CPO-RL: CoT-Policy Co-Optimization for Embodied Reinforcement Learning

Anonymous authors
Paper under double-blind review

Abstract

Reinforcement learning (RL) can adapt Vision-Language-Action (VLA) models through interaction, but reward feedback alone does not explicitly identify the contact regions needed for precise manipulation. We propose CPO-RL, a CoT–Policy Co-Optimization framework that jointly trains a contact-grounded intermediate representation and a continuous control policy. CPO-RL uses dynamic 2D contact masks as supervision for an embodied Chain-of-Thought (CoT) latent. An input-dependent contact token conditions action prediction, while an auxiliary decoder combines this latent with RGB features to predict the contact mask. Contact-SFT first initializes the policy and the contact representation from demonstrations. During online learning, PPO uses all collected trajectories, while valid contact targets derived from successful simulated interactions supervise the auxiliary branch and its shared policy representation. This couples policy optimization with refinement of contact localization without requiring predicted masks or the auxiliary decoder at deployment. On four ManiSkill manipulation tasks, CPO-RL achieves 94.2% average success, improving over the official RLinf baseline by 19.3 percentage points and reducing mean episode length from 68.5 to 52.7 steps. A controlled component ablation shows that contact supervision contributes beyond adding the latent token alone. After sim-real co-training, deployment on a physical Franka robot yields 66.9% average success across original and distractor settings, compared with 29.4% for SFT. These results support contact-grounded representation learning as a practical complement to reward-based VLA post-training.

1 Introduction

Vision-Language-Action (VLA) models combine visual perception, language understanding, and continuous action generation (Intelligence et al., 2025; Team et al., 2025; Kim et al., 2024; Team et al., 2024). Pre-training provides transferable representations (Collaboration et al., 2023; Walke et al., 2023), but downstream manipulation still requires adaptation to task-specific objects and contact dynamics. Reinforcement learning (RL) complements supervised fine-tuning (SFT) by improving the policy through interaction (Li et al., 2025b; Lu et al., 2025; Zhai et al., 2025; Xu et al., 2026; Zhang et al., 2025). A remaining challenge is to make these interactions informative for precise contact control.

Consider peg insertion: the relevant contact region changes from the grasping surface to the insertion opening. A scalar reward evaluates success or shaped progress, but does not directly provide an image-aligned target for learning where interaction should occur. The policy must learn this association through continuous actions. We therefore seek a contact-grounded intermediate representation that complements reward feedback and can be refined from the agent's own successful interactions.

CoT-based reasoning and visual generation motivate jointly optimizing an intermediate representation and the model's output (Guo et al., 2025; Jaech et al., 2024; Jiang et al., 2025). Embodied reasoning methods extend this idea to action generation (Liu et al., 2026; Zawalski et al., 2024). For manipulation, we use dynamic 2D contact masks as a spatial supervision target that changes with the interaction phase. These masks localize task-relevant contact regions and adapt their spatial focus as manipulation proceeds. This leads to our question: **Can a contact-grounded intermediate representation and a VLA policy improve together during online RL?**

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

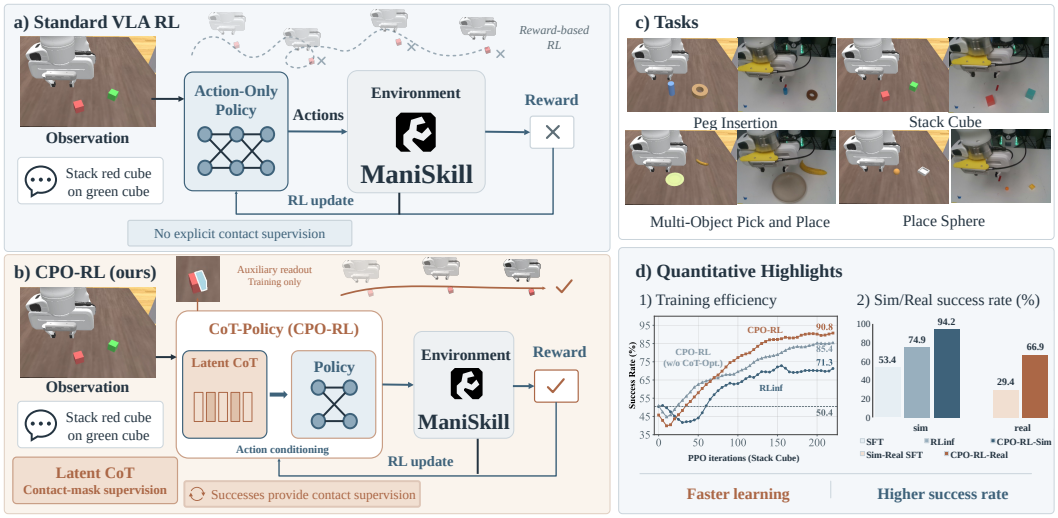


Figure 1: Overview of CPO-RL. Latent CoT conditions action prediction and receives dynamic contact-mask supervision through a training-only auxiliary decoder. Successful simulated rollouts provide contact targets during online co-optimization. Exploration paths (a–b) are schematic.

We propose **CPO-RL**, a CoT–Policy Co-Optimization framework. A learned query becomes an input-dependent contact latent that conditions action prediction. An auxiliary decoder combines this latent with RGB features to predict a contact mask. Embodied CoT here denotes a contact-grounded latent intermediate followed by action realization. The auxiliary mask-prediction loss supervises representation learning; predicted masks are not policy inputs, and the decoder is unnecessary at deployment.

Contact-SFT first initializes the policy and contact representation from demonstrations. Online co-optimization then combines PPO on all trajectories with contact-mask supervision from successful simulated interactions. The contact latent is trained alongside the evolving policy: successful contacts supply spatial targets, while both successful and failed attempts inform policy optimization. The auxiliary objective complements the task reward without discarding unsuccessful experience. We evaluate this design by separating the token architecture, offline contact initialization, and online contact supervision, and by tracking mask localization on held-out interaction frames alongside task success.

On four ManiSkill manipulation tasks (Mu et al., 2021; Tao et al., 2024), CPO-RL achieves 94.2% average success versus 74.9% for RLinf under matched interaction budgets. After sim-real co-training (Shi et al., 2026), a Franka robot achieves 66.9% average success across original and distractor settings versus 29.4% for SFT. In simulation, the largest gain is on Peg Insertion, where success rises from 59.4% to 95.6%. Removing online contact supervision reduces average success from 94.2% to 86.0% under the same initialization and interaction budget. A controlled component ablation distinguishes these gains from the effect of merely adding a token, and held-out mask evaluations track spatial localization during training. Together, these analyses connect task performance with contact representation learning, while the real-robot study tests the complete pipeline in physical scenes. Our contributions are:

- A contact-grounded latent representation that conditions the action policy and is supervised by dynamic 2D contact masks during training.
- An online co-optimization scheme that combines PPO with contact supervision from successful interactions, retaining all trajectories for reinforcement learning.
- Simulation and real-world evaluations, including a controlled component-wise ablation separating the effects of the CoT token, Contact-SFT, and online contact loss.

2 Related Work

2.1 VLA Models and RL Post-Training

VLA models have recently become a promising paradigm for generalist robot control. Representative systems such as RT-1 (Brohan et al., 2022), RT-2 (Zitkovich et al., 2023), Open X-Embodiment/RT-X (Collaboration et al., 2023), OpenVLA (Kim et al., 2024), and the π_0 family (Black et al., 2024; Intelligence et al., 2025) show that large-scale robot data and vision-language pretraining can improve instruction following, semantic generalization, and continuous action generation. However, supervised fine-tuning remains sensitive to demonstration quality and downstream distribution shifts. Recent RL post-training frameworks further improve VLA policies through online interaction and scalable policy optimization, including SimpleVLA (Li et al., 2025b), RLinf (Shi et al., 2026; Zang et al., 2025; Yu et al., 2025), and TwinRL (Xu et al., 2026). These works mainly focus on scaling RL training, improving optimization pipelines, or bridging sim-real deployment. In contrast, CPO-RL studies how to structure the exploration process itself by co-optimizing the policy with an embodied CoT signal.

2.2 Grounded Embodied Reasoning for Manipulation

CoT reasoning has been widely explored as a mechanism to improve the reasoning ability of foundation models (Wei et al., 2022). Recent reasoning models show that reinforcement learning can elicit multi-step reasoning behaviors, while T2I-R1 extends this idea to visual generation by optimizing semantic-level and token-level CoT during RL (Guo et al., 2025; Jiang et al., 2025). Recent embodied reasoning works further introduce CoT-like intermediate representations into robotic policies. ECoT generates grounded reasoning traces before action prediction, while CoT-VLA and CoA-VLA use visual goals or visual-text affordance chains to guide VLA action generation (Zawalski et al., 2024; Li et al., 2025c; Zhao et al., 2025). ThinkAct compresses reinforced multimodal reasoning plans into a visual latent that conditions a downstream action model (Huang et al., 2025a). MolmoAct connects depth-aware perception, spatial trajectory traces, and action prediction (Lee et al., 2025). These methods share the use of intermediate representations for control. CPO-RL focuses on contact-grounded latent conditioning during online policy optimization, using simulator-derived contact targets to co-train that representation with the policy.

Spatial grounding has also been explored as an intermediate structure for connecting vision-language reasoning with manipulation. MOKA and RoboPoint predict language-conditioned keypoint affordances, while RoboGround and Aff-Grasp use grounding or affordance masks to provide dense spatial guidance for robot policies or affordance-oriented grasping (Liu et al., 2024; Yuan et al., 2024; Huang et al., 2025b; Li et al., 2025a). These methods highlight the importance of spatial grounding, but the resulting priors are typically used as fixed perception outputs, visual prompting interfaces, or supervised policy inputs. In contrast, CPO-RL trains a contact-grounded latent with dynamic 2D mask supervision and jointly optimizes this representation with the continuous control policy during online RL.

3 Methodology

3.1 Preliminaries and Problem Formulation

We consider a robotic agent interacting with its physical environment to perform high-precision manipulation tasks guided by natural language instructions. This process is formally modeled as a Markov Decision Process (MDP), defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, respectively. At each time step t, the agent perceives a state $s_t \in \mathcal{S}$, which comprises a visual observation v_t, robot state, and a language instruction l describing the task goal. Based on s_t, the agent generates a continuous action $a_t \in \mathcal{A}$ according to its parameterized policy $\pi_\theta(a_t|s_t)$. The environment then transitions to the next state s_{t+1} following the transition dynamics $\mathcal{P}(s_{t+1}|s_t, a_t)$ and provides a scalar reward $r_t = \mathcal{R}(s_t, a_t)$. The objective of the agent is to find an optimal policy π_θ^* that maximizes the expected discounted return: $J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\gamma^t r_t\right]$, where τ denotes a trajectory and $\gamma \in [0, 1)$ is the discount factor.

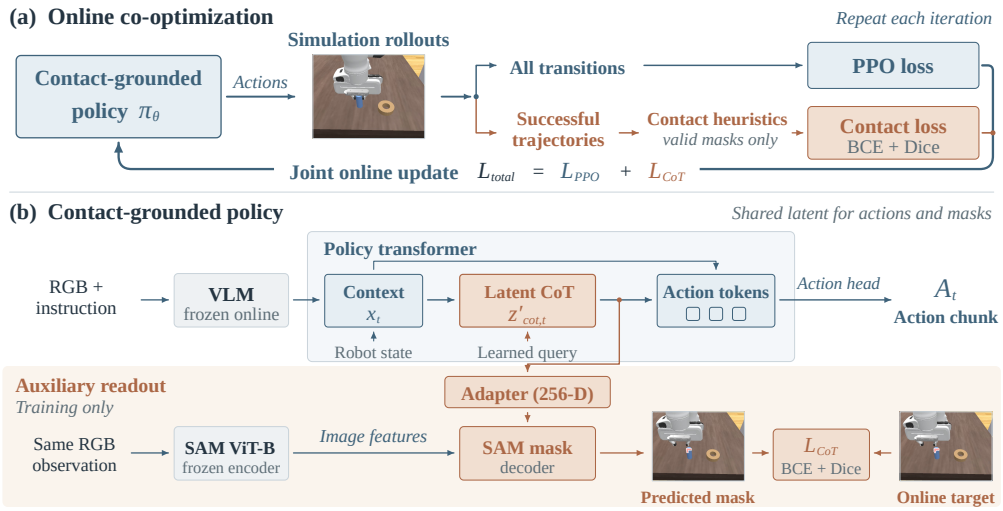


Figure 2: The framework of CPO-RL. A causal attention mask lets action prediction attend to the CoT query hidden state and the original context. This latent also conditions the SAM decoder alongside frozen image features. Successful trajectories provide auxiliary mask supervision (Appendix B.1).

Proximal Policy Optimization (PPO) To fine-tune the pretrained VLA model in downstream environments, we employ Proximal Policy Optimization (PPO) (Schulman et al., 2017) as our base reinforcement learning algorithm. PPO optimizes a clipped surrogate objective to promote stable policy updates:

$$\mathcal{L}_{\text{PPO}}(\theta) = -\mathbb{E}_t \left[\min \left(\rho_t(\theta)\hat{A}_t, \text{clip}(\rho_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right], \tag{1}$$

where $\rho_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio, $\hat{A}_t$ is the estimated advantage at time t, and ϵ is a hyperparameter for clipping.

3.2 Dual-Stream Architecture for Embodied CoT

We represent embodied CoT as a **contact-grounded latent** supervised by dynamic 2D contact masks. These targets identify regions relevant to the current interaction phase. A dedicated policy token carries the latent, while an auxiliary decoder predicts masks for the co-training loss. This **"implicit reasoning, explicit co-training"** architecture preserves efficient action generation; the VLM remains frozen during online RL.

Contact-grounded latent conditioning (the policy stream) Let x_t denote the observation–language context and robot state. A learnable query z_{cot} is contextualized to form an input-dependent contact latent $z'_{\text{cot},t}$, which is appended to the conditioning context before predicting an action chunk A_t:

$$z'_{\text{cot},t} = f_\theta(x_t, z_{\text{cot}}), \qquad A_t \sim \pi_\theta(\cdot \mid x_t, z'_{\text{cot},t}). \tag{2}$$

The query is a learned parameter; its contextualized hidden state forms the input-dependent latent token. A causal attention mask places latent formation before action prediction: action tokens attend to this latent and the original context, while the query cannot attend to subsequent action tokens. Thus, the latent is a *soft control interface*, not an exclusive information bottleneck. We use latent CoT to denote this contact-grounded intermediate followed by action realization. Appendix B.1 details the control and auxiliary pathways.

Contact-mask readout (the decoding stream) A frozen SAM ViT-B image encoder (Kirillov et al., 2023) maps the RGB observation to an image embedding. A trainable adapter maps $z'_{\text{cot},t}$ to a 256-dimensional prompt, and a trainable SAM-based mask decoder combines both inputs to predict M_t. The latent prompt is the sole semantic prompt: no point, box, or mask prompts are supplied. The mask is therefore a joint function of visual evidence and the contact latent, rather than

a reconstruction from the latent alone. The auxiliary mask loss shapes the latent through the adapter and decoder. At inference, the latent remains in the action-conditioning path, while the auxiliary SAM image encoder and mask decoder are unnecessary; predicted masks are never fed back into the policy.

3.3 Online Dual-Stream Co-Optimization

Online co-optimization couples reward-based policy updates with contact-grounded representation learning. PPO updates the action policy from environment feedback, while successful interactions provide targets for the auxiliary contact branch. Because the contact latent also conditions action prediction, the auxiliary objective can shape features used for control. As the policy collects further successful interactions, these rollouts supply additional contact supervision. The two objectives therefore share a representation and a source of training experience, while using different eligibility rules for their supervision.

Concurrent Gradient Updates with Contact Heuristics Online RL freezes the VLM and jointly updates the policy network, learnable query z_{cot}, adapter, and auxiliary mask decoder using PPO feedback ($\mathcal{L}_{\mathrm{PPO}}$) and contact-mask supervision ($\mathcal{L}_{\mathrm{CoT}}$).

We construct contact-heuristic targets from successful simulated trajectories using privileged segmentation and contact information. For grasping and object-object interactions, we intersect relevant 2D masks rendered separately at different time steps within the same episode. Peg Insertion additionally uses dilation to estimate the slot's central opening for the insertion target. The selected target changes with the manipulation phase, from gripper-object contact to the relevant placement or insertion region. These heuristic masks $\hat{M}_t$ supervise the auxiliary contact objective; they are not manually annotated ground truth. All-zero targets are skipped, and a valid mask is reused only within its corresponding task phase (Appendix A.4). Together with PPO gradients from environmental rewards, this dense geometric supervision enables the policy stream and the CoT decoding stream to be optimized jointly. Joint optimization encourages the contact latent $z'_{\mathrm{cot},t}$ to retain information relevant to successful interaction. We evaluate its effect through task success, learning curves, and held-out mask localization.

3.4 Training Pipeline and Overall Objective

Stage 1: Offline Pre-training We initialize online RL by supervised fine-tuning on limited offline demonstrations with a flow-matching action objective and contact-mask supervision (Appendix A.2). During this Contact-SFT stage, the VLM, policy head, learnable CoT token, adapter, and mask decoder are updated, while the SAM image encoder remains frozen.

Stage 2: Online Co-Optimization During the online RL phase, we collect parallel rollout data directly from the environment. For trajectories that successfully complete the task, we construct contact-mask targets $\hat{M}_t$ using the phase-specific heuristics above.

The joint objective combines policy optimization with geometric supervision on eligible samples. To mitigate the pixel imbalance problem inherent in spatial mask prediction, the contact-mask prediction loss is formulated as an unweighted sum of the Binary Cross-Entropy (BCE) loss and the Dice loss (Milletari et al., 2016):

$$\mathcal{L}_{\mathrm{CoT}}(\theta, \phi) = \mathcal{L}_{\mathrm{BCE}}(M_t, \hat{M}_t) + \mathcal{L}_{\mathrm{Dice}}(M_t, \hat{M}_t). \tag{3}$$

The policy-surrogate and contact terms of online co-optimization are:

$$\mathcal{L}_{\mathrm{total}}(\theta, \phi) = \mathcal{L}_{\mathrm{PPO}}(\theta) + \mathcal{L}_{\mathrm{CoT}}(\theta, \phi), \tag{4}$$

Here, θ denotes the trainable policy-stream parameters, including the contact query, and ϕ denotes the adapter and mask-decoder parameters. The contact loss depends on both through the shared latent. Value-function updates and the entropy bonus are omitted from this expression (Appendix A.3). $\mathcal{L}_{\mathrm{PPO}}$ is the negative clipped PPO surrogate loss and uses all collected transitions; the auxiliary loss uses only valid, non-zero masks from successful trajectories. The VLM backbone and SAM image encoder remain frozen during online optimization.

4 EXPERIMENTS

We evaluate CPO-RL in simulation and on a physical robot. Three research questions (RQs) examine task performance, contact-representation refinement, and real-world deployment after sim-real co-training:

- **RQ1 (Performance & Efficiency):** Does CPO-RL improve task success, training and execution efficiency compared with SFT and RL baselines? We systematically investigate this by designing a suite of manipulation tasks ranging from foundational skills to high-precision requirements.
- **RQ2 (CoT-Guided Policy Learning & Progressive CoT Refinement):** Does explicit CoT optimization improve policy learning, and can the growing number of successful trajectories during online RL progressively refine the embodied CoT into more precise and task-aligned contact guidance?
- **RQ3 (Real-World Deployment after Sim-Real Co-Training):** Can CPO-RL achieve reliable real-world manipulation after sim-real RL co-training, and does it outperform the SFT baseline under both original and distractor settings?

4.1 EXPERIMENT SETUP

Environments and Tasks We construct four ManiSkill digital-twin tasks corresponding to the physical Franka setup: Multi-object Pick and Place, Stack Cube, Place Sphere, and Peg Insertion. These tasks span object relocation, stacking, containment, and precise insertion. Each simulation task uses 128 offline demonstrations, followed by online RL under a matched interaction budget. For real-world deployment, sim-real RL co-training additionally uses 50 real-world demonstrations per task as offline samples. These real-world samples do not supply simulator-derived contact targets. We evaluate the resulting policy on all four physical tasks in original and distractor settings; Appendix A.5 gives further details.

Baselines and Ablations We compare CPO-RL against the following baselines and ablation variants to isolate the contribution of our dual-stream co-optimization mechanism:

- **OpenPI (SFT Only):** The pretrained $\pi_{0.5}$ model adapted using only offline supervised fine-tuning. This represents the performance of pure imitation learning without online exploration.
- **RLinf (Official RL Baseline):** The official reinforcement learning post-training pipeline for embodied foundation models. It optimizes the policy with environmental rewards, without the added contact latent or auxiliary contact-mask supervision.
- **CPO-RL (w/o CoT-Opt):** In this setup, we no longer dynamically generate contact-heuristic masks to supervise the decoding stream during the online RL phase. Instead, the learned query and the policy modules producing latent CoT are updated through PPO without the auxiliary contact-mask loss. This variant uses the same Contact-SFT checkpoint as full CPO-RL; only online contact supervision is disabled.
- **CPO-RL (Ours):** Our complete dual-stream co-optimization framework. It extracts successful trajectories on the fly to construct contact-heuristic targets, jointly optimizing the action policy and the contact-grounded latent.

Evaluation Protocol For simulation experiments, each trained policy is evaluated using five evaluation seeds, with 128 episodes per seed and task (640 episodes per task in total). We report the mean success rate across these five evaluation seeds, which vary the evaluation conditions rather than the training initialization. Simulation tables in the main text round SR to integers; numerical comparisons use the values before display rounding. The five evaluation runs use non-overlapping episode-ID ranges starting from 2026. Each environment seed is the GPU rank ID plus the episode ID. For real-world deployment, we conduct 20 trials for each task and each setting on a physical Franka robot, covering both original scenes and distractor scenes with task-irrelevant objects. Success rates are the percentage of trials that satisfy the task-specific success criterion within the evaluation horizon. CoT-Opt denotes Chain-of-Thought optimization. Within each task, RLinf, CPO-RL w/o CoT-Opt, and CPO-RL share the same PPO settings, interaction budget, and stopping point; we report final evaluations rather than independently selecting each method's best checkpoint. Appendix A.3 reports exact budgets, and Appendix C describes the environments.

Table 1: **Main results on robotic manipulation tasks.** We report success rate (SR) and mean episode length (Step), counting failed episodes at the maximum task horizon. SR is rounded to the nearest integer for display. For OpenPI (SFT), "-/H" indicates that the execution-step statistic is not output by the baseline, and H denotes the maximum episode horizon.

Models	Multi-object PnP		Stack Cube		Place Sphere		Peg Insertion	
	SR (%) ↑	Step ↓	SR (%) ↑	Step ↓	SR (%) ↑	Step ↓	SR (%) ↑	Step ↓
OpenPI (SFT)	69	-/80	50	-/100	55	-/100	40	-/100
RLinf (Official)	85	57.4	75	72.1	79	78.9	59	65.7
CPO-RL (w/o CoT-Opt.)	89	54.6	88	65.5	83	67.2	84	57.2
CPO-RL (Ours)	**95**	**50.8**	**93**	**55.6**	**94**	**57.4**	**96**	**46.9**

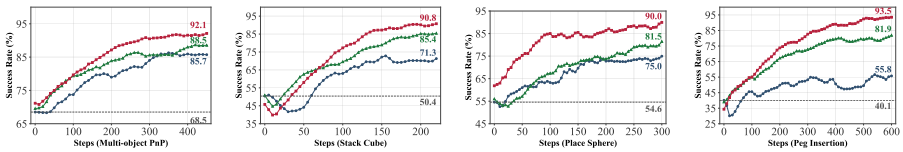


Figure 3: Training rollout success on four tasks, plotted against PPO iterations; final evaluation success is reported separately in Table 1. CPO-RL improves success rates over RLinf and CPO-RL w/o CoT-Opt.; dashed lines indicate the OpenPI (SFT) baseline.

4.2 Performance and Training Efficiency (Answers RQ1)

To answer RQ1, we evaluate CPO-RL on four robotic manipulation tasks with progressively increasing precision requirements. As shown in Table 1, CPO-RL consistently achieves the best success rate across all tasks. Averaged over the four tasks, CPO-RL reaches a success rate of 94.2%, outperforming OpenPI (SFT), RLinf, and CPO-RL w/o CoT-Opt by 40.8, 19.3, and 8.2 percentage points, respectively. The largest gain occurs on the precision-demanding Peg Insertion task. On Multi-object PnP, CPO-RL improves the success rate over RLinf from 85.3% to 94.5%. For the most challenging Peg Insertion task, the improvement is substantially greater, increasing the success rate from 59.4% to 95.6%. This pattern supports the value of contact-grounded supervision for precision-demanding manipulation.

CPO-RL also reduces mean episode length from 68.5 to 52.7 steps across the four tasks, with Peg Insertion decreasing from 65.7 to 46.9 steps. This statistic includes all evaluation episodes, with failures counted at the maximum task horizon. It combines task completion and episode duration, and does not isolate duration among successful trials. The lower episode length, together with the higher success rate, indicates improved overall evaluation efficiency. Figure 3 reports training rollout success over PPO iterations; Table 1 reports separate final evaluations. Curve endpoints therefore need not equal table entries. CPO-RL achieves higher final training success than both RLinf and the variant without online contact loss across all four tasks, and exceeds the dashed OpenPI baselines.

4.3 Policy Learning and CoT Refinement (Answers RQ2)

CoT optimization improves policy learning As shown in Table 1, removing explicit CoT optimization consistently degrades performance across all four manipulation tasks. Adding online contact supervision raises average success from 86.0% to 94.2%, an 8.2 percentage-point gain. The largest improvements occur on Place Sphere and Peg Insertion, at 10.4 and 11.2 points, respectively. Mean episode length also decreases by 9.9 steps on Stack Cube, 9.8 on Place Sphere, and 10.3 on Peg Insertion. Since the two variants share the same Contact-SFT initialization and interaction budget, this comparison supports the contribution of online contact supervision. Together, the success-rate and episode-length improvements demonstrate more effective policy learning with online contact supervision.

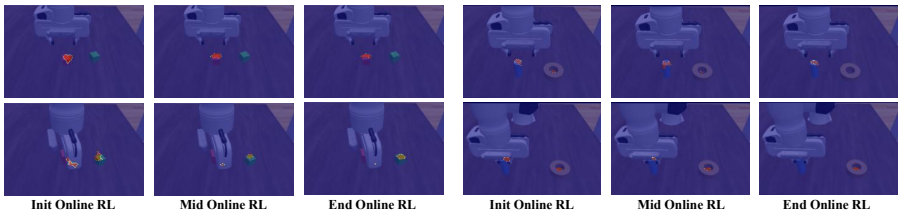


Figure 4: Representative contact-mask predictions during online co-optimization. Different rollout frames illustrate predictions at initialization, intermediate training, and the end of training for Stack Cube and Peg Insertion.

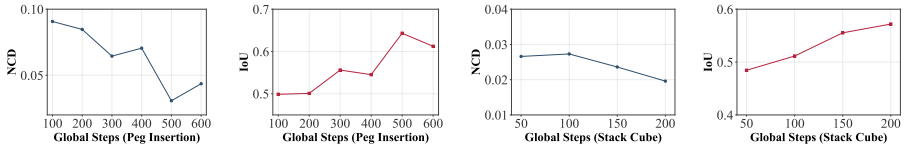


Figure 5: Quantitative evaluation of the auxiliary contact-mask readout during online RL. Mask IoU and NCD on fixed held-out samples for Peg Insertion and Stack Cube at selected PPO iterations (labeled Global Steps).

Component ablation on Peg Insertion. We ablate the CoT token, Contact-SFT, and online contact loss after 600 PPO iterations (7.680M simulator transitions; Table 2). The token alone raises success from 59.4% to 68.6%. Contact-SFT and online contact loss separately reach 84.4% and 91.7%, respectively, and together achieve 95.6%.

Table 2: Component ablation on Peg Insertion.

Configuration	SR (%)
RLinf	59
RLinf + CoT Token	69
RLinf + CoT Token + Contact Loss	92
CPO-RL w/o CoT-Opt.	84
CPO-RL	**96**

In this study, contact supervision improves performance beyond the token architecture, with complementary offline and online contributions. Appendix B reports intermediate training measurements.

Successful trajectories progressively refine embodied CoT Figure 4 provides qualitative evidence of this refinement process, which is continuously shaped by successful interaction experience collected during online RL. Across Stack Cube and Peg Insertion, the predicted masks evolve from relatively coarse or diffuse activations at the beginning of online RL to more localized and stage-aligned contact regions after training. For Stack Cube, the masks focus on gripper-object and object-object contact regions across the pick and stack stages. For Peg Insertion, the masks become increasingly concentrated around the peg-slot interaction area, a crucial factor for high-precision insertion. These examples illustrate more localized mask predictions after online training, complementing the quantitative localization analysis below.

Quantitative localization analysis To further evaluate CoT refinement, we construct a held-out evaluation set from interaction episodes generated with unseen random seeds. We randomly sample 100 frames from this held-out set and use the same samples to evaluate all model checkpoints. For each checkpoint, we run model inference and compare the thresholded predicted contact masks with the corresponding contact-heuristic reference masks using two proxy localization metrics: mask IoU and normalized centroid distance (NCD). The predicted mask is binarized using a threshold of $\tau = 0.5$. NCD is the Euclidean distance between the predicted and reference mask centroids, divided by the image diagonal length. As shown in Figure 5, the predicted contact masks become better aligned with contact-relevant regions as online training proceeds. On Peg Insertion, the normalized centroid distance decreases overall, while the IoU improves from early to later checkpoints, indicating stronger localization around the peg-slot interaction region. On Stack Cube, the centroid distance decreases overall, and IoU increases steadily as training progresses, showing that the auxiliary contact predictions become better aligned with the heuristic targets for grasping and stacking. These metrics quantify agreement with simulator-derived heuristic contact targets on held-out interactions. Task

Table 3: Real-world deployment results after sim-real RL co-training. We report the SR (%) of CPO-RL and the SFT baseline across four manipulation tasks under original and distractor settings.

Model	Multi-object PnP		Stack Cube		Peg Insertion		Place Sphere	
	Original	Distractor	Original	Distractor	Original	Distractor	Original	Distractor
Sim-Real SFT	55	35	35	15	25	10	35	25
CPO-RL (Ours)	**80**	**75**	**75**	**65**	**75**	**60**	**60**	**45**

success, execution length, and the component-wise ablation provide complementary evidence about control performance.

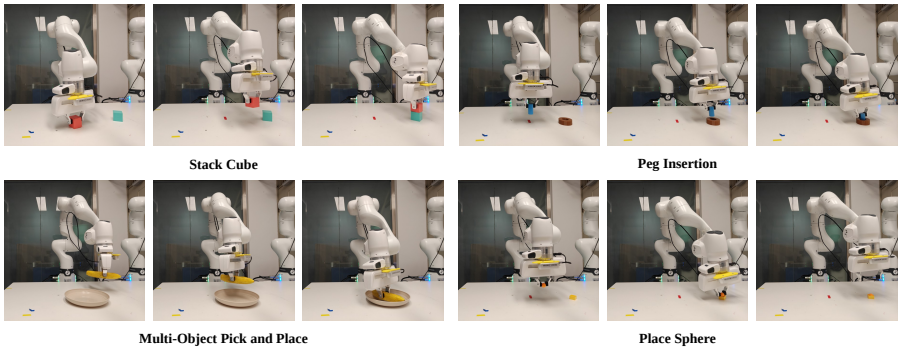


Figure 6: Real-world execution key frames of CPO-RL after sim-real RL co-training. These examples show successful executions on a physical Franka robot across the four manipulation tasks.

4.4 Real-World Robot Deployment (Answers RQ3)

To answer RQ3, we deploy CPO-RL on a physical Franka robot after sim-real RL co-training. We evaluate four manipulation tasks, including Multi-object Pick and Place, Stack Cube, Peg Insertion, and Place Sphere, under both original and distractor settings.

As shown in Table 3, CPO-RL markedly outperforms the SFT baseline across all task-setting pairs, averaging 66.9% success versus 29.4% with a gain of 37.5 percentage points. Under the original setting, average success rises from 37.5% to 72.5%. With distractors, it climbs from 21.3% to 61.3%, showing stronger robustness to irrelevant objects. The gains are particularly large on precision tasks: Peg Insertion improves from 25% to 75% in the original setting and from 10% to 60% with distractors, while Stack Cube improves from 15% to 65% with distractors. These results demonstrate improved real-world task success over SFT across both original and distractor scenes, supporting deployment of the co-trained policy on the physical robot.

Figure 6 provides qualitative evidence of successful real-world execution. Across the four tasks, CPO-RL produces coherent manipulation sequences for the physical Franka robot, including grasping, placing, stacking, inserting, and relocating objects. Together with the quantitative results, these key frames demonstrate that CPO-RL can be effectively deployed in real-world manipulation scenarios.

5 Conclusion

We presented **CPO-RL**, which co-optimizes a contact-grounded latent and an action policy. Contact-SFT initializes the representation; online learning combines PPO on all trajectories with contact-mask supervision from successful simulated interactions. Across four ManiSkill tasks, CPO-RL improves success and mean episode length under matched interaction budgets. Component ablations support the contribution of contact supervision beyond the token architecture. Franka deployment after sim-real co-training shows gains over SFT in original and distractor scenes. The learned latent conditions action prediction at deployment without requiring the auxiliary mask branch.

AI USE STATEMENT

Generative AI tools assisted with manuscript drafting, language polishing, organization of technical descriptions, and code development. For writing, they helped refine wording, improve readability, and clarify the presentation of the method and experiments. For code development, they provided implementation suggestions and debugging assistance. The authors take full responsibility for the final manuscript and reported results.

ETHICS STATEMENT

This work studies robotic manipulation in simulation and on a physical robot. Deployment beyond the evaluated settings requires additional safety validation and appropriate human oversight. Potential safety risks and broader impacts are discussed in Appendix D.

REPRODUCIBILITY STATEMENT

Appendix A documents the model setup, training hyperparameters, contact-mask generation, evaluation protocols, and compute resources. Appendix B clarifies the latent control path and controlled ablations, and Appendix C describes the simulation tasks and rewards.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

OX-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 1(2), 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. ThinkAct: Vision-language-action reasoning via reinforced visual latent planning. *arXiv preprint arXiv:2507.16815*, 2025a.

Haifeng Huang, Xinyi Chen, Yilun Chen, Hao Li, Xiaoshen Han, Zehan Wang, Tai Wang, Jiangmiao Pang, and Zhou Zhao. Roboground: Robotic manipulation with grounded vision-language priors. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 22540–22550, 2025b.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $pi_\{0.5\}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Dongzhi Jiang, Ziyu Guo, Renrui Zhang, Zhuofan Zong, Hao Li, Le Zhuo, Shilin Yan, Pheng-Ann Heng, and Hongsheng Li. T2i-r1: Reinforcing image generation with collaborative semantic-level and token-level cot. *arXiv preprint arXiv:2505.00703*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, Winson Han, Wilbert Pumacay, Angelica Wu, Rose Hendrix, Karen Farley, Eli VanderBilt, Ali Farhadi, Dieter Fox, and Ranjay Krishna. MolmoAct: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

Gen Li, Nikolaos Tsagkas, Jifei Song, Ruaridh Mon-Williams, Sethu Vijayakumar, Kun Shao, and Laura Sevilla-Lara. Learning precise affordances from egocentric videos for robotic manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10581–10591, 2025a.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, et al. Simplevla-rl: Scaling vla training via reinforcement learning. *arXiv preprint arXiv:2509.09674*, 2025b.

Jinming Li, Yichen Zhu, Zhibin Tang, Junjie Wen, Minjie Zhu, Xiaoyu Liu, Chengmeng Li, Ran Cheng, Yaxin Peng, Yan Peng, et al. Coa-vla: Improving vision-language-action models via visual-text chain-of-affordance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9759–9769, 2025c.

Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-world robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

Zhuoyang Liu, Jiaming Liu, Hao Chen, Jiale Yu, Ziyu Guo, Chengkai Hou, Chenyang Gu, Xiangju Mi, Renrui Zhang, Kun Wu, et al. Last _{0}: Latent spatio-temporal chain-of-thought for robotic vision-language-action model. *arXiv preprint arXiv:2601.05248*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. Vla-rl: Towards masterful and general robotic manipulation with scalable reinforcement learning. *arXiv preprint arXiv:2505.18719*, 2025.

Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pp. 565–571. Ieee, 2016.

Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. *arXiv preprint arXiv:2107.14483*, 2021.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-kai Chan, et al. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *arXiv preprint arXiv:2410.00425*, 2024.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pp. 1723–1736. PMLR, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11097–11107, 2020.

Qinwen Xu, Jiaming Liu, Rui Zhou, Shaojun Shi, Nuowei Han, Zhuoyang Liu, Chenyang Gu, Shuo Gu, Yang Yue, Gao Huang, et al. Twinrl-vla: Digital twin-driven reinforcement learning for real-world robotic manipulation. *arXiv preprint arXiv:2602.09023*, 2026.

Chao Yu, Yuanqing Wang, Zhen Guo, Hao Lin, Si Xu, Hongzhi Zang, Quanlu Zhang, Yongji Wu, Chunyang Zhu, Junhao Hu, et al. Rlinf: Flexible and efficient large-scale reinforcement learning via macro-to-micro flow transformation. *arXiv preprint arXiv:2509.15965*, 2025.

Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.

Hongzhi Zang, Mingjie Wei, Si Xu, Yongji Wu, Zhen Guo, Yuanqing Wang, Hao Lin, Liangzhi Shi, Yuqing Xie, Zhexuan Xu, et al. Rlinf-vla: A unified and efficient framework for vla+ rl training. *arXiv preprint arXiv:2510.06710*, 2025.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024.

Shaopeng Zhai, Qi Zhang, Tianyi Zhang, Fuxian Huang, Haoran Zhang, Ming Zhou, Shengzhe Zhang, Litao Liu, Sixu Lin, and Jiangmiao Pang. A vision-language-action-critic model for robotic real-world reinforcement learning. *arXiv preprint arXiv:2509.15937*, 2025.

Jiahui Zhang, Yusen Luo, Abrar Anwar, Sumedh Anand Sontakke, Joseph J Lim, Jesse Thomason, Erdem Biyik, and Jesse Zhang. Rewind: Language-guided rewards teach robot policies without new demonstrations. *arXiv preprint arXiv:2505.10911*, 2025.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 1702–1713, 2025.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A Training and Implementation Details

A.1 Model and observation setup

CPO-RL is built on the released $\pi_{0.5}$ checkpoint. The policy takes RGB observations, robot state, and language instructions as input and predicts action chunks in the end-effector control space. Each action consists of an end-effector delta pose and a binary gripper command in $\{-1, 1\}$. The online RL action chunk size is fixed to 5, and both simulation and real-world execution run at 5 Hz.

For simulation, we use a single RGB camera view with resolution 480×640. For real-world sim-real co-training and deployment, we use two RGB camera views: a front-overhead third-person camera and a wrist-mounted camera, both at 480×640. Depth and segmentation are not used as policy inputs. Simulator segmentation and contact information are only used to construct contact-mask supervision for the embodied CoT.

During Contact-SFT, the VLM backbone, policy modules, CoT token, adapter, and mask decoder are updated. During online RL, the VLM backbone is frozen, while the task-specific policy modules, CoT token, adapter, mask decoder, and RL modules are optimized. The SAM image encoder remains frozen in both stages.

A.2 Offline supervised fine-tuning

We perform task-specific offline supervised fine-tuning before online RL. For each simulation task, we collect 128 demonstrations using rule-based expert scripts. For real-world sim-real co-training, we collect 50 real-world demonstrations per task using a space mouse.

We train two SFT variants. The first is the original $\pi_{0.5}$ SFT baseline trained with the standard flow-matching action loss. The second is a CoT-augmented SFT model (Contact-SFT), which additionally trains the learnable CoT token, adapter, and auxiliary mask decoder with contact-mask supervision. Full CPO-RL and CPO-RL w/o CoT-Opt start from the same Contact-SFT checkpoint. Its objective is:

$$\mathcal{L}_{\mathrm{SFT}} = 10\mathcal{L}_{\mathrm{flow}} + 2\mathcal{L}_{\mathrm{CoT}}, \tag{5}$$

where $\mathcal{L}_{\mathrm{CoT}}$ is the sum of Binary Cross-Entropy (BCE) and Dice losses. Offline contact-mask targets are generated from simulator segmentation and contact information.

All SFT experiments use AdamW(Loshchilov & Hutter, 2017) and a cosine decay learning-rate schedule with 1,000 warm-up steps, 10,000 decay steps, peak learning rate 1×10^{-5}, and final learning rate 1×10^{-6}. We use a per-GPU batch size of 4 and train each task for 5 epochs with bfloat16 precision.

A.3 Online RL co-optimization

Online RL is implemented based on RLinf. We use PPO to optimize the policy and synchronously update the auxiliary contact-mask branch and the shared contact representation using contact-mask supervision from successful trajectories. The VLM backbone remains frozen during online RL.

We use 128 parallel simulation environments. Table 4 reports the task-specific stopping points used for RLinf, CPO-RL w/o CoT-Opt, and full CPO-RL. Each method uses the same number of PPO iterations and training simulator transitions within a task. The iteration limit in the training configuration is 1,000, but the reported runs stop at the fixed task-specific budgets below. No method-specific best-checkpoint selection is used. Evaluation interactions are excluded from these training counts and kept identical across methods. Within a task, PPO iteration k corresponds to $128 \times H \times k$ training transitions, where H is the rollout horizon.

The main online RL hyperparameters are summarized in Table 5. We use $\gamma = 0.99$, GAE parameter $\lambda_{\mathrm{GAE}} = 0.95$, PPO clip range $\epsilon = 0.2$, entropy bonus coefficient 0.005, 5 PPO epochs per update, minibatch size 40, and maximum gradient norm 1.0. Exploration uses a learnable action noise standard deviation with log-variance range $[0.08, 0.16]$.

Table 4: Matched training interaction budgets across RL methods within each task. All runs use 128 parallel environments. Transitions equal environments × rollout horizon × PPO iterations.

Task	Horizon	PPO iterations	Transitions
Multi-object PnP	80	450	4,608,000
Stack Cube	100	220	2,816,000
Place Sphere	100	300	3,840,000
Peg Insertion	100	600	7,680,000

Table 5: Main training hyperparameters for CPO-RL.

Hyperparameter	Value
Base checkpoint	Released $\pi_{0.5}$ checkpoint
Observation resolution	480×640 RGB
Action space	End-effector delta pose + gripper $\{-1, 1\}$
Control frequency	5 Hz
Online action chunk size	5
Offline optimizer	AdamW
Offline LR schedule	Cosine decay
Peak / final learning rate	1×10^{-5} / 1×10^{-6}
Warm-up / decay steps	1,000 / 10,000
Offline batch size	4 per GPU
Offline epochs	5
PPO discount factor γ	0.99
GAE parameter λ_{GAE}	0.95
PPO clip range ϵ	0.2
Entropy bonus coefficient	0.005
PPO epochs per update	5
PPO minibatch size	40
Max gradient norm	1.0
Learnable noise log-variance range	$[0.08, 0.16]$
Parallel simulation environments	128
Auxiliary contact-mask loss	BCE + Dice

A.4 CONTACT-MASK GENERATION

The contact-grounded latent is supervised through an auxiliary dynamic 2D contact-mask prediction at the input image resolution. The auxiliary decoder is a lightweight SAM-based mask decoder. Before the decoder, we use an adapter:

$$\mathrm{Linear}(D, D) \rightarrow \mathrm{ReLU} \rightarrow \mathrm{Linear}(D, 256), \tag{6}$$

where $D = 1024$.

For successful trajectories, contact masks are generated using simulator segmentation and contact information. Failed trajectories are still used for PPO updates but are excluded from the auxiliary contact-mask loss. If the generated mask is all-zero, the corresponding supervision is skipped.

Masks are computed at valid contact steps and reused only within the corresponding task phase. The phase boundary is detected from the gripper open/close state. For Peg Insertion, observations before gripper closure belong to peg grasping, and those after closure belong to post-grasp insertion; a mask is not propagated across this boundary. For grasping and other object-object interactions, we take the bitwise AND of relevant 2D masks rendered separately at different time steps within the same episode. This is a temporal intersection, not an intersection of mutually exclusive instance labels in a single segmentation image. For the insertion phase of Peg Insertion, the slot target additionally requires a computer-vision dilation procedure to extract the central hollow circular region. This dilation is specific to the insertion target; the other contact targets use direct intersection. No smoothing or post-processing beyond these target-construction steps is applied.

Table 6: Task horizons and data sizes used in simulation experiments. Evaluation uses five seeds, with 128 episodes per seed (640 episodes per task).

Task	Horizon	Eval. Episodes / Seed	SFT Demos
Multi-object Pick and Place	80	128	128
Stack Cube	100	128	128
Place Sphere	100	128	128
Peg Insertion	100	128	128

A.5 Simulation and real-world evaluation

We use ManiSkill3 for simulation. Following RLinf-style digital-twin environment construction, we build four manipulation tasks aligned with the physical Franka setup: Multi-object Pick and Place, Stack Cube, Place Sphere, and Peg Insertion. The task horizons and data sizes are summarized in Table 6. Reward functions and task-specific success criteria are provided in Appendix C.

For real-world deployment, we use a Franka Research 3 robot with a Franka Hand gripper. We collect 50 real-world demonstration trajectories per task and use them as offline samples during sim-real RL co-training. Real-world samples are not used to generate simulator-based contact-mask supervision. The final real-world policy is obtained after sim-real RL co-training.

For each real-world task and setting, we conduct 20 trials and reset object poses before each trial. In the distractor setting, 2–3 task-irrelevant objects are placed in the workspace, sampled from a pool of approximately 5–10 distractors, including fruits, kitchenware, and food-related objects. During deployment, we use an emergency stop mechanism and execute the policy at 5 Hz.

A.6 Compute resources

Offline SFT for each task is trained on 2 NVIDIA A800 GPUs and takes approximately 4–6 hours. Online RL co-optimization for each task is trained on 8 NVIDIA A800 GPUs and takes approximately 2 days. This corresponds to approximately 8–12 GPU-hours per task for offline SFT and approximately 384 GPU-hours per task for online RL. Across the four reported tasks, the main experiments require approximately 32–48 GPU-hours for offline SFT and approximately 1,536 GPU-hours for online RL. These estimates do not include preliminary experiments or failed exploratory runs.

B Control Path and Component-wise Analysis

B.1 Latent control path and auxiliary decoder

The learnable query z_{cot} is shared across inputs, whereas $z'_{\text{cot},t}$ is generated from the current context and appended before action prediction. Equation 2 describes context $\rightarrow$ contact-grounded latent $\rightarrow$ action chunk, retaining a direct context-to-action connection. A causal attention mask prevents the query from attending to subsequent action tokens, while action prediction attends to its hidden state and the original context. Thus, the latent is an intermediate control interface, not the sole route for visual information.

The auxiliary branch processes the RGB image through a frozen pretrained SAM ViT-B image encoder. In parallel, the trainable adapter $\mathrm{Linear}(1024, 1024) \rightarrow \mathrm{ReLU} \rightarrow \mathrm{Linear}(1024, 256)$ maps the contextualized latent to a prompt embedding. The SAM-based decoder receives the image embedding and this learned prompt, with no point, box, or mask prompts. It predicts the contact mask used by the BCE and Dice objectives. Consequently, image features supply spatial evidence while the latent specifies task-relevant contact information. The mask loss updates the adapter, decoder, and the trainable modules producing the latent, but not the frozen SAM image encoder. At deployment, neither privileged simulator information nor the auxiliary mask branch is required; robot observations, state, and language condition the policy through the retained latent pathway.

Table 7: Training rollout success (%) at PPO iterations 0, 200, 400, and 600, and final evaluation success (%) for the controlled Peg Insertion study.

Configuration	Train@0	@200	@400	@600	Eval@600
RLinf	40.1	49.8	50.6	55.8	59.4
Token only	31.4	55.4	63.9	65.2	68.6
Token + online loss	27.8	64.3	84.8	90.0	91.7
CPO-RL w/o CoT-Opt.	39.0	67.2	79.6	81.9	84.4
CPO-RL	34.3	73.8	89.0	93.5	95.6

B.2 Controlled Peg Insertion study

Each configuration uses one training seed, shared across configurations. All five configurations in Table 2 use the same base initialization, PPO hyperparameters, simulator-interaction budget, and evaluation protocol, apart from the explicitly varied Contact-SFT initialization. Each run uses 128 environments, a rollout horizon of 100, and 600 PPO iterations (7.680M training transitions). Full CPO-RL and CPO-RL w/o CoT-Opt share the same Contact-SFT checkpoint. The token-only and token-plus-online-loss variants omit Contact-SFT. Table 7 separates training rollout success from the final evaluation success; these are distinct measurements and need not match.

The token-only configuration improves final evaluation success by 9.2 percentage points over RLinf. With the token architecture fixed, Contact-SFT without online loss adds 15.8 points, while online loss without Contact-SFT adds 23.1 points. Adding online loss to the Contact-SFT initialization yields a further 11.2 points. Contact-SFT also improves intermediate training success when online loss is enabled: 73.8% versus 64.3% at iteration 200, and 89.0% versus 84.8% at iteration 400. These results support complementary effects of initialization and online supervision in this setting.

The auxiliary decoder and its contact-supervision objective are evaluated together as a training component. Both successful and unsuccessful trajectories remain in PPO policy and value updates; the success filter applies only to the auxiliary contact loss.

C Simulation Environments

This section summarizes the simulation environments used in our benchmark, focusing on the physical setup, success predicates, and reward design for each task. All tasks are implemented in ManiSkill3 with SAPIEN/PhysX dynamics(Xiang et al., 2020) and follow a tabletop manipulation setting. The simulator runs at 500 Hz and the policy is executed at 5 Hz. The benchmark contains four tasks with increasing spatial precision requirements: Multi-object Pick and Place(Following (Zang et al., 2025) settings, we modified the camera and the scene), Stack Cube, Place Sphere, and Peg Insertion. The first task evaluates robust object relocation under object and background variation. Stack Cube and Place Sphere require accurate object-object or object-container alignment. Peg Insertion requires the finest geometric precision because the clearance between the peg and the hole is only a few millimeters.

C.1 Multi-object Pick and Place

Environment setup The scene contains a Franka robot, a tabletop arena, one source object, and one target plate. The source object is sampled from a set of object meshes, while the target is a tray-like plate. At reset, object and plate poses are selected from a predefined grid of feasible tabletop configurations. The object and plate are placed at distinct grid locations with sufficient separation, and their in-plane orientations are sampled from a discrete set of yaw rotations. After object placement, the simulator is briefly settled before resetting the robot to its initial configuration. The language instruction asks the robot to pick up the object on the table and place it into the white tray.

Success criterion Success requires the source object to be placed on the target plate. The predicate first checks whether the horizontal distance between the source object and the target plate lies within the target object's bounding-box tolerance. It then checks whether the source object is vertically above the target plate with a valid height offset. When full placement verification is enabled, the

predicate additionally requires a non-trivial contact force between the source object and the target plate:

$$\|\Delta_{xy}\|_2 \leq r_{\text{target}} + 0.01, \quad \Delta_z > 0, \quad \Delta_z - h_{\text{target}} - h_{\text{source}} \leq 0.05, \tag{7}$$

together with a source-target contact force larger than 0.03 N.

Reward design The reward follows a shared pick-and-place shaping design. It first encourages the end-effector to approach the source object. When the source object is currently grasped, the reward adds a grasp bonus and activates a placement term that encourages the source object to move toward the target region. If the success predicate is satisfied, the dense reward is overwritten by a terminal success bonus of 5.0. The reward also penalizes movement of the target plate to discourage solving the task by disturbing the receptacle. The normalized reward is obtained by dividing the dense reward by 5.0.

C.2 Stack Cube

Environment setup The scene contains two dynamic cubes with half-side length 1.8 cm. One cube is the movable object and the other is the stacking target. At reset, the target cube is placed near a randomized tabletop center, while the movable cube is initialized on one side of it. The tabletop center is randomly perturbed, each cube receives additional XY jitter, and the relative lateral distance is constrained so that the objects remain reachable and visible from the camera. Both cubes receive independent random yaw rotations sampled from $[-\pi/2, \pi/2]$. The scene is briefly settled before the robot is reset. The language instruction asks the robot to stack the movable cube onto the target cube.

Success criterion Success requires the movable cube to be stacked on top of the target cube. Let $\Delta = p_{\text{mov}} - p_{\text{tar}}$ be the relative displacement between the two cube centers. The active success predicate is

$$\|\Delta_{xy}\|_2 \leq 0.018, \quad |\Delta_z - 0.036| \leq 0.005. \tag{8}$$

Thus, the movable cube must be horizontally aligned within one cube half-side and vertically displaced by one cube height, with a 5 mm tolerance.

Reward design This task reuses the shared pick-and-place reward.

C.3 Place Sphere

Environment setup The Place Sphere task requires the robot to pick up a small sphere and place it into a square bin. The scene contains a dynamic sphere with radius $r_{\text{sphere}} = 2$ cm and a kinematic bin. The bin is composed of a bottom plate and four side walls, with inner half-side length $L_{\text{bin}} = 2$ cm, resulting in a 4 cm by 4 cm opening. At reset, the sphere and bin are initialized around a randomized tabletop center and separated primarily along the tabletop y axis. The sphere is placed on one side of the workspace and the bin on the other, with their separation sampled from $[0.15, 0.30]$ m. Both objects receive independent XY jitter of up to 1.5 cm and independent yaw rotations sampled uniformly from $[-\pi/2, \pi/2]$. The language instruction is "Place the sphere on the bin."

Success criterion The success predicate checks the sphere center relative to the bin, its vertical placement height, and physical contact with the bin. Let $\Delta = p_{\text{sphere}} - p_{\text{bin}}$ denote the relative position between the sphere and the bin. The pose condition is

$$\|\Delta_{xy}\|_2 \leq L_{\text{bin}}, \quad |\Delta_z - r_{\text{sphere}} - h_{\text{bin}}| \leq 0.005, \tag{9}$$

where $L_{\text{bin}} = 0.02$ m, $r_{\text{sphere}} = 0.02$ m, and $h_{\text{bin}} = 0.0025$ m. In addition, the sphere-bin contact force must be larger than 0.01 N. These conditions check center alignment, placement height, and contact; the center-distance bound alone does not guarantee full geometric containment.

Reward design This task inherits the shared pick-and-place dense reward.

C.4 PEG INSERTION

Environment setup The scene contains a dynamic cylindrical peg and a kinematic slot with a circular hole. The peg has radius 1.6 cm and height 8 cm. The hole radius is 2 cm and the board thickness is 2.5 cm, giving a single-side clearance of 4 mm. The board is constructed as a ring of 24 thin collision and visual segments around the hole. At reset, the board is placed at a fixed orientation, while the peg is initialized on one side of the board with XY jitter of up to 1.5 cm. The peg-board separation along the tabletop y axis is randomly sampled from $[0.12, 0.22]$ m. Because the peg is rotationally symmetric, no yaw randomization is needed. The scene is settled before the robot is reset. The language instruction asks the robot to pick up the block and insert it into the hole on the board.

Success criterion The success predicate checks horizontal alignment and the peg center height relative to the board. Let $\Delta = p_{\text{peg}} - p_{\text{board}}$. The active success predicate is

$$\|\Delta_{xy}\|_2 \leq 0.9 r_{\text{hole}}, \quad \Delta_z \leq 0. \tag{10}$$

Since $r_{\text{hole}} = 2$ cm, the horizontal tolerance is 1.8 cm. The vertical condition requires the peg center to reach or pass below the board center height. The center-distance tolerance is distinct from the physical clearance; these inequalities alone do not characterize collision-free insertion.

Reward design The reward is designed to enforce a grasp-before-insertion behavior. Before success, the reward is zero when the peg is not grasped. After grasping, the reward includes a fixed grasp bonus and an XY alignment term between the peg and the board center. When the insertion success predicate is satisfied, the reward is set to 10.0:

$$r = \begin{cases} 10, & \text{if the task succeeds,} \\ 0, & \text{otherwise, if the peg is not grasped,} \\ 1 + 4\left(1 - \tanh(5 d^{xy}_{\text{peg,board}})\right), & \text{otherwise.} \end{cases} \tag{11}$$

The normalized reward is

$$r_{\text{norm}} = r / 10.0. \tag{12}$$

D BROADER IMPACTS

CPO-RL aims to improve the sample efficiency and reliability of reinforcement learning for robotic manipulation. By augmenting reward-based policy learning with contact-supervised latent conditioning, the proposed framework may reduce unnecessary environment interactions and make robot learning more efficient in controlled settings. These benefits could support applications such as manufacturing, laboratory automation, logistics, and assistive robotics, where reliable manipulation and reduced training cost are important.

At the same time, more capable robotic manipulation policies may introduce potential safety risks if deployed without sufficient validation. For example, failures in perception, contact estimation, or policy execution could lead to unintended object motion, equipment damage, or unsafe interaction with humans in shared workspaces. Since CPO-RL relies on learned policies for physical control, deployment should be restricted to controlled environments with appropriate workspace constraints, emergency stop mechanisms, collision monitoring, and task-specific safety checks.