ROBOCTR: CROSS-MODAL TRAJECTORY REWARDING FOR WORLD MODEL-BASED ROBOTIC PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Action-conditioned world models require a planning score that evaluates predicted trajectories with respect to the task goal. We propose RoboCTR, a Cross-Modal Trajectory Reward (CTR) interface that scores a latent rollout against the available text, goal-image, and demonstration-video goals. Stage I learns a goal-conditioned per-step critic; Stage II freezes this critic and uses its gradients to shape the V-JEPA2 predictor; receding-horizon MPC then reuses the same critic to rank candidate action sequences. We further introduce RoboTrain, an eight-task simulation collection with keyframe-aligned subtask instructions and positive and negative trajectory segments. On seven RoboTwin2 tasks, RoboCTR achieves 84.3% macro success versus 79.7% for $\pi 0$, a gain of 4.6 percentage points, with the largest gains on Pick Dual Bottles and Place Object in Basket. Additional ablations examine the training stages, reward interface, goal modalities, task decomposition, and guidance weight. We also present qualitative demonstrations on a physical manipulator.

1 INTRODUCTION

World models predict future observations or latent states, but planning also requires task-dependent evaluation of those futures Bardes et al. (2024); Assran et al. (2025); Cen et al. (2025). Endpoint image distance, language similarity, and terminal success provide useful cues; our focus is a shared progress score over an action-conditioned trajectory.

RoboCTR scores each step of an ordered latent rollout against text, goal-image, and demonstration-video inputs. The differentiable critic supplies progress feedback for predictor adaptation and is reused to evaluate candidate rollouts during receding-horizon MPC, following the visual-planning paradigm Finn & Levine (2017); Ebert et al. (2018).

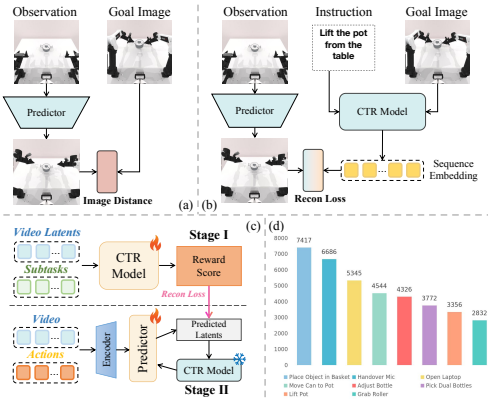


Figure 1: One goal-conditioned trajectory score for predictor shaping and MPC selection.

Figure 2 summarizes this interface. Goals condition the critic, while observations and actions condition the world-model rollout. Stage I learns per-step scores from local progress, instruction contrast, and trend supervision (Figure 3). The frozen critic then guides predictor adaptation and ranks MPC candidates.

We also introduce **RoboTrain**, an eight-task collection of positive and negative trajectory segments with keyframe-aligned subtask instructions and local progress labels. It provides the demonstrations used for critic learning and predictor adaptation.

On the seven reported RoboTwin2 tasks, RoboCTR achieves 84.3% macro success versus 79.7% for $\pi 0$ (+4.6 percentage points), improving on four tasks and trailing on three (Table 1). Additional studies compare training schedules, reward interfaces, goal modalities, task decomposition, and Stage-II guidance weights. The main contributions of our paper are:

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.

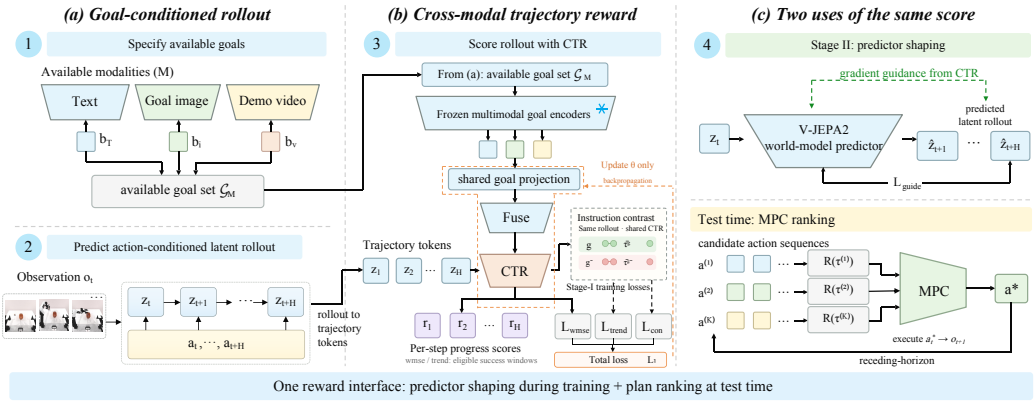


Figure 2: RoboCTR overview. An action-conditioned world model predicts candidate latent trajectories, which a multimodal goal-conditioned CTR evaluates for predictor shaping and MPC selection.

- We formulate trajectory-level cross-modal reward as a shared interface between latent world-model prediction and MPC, scoring action-conditioned future rollouts against an available subset of language, image, and video goals.
- We introduce a two-stage optimization scheme in which Stage I learns the critic, Stage II freezes it while adapting the predictor, and deployment reuses it for MPC selection.
- We provide RoboTrain and corresponding RoboTwin2 experiments that evaluate task-wise performance, component choices, multimodal goals, and predictor-guidance strength, together with qualitative physical-platform demonstrations.

2 Related Work

2.1 World Models

World models learn dynamics for planning or policy learning Ha & Schmidhuber (2018); Hafner et al. (2019; 2020). DreamerV3 scales learning in imagination, while TD-MPC and TD-MPC2 combine latent prediction with value-guided planning Hafner et al. (2025); Hansen et al. (2022; 2024). I-JEPA, V-JEPA, and V-JEPA 2 learn predictive visual representations Assran et al. (2023); Bardes et al. (2024); Assran et al. (2025); DINO-WM predicts pretrained patch features for goal-image planning Zhou et al. (2025). RoboCTR builds on latent prediction but learns a separate goal-conditioned trajectory critic.

2.2 Goal-Conditioned Rewards and Progress

CLIP enables image–text matching Radford et al. (2021), and R3M learns transferable robotic representations from human videos Nair et al. (2023). TCN and temporal cycle-consistency learn temporal correspondences Sermanet et al. (2017); Dwibedi et al. (2019); XIRL uses aligned representations for cross-embodiment visual rewards Zakka et al. (2022). VIP and LIV learn visual and language–image value representations for reward specification Ma et al. (2023b;a). RoboCLIP and VLM-derived rewards use pretrained multimodal models to specify tasks without manual reward design Sontakke et al. (2023); Baumli et al. (2023).

GenRL aligns foundation VLM representations with world-model latents and learns prompted behaviors in imagination Mazzaglia et al. (2024). RoboCTR instead trains a progress critic, freezes it for predictor guidance, and reuses it for MPC; its local time template is not an independently measured completion value. Generalist policies such as RT-2, Octo, and OpenVLA directly learn goal-conditioned action generation Zitkovich et al. (2023); Ghosh et al. (2024); Kim et al. (2025), whereas our contribution concerns scoring predicted trajectories.

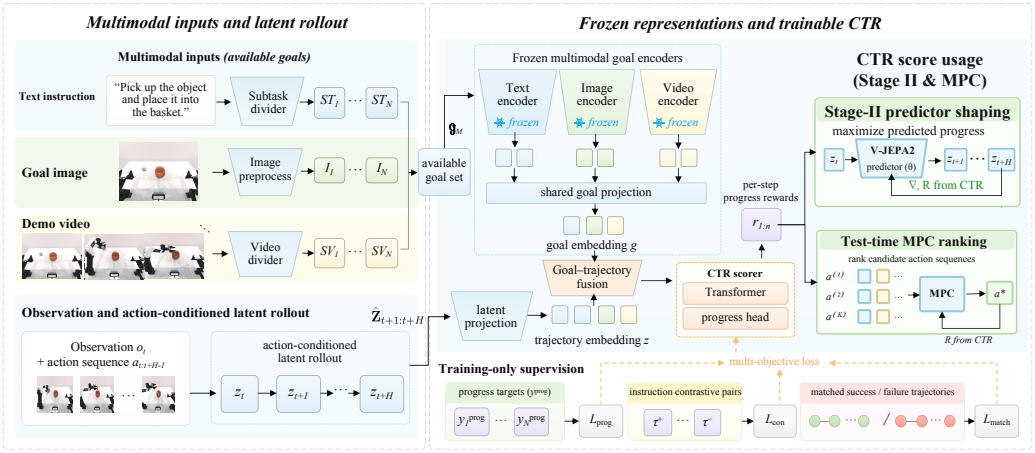


Figure 3: Stage-I CTR training. New goal and latent projections, fusion, temporal modules, and the scoring head are trained; pretrained encoders are frozen. The complete learned critic is frozen for Stage II and MPC.

3 METHOD

RoboCTR is organized around one interface rather than three independent modules. Let $\hat{\mathbf{Z}}_{t+1:t+H}$ denote an action-conditioned latent rollout and let $\mathcal{G}_M = \{g_m : m \in M\}$ denote the subset of available goal modalities. We write the CTR score abstractly as

$$\hat{\mathbf{r}}_{1:H} = r_\phi\Big(\hat{\mathbf{Z}}_{t+1:t+H}, \mathcal{G}_M\Big) = \mathcal{S}_\phi\Big(\hat{\mathbf{Z}}_{t+1:t+H}, \{q_m(g_m)\}_{m\in M}\Big), \tag{1}$$

where $q_m(g_m) = p_m(f_m(g_m))$ separates the frozen pretrained encoder f_m from the newly trained goal projection p_m. The critic also projects latent inputs as $\tilde{z}_k = p_z(z_k)$. Writing ψ_m, ψ_z for these projection parameters, its trainable parameters are

$$\phi = \big\{\{\psi_m\}_{m\in\mathcal{M}}, \psi_z, \phi_{\text{fusion}}, \phi_{\text{temporal}}, \phi_{\text{head}}\big\}. \tag{2}$$

Function notation accommodates biases or nonlinear layers. The new reward-side projections are distinct from pretrained encoders' frozen internal projections. Missing modalities are omitted from $\mathcal{G}_M$; each subset is trained separately rather than masked at test time.

Stage I trains all critic parameters in Eq. 2, with P_θ frozen. Stage II freezes the complete critic and updates only P_θ, including its action/state and input/output projections. Pretrained visual, target, and modality encoders remain frozen throughout; deployment updates no parameters (Figure 2(c)). Appendix B details feature dimensions and normalization.

3.1 ACTION-DRIVEN V-JEPA2 WITH CTR GUIDANCE

Given RGB observations $x_{1:T}$, actions $a_{1:T-1}$, robot states $s_{1:T}$, camera extrinsics $e_{1:T}$, and instruction g, the frozen visual encoder produces $h_t = E(x_t)$. At planning time, an action-conditioned predictor rolls out $H = 4$ future latent states from the current proprioceptive state, the latest calibrated camera extrinsics, and a candidate action sequence:

$$\hat{h}_{t+1:t+H} = \text{Rollout}(P_\theta, h_t, s_t, e_t, a_{t:t+H-1}). \tag{3}$$

The predictor recursively generates the future latent sequence; the evaluated episode's future robot states, camera poses, and observations are never supplied to the planner. Camera extrinsics are treated as fixed over the short planning horizon and refreshed with the next observation before replanning. We average-pool the tokens of each predicted frame into $\hat{Z}_{t+1:t+H}$ and score the segment with the CTR critic:

$$\hat{\mathbf{r}}_{1:H} = r_\phi\Big(\hat{\mathbf{Z}}_{t+1:t+H}, \mathcal{G}_M\Big) \in \mathbb{R}^H. \tag{4}$$

The instruction is encoded by frozen CLIP and projected to the 512-dimensional reward head. Image and video goal features enter through their corresponding projections; missing modalities are omitted

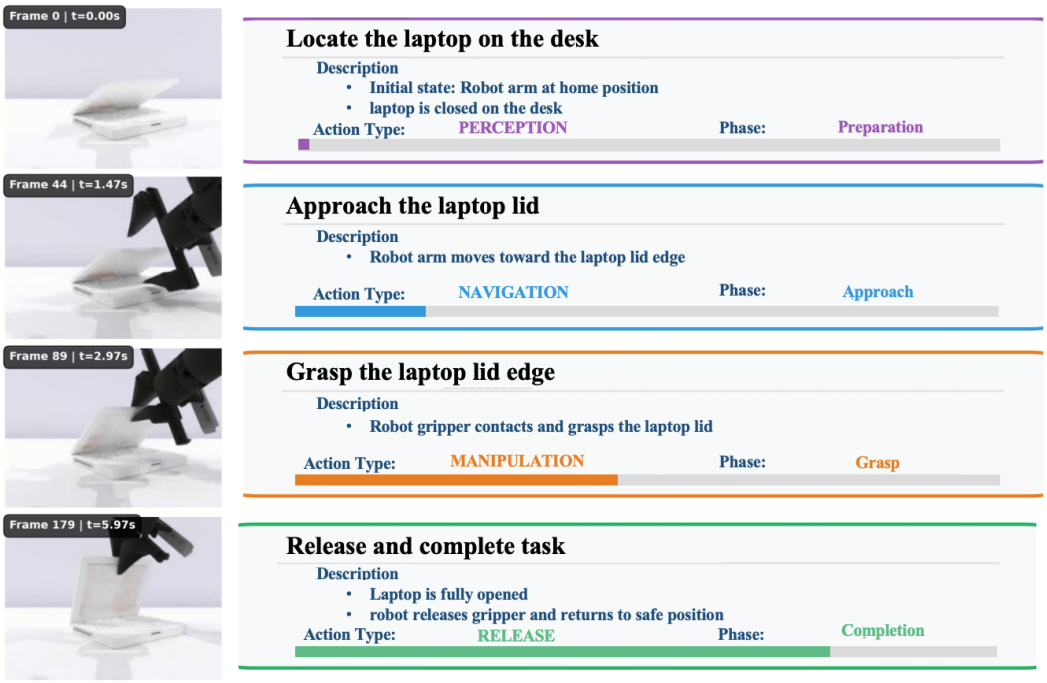


Figure 4: Open Laptop keyframes used for offline subtask annotation.

through the availability set in Eq. 1. Frozen EMA targets, token dimensions, and feature alignment are given in Appendix B.

Pooling compresses spatial tokens within each frame while preserving the ordered H-step sequence. Goal and latent projections align the two input types for the fusion and temporal modules, which produce one score per predicted step. Stage I supervises this sequence, Stage II differentiates through it, and MPC averages its entries to compare candidate trajectories.

Annotated keyframes decompose text instructions and video demonstrations into an offline subgoal sequence (Figures 3 and 4). Frozen encoders supply mean-centered, normalized goal features to the critic. Execution invokes no language model; only observed completion, not a high predicted CTR score, advances the subgoal sequence.

Windows sample the current observation and $H = 4$ future observations at $t, t + \Delta t, \ldots, t + 4\Delta t$, with actions aligned to the four intervening intervals. They do not cross subtask boundaries. All valid recorded transitions support dynamics prediction. Only successful subtask windows with reliable monotonic labels contribute progress regression, trend regularization, and Stage-II guidance. Other successful windows and failed, stalled, or backtracking windows receive neither forced increasing targets nor blanket zero labels.

Without ground-truth environment rewards, we retain the local progress coordinate $u_k = (k - 1)/(H - 1)$ and proxy target

$$y_k = 0.5 + 0.5 \cdot \frac{e^{2u_k} - 1}{e^2 - 1}, \quad k = 1, \ldots, H \tag{5}$$

and combine three terms: time-weighted regression fits the local position template on keyframe-delimited successful subtask segments; an instruction-contrastive margin separates the correct instruction from one mismatched instruction g^- sampled from a different task in the same mini-batch; and a trend regularizer imposes a local monotonic preference:

$$\mathcal{L}_{\text{CTR}} = \mathcal{L}_{\text{wmse}} + 0.2\mathcal{L}_{\text{con}} + 0.1\mathcal{L}_{\text{trend}} \tag{6}$$

Regression fixes the score scale, instruction contrast compares mean scores under the correct and mismatched goals, and the trend term penalizes decreases between consecutive steps. Regression and trend average eligible successful windows; contrastive eligibility follows the sampler. Instructions

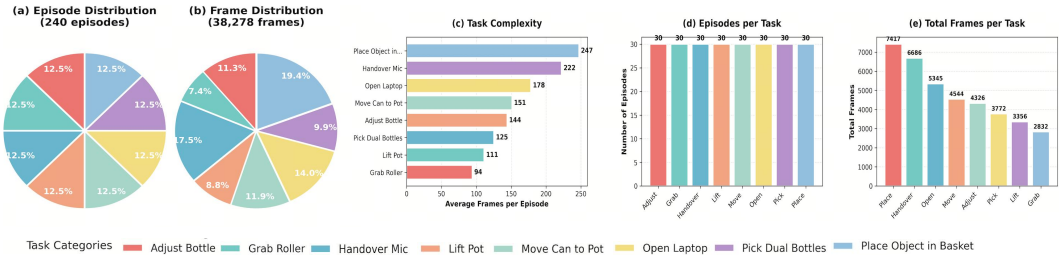


Figure 5: RoboTrain composition: episode counts, frame counts, and trajectory lengths across eight manipulation tasks.

from the same task or subtask are excluded from the mismatch pool, and m sets the margin. This is an instruction contrast, not a success–failure ranking loss. The local time template specifies within-window ordering, not an absolute task-completion percentage. Appendix B expands the per-window terms.

3.2 Data Collection

RoboTrain contains 240 demonstration episodes and 38,278 frames across eight manipulation tasks, with 30 episodes per task (Figure 5). Equal episode counts yield unequal frame counts because trajectory lengths differ. Keyframe-aligned subtask instructions and trajectory segments support both training stages; Appendix A details temporal window construction.

Observations are rendered at 720×720 resolution with an 80° field of view. RoboTrain supplies simulation trajectories for CTR training and predictor adaptation; the visual encoder remains frozen. Goal images and demonstration videos are references supplied to the reward interface and are distinct from the evaluated episode's future observations. The physical-platform sequences are used only for the qualitative transfer examples.

3.3 Stage II: Fine-Tuning V-JEPA2

Stage II freezes the complete learned critic $\phi^\star$, including all reward-side projections, and updates only the dynamics predictor θ. Pretrained visual, target, and goal encoders remain frozen. A batch contains B valid windows, each with the current observation and $H = 4$ future observations:

$$\begin{aligned} \hat{h}_{i,1:H} &= \mathrm{Rollout}(P_\theta, h_{i,0}, s_{i,0}, e_{i,0}, a_{i,0:H-1}), \\ h^\star_{i,k} &= \mathrm{sg}\big(\bar{E}(x_{i,k})\big)\,. \end{aligned} \tag{7}$$

Dynamics prediction. All valid windows, including failures and nonmonotonic motion, contribute

$$\mathcal{L}_{\mathrm{rec}}(\theta) = \frac{1}{BH} \sum_{i=1}^{B} \sum_{k=1}^{H} \left\| \hat{h}_{i,k} - h^\star_{i,k} \right\|_F^2 . \tag{8}$$

The Frobenius norm sums over token/channel dimensions; the outer reduction averages windows and prediction steps, without introducing new latent normalization.

Progress guidance. Let $\mathcal{I}_+$ contain only successful subtask windows with reliable monotonic progress labels, and $B_+ = |\mathcal{I}_+|$. The fixed critic scores predicted latents:

$$\hat{r}_{i,1:H} = r_{\phi^\star}\Big(\mathrm{Pool}(\hat{h}_{i,1:H}), \mathcal{G}_i\Big)\,, \qquad \mathcal{L}_{\mathrm{guide}}(\theta) = \frac{\displaystyle\sum_{i \in \mathcal{I}_+} \sum_{k=1}^{H} w_k (\hat{r}_{i,k} - y_{i,k})^2}{H \max(B_+, 1)}. \tag{9}$$

Here $w_k = \exp[-0.2(k-1)]$ and $y_{i,k}$ retains Eq. 5's label rule. If $B_+ = 0$, the empty sum makes guidance zero while prediction training continues.

Separate reductions keep the guidance coefficient independent of the fraction of eligible windows in a batch. A valid failure transition can therefore train dynamics without acquiring a progress target. With $\lambda_{\mathrm{CTR}} = 0$, the predictor still learns from reconstruction; only the critic-derived gradient is removed.

Table 1: RoboTwin 2.0 task-wise success rates. Baselines are ACT Zhao et al. (2023), Diffusion Policy (DP) Chi et al. (2023), 3D Diffusion Policy (DP3) Ze et al. (2024), RDT Liu et al. (2024), and $\pi 0$ Black et al. (2024). The final row is the macro average over the seven displayed tasks; colored arrows show percentage-point differences from $\pi 0$.

Simulation Task	ACT	DP	DP3	RDT	$\pi 0$	**RoboCTR (ours)**
Grab Roller	66.0%	**98.0%**	77.0%	74.0%	96.0%	90.0%↓ 6.0
Hand over Mic	9.0%	53.0%	93.0%	98.0%	**100.0%**	98.0%↓ 2.0
Lift Pot	7.0%	37.0%	**85.0%**	82.0%	84.0%	80.0%↓ 4.0
Move Can Pot	0.0%	42.0%	28.0%	47.0%	74.0%	**78.0%**↑ 4.0
Open Laptop	32.0%	46.0%	54.0%	61.0%	85.0%	**88.0%**↑ 3.0
Pick Dual Bottles	4.0%	26.0%	25.0%	42.0%	57.0%	**72.0%**↑ 15.0
Place Object Basket	0.0%	17.0%	42.0%	42.0%	62.0%	**84.0%**↑ 22.0
Displayed-task macro	16.9%	45.6%	57.7%	63.7%	79.7%	**84.3%**↑ 4.6

Stage-II objective. The two terms use their own valid sample counts:

$$\min_{\theta} \mathcal{L}_{\text{StageII}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{CTR}} \mathcal{L}_{\text{guide}}, \qquad \phi = \phi^{\star} \text{ fixed}, \quad \lambda_{\text{CTR}} = 0.10. \tag{10}$$

Gradients follow $\mathcal{L}_{\text{guide}} \to r_{\phi^{\star}} \to \hat{Z} \to P_{\theta}$. Freezing $\phi^{\star}$ fixes the critic's parameters, not its derivative with respect to predicted inputs: the critic forward pass remains differentiable, while target features are stop-gradient. Reconstruction fits observed future latents; guidance moves the predicted trajectory toward the fixed target-score sequence. It is nonnegative target regression, not maximization of the critic score. Thus Stage II shapes predictions and MPC ranks actions using the same critic but different objectives.

The visual encoder uses the released V-JEPA2 ViT-Giant checkpoint; the predictor uses V-JEPA2-AC initialization, including action and proprioceptive conditioning. Deployment updates no parameters.

3.4 CTR-Guided Model Predictive Control

Following sampling-based model-based control Chua et al. (2018), MPC samples H-action sequences from a multivariate Gaussian, predicts H future latent states, and scores each trajectory with CTR. We average the per-step CTR scores and penalize the second finite difference of the normalized delta end-effector commands:

$$\mathcal{A}(\mathbf{r}_{1:H}) = \frac{1}{H} \sum_{k=1}^{H} r_k, \qquad J(\mathbf{a}_{t:t+H-1}) = \sum_{k=1}^{H-2} \|\mathbf{a}_{t+k+1} - 2\mathbf{a}_{t+k} + \mathbf{a}_{t+k-1}\|_2^2 . \tag{11}$$

The selection criterion is

$$R(\tau, \mathcal{G}_M) = \mathcal{A}\Big(r_{\phi}(\hat{\mathbf{Z}}_{t+1:t+H}, \mathcal{G}_M)\Big) - \lambda_{\text{jerk}} J(\mathbf{a}_{t:t+H-1}), \tag{12}$$

MPC executes the first action of the highest-scoring sequence and replans after the next observation. The reported configuration uses $H = 4$, 512 candidate sequences, the shifted previous solution to initialize μ, $\Sigma = 0.1I$, and $\lambda_{\text{jerk}} = 0.01$. Actions use the simulator's normalized delta end-effector convention, and action limits are applied before latent rollout. With sampling interval Δt, the physical horizon is $4\Delta t$, not a fixed number of seconds. Algorithm 1 uses the same H-action/H-state indexing.

Subtask completion is determined by $c_j(o_{\le t}, s_t) \in \{0, 1\}$ from actual observation history and available proprioceptive state. The next offline subgoal is activated only when its configured completion condition is satisfied; predicted CTR scores never trigger switching. If consecutive confirmation is configured, the condition must hold for q observations. Subtask or task timeout terminates the rollout as a failure, without skipping a stage. After the final subtask completes, the original task-success predicate determines success.

4 Experiments

We evaluate RoboCTR on the RoboTwin2 benchmark Chen et al. (2025) along three axes: task-wise planning success, the contribution of its training and reward components, and sensitivity to goal modalities and Stage-II guidance.

Table 2: Recorded data and training variants. I/II denote training stages; V-J2 denotes V-JEPA2. These entries are not substituted for the seven-task macro in Table 1.

(a) Data and guidance controls

Method	Data	CTR	Succ. ↑
$\pi0$	–	none	64.9
$\pi0$ + pos.	240 pos.	none	∼64.9
$\pi0$ + pos./neg.	240 p+n	none	64.4
V-JEPA2 only	240 p+n	none	62.8
CTR Stage I	240 p+n	Stage I	74.5
RoboCTR	240 p+n	I+II+MPC	**84.3**

(b) Training schedule

Strategy	I	II	Succ. ↑
Initialization			
No training	–	–	45.2
V-JEPA2 only	–	pretrain	62.8
Optimization			
Joint training	joint	joint	71.5
Two-stage	CTR	V-J2+CTR	**84.3**

Table 3: Recorded CTR component variants. Loss columns identify objective families; test scores are listed separately. Gray highlights group leaders.

Variant	Stage-I loss	Stage-II loss	Test score	Decomp.	Succ. ↑
Stage separation					
V-JEPA2-only	–	$\mathcal{L}_{\mathrm{rec}}$	goal dist.	ordered	62.8
Stage II removed	$\mathcal{L}_{\mathrm{CTR}}$	–	CTR	ordered	**74.0**
Reward interface					
Image-distance	–	rec.+image	image dist.	ordered	**76.7**
Language–CLIP	–	rec.+CLIP	CLIP sim.	ordered	23.3
Task decomposition					
No decomposition	$\mathcal{L}_{\mathrm{CTR}}$	rec.+guide	CTR	none	68.2
Static decomposition	$\mathcal{L}_{\mathrm{CTR}}$	rec.+guide	CTR	fixed	**74.7**
RoboCTR (ours)	$\mathcal{L}_{\mathrm{CTR}}$	rec.+guide	**CTR**	**ordered**	**84.3**

Evaluation settings. The main benchmark uses RoboTwin2 Easy with 100 rollouts per displayed task and equal task weights: RoboCTR achieves 84.3% versus 79.7% for $\pi0$ (+4.6 percentage points). Tables 2 and 3 retain the recorded data, schedule, and component variants.

4.1 Implementation Details

Training data and quantitative evaluation use RoboTwin2. Unless otherwise stated, training uses two NVIDIA A100 GPUs, while planning uses $H = 4$, 512 candidate action sequences, $\Sigma = 0.1I$, and $\lambda_{\mathrm{jerk}} = 0.01$. The physical demonstration uses an ALOHA dual-arm platform with two 6-DoF Piper arms and a single RGB camera. Green and red deltas indicate improvements and regressions in percentage points relative to each table's reference.

4.2 Overall Performance

Table 1 reports seven evaluation tasks. RoboCTR obtains the highest success on Move Can Pot, Open Laptop, Pick Dual Bottles, and Place Object Basket, with gains of 3–22 percentage points over $\pi0$. It is lower on Grab Roller, Hand over Mic, and Lift Pot, where DP, $\pi0$, and DP3 provide the strongest task-wise results. The displayed-task macro is 84.3% for RoboCTR and 79.7% for $\pi0$, a gain of 4.6 percentage points; all seven rows, including the three regressions, contribute to this average. The largest gains occur on Pick Dual Bottles and Place Object Basket, whereas near-ceiling baseline success on Grab Roller and Hand over Mic leaves substantially less room for improvement.

Table 2(a) shows little change between the reported positive-only and positive/negative data variants, at approximately 64.9% and 64.4%. The schedule variants in Table 2(b) report 45.2% without training, 62.8% for V-JEPA2-only, and 71.5% for joint training. These comparisons concern complete configurations, rather than an isolated guidance coefficient.

4.3 Ablation Summary

Figure 6(b) reports 76.0–77.5% for single modalities and 79.5–80.5% for pairs, with 82.7% in the recorded all-modality setting. Each modality subset is trained separately. A wrong-goal diagnostic instead fixes one checkpoint, replaces goals only at test time, and judges success against the original

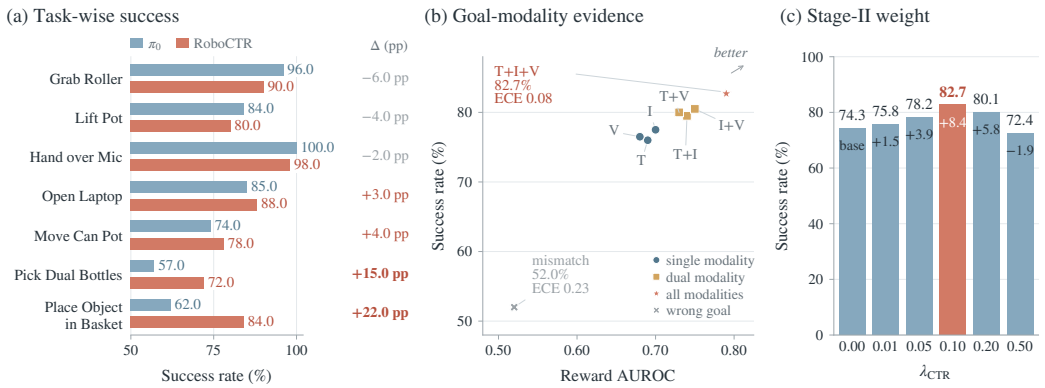


Figure 6: (a) Seven-task benchmark comparison (Table 1); bars start at zero. (b) Separately trained modality variants. (c) Stage-II weight sweep; deltas are relative to $\lambda_{\text{CTR}} = 0$. Panels (b,c) retain the recorded modality and weight-sweep values; they do not redefine the seven-task benchmark macro.

Table 4: Recorded MPC parameter sweep. Gray marks the default configuration; Δ is the percentage-point difference from it.

Configuration	Candidates	H	λ_{jerk}	Success (%)
Smaller candidate set	128	4	0.010	77.1 −5.6
	256	4	0.010	80.3 −2.4
Default	512	4	0.010	82.7
Larger candidate set	1024	4	0.010	**82.9** +0.2
Shorter horizon	512	2	0.010	76.4 −6.3
Longer horizon	512	8	0.010	80.1 −2.6
Lower jerk weight	512	4	0.005	82.1 −0.6
Higher jerk weight	512	4	0.050	81.7 −1.0

task. Whole-goal replacement and single-modality conflict are distinct interventions. The plotted wrong-goal success of 52.0% is not a chance-level estimate.

For the guidance-weight comparison, predictor initialization, data, training steps, optimizer, frozen CTR, test goals, candidate budget, and evaluation initializations are held fixed; only λ_{CTR} changes. At $\lambda_{\text{CTR}} = 0$, prediction-loss training continues, so this is not Stage-II removal. The recorded sweep peaks at 0.10 and declines at larger weights, characterizing task-success sensitivity rather than prediction fidelity.

Table 3 separates training objectives from test scoring. Basic distance scores compare predicted futures with available test-goal representations, never with the evaluated episode's actual future. Image-distance and language–CLIP variants change both training and scoring, reaching 76.7% and 23.3%; they are not test-reward-only interventions. No decomposition and static splitting yield 68.2% and 74.7%. The historical 74.5% Stage-I entry and 74.0% Stage-II-removed entry remain separate.

4.4 Planning and Training Sensitivity

Tables 4 and 5 examine the planning budget and Stage-I loss weights around the recorded default settings.

Candidate budget and horizon. Success increases from 77.1% with 128 candidates to 80.3% with 256 and 82.7% with 512. Doubling the default budget to 1,024 yields only 0.2 additional points, indicating diminishing success gains over this range. At 512 candidates, $H = 2$ and $H = 8$ underperform $H = 4$ by 6.3 and 2.6 points, respectively; extending the rollout is not uniformly beneficial. Changing λ_{jerk} from 0.01 to 0.005 or 0.05 gives 82.1% and 81.7%, smaller changes than reducing the candidate budget or using the shortest horizon.

Table 5: Stage-I loss-weight ablation. Task success is evaluated with different weights for instruction contrast and temporal trend regularization to examine the influence of these two training objectives.

Configuration	Contrastive weight	Trend weight	Success (%)
No instruction contrast	0.0	0.1	74.1 −8.6
Lower contrastive weight	0.1	0.1	79.3 −3.4
Default	0.2	0.1	**84.3**
Higher contrastive weight	0.5	0.1	79.8 −2.9
No trend term	0.2	0.0	80.9 −1.8
Higher trend weight	0.2	0.2	81.4 −1.3
	0.2	0.5	78.6 −4.1



Real World Task 1: Pick Dual Bottles



Real World Task 2: Place Object Basket

Figure 7: Qualitative physical-platform rollouts on an ALOHA dual-arm system for picking dual bottles and placing an object into a basket.

Critic supervision. In the recorded Stage-I sweep, removing instruction contrast gives 74.1%, and removing the trend term gives 80.9%, versus 82.7% at the default weights. Increasing either coefficient beyond the default also lowers success. Moderate weights balance instruction discrimination, local ordering, and progress regression; these one-factor sweeps do not identify interactions between losses.

4.5 Real-world Performance

We transfer the learned policy to an ALOHA dual-arm platform for bottle-picking and basket-placement rollouts. Figure 7 shows successive stages of these physical manipulations; the evidence is qualitative rather than a quantitative sim-to-real comparison. The successive frames make the approach, grasping, and object transport stages visible, complementing the simulation success rates with a view of intermediate configurations and coordinated manipulation on physical hardware. These views connect trajectory scoring to physical execution.

5 Conclusion

RoboCTR turns a predicted latent trajectory into a goal-conditioned progress sequence and reuses that interface across reward learning, predictor adaptation, and MPC. Stage I learns the critic from local progress, instruction contrast, and a monotonic trend prior; Stage II freezes the critic and adapts only the predictor; deployment uses the same critic to rank candidate rollouts. This handoff provides a single cross-modal reward interface without conflating critic learning with dynamics fitting. Dynamics training uses all valid transitions, while progress supervision is restricted to successful windows with reliable monotonic labels.

On the seven displayed RoboTwin2 tasks, RoboCTR exceeds $\pi 0$ on four tasks and trails it on three, with macro success of 84.3% versus 79.7%, a 4.6-percentage-point gain. Component and modality studies describe the behavior of the reported variants, while the guidance-weight sweep favors a moderate coefficient. These results motivate trajectory-level scoring without establishing a guidance-only causal effect or prediction fidelity. Physical-platform demonstrations illustrate qualitative transfer.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-Supervised Learning From Images With a Joint-Embedding Predictive Architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15619–15629, 2023. URL https://arxiv.org/abs/2301.08243.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, et al. V-JEPA 2: Self-Supervised Video Models Enable Understanding, Prediction and Planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting Feature Prediction for Learning Visual Representations from Video. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2404.08471.

Kate Baumli, Satinder Baveja, Feryal Behbahani, Harris Chan, Gheorghe Comanici, Sebastian Flennerhag, Maxime Gazeau, Kristian Holsheimer, Dan Horgan, Michael Laskin, et al. Vision-Language Models as a Source of Rewards. *arXiv preprint arXiv:2312.09187*, 2023. URL https://arxiv.org/abs/2312.09187.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A Vision-Language-Action Flow Model for General Robot Control. *arXiv preprint arXiv:2410.24164*, 2024. URL https://arxiv.org/abs/2410.24164.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. WorldVLA: Towards Autoregressive Action World Model. *arXiv preprint arXiv:2506.21539*, 2025. URL https://arxiv.org/abs/2506.21539.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. RoboTwin 2.0: A Scalable Data Generator and Benchmark with Strong Domain Randomization for Robust Bimanual Robotic Manipulation. *arXiv preprint arXiv:2506.18088*, 2025. URL https://arxiv.org/abs/2506.18088.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://www.roboticsproceedings.org/rss19/p026.html.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://arxiv.org/abs/1805.12114.

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal Cycle-Consistency Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1801–1810, 2019. URL https://arxiv.org/abs/1904.07846.

Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual Foresight: Model-Based Deep Reinforcement Learning for Vision-Based Robotic Control. *arXiv preprint arXiv:1812.00568*, 2018. URL https://arxiv.org/abs/1812.00568.

Chelsea Finn and Sergey Levine. Deep Visual Foresight for Planning Robot Motion. In *IEEE International Conference on Robotics and Automation*, 2017. URL https://arxiv.org/abs/1610.00696.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.090. URL https://www.roboticsproceedings.org/rss20/p090.html.

David Ha and Jürgen Schmidhuber. Recurrent World Models Facilitate Policy Evolution. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://arxiv.org/abs/1809.01999.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 2555–2565. PMLR, 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/1912.01603.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025. doi: 10.1038/s41586-025-08744-2. URL https://www.nature.com/articles/s41586-025-08744-2.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, Robust World Models for Continuous Control. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

Nicklas A Hansen, Hao Su, and Xiaolong Wang. Temporal Difference Learning for Model Predictive Control. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 8387–8406. PMLR, 2022. URL https://proceedings.mlr.press/v162/hansen22a.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An Open-Source Vision-Language-Action Model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: A Diffusion Foundation Model for Bimanual Manipulation. *arXiv preprint arXiv:2410.07864*, 2024. URL https://arxiv.org/abs/2410.07864.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. LIV: Language-Image Representations and Rewards for Robotic Control. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 23301–23320. PMLR, 2023a. URL https://proceedings.mlr.press/v202/ma23b.html.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: Towards Universal Visual Reward and Representation via Value-Implicit Pre-Training. In *International Conference on Learning Representations*, 2023b. URL https://arxiv.org/abs/2210.00030.

Pietro Mazzaglia, Tim Verbelen, Bart Dhoedt, Aaron Courville, and Sai Rajeswar. GenRL: Multimodal-foundation world models for generalization in embodied agents. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://doi.org/10.52202/079017-0865.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A Universal Visual Representation for Robot Manipulation. In *Proceedings of the 6th Conference on Robot Learning*, volume 205, pp. 892–909. PMLR, 2023. URL https://proceedings.mlr.press/v205/nair23a.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-Contrastive Networks: Self-Supervised Learning from Video. *arXiv preprint arXiv:1704.06888*, 2017. URL https://arxiv.org/abs/1704.06888.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and Characterizing Reward Gaming. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://doi.org/10.52202/068431-0687.

Sumedh Sontakke, Jesse Zhang, Séb Arnold, Karl Pertsch, Erdem Bıyık, Dorsa Sadigh, Chelsea Finn, and Laurent Itti. RoboCLIP: One Demonstration is Enough to Learn Robot Policies. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://doi.org/10.52202/075280-2430.

Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. XIRL: Cross-embodiment Inverse Reinforcement Learning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164, pp. 537–546. PMLR, 2022. URL https://proceedings.mlr.press/v164/zakka22a.html.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D Diffusion Policy: Generalizable Visuomotor Policy Learning via Simple 3D Representations. *arXiv preprint arXiv:2403.03954*, 2024. URL https://arxiv.org/abs/2403.03954.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.016. URL https://www.roboticsproceedings.org/rss19/p016.html.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World Models on Pre-trained Visual Features enable Zero-shot Planning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 79115–79135. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhou25t.html.

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 *Proceedings of the 7th Conference on Robot Learning*, volume 229, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

RoboCTR: Cross-Modal Trajectory Rewarding for World Model-Based Robotic Planning

Supplementary Material

Anonymous submission

Contents

A RoboTrain and Temporal Window Construction . . . 13
B Feature Alignment and Critic Learning . . . 14
C Predictor Guidance and Closed-Loop Execution . . . 15
C.1 Stage-II reduction and gradient flow . . . 15
C.2 Prediction, action selection, and observed completion . . . 15
D Additional Comparisons and Their Interpretation . . . 16
E Qualitative Manipulation Sequences . . . 17
F Statements . . . 17

A RoboTrain and Temporal Window Construction

RoboTrain contains 240 simulation episodes and 38,278 frames across eight tasks, with 30 episodes per task (Figure 5). The tasks cover object lifting, handover, container manipulation, and coordinated dual-arm actions. Equal episode counts do not imply equal frame counts: a longer manipulation contributes more observations than a shorter one. Consequently, the episode and frame distributions describe different aspects of the collection. The seven-task benchmark in Table 1 is an evaluation summary, not an eight-task average of these training-data statistics.

Keyframes and local goals. An episode is annotated offline with an ordered sequence of subtask goals and their keyframe boundaries. The Open Laptop example in Figure 4 illustrates this correspondence between visual changes and local instructions. The annotations separate the goal currently being pursued from the complete task description. A goal image or demonstration segment supplies a reference for that goal; it is not a future observation from the episode being evaluated. This distinction is important when the same goal-conditioned critic is applied to multiple candidate futures.

Aligned observation–action windows. A window contains observations at $t, t + \Delta t, \ldots, t + 4\Delta t$ and the four intervening action intervals. The current observation conditions prediction, and the following four observations provide latent targets. Each action must correspond to the transition between its adjacent observations, rather than to the next or previous interval. Windows remain inside one annotated subtask, so that all four target steps refer to the same active goal. Fixed-interval sampling gives the horizon the physical duration $4\Delta t$; the number of prediction steps alone does not specify seconds.

Dynamics coverage and progress supervision. Validity for dynamics and eligibility for progress are separate properties. A recorded failure can still contain a valid action-conditioned transition and therefore train the predictor. Conversely, successful task completion does not make every intermediate window monotonic: a successful episode can include waiting, repositioning, or recovery. Progress regression, trend regularization, and Stage-II guidance use only successful subtask windows with reliable monotonic labels. Neither failure nor lack of eligibility implies a zero progress target. This separation retains diverse motion for dynamics fitting without forcing recovery motion into an increasing-time template.

Local rather than global progress. Equation 5 supplies an increasing coordinate within an eligible window, not an absolute fraction of the entire task completed. Its endpoints are 0.5 and 1.0, and its exponential shape assigns larger increments toward the end of the window. The labels do not assert that an object has traversed a corresponding fraction of its physical path. Likewise, reaching the end of a predicted progress sequence is not an execution-time completion event; actual observations determine subtask transitions.

Table 6: Sample use for dynamics and progress objectives.

Window type	Dynamics	Progress/trend	Guidance
Eligible monotonic success	Yes	Yes	Yes
Other valid success	Yes	–	–
Valid failure, stall, or backtracking	Yes	–	–

B Feature Alignment and Critic Learning

Frozen representations and learned alignment. For V-JEPA2 ViT-Giant, the visual representation is $h_t = E(x_t) \in \mathbb{R}^{K\times d}$, with $K = 256$ tokens and $d = 1408$. Target features are $h^\star_{t+k} = \mathrm{sg}(\bar{E}(x_{t+k}))$ from the frozen target encoder. Dynamics prediction operates on these token features. CTR instead receives their frame-wise pooled representations through its own latent projection p_z; token pooling does not replace the token-level prediction loss.

For frozen text and video features, the alignment operation centers features at the corresponding annotation granularity and normalizes the residual. Let $\mathcal{T}_g$ contain the N_g text segments at granularity g, and let e_{SV_i} be the features of the L video segments used for video centering. The means and aligned features are

$$M_{T,g} = \frac{1}{N_g}\sum_{c\in\mathcal{T}_g} e_{ST_c}, \qquad M_V = \frac{1}{L}\sum_{i=1}^{L} e_{SV_i}, \tag{13}$$

$$F_{T,g} = \frac{e_{ST_g} - M_{T,g}}{\|e_{ST_g} - M_{T,g}\|_2}, \qquad F_{V,g} = \frac{e_{SV_g} - M_V}{\|e_{SV_g} - M_V\|_2}. \tag{14}$$

Centering and normalization align frozen features; they do not imply that the newly introduced goal projections are frozen. Stage I trains those projections, the latent projection, fusion and temporal modules, and scoring head together as ϕ. The pretrained encoders remain unchanged.

Per-window objectives. For an eligible window with $H = 4$, define $w_k = \exp[-0.2(k-1)]$. The three terms of Eq. 6 are

$$\mathcal{L}_{\mathrm{wmse}} = \frac{1}{H}\sum_{k=1}^{H} w_k(\hat{r}_k - y_k)^2, \tag{15}$$

$$\mathcal{L}_{\mathrm{con}} = \max\left(0, m - \left[\frac{1}{H}\sum_{k=1}^{H}\hat{r}^{g}_{k} - \frac{1}{H}\sum_{k=1}^{H}\hat{r}^{g^-}_{k}\right]\right), \tag{16}$$

$$\mathcal{L}_{\mathrm{trend}} = \frac{1}{H-1}\sum_{k=1}^{H-1}\max(0, \hat{r}_k - \hat{r}_{k+1}). \tag{17}$$

Regression fixes the local score scale, while the trend term penalizes decreases without prescribing an additional step size. Both terms average over eligible successful windows (Table 6). The contrastive term evaluates the same trajectory under the correct instruction and a mismatched instruction from another task in the mini-batch. Its eligibility follows the contrastive sampler; it is not a success–failure ranking loss.

What the terms distinguish. A constant score incurs no trend penalty but generally fails to fit the progress template. A score depending only on timestep can fit that template, but cannot distinguish two instructions for the same segment. Instruction contrast therefore addresses a different property from local ordering. It does not, by itself, establish semantic progress within a task. The margin m controls separation of instruction-conditioned mean scores, whereas the coefficients 0.2 and 0.1 in Eq. 6 weight contrast and trend relative to regression.

Algorithm 1 CTR-Guided MPC

Require: Observations $o_{\le t}$, state s_t, extrinsics e_t, ordered goals, configured completion/timeout rules

Set active subtask $j \leftarrow 1$ and horizon $H \leftarrow 4$
while task not completed **do**
  Encode $o_t \to z_t$ and sample $\mathbf{a}_{t:t+H-1} \sim \mathcal{N}(\mu, \Sigma)$
  for each candidate $\mathbf{a}_{t:t+H-1}$ **do**
    Predict $\hat{\mathbf{Z}}_{t+1:t+H}$ using Eq. 3
    Score $\mathbf{r}_{1:H} = r_{\phi^\star}(\hat{\mathbf{Z}}_{t+1:t+H}, \mathcal{G}_j)$
    $R \leftarrow \frac{1}{H}\sum_{k=1}^{H} r_k - \lambda_{\text{jerk}} \sum_{k=1}^{H-2} \|\mathbf{a}_{t+k+1} - 2\mathbf{a}_{t+k} + \mathbf{a}_{t+k-1}\|_2^2$
  end for
  Select $\mathbf{a}^*_{t:t+H-1} = \arg\max R$ and execute $\mathbf{a}^*_t$
  Observe the next state and image, extend the history, and set $t \leftarrow t+1$
  If subtask or task timeout is reached, **return** failure
  If $c_j(o_{\le t}, s_t)$ has not met the configured confirmation rule, continue
  If j is final, **return** the original task-success outcome; otherwise set $j \leftarrow j+1$
end while

C Predictor Guidance and Closed-Loop Execution

C.1 Stage-II reduction and gradient flow

The reconstruction and guidance losses in Eqs. 8–9 average different sample sets. Reconstruction uses all B valid windows; guidance uses the B_+ eligible successful windows. Dividing both sums by B would additionally scale guidance by the batch's eligible fraction. The separate reductions avoid that implicit reweighting. If $B_+ = 0$, guidance is zero and reconstruction still updates the predictor.

Freezing the critic fixes a function, not its derivative with respect to predicted inputs. To make the distinction explicit, stack the predicted token features into $\hat{h}(\theta)$ and write $f = r_{\phi^\star} \circ \text{Pool}$. The guidance gradient has the chain-rule form

$$\nabla_\theta \mathcal{L}_{\text{guide}} = J_{\hat{h},\theta}^{\mathsf{T}} J_{f,\hat{h}}^{\mathsf{T}} v, \qquad v_{i,k} = \begin{cases} \dfrac{2w_k(\hat{r}_{i,k} - y_{i,k})}{H \max(B_+, 1)}, & i \in \mathcal{I}_+, \\ 0, & i \notin \mathcal{I}_+. \end{cases} \tag{18}$$

The critic Jacobian remains in the computation even though $\phi^\star$ is not updated. Detaching predicted latents would remove this guidance path. In contrast, target features are stop-gradient by definition. Keeping reconstruction explicit separates fitting observed dynamics from optimizing a surrogate score Skalse et al. (2022).

For $\lambda_{\text{CTR}} \ge 0$, the same-batch objectives satisfy $\mathcal{L}_{\text{StageII}} - \mathcal{L}_{\text{rec}} = \lambda_{\text{CTR}} \mathcal{L}_{\text{guide}} \ge 0$. Guidance fits a target score sequence; it is not minimization of negative reward. Thus sharing a critic between training and planning does not make their optimization objectives identical.

C.2 Prediction, action selection, and observed completion

Algorithm 1 separates candidate evaluation from execution. Predicted rollouts are scored against the active subgoal; only the selected first action is executed before observing and replanning. The mean CTR score favors progress over the predicted segment, while the action penalty discourages large second finite differences of normalized commands. This discrete penalty is not a measurement of physical jerk in SI units.

The ordered goal list is fixed offline. A high candidate score can select an action but cannot activate the next subgoal. Switching requires the configured completion predicate on actual observations and state, including consecutive confirmation if configured. Timeouts terminate failure rather than skip a stage. This prevents predicted completion from being treated as observed task success, while leaving the original task-success criterion intact.

Table 7: Reward-interface comparison on the recorded ablation setting.

Reward interface	Scoring cue	Success (%)
Image distance (V-JEPA)	Visual goal proximity	76.7
Language–CLIP	Language–visual similarity	23.3
Concat + masking	Concatenated goal features	64.0
CTR (ours)	**Goal-conditioned trajectory progress**	**82.7**

Table 8: Guidance-weight sensitivity and subtask decomposition.

(a) Guidance weights

λ_{CTR}	0.00	0.01	0.05	0.10	0.20	0.50
Rec.	1.0	1.0	1.0	1.0	1.0	1.0
Guide	0.00	0.01	0.05	0.10	0.20	0.50
Success	74.3	75.8	78.2	**82.7**	80.1	72.4

(b) Decomposition

Strategy	Success (%)
None	68.2
Static split	74.7
Ordered CTR	**82.7**

D Additional Comparisons and Their Interpretation

Reward interfaces. Table 7 collects the recorded reward-interface results, including the concatenation-and-masking variant not displayed in the main component table. CTR reaches 82.7%, followed by image distance at 76.7%, concatenation and masking at 64.0%, and language–CLIP at 23.3%. Visual proximity provides a useful goal cue, but does not explicitly specify temporal progress. Likewise, language–visual similarity and feature concatenation do not themselves define a learned ordering over a predicted segment. These distinctions motivate the shared temporal scorer; the comparison does not isolate architecture from every training choice. In particular, the image-distance and language–CLIP configurations also change predictor training, as shown in Table 3.

Guidance weight. Table 8(a) transposes the recorded sweep so that the common reconstruction weight and changing guidance coefficient can be read across one row. Success rises from 74.3% at zero guidance to 82.7% at 0.10, then declines to 80.1% at 0.20 and 72.4% at 0.50. The sweep is nonmonotonic: stronger guidance is not uniformly better, and the largest coefficient underperforms zero guidance. These values measure downstream success, not latent prediction error or reward calibration. The matched comparison varies the guidance weight while retaining predictor training with reconstruction at zero weight; it must not be interpreted as removing Stage II.

Subtask decomposition. Table 8(b) brings together the no-decomposition and static-split entries with the ordered CTR entry already reported in Table 3. Their respective success rates are 68.2%, 74.7%, and 82.7%. The ordered formulation assigns a local goal to the current stage and advances using observed completion. A static partition instead fixes the segmentation, whereas removing decomposition evaluates without that local goal sequence. This comparison motivates stage-specific goal conditioning, but does not separately measure the accuracy of keyframe boundaries or completion detectors.

Modality availability and goal mismatch. Figure 6(b) compares separately trained modality subsets. That comparison changes available information during learning as well as evaluation. The wrong-goal intervention asks a different question: with a fixed checkpoint, replace the test goal and evaluate success on the original task. A drop under that intervention indicates sensitivity to goal content, but does not alone establish calibration or a chance-level baseline. Whole-goal replacement and a conflicting individual modality should therefore not be conflated.

The ablation values above are retained as their recorded comparisons, rather than pooled with the seven-task benchmark. The benchmark's 84.3% and 79.7% are macro averages of its displayed task rows. Neither those averages nor selected rollout images supply uncertainty estimates for the ablation scores.

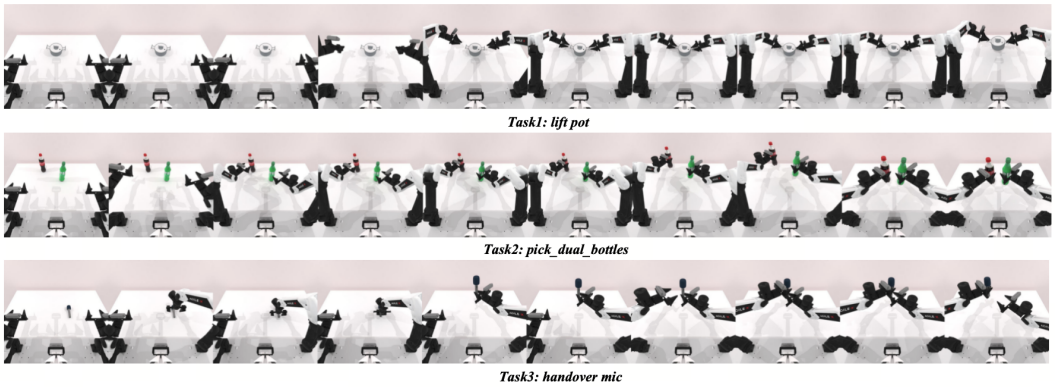


Figure 8: Qualitative simulation sequences for Lift Pot, Pick Dual Bottles, and Hand over Mic. Frames in each row follow the displayed manipulation from left to right.

E Qualitative Manipulation Sequences

Figure 8 shows three simulated manipulations, illustrating the temporal structure that a trajectory-level reward is intended to evaluate. The images expose intermediate configurations absent from terminal success rates. They are qualitative examples, not additional trials.

Temporal structure across tasks. In Lift Pot, the arms approach the pot and raise it from the table. Pick Dual Bottles shows coordinated approach and lifting of two objects; Hand over Mic shows acquisition and transfer of the microphone. In each case, intermediate configurations must lead to the required final object relation. Successive states expose relationships absent from a single appearance match, without establishing a causal contribution of CTR to the displayed movement.

Relation to quantitative and physical evidence. RoboCTR improves over $\pi 0$ on Pick Dual Bottles but trails it on Lift Pot and Hand over Mic. A displayed sequence therefore does not imply superiority on every illustrated task. Figure 7 separately presents physical-platform bottle-picking and basket-placement examples. These images are not combined with the simulation benchmark into a simulation-to-real success statistic.

F Statements

Ethics. Physical deployment requires supervision and safety measures; qualitative examples do not establish general reliability. **AI use.** Generative AI assisted with language, LaTeX, consistency checks, and presentation, not experimental measurements. The authors are responsible for the final manuscript. **Reproducibility.** Sections 3–4 specify the method and evaluation; this appendix expands data alignment, sample eligibility, losses, gradient flow, execution, and existing comparisons.