomy. This step enables the system to reason about safety-critical conditions grounded in real-world semantics. **(2) Schedule cost terms with keypoints.** Conditioned on the predicted unsafe pattern, the VLM schedules a subset of relevant safety cost terms and identifies the corresponding relational keypoints. It activates only the necessary costs $\mathcal{C}_i$ via a binary mask and assembles them into $\mathcal{C}_{\text{sched}}(A_{0|k}, \mathcal{P}, s_t)$.

For keypoints proposal, our relational keypoint set $\mathcal{P}$ consists of: (i) semantically meaningful object keypoints in Cartesian space extracted using ReKeP [37], tracked over time with CoTracker3 [51] for temporal consistency; (ii) gripper tip positions $p_\ell^{\text{tip}}, p_r^{\text{tip}}$ derived from forward kinematics; and (iii) object pose centers estimated via Omni6DPose [52]. Detailed are provided in the Appendix D.3.

For efficiency, we adopt a DDIM denoising strategy and apply safety guidance only in the final denoising steps ($k \leq 3$), during which the trajectory is refined using the gradient of the scheduled cost as shown in Eq. 2, progressively steering generation toward trajectories A_t^{safe} that are both task-effective and physically safe.

4 Experimental Results

We evaluate SafeBimanual through extensive experiments in both simulated and real-world settings. Our experiments addresses three key questions: **(1)** How effectively does SafeBimanual optimize diverse bimanual tasks while ensuring safety and task success (Section 4.2)? **(2)** Do the five safety cost terms sufficiently cover unsafe patterns, and can the scheduler adaptively select constraints across task stages (Section 4.3)? **(3)** What practical safety and generalization benefits does SafeBimanual bring in real-world applications (Section 4.4)?

4.1 Experiment Setup

Simulation Benchmark. We evaluate our framework on eight representative bimanual manipulation tasks from the RoboTwin [44], covering diverse dual-arm coordination patterns that span nearly all types of unsafe interactions. See Appendix B.1 for detailed task descriptions.

Baselines. We apply SafeBimanual framework to optimize several representative diffusion-based policies, including 2D Diffusion Policy (DP), which predicts actions from images, and 3D Diffusion Policy (DP3), which uses compact 3D point cloud representations for imitation learning. We also include RDT-1b, a diffusion-based bimanual foundation model, as a strong baseline. All baselines are trained or fine-tuned with 50 expert demonstrations per task. See Appendix D.2 for more details.

Evaluation Matrix. Success Rate (SR) and **Danger Rate (DR)**. **SR** is the percentage of episodes (out of 100) completed successfully without any unsafe behavior. **DR** is the proportion of episodes where at least one unsafe interaction occurs, as defined by the taxonomy in Section 3.3, which includes: 1) Object–Object Collision and 2) Gripper–Gripper Collision: detected via continuous collision monitoring in simulation. 3) Gripper Poking Object: triggered by unintended surface contact during grasping (e.g., poking or scratching). 4) Objects behavior misalignment: detected when the projected distance between two task-relevant keypoints on the plane orthogonal to the alignment axis exceeds the threshold $d_{\text{align}} = 0.03$ m. 5) Gripper tearing object: flagged when the change in gripper tips distance for a rigidly held object exceeds $d_{\text{tear}} = 0.04$ m.

4.2 Simulation Results

Quantitative Experiments. Experiment results are shown in Table 1. (1) SafeBimanual consistently improves the performance of three state-of-the-art diffusion-based methods (DP, DP3, and RDT-1b) across eight simulated tasks in the RoboTwin environment. Specifically, it increases the success rate from 39.6% to 50.7% and reduces the danger rate from 43.9% to 25.0%, demonstrating

Table 1: **Multi-Task Test Results in Simulator.** SR ↑ (Success Rate) and DR ↓ (Danger Rate) are evaluated over 100 episodes. SafeBimanual enables plug-and-play transfer for multiple diffusion-based policies, achieving a 13.8% increase in success rate and a 18.9% reduction in unsafe interactions.

Method	Dual Bottles Pick (Easy)		Dual Bottles Pick (Hard)		Block Handover		Blocks Stack (Easy)		Pour Water	
	SR↑	DR↓	SR↑	DR↓	SR↑	DR↓	SR↑	DR↓	SR↑	DR↓
DP [12]	33%	23%	45%	38%	19%	13%	0%	40%	26%	30%
DP + SafeBimanual	**46%**	**9%**	**60%**	**23%**	**25%**	**4%**	**5%**	**20%**	**40%**	**14%**
DP3 [14]	59%	19%	45%	37%	83%	17%	25%	46%	79%	21%
DP3 + SafeBimanual	**68%**	**10%**	**58%**	**24%**	**90%**	**7%**	**30%**	**28%**	**89%**	**11%**
RDT-1b [53]	56%	40%	39%	48%	94%	2%	14%	33%	41%	55%
RDT-1b + SafeBimanual	**66%**	**30%**	**47%**	**39%**	**96%**	**0%**	**21%**	**21%**	**59%**	**41%**

Method	Pick Apple Handover		Dual Shoes Place		Diverse Bottles Pick		**Average**	
	SR↑	DR↓	SR↑	DR↓	SR↑	DR↓	SR↑	DR↓
DP [12]	31%	63%	5%	88%	7%	70%	20.7%	45.6%
DP + SafeBimanual	**57%**	**23%**	**31%**	**34%**	**24%**	**40%**	**36.0%** (↑ 15.3)	**20.9%** (↓ 24.7)
DP3 [14]	82%	18%	24%	70%	34%	54%	53.9%	35.3%
DP3 + SafeBimanual	**94%**	**3%**	**53%**	**38%**	**56%**	**32%**	**67.3%** (↑ 13.4)	**19.1%** (↓ 16.2)
RDT-1b [53]	33%	57%	7%	90%	7%	81%	36.4%	50.8%
RDT-1b + SafeBimanual	**45%**	**41%**	**26%**	**56%**	**31%**	**54%**	**48.9%** (↑ 12.5)	**35.3%** (↓ 15.5)

the effectiveness of imposing safety constraints into the bimanual manipulation. (2) SafeBimanual generalizes across diverse task types by handling different forms of unsafe interactions. Notably, it shows greater improvements in tasks involving multiple unsafe modes, such as Dual Shoes Place, Dual Bottles Pick (Hard), and Pour Water, where failures often result from unsafe interactions of gripper poking, object collisions, and behavioral misalignment. In contrast, for tasks like Blocks Stack (Easy), where failures are primarily due to semantic mistakes (e.g., incorrect stacking order) rather than physical safety violations, SafeBimanual still reduces the danger rate (from 39.67% to 23%) but has limited impact on task success. This highlights that our method specifically addresses safety-critical failure modes. Overall, SafeBimanual sets a new performance benchmark in bimanual manipulation and demonstrates strong robustness across varying task complexities and unsafe interaction patterns.

Qualitative Analysis. As shown in Figure 3, we visualize representative executions in both real-world and simulated environments. SafeBimanual successfully avoids diverse unsafe interaction patterns including collisions, misalignment, and gripper poking, while maintaining coordinated dual-arm behavior across different tasks.

4.3 Ablations

SafeBimanual integrates two key components: a set of safety cost functions and an adaptive scheduling mechanism for selecting appropriate constraints during execution. We conduct ablations by applying SafeBimanual to optimize diffusion policy [12].

Safety Cost Ablation. We perform leave-one-out ablations over the five safety cost terms and report results in Table 2. Removing any single cost consistently increases the overall Danger Rate, confirming the importance of each constraint. The object–object collision cost has the most significant impact, as such collisions are common across all tasks. In contrast, the gripper tearing cost has a smaller effect, since tearing mainly arises in specific tasks like Block Handover and Pick Apple Handover.

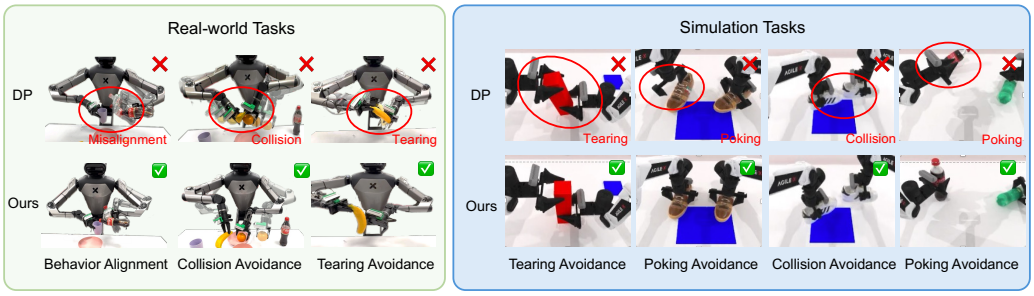

Figure 3: **SafeBimanual** integrates safety constraints to ensure robust and safe bimanual manipulation across diverse tasks.

Cost Scheduling Ablation. We compare three scheduling strategies: (1) a fixed-weight baseline that activates all costs equally, (2) a VLM-based scheduler without chain-of-thought (CoT) reasoning, and (3) our proposed CoT-VLM scheduler that dynamically selects cost terms based on inferred unsafe patterns. As shown in Table 2, fixed-weight scheduling often fails to adapt to task-specific safety needs and may introduce conflicting constraints, which can hinder task execution and reduce success rates. In contrast, our adaptive CoT scheduler consistently improves success rate and reduces Danger Rate by selecting stage-appropriate cost combinations.

Table 2: **Ablation.** Each ablation setting is tested over 20 episodes across 8 tasks.

Method	**Average**	
	SR (%)↑	DR (%)↓
w/o $\mathcal{C}_1$	31.8	24.4
w/o $\mathcal{C}_2$	33.8	23.1
w/o $\mathcal{C}_3$	25.0	37.5
w/o $\mathcal{C}_4$	33.1	21.3
w/o $\mathcal{C}_5$	30.0	24.4
w/o VLM (fixed weights)	19.4	42.5
w/ VLM w/o CoT	25.6	36.9
SafeBimanual	**35.5**	**19.8**

4.4 Real-World Experiments

We evaluate SafeBimanual on the Galaxea-R1 humanoid robot using four real-world bimanual tasks that span all unsafe interaction types in our taxonomy. SafeBimanual is deployed to optimize pre-trained diffusion policies, resulting in higher success rates and fewer unsafe interactions compared to the original policies as shown in Figure 4. To further assess long-horizon performance, we introduce a challenging *Prepare-Breakfast* benchmark that combines all five unsafe interaction modes. Fine-tuning iDP3 [54] with SafeBimanual enhances object-level generalization and reduces compounding errors over extended sequences. Detailed task settings are provided in the Appendix B.2.

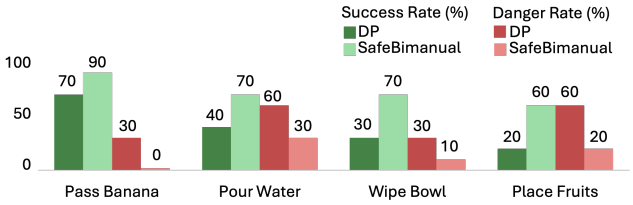

Figure 4: **SafeBimanual** Real-World Results.

5 Conclusion

In this work, we present SafeBimanual, a test-time trajectory optimization framework that enhances the safety of diffusion-based bimanual manipulation policies to enable safe interactions. We design safety constraints to guide the trajectory denoising process and prevent unsafe behaviors. In addition, we leverage a vision-language model (VLM) to dynamically schedule optimal constraint combinations based on keypoints and their relational structures, avoiding unsafe interactions such as collisions, tearing, and misalignment. Our approach generalizes across diverse dual-arm cooperation patterns and achieves consistent performance improvements in both simulated and real-world tasks. These results highlight the practical value of incorporating physical safety reasoning into policy optimization for reliable robotic manipulation in complex environments.

6 Limitations

While SafeBimanual significantly reduces unsafe interaction rates and improves task success across a wide range of scenarios, it does not guarantee complete safety under all conditions. In extreme or highly cluttered environments, residual unsafe behaviors may still occur due to the limitations of policy expressiveness or constraint coverage. Moreover, SafeBimanual is specifically designed to mitigate unsafe physical interactions (e.g., collisions, poking, tearing) and is less effective in addressing semantic task failures, such as incorrect execution order—for instance, placing blocks in the wrong color sequence. Our method also relies on visual keypoints and estimated poses, which may suffer from occlusion or noise. To mitigate this, we employ 2D keypoint tracking with a fixed grasp-offset assumption to improve robustness once an object is held; implementation details are provided in the Figure 6. Additionally, the quality of constraint scheduling depends on the reasoning ability of the vision-language model (VLM). While the CoT-based scheduler shows strong performance overall, it occasionally generates incomplete or suboptimal constraints due to limitations in VLM understanding. We expect this aspect to improve with continued progress in large-scale pre-trained models. Lastly, we observe that in subjective tasks such as object folding or table setting, the VLM may miss fine-grained steps aligned with human preferences, which are inherently ambiguous and task-dependent.

Acknowledgments

This work was supported by the MoE AcRF Tier 1 Seed Grant (RS17/24) and the NTU EEE Ignition Research Grant (024920-00001).

References

[1] M. Grotz, M. Shridhar, Y.-W. Chao, T. Asfour, and D. Fox. Peract2: Benchmarking and learning for robotic bimanual manipulation tasks. In *Conference on Robot Learning (CoRL)*, 2024.

[2] T. Zhang, D. Li, Y. Li, Z. Zeng, L. Zhao, L. Sun, Y. Chen, X. Wei, Y. Zhan, L. Li, et al. Empowering embodied manipulation: A bimanual-mobile robot manipulation dataset for household tasks. *arXiv preprint arXiv:2405.18860*, 2024.

[3] G. Lu, T. Yu, H. Deng, S. S. Chen, Y. Tang, and Z. Wang. Anybimanual: Transferring unimanual policy for general bimanual manipulation. *arXiv preprint arXiv:2412.06779*, 2024.

[4] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[5] F. Krebs and T. Asfour. A bimanual manipulation taxonomy. *IEEE Robotics and Automation Letters (RAL)*, 7(4):11031–11038, 2022.

[6] J. Grannen, Y. Wu, B. Vu, and D. Sadigh. Stabilize to act: Learning to coordinate for bimanual manipulation. In *Conference on Robot Learning (CoRL)*, pages 563–576, 2023.

[7] Z. J. Hu, Z. Wang, Y. Huang, A. Sena, F. Rodriguez y Baena, and E. Burdet. Towards human-robot collaborative surgery: Trajectory and strategy learning in bimanual peg transfer. *IEEE Robotics and Automation Letters (RAL)*, 8(8):4553–4560, 2023. doi:10.1109/LRA.2023.3285478.

[8] J. F. Buhl, R. Grønhøj, J. K. Jørgensen, G. Mateus, D. Pinto, J. K. Sørensen, S. Bøgh, and D. Chrysostomou. A dual-arm collaborative robot system for the smart factories of the future. *Procedia manufacturing*, 38:333–340, 2019.

[9] A. Gupta, V. Kumar, C. Lynch, S. Levine, and K. Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*, 2019.

[10] X. B. Peng, Y. Guo, L. Halper, S. Levine, and S. Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions On Graphics (TOG)*, 41(4):1–17, 2022.

[11] E. Triantafyllidis, F. Acero, Z. Liu, and Z. Li. Hybrid hierarchical learning for solving complex sequential tasks using the robotic manipulation network roman. *Nature Machine Intelligence*, 5(9):991–1005, 2023.

[12] 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*, page 02783649241273668, 2023.

[13] S. Yan, Z. Zhang, M. Han, Z. Wang, Q. Xie, Z. Li, Z. Li, H. Liu, X. Wang, and S.-C. Zhu. M 2 diffuser: Diffusion-based trajectory optimization for mobile manipulation in 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[14] 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.

[15] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024.

[16] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic. Dual arm manipulation—a survey. *Robotics and Autonomous systems*, 60(10):1340–1353, 2012.

[17] M. Abbas, J. Narayan, and S. K. Dwivedy. A systematic review on cooperative dual-arm manipulators: modeling, planning, control, and vision strategies. *International Journal of Intelligent Robotics and Applications*, 7(4):683–707, 2023.

[18] A. Lobbezoo, Y. Qian, and H.-J. Kwon. Reinforcement learning for pick and place operations in robotics: A survey. *Robotics*, 10(3):105, 2021.

[19] A. Iriondo, E. Lazkano, L. Susperregi, J. Urain, A. Fernandez, and J. Molina. Pick and place operations in logistics using a mobile manipulator controlled with deep reinforcement learning. *Applied Sciences*, 9(2):348, 2019.

[20] T. Zhang, D. Li, Y. Li, Z. Zeng, L. Zhao, L. Sun, Y. Chen, X. Wei, Y. Zhan, L. Li, et al. Empowering embodied manipulation: A bimanual-mobile robot manipulation dataset for household tasks. *arXiv preprint arXiv:2405.18860*, 2024.

[21] Q. Wu, M. Li, X. Qi, Y. Hu, B. Li, and J. Zhang. Coordinated control of a dual-arm robot for surgical instrument sorting tasks. *Robotics and Autonomous Systems*, 112:1–12, 2019.

[22] R. Li and H. Qiao. A survey of methods and strategies for high-precision robotic grasping and assembly tasks—some new trends. *IEEE/ASME Transactions on Mechatronics*, 24(6): 2718–2732, 2019.

[23] F. Xie, A. Chowdhury, M. De Paolis Kaluza, L. Zhao, L. Wong, and R. Yu. Deep imitation learning for bimanual robotic manipulation. *Advances in neural information processing systems*, 33:2327–2337, 2020.

[24] Z. Xian and N. Gkanatsios. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *Conference on Robot Learning/Proceedings of Machine Learning Research*. Proceedings of Machine Learning Research, 2023.

[25] Z. Xing, Q. Feng, H. Chen, Q. Dai, H. Hu, H. Xu, Z. Wu, and Y.-G. Jiang. A survey on video diffusion models. *ACM Computing Surveys*, 57(2):1–42, 2024.

[26] Y. Wu, Z. Chen, F. Wu, L. Chen, L. Zhang, Z. Bing, A. Swikir, S. Haddadin, and A. Knoll. Tacdiffusion: Force-domain diffusion policy for precise tactile manipulation. *arXiv preprint arXiv:2409.11047*, 2024.

[27] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

[28] G. Lu, Z. Gao, T. Chen, W. Dai, Z. Wang, W. Ding, and Y. Tang. Manicm: Real-time 3d diffusion policy via consistency model for robotic manipulation. *arXiv preprint arXiv:2406.01586*, 2024.

[29] K. Frans, D. Hafner, S. Levine, and P. Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.

[30] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. Stomp: Stochastic trajectory optimization for motion planning. In *2011 IEEE international conference on robotics and automation*, pages 4569–4574. IEEE, 2011.

[31] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel. Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research*, 33(9):1251–1270, 2014.

[32] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa. Chomp: Covariant hamiltonian optimization for motion planning. *The International journal of robotics research*, 32(9-10):1164–1193, 2013.

[33] Z. Manchester and S. Kuindersma. Variational contact-implicit trajectory optimization. In *Robotics Research: The 18th International Symposium ISRR*, pages 985–1000. Springer, 2019.

[34] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13438–13444. IEEE, 2021.

[35] J. Urain, N. Funk, J. Peters, and G. Chalvatzaki. Se (3)-diffusionfields: Learning smooth cost functions for joint grasp and motion optimization through diffusion. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5923–5930. IEEE, 2023.

[36] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[37] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024.

[38] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR, 2022.

[39] D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, W. Huang, Y. Chebotar, P. Sermanet, D. Duckworth, S. Levine, V. Vanhoucke, K. Hausman, M. Toussaint, K. Greff, A. Zeng, I. Mordatch, and P. Florence. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023. URL https://doi.org/10.48550/arXiv.2303.03378.

[40] Z. Wu, Y. Zhou, X. Xu, Z. Wang, and H. Yan. Momanipvla: Transferring vision-language-action models for general mobile manipulation. *arXiv preprint arXiv:2503.13446*, 2025.

[41] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[42] J. Yu, Y. Wang, C. Zhao, B. Ghanem, and J. Zhang. Freedom: Training-free energy-guided conditional diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23174–23184, 2023.

[43] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

[44] Y. Mu, T. Chen, S. Peng, Z. Chen, Z. Gao, Y. Zou, L. Lin, Z. Xie, and P. Luo. Robotwin: Dual-arm robot benchmark with generative digital twins (early version). *arXiv preprint arXiv:2409.02920*, 2024.

[45] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning (CoRL)*, 2021.

[46] N. Chernyadev, N. Backshall, X. Ma, Y. Lu, Y. Seo, and S. James. Bigym: A demo-driven mobile bi-manual manipulation benchmark. *arXiv preprint arXiv:2407.07788*, 2024.

[47] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, 2023. doi:10.1109/LRA.2023.3270034.

[48] L. Manuelli, W. Gao, P. Florence, and R. Tedrake. kpam: Keypoint affordances for category-level robotic manipulation. In *The International Symposium of Robotics Research*, pages 132–157. Springer, 2019.

[49] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[50] R. Zhang, B. Zhang, Y. Li, H. Zhang, Z. Sun, Z. Gan, Y. Yang, R. Pang, and Y. Yang. Improve vision language model chain-of-thought reasoning. *arXiv preprint arXiv:2410.16198*, 2024.

[51] N. Karaev, I. Makarov, J. Wang, N. Neverova, A. Vedaldi, and C. Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In *Proc. arXiv:2410.11831*, 2024.

[52] J. Zhang, W. Huang, B. Peng, M. Wu, F. Hu, Z. Chen, B. Zhao, and H. Dong. Omni6dpose: A benchmark and model for universal 6d object pose estimation and tracking. In *European Conference on Computer Vision*, pages 199–216. Springer, 2024.

[53] 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. *arXiv preprint arXiv:2410.07864*, 2024.

[54] Y. Ze, Z. Chen, W. Wang, T. Chen, X. He, Y. Yuan, X. B. Peng, and J. Wu. Generalizable humanoid manipulation with 3d diffusion policies. *arXiv preprint arXiv:2410.10803*, 2024.

[55] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[56] M. Pan, J. Zhang, T. Wu, Y. Zhao, W. Gao, and H. Dong. Omnimanip: Towards general robotic manipulation via object-centric interaction primitives as spatial constraints. *arXiv preprint arXiv:2501.03841*, 2025.

Appendix

A Video Demo

A short video shows **SafeBimanual** imposing safety constraints to a diffusion-based policy for bimanual manipulation. The clip explains the idea, shows each step of the method, and ends with the long-horizon *Prepare Breakfast* task completed safely. Enjoy!

B Task Description

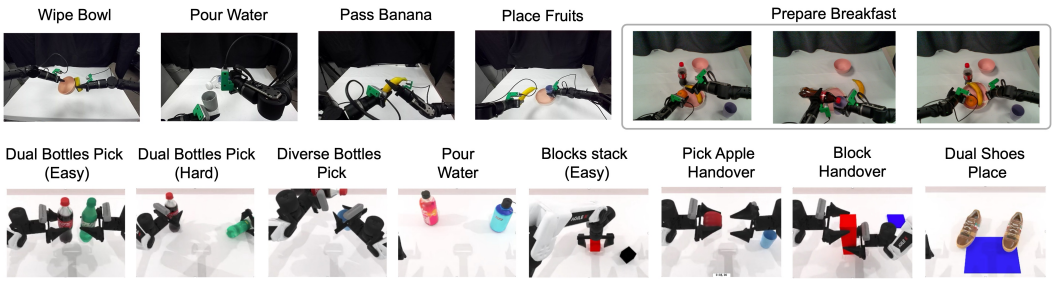

Figure 5: Snapshot of all simulation(second line) and real-world tasks(first line).

B.1 Simulation Tasks

Dual Bottles Pick (Easy): Two upright bottles are randomly placed on the left and right sides of the table. Each arm must grasp its assigned bottle, lift it, and transport both to a shared target area. Success means both bottles end up upright and stably positioned within the target zone without dropping.

Dual Bottles Pick (Hard): Two bottles are randomly oriented (upright, tilted, or lying down) and placed left and right. Each arm must identify, grasp, and lift its bottle despite the varied poses, then place both upright in the target area. Success requires both bottles to be upright and stably located in the goal region.

Block Handover: A long rectangular block lies on the left side of the table. The left arm must pick it up, transfer it to the right arm, and the right arm must place it at a marked goal on the right. Success is achieved when the block is handed over and placed stably at the goal.

Blocks Stack (Easy): One red and one black cube are scattered randomly. The robot must stack the black cube on top of the red cube in the correct order. Success is defined by a stable vertical stack in the specified color sequence.

Pour Water: A filled bottle sits on the left and an empty cup on the right. The left arm must grasp and tilt the bottle over the cup held by the right arm to pour water. Success requires water to transfer into the cup without spilling outside.

Pick Apple Handover: An apple is placed at a random location on the left side of the table. The left arm must grasp the apple, transfer it to the right arm, and fully open its gripper to complete the handover. Success is achieved when the apple is securely held by the right arm and the left gripper is fully open with no residual contact.

Dual Shoes Place: Two shoes from the same pair, each with a distinct design, are placed randomly on the left and right sides of the table. The robot's left and right arms must each grasp their assigned shoe and deposit it into the blue target area, ensuring the toe of each shoe points toward the table's left or right edge.

Diverse Bottles Pick: Two bottles of different shapes and sizes are randomly positioned. Each arm must pick its assigned bottle and deliver both to a common target region. Success means both bottles are placed upright and stably in the goal area.

B.2 Real-World Tasks

Pass Banana: A banana lies on the left side of the workspace. The left arm must grasp the banana, pass it to the right arm, which places it into the other side of table. Success is achieved when the banana is transferred intact and stably positioned within the target region.

Pour Water (Real): A filled bottle and an empty cup are arranged on a table. The robot must coordinate both arms to pour water from the bottle into the cup. Success requires sufficient water transfer with minimal spillage.

Wipe Bowl: A bowl sits on the left and a sponge on the right. The left arm must lift and stabilize the bowl while the right arm wipes its interior with the sponge. Success is defined by a clean bowl and no drops or excessive force.

Place Fruits: A juice cup is positioned to the right of a plate and a banana to the left. The right arm must first grasp and place the cup onto the plate, and then the left arm must grasp and place the banana beside it. Success is achieved when both the cup and banana rest stably in their intended positions on the plate.

Prepare Breakfast: A cola bottle, an empty cup, an orange, and a banana are arranged. In Phase 1 the robot pours water from the bottle into the cup; in Phase 2 it places the orange and banana onto a plate; in Phase 3 it lifts and carries the loaded plate to a target area. Success requires each phase to complete without spillage, misplacement, or dropping.

C Pseudo-code for SafeBimanual

Algorithm Workflow. At runtime (Algorithm 1), SafeBimanual first decomposes the task into discrete stages $\{s_t\}$, each governed by a termination condition. For simple tasks, we pre-select all stage-specific safety constraints and their associated termination checks, automatically switching to the next stage's cost guidance when its condition is met. For long-horizon tasks, we instead re-invoke the VLM at each stage boundary to generate fresh safety primitives—adding delay but enhancing adaptability to dynamic scenes.

Once the active stage s_t is determined, SafeBimanual applies CoT-VLM reasoning to the current observation $\mathcal{O}_t$ and keypoints $\mathcal{P}$ to identify the prevailing unsafe interaction pattern. The Adaptive Safety Cost Scheduler uses this result to produce a binary mask α that activates only the relevant cost terms $\{\mathcal{C}_i\}$. We then sample initial noise A_t^K and run DDIM denoising from $k = K$ down to 1. To balance exploration and safety, we inject cost-gradient guidance only in the final M steps ($k \leq M = 0.3K$); early steps remain pure denoising to preserve trajectory diversity, while later guidance steers the trajectory toward the safe output A_t^{safe} once noise levels are sufficiently low.

D Implementation Details

D.1 Real World Setup

All experiments use the Galaxea R1 humanoid robot, featuring two 7-DoF arms (70 cm reach, 100 N force), a 4-DoF waist, as shown in Figure 6. Scene rgb perception is provided by a ZED head-camera and wrist-mounted Intel D435i camera, supplemented by an Intel L515 for high-fidelity point clouds. The real-world experiments are performed on a single RTX 4080s GPU.

D.2 Training Settings of Baseline Methods

The key training setup for our baseline policies is detailed in Table 3. In simulation experiments, we evaluate 2D Diffusion Policy (DP), 3D Diffusion Policy (DP3), and RDT-1b; for real-world

Algorithm 1 Safety-Guided Sampling Process for Bimanual Manipulation

Require: Pretrained diffusion-based policy ϵ_θ [42], safety costs $\{\mathcal{C}_i\}$, taxonomy $\mathcal{S}$, keypoints $\mathcal{P}$, scheduler VLM [49, 50], total denoising steps K, guided denoising steps M, execution horizon m

$t \leftarrow 1$
while $t \leq T$ **do**
  $s_t \leftarrow$ current task stage
  $\hat{p}_t \leftarrow \arg\max_{p \in \mathcal{S}} \Pr(p \mid \mathcal{O}_t, \mathcal{P}, s_t)$ ▷ CoT-VLM unsafe pattern
  $\alpha \leftarrow$ Scheduler.VLM($\hat{p}_t, \mathcal{P}, s_t$) ▷ Cost-term mask
  sample initial noise $A_t^K \sim \mathcal{N}(0, I)$
  for $k = K$ **down to** 1 **do** ▷ DDIM denoising
    $\mu \leftarrow \mu(A_t^k, \mathcal{O}_t, k)$
    $A_{0|k} \leftarrow$ Estimate clean chunk via ϵ_θ
    if $k \leq M$ **then** ▷ apply safety guidance in final M steps
      $\mathcal{C}_{\text{sched}} \leftarrow \sum_i \alpha_i \mathcal{C}_i(A_{0|k}, \mathcal{P}, s_t)$
      $A_t^{k-1} \leftarrow \mu - \rho_k \nabla_{A_k} \mathcal{C}_{\text{sched}} + \sigma_k z,\ z \sim \mathcal{N}(0, I)$
    else
      $A_t^{k-1} \leftarrow \mu + \sigma_k z,\ z \sim \mathcal{N}(0, I)$
    end if
  end for
  Execute A_t^0 for the next m steps
  $t \leftarrow t + m$
end while

tasks, we use DP and improved 3D Diffusion Policy (iDP3) as baselines. All models are trained or fine-tuned with 50 episodes per task.

Table 3: Hyper-parameter Settings for DP, DP3, RDT-1b, and iDP3 Algorithms.

Parameter	**DP [12]**	**DP3 [14]**	**RDT-1b [53]**	**iDP3 [54]**
horizon	8	8	64	16
n.obs_steps	3	3	2	2
n.action_steps	6	6	30	15
num.inference_steps	100	10	5	10
dataloader.batch_size	128	256	32	64
dataloader.num_workers	0	8	8	8
dataloader.shuffle	True	True	True	True
dataloader.pin_memory	True	True	True	True
dataloader.persistent_workers	False	False	False	False
optimizer.target	AdamW	AdamW	AdamW	AdamW
optimizer.lr	1.0e-4	1.0e-4	1.0e-4	1.0e-4
optimizer.betas	[0.95, 0.999]	[0.95, 0.999]	[0.9, 0.999]	[0.95, 0.999]
training.lr.scheduler	cosine	cosine	constant	cosine
training.lr.warmup_steps	500	500	500	500
training.num_epochs	300	3000	10000	3000
training.gradient.accumulate.every	1	1	1	1
training.use.ema	True	True	True	True
training.gpu	6000 Ada	6000 Ada	A800	6000 Ada

D.3 Implementation Details of Keypoint Proposal and Tracking

Keypoint Proposal. We refer to these keypoints as *safety constraint primitives*. In SafeBimanual, we define safety constraint primitives at both the object and gripper levels to capture all relevant interaction cues (see Figure 6):

- **Object-level keypoint:** Local surface keypoints are extracted from RGB-D images using DINOv2 features, SAM2 segmentation [55], PCA reduction, k-means clustering ($k = 5$), and back-projection into world coordinates (see ReKeP [37] for more details). Candidates

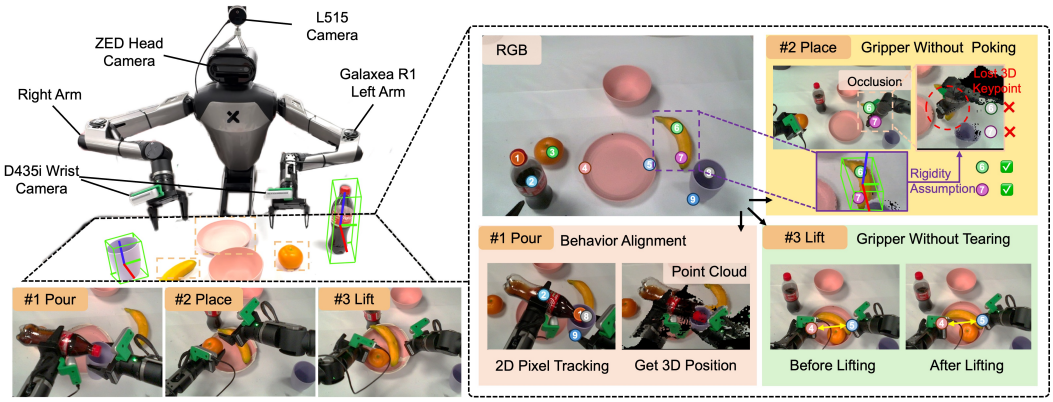

Figure 6: Real-world experimental platforms (Left) and Keypoint Proposal and Tracking (Right) in SafeBimanual.

within 4cm of any existing keypoint are then filtered out to ensure spatial diversity. The object center and primary axis come from the translation and principal axis of the 6D pose estimated by Omni6DPose [52], with axis grounding as in OmniManip [56].

- **Gripper-level keypoint:** Gripper tip positions $p_i^{\text{tip}} = \mathcal{F}(q^i)\, T_i^{\text{tip}}$ are computed via differentiable forward kinematics, where T_i^{tip} is the fixed offset from end-effector to tip. We do not perform surface segmentation or generate additional local keypoints on the gripper itself, which simplifies computation and analysis while still ensuring effective prevention of gripper collisions.

These object-level and gripper-level primitives are used to compute our five differentiable safety costs, enabling modular, interpretable, and dynamically scheduled safety constraints during diffusion-guided trajectory optimization.

Keypoint Tracking Under Occlusion. Although CoTracker3 [51] provides robust 2D pixel-level tracking under heavy occlusions, direct 3D point-cloud measurements can still be lost. To recover occluded 3D keypoints, we assume each local surface keypoint remains fixed in the object's frame. Concretely, let $T_{wo}(t) \in \mathrm{SE}(3)$ be the object's pose in the world frame at time t (from Omni6DPose [52]), and let $p_k^o \in \mathbb{R}^3$ be the keypoint coordinates in that object frame. Then its world coordinate is $k_t = T_{wo}(t)\, p_k^o$. When first observed at t_0, we initialize $p_k^o = T_{wo}(t_0)^{-1} k_{t_0}$. For any later timestep t where the point is missing, we reconstruct it as $\widehat{k}_t = T_{wo}(t)\, p_k^o$. If a noisy observation k_t^{obs} becomes available, we optionally refine via exponential smoothing:

$$p_k^o \leftarrow \beta\, p_k^o + (1-\beta)\, T_{wo}(t)^{-1} k_t^{\text{obs}}, \quad \beta \in [0,1]. \tag{9}$$

This static-offset assumption ensures accurate, temporally consistent 3D keypoint estimates even under severe occlusions.

D.4 CoT-based Vision-Language Model Prompts

After identifying keypoint candidates, we number them $\{0, \ldots, K-1\}$ and overlay them on the key-frame RGB image, which is then input, along with a high-level task instruction, into the Vision-Language Model (VLM). We adopt a code-style prompt template centered around a `stage_n_llm_output` dictionary and a `stage_n_end()` function skeleton. Only minimal examples are given to illustrate stage structure and constraint formats; all remaining fields (e.g., number of stages, keypoints per stage, termination logic) are inferred autonomously by the VLM through physical priors and visual understanding.

In this work, GPT-4o is used as the default VLM. As newer VLMs emerge, they can be directly substituted to enhance performance without modifying downstream code. Hence, our goal is not prompt engineering per task type, but to establish a sustainable, end-to-end pipeline: as long as key names and function interfaces remain consistent, newer VLMs can iteratively generate stage-wise

constraints that conform to the template format and drive robotic execution in increasingly complex tasks. To illustrate the CoT-VLM scheduling process, we present a concrete prompt instance that aligns with our two-stage procedure:

```
## Instructions
## Universal VLM-Prompt Template
# ==============================================================
# Suppose you are helping a bimanual robot to ensure the safety
# during performing manipulation tasks by output a python dictionary
# and stage_end function. The manipulation task is given as an image
# of the environment, overlayed with keypoints marked with their
# indices, along with coordinates of keypoints. Your job should be
# focus on avoiding unsafe behaviors between the objects.

# Inputs: - system prompt(you are safety assistant)
#         - user RGB image(keypoints numbered)
#         - user np.array of keypoint XYZ coordinates

### ---- 1. KEYPOINT ALIAS TABLE -----------------------------
KEYPOINTS = {
    -1 : "left_gripper_tip",
    -2 : "right_gripper_tip",
    -3 : "apple_centre",
    -4 : ...
    # Additional ids are defined by the annotated image
}

# ==============================================================

## GENERAL NOTES
# - Keypoints coordinates input, is an np.array of three-dimensional
#   point coordinates, each row represents the X Y Z coordinates of a
#   point, the first row represents the point 1, the second represents
#   the point 2, and so on.
# - Choose points by rows, which is point 1, 2, 3... And -1 is
#   left_gripper_tip while -2 is right_gripper_tip
# - Decide which arm is nearer if only one should act.
# - Typical guidance types: poking / tear / align / gripper collision /
#   object collision.
# - stage_end is omitted for final stage unless needed.

## STEP-BY-STEP INSTRUCTION FOR GPT4o
# 1. Determine how many stages are involved in the task. Grasping
#    must be an independent stage.
Some examples:
  - "dual_bottle_easy":
    - 2 stages: "grasp bottles", "move and place bottles"
# 2. For **each** stage:
#   a. Choose guidance type & fill llm_output. (poking, objects-collision, grippers-
    collision, tear, align)
#       if gripper is going to grasp object:
#           enable poking_guidance
#       if two objects are both grasped and moving:
#           enable collision_guidance (between objects)
#       if object is not successfully grasped:
#           enable collision_guidance (between grippers)
#       if both grippers are holding the same object:
#           enable tear_guidance
#       if two points on the different objects need to be aligned:
#           enable align_guidance

# llm_output example:
llm_output = {
        "enable_?_guidance":
        {
```

```
                "enable": True/False,
                "enable_left_arm":
                {
                "enable": True/False,
                "point": "?",
                },
                "enable_right_arm":
                {
                "enable": True/False,
                "point": "?",
                }
        },
}

#   b. If the stage needs a finish condition, implement
def stage_N_end(self):
        ...
        return ...
#   **Nmote:**
#   - Consider use bool to determin the start and end of the stage, in
#     case the stage is enable more than once.
#   c. Write one-line comment explaining key-point choices.
# 3. Preserve ordering: Stage 1,2,3...
# 4. Return only code no extra narration.

# ==============================================================

## Example A "apple handover"
# Stages: 1 grasp apple, 2 handover
### stage_1 "grasp apple"
# left arm nearer to apple (choose apple centre)
llm_output = {
    "enable_poking_guidance":{
        "enable": True,
        "enable_left_arm":  {"enable": False, "point": "-3"},
        "enable_right_arm": {"enable": True,  "point": null}
    }
}
def stage_1_end(self):
    return self.is_left_gripper_close() and (self.apple_pose[2] > 0.85)

# both grippers hold apple centre; tips are "-1" (L) & "-2" (R)
llm_output_2T = {
    "enable_tear_guidance":{
        "enable": True,
        "enable_left_arm":  {"enable": True, "point": "-1"},
        "enable_right_arm": {"enable": True, "point": "-2"}
    }
}

# both grippers hold apple centre; tips are "-1" (L) & "-2" (R)
llm_output_2F = {
    "enable_gripper_collision_guidance":{
        "enable": True,
        "enable_left_arm":  {"enable": True, "point": "-1"},
        "enable_right_arm": {"enable": True, "point": "-2"}
    }
}

def stage_tear_guidance_start(self):
    return self.is_left_gripper_close() and self.is_right_gripper_close() and (self.
    apple_pose[2] > 0.85)

def stage_gripper_collision_guidance_start(self):
```

```
    return (self.is_left_gripper_close() and self.left_gripper.pose()[2] > 0.85 and
    self.apple_pose[2] < 0.8)

def get_current_guidance(self, llm_output_2T, llm_output_2F):
    # Decide which guidance dictionary to return
    if self.stage_tear_guidance_start():
        return llm_output_2T
    elif self.stage_gripper_collision_guidance_start():
        return llm_output_2F

## Example B "pour coke into cup"
# Stages: 1 grasp objects, 2 align, 3 pour
### stage_1 "grasp coke and cup"
# left gripper: kp-3 (coke mid), right: kp-5 (cup mid)
llm_output = {
    "enable_poking_guidance":{
        "enable": True,
        "enable_left_arm":  {"enable": True, "point": "3"},
        "enable_right_arm": {"enable": True, "point": "5"}
    }
}
def stage_1_end(self):
    return self.left_gripper_close() and self.right_gripper_close()

### stage_2 "align coke with cup"
# collision points kp-1 (coke neck) & kp-4 (cup rim)
llm_output = {
    "enable_collision_guidance":{
        "enable": True,
        "enable_left_arm":  {"enable": True, "point": "1"},
        "enable_right_arm": {"enable": True, "point": "4"}
    }
}
def stage_2_end(self):
    height = abs(self.object1_pose[2] - self.object2_pose[2])
    return height > 0.10   # coke 10cm above cup

### stage_3 "pour coke"
# align guidance kp-1 & kp-4 (same as above)
llm_output = {
    "enable_align_guidance":{
        "enable": True,
        "enable_left_arm":  {"enable": True, "point": "1"},
        "enable_right_arm": {"enable": True, "point": "4"}
    }
}


# ----- END OF EMPTY TEMPLATE -----
```

As an example, Figure 6 illustrates a sub-task of pouring water. After prompting, the Vision-Language Model (VLM) analyzes a keyframe image with keypoints mask and correctly infers the high-risk scenario of **Objects Behavior Misalignment**, activating the cost term $\mathcal{C}_2$. Based on its interpretation, the VLM identifies and returns two keypoints: one on the top of the bottle and one on the top of the cup. To initiate a safe and effective pouring motion, a vector $\vec{a}$ is first constructed between the top point of the cup and the central axis point of the cup. This vector is designed to have the minimum angle with the world frame's vertical z-axis. Next, a vector $\vec{b}$ is constructed between the top of the cup and the top of the bottle. The robot then executes the pouring motion by aligning $\vec{b}$ with $\vec{a}$, thereby ensuring the bottle is tilted in a direction that minimizes spillage and avoids collision. This behavior alignment cost ensures that dynamic constraints are grounded in visual observations and can be flexibly scheduled based on scene semantics and geometry.

E Additional Results

E.1 Visualization of the Generated Safe Bimanual Trajectories in RoboTwin

In Figure 7, we compare Diffusion Policy outputs against trajectories refined by SafeBimanual across three RoboTwin tasks. **Block Handover.** During the handover stage, SafeBimanual correctly identifies the risk of gripper tearing and activates $\mathcal{C}_4$ to maintain a safe inter-tip distance. As a result, the optimized trajectory avoids the tearing seen in the unmodified Diffusion Policy rollout. **Blocks Stack (Easy).** At the pick stage, SafeBimanual detects potential gripper poking and applies $\mathcal{C}_3$ to enforce directional alignment between the gripper tip and the cube keypoint. This prevents surface damage and enables a stable grasp. **Dual Shoes Place.** While lifting the shoes, SafeBimanual recognizes an imminent object–object collision and selects the shoe-head (left) and shoe-heel (right) keypoint pair to compute $\mathcal{C}_1$. The guided trajectory steers clear of high-risk regions, ensuring both shoes are moved safely to their target areas.

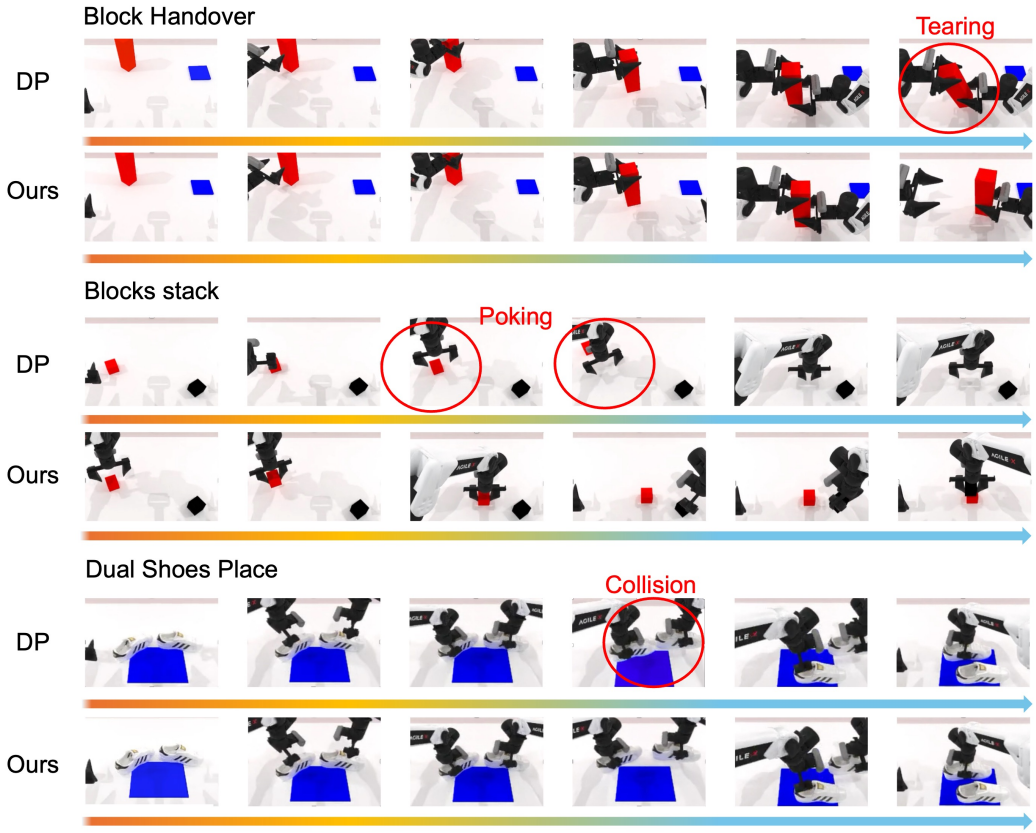

Figure 7: Simulation Tasks Visualization.

E.2 Visualization of the Generated Safe Bimanual Trajectories in Real-world

Figure 8, we compare Diffusion Policy outputs against trajectories refined by SafeBimanual across three real-world tasks: **Pour Water.** During the pouring phase, SafeBimanual infers misalignment and activates $\mathcal{C}_2$, selecting the bottle mouth and cup mouth keypoints. The guided trajectory corrects the tilt to align spout and cup, preventing spillage. **Pass Banana.** In the handover stage, SafeBimanual detects potential tearing and enables $\mathcal{C}_4$, enforcing a fixed inter-tip distance as both grippers hold the banana. This avoids relative motion that would damage the fruit. **Place Fruits.** At the final placement phase, SafeBimanual triggers the objects collision cost $\mathcal{C}_1$, choosing a surface keypoint on the banana and one on the cup. The resulting trajectory maintains safe separation, preventing the banana from knocking over the juice cup.

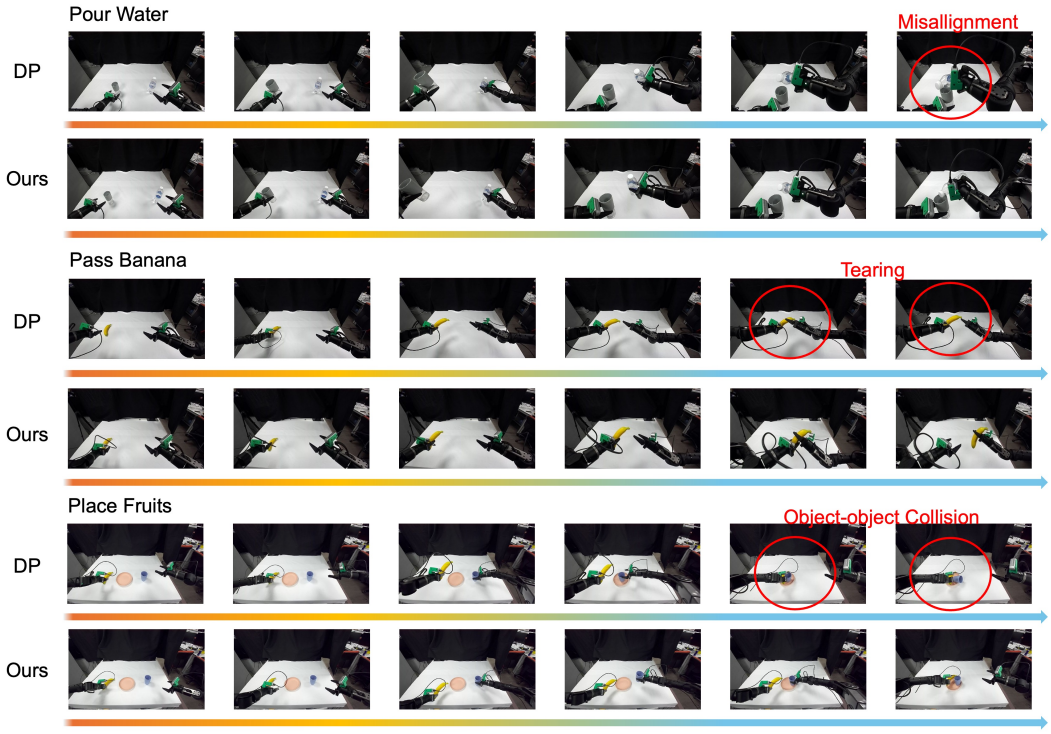

Figure 8: Real-world Tasks Visualization.