IROHA: LONG-HORIZON VISUAL REASONING WITH AN AGENTIC HARNESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual reasoning is the process of deriving new task-relevant states, relations, conclusions, or actions through structured operations over visual representations. A video generator produces a candidate solution but does not determine whether it satisfies the task requirements or how to recover from failure. Long-horizon tasks also require checking whether earlier actions have established the conditions needed for later ones, which current video generation models struggle to do. We introduce v**I**sual **R**easoning **O**riented **H**arness for **A**gents (**IROHA**), which makes this control explicit around frozen video and world-action models. Planning defines task objectives, while executable tools produce candidate visual solutions and first-frame guidance for structured tasks, or visual grounding for natural scenes. Verification checks whether generated outcomes satisfy these requirements; its feedback guides candidate selection or revision of the task analysis and generation setup. With frozen state-of-the-art video and world-action backbones, IROHA improves the VBVR score from 0.65 to 0.78 and the LIBERO-Plus simulator success rate from 72.8% to 74.8%. It also improves the RBench scores across three backbones, including gains on long-horizon instructions. Intriguingly, our RBench analysis shows substantial gains in task accomplishment alongside nearly unchanged rendering consistency. We further demonstrate how IROHA assembles multistep video solutions to tasks such as Tower of Hanoi in realistic, synthetically generated scenes, using state checks, retries, and backtracking. These findings show that explicit reasoning and feedback unlock the potential of existing generative capabilities in complex tasks, without additional training or parameter updates.

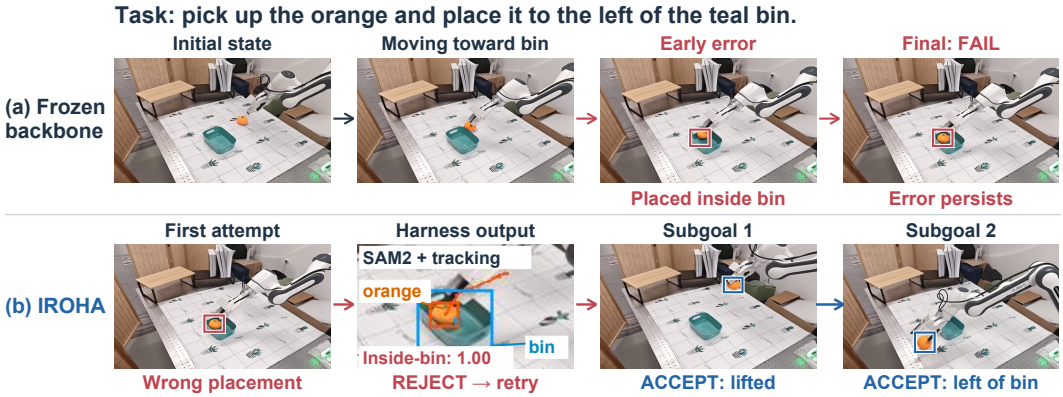


Figure 1: An early error and recovery on RBench. Top: the frozen backbone puts the orange inside the bin and ends in the wrong state. Bottom: IROHA's recorded tracking output shows inside-bin placement. Tool and verifier checks reject the clip and trigger a retry; only accepted lift and left-placement segments enter the final video. The demonstration is detailed in Appendix F.2. This example uses multistep execution with retries. In the main RBench evaluation, the verifier selects one of three candidate videos for each episode.

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.

1 INTRODUCTION

Visual reasoning over evolving scenes requires connecting local state changes to a long-horizon goal. Video generation models provide one basis for modeling these changes by generating candidate state transitions and their resulting visual states (Team Wan et al., 2025; Yang et al., 2024; NVIDIA et al., 2025). However, visual plausibility alone does not determine whether a candidate state satisfies the task-relevant conditions needed for future progress toward the goal. Long-horizon reasoning therefore requires identifying necessary intermediate states, checking which have been achieved, and deciding how to proceed. These reasoning steps depend on one another over time: an early state change determines whether later actions can succeed. The difficulty in visual reasoning is that a plausible local transition need not make correct progress toward the task. A generated transition should be evaluated by the task-relevant conditions it establishes. A visually plausible placement can still establish an incorrect spatial relation and cause a future action to fail. In Figure 1, the frozen backbone puts an orange *inside* a bin; the target relation requires the orange to be *to* the left of the bin. This error persists to the final frame. The central reasoning problem is determining whether the resulting visual state establishes the conditions needed for task completion. Methods such as NEWTON and PhysAgent use planning and verification to improve the physical plausibility of generated videos (Feng et al., 2026; Li et al., 2026b). The tasks we study impose requirements beyond physical plausibility, including selecting a shortest path, satisfying discrete manipulation rules, and choosing robot actions that establish the states needed to complete an instruction. Satisfying these logical constraints and rendering the resulting solution requires search, reasoning about action dependencies, and progress assessment. We investigate whether executable task structure and visual outcome checks can improve visual task completion with the generator held fixed.

We present IROHA, a harness for visual generation and reasoning that adapts the execution, state management, and feedback cycle of coding-agent harnesses (Weng, 2026). Visual generation requires this cycle to connect logical requirements to image content. Tools translate executable solutions into rendered candidates or first-frame guidance; grounding and tracking recover relevant entities and state changes from generated frames; and verification uses task and state checks to guide candidate selection or revision. The harness coordinates short-horizon generation with long-horizon planning and analysis while keeping the backbone frozen. This loop turns visual outcome checks into decisions about how execution should continue. Returning to Figure 1, IROHA rejects the incorrect placement and retries from the initial state; only the accepted lift advances to left-side placement (Appendix F.2). The checked visual state determines how this demonstration continues. Across the benchmark protocols, the harness supports segment-level revision on VBVR-Bench, episode-level improvement on RBench, and trajectory selection per control block on LIBERO-Plus. Across these benchmarks, IROHA improves VBVR-Bench by 0.131, RBench scores by $0.084/0.149/0.106$ across three backbones, and LIBERO-Plus success by 2.02 percentage points (Wang et al., 2026a; Deng et al., 2026; Fei et al., 2026). In the tool-free LingBot-Video diagnostic, task accomplishment improves by 0.193, while pooled rendering consistency changes by 0.005. This observation motivates evaluating how generated outcomes satisfy a task alongside how consistently they are rendered. Rollout-count controls and tool and verifier ablations examine the contributions of candidate sampling, executable guidance and feedback. To examine long-horizon reasoning more directly, we introduce the Real-world Reasoning Benchmark (R^2bench) to evaluate visual-state interpretation and sequential planning under explicit manipulation rules. Multistep Hanoi and brick-assembly video demonstrations illustrate state checks and backtracking, while a held-out V-ReasonBench study evaluates test-time tool synthesis. Together, these studies examine how task constraints and outcome evidence extend the use of frozen generative models for visual reasoning.

Contributions.

❑ We introduce a tool-based interface that connects possible solutions and visual measurements to candidate generation, selection and revision around frozen backbones.

❑ We demonstrate task-completion gains across video and world-action benchmarks, and analyze their relation to rendering consistency, candidate sampling, tool guidance and verification.

❑ We provide R^2bench for planner comparisons, together with multistep execution demonstrations and a held-out evaluation of test-time tool synthesis.

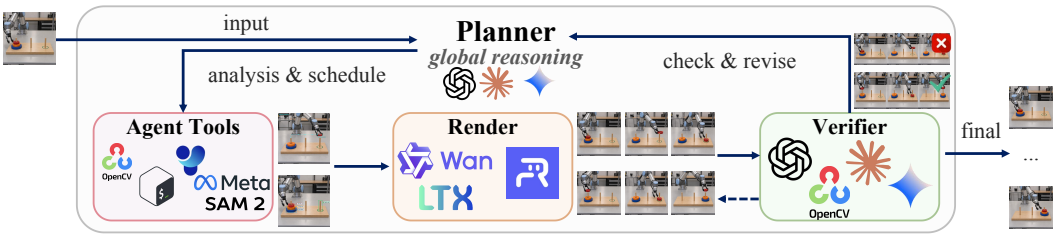


Figure 2: IROHA's planning, tools, frozen backbone and verifier, illustrated on Hanoi. Tools supply first-frame guidance, object grounding and SAM 2 (Ravi et al., 2025) segmentation (Section 4.2).

2 Related Work

Reasoning with video models. Diffusion and flow-based video models (Yang et al., 2025b; Kong et al., 2024), together with other image-to-video generators (HaCohen et al., 2026; Team Wan et al., 2025; OpenAI, 2025; Google DeepMind, 2025; Team Seedance, 2026; Runway, 2025; Kuaishou Technology, 2025), generate clips without exposing an intermediate symbolic task state. Existing approaches incorporate reasoning through fine-tuning on a reasoning corpus (Video-Reason Team, 2026), training with verifiable rewards (Zhu et al., 2026), or ensembling latent trajectories (Wang et al., 2026b). These approaches improve in-domain scores while retaining an out-of-domain performance gap. Other methods keep the generator frozen and use a language module to produce its conditioning text (Huang et al., 2025; Zhang et al., 2025b; Fei et al., 2024; Tong et al., 2026; Chen et al., 2026). Text conditioning alone does not ensure that the generated frames satisfy the inferred task solution. VLM-as-Teacher (Cheng et al., 2026) updates a per-instance LoRA adapter at test time.

Agentic harnesses. Language agents use external tools and search over intermediate steps (Yao et al., 2023b;a; Wang et al., 2023a); Toolformer learns API use through model training (Schick et al., 2023). Visual programming agents compose modules for visual reasoning (Gupta & Kembhavi, 2023; Surís et al., 2023), including image editing in VISPROG. Generated videos must realize the inferred solution. NEWTON (Feng et al., 2026) and PhysAgent (Li et al., 2026b) use planning and verification around a generator or simulator for physical plausibility. Harness VLA (Zhang et al., 2026c) composes frozen VLA contact skills with analytic primitives and memory.

Self-improving agents. Work on self-improving agents distinguishes changes to external components from parameter updates (FrontisAI, 2026). External changes can affect prompts, context, workflows, or harness code (Weng, 2026). Open-ended systems modify the code itself (Zhang et al., 2025a), and Self-Harness (Zhang et al., 2026a) uses regression tests to accept or reject agent-proposed code changes.

3 Method

3.1 Problem Formulation and Overview

A task provides an initial observation I_0 and an instruction p. The harness coordinates a frozen backbone to produce a task outcome, which is evaluated by the benchmark's scorer. For video tasks, we write an episode as (I_0, p, s), where s evaluates a T-frame video V and the frozen image-to-video generator $\mathcal{G}_\theta$ produces candidate videos. For LIBERO-Plus, the frozen world-action model proposes trajectories and the simulator evaluates the executed actions. The benchmark scorer is reserved for evaluation; the harness uses its own verification signals. No model parameters are updated.

A scheduler coordinates the planner, tools, frozen backbone and verifier (Figure 2). VBVR-Bench uses segment-level revision; RBench selects episode-level candidates; LIBERO-Plus selects trajectories per control block (Section 4.2).

For VBVR-Bench, the planner returns a task analysis and a rewritten prompt. A matched tool renders a candidate solution. Rejections trigger prompt revision under budget B; the last candidate is retained when the budget is exhausted. Algorithm 1 details candidate generation and revision. Ground-truth answers and benchmark scores are excluded from these calls.

3.2 COMPONENTS

Planner. The planner interprets the instruction and prepares the backbone's input. In the VBVR-Bench configuration, a single structured MLLM call reads (I_0, p) and returns an analysis a: a typed scene description containing the inferred task, expected answer, object inventory and entities to preserve. It also returns a rewritten instruction $p^\star$ that states the task constraints explicitly.

Tools. Tools are executable programs with declared input signatures. The scheduler searches the existing registry for a tool compatible with the planner's task analysis. Tools produce candidate clips or first-frame guidance. If none applies, the initial fallback uses (I_0, p); retries use feedback-revised prompts. Neither tool matching nor tool execution receives benchmark scores or ground-truth answers. Appendix E distinguishes the offline registry from the separate test-time synthesis experiment.

Frozen backbone. For VBVR-Bench, $\mathcal{G}_\theta$ generates a candidate video conditioned on an input image and a text prompt. The seed stays fixed across revisions, while planner feedback changes the prompt. For RBench, $\mathcal{G}_\theta$ samples three candidate rollouts, and the verifier selects one (Section 4.2.2). For LIBERO-Plus, the frozen world-action model proposes candidate trajectories for each control block (Section 4.2.3).

Verifier. For VBVR-Bench, a multi-frame MLLM judge with three samples scores a candidate (Wei et al., 2022; Wang et al., 2023b). Its median score $m \in [0, 10]$ is combined with a four-term proxy $q \in [0, 1]$ as $v = 0.7m + 0.3(10q)$. A generated video is accepted if $v \geq 6.0$, $q \geq 0.50$, and the judge reports a correct endpoint relative to the planner's answer. Appendix G defines the proxy and gates; rejections trigger prompt revision.

Verification signals and decisions depend on the setting. For RBench, the verifier selects a rollout using task-completion and physical-plausibility scores; its full configuration adds detector permanence and segment-identity checks. For the action-based LIBERO-Plus benchmark, it ranks imagined trajectories using blended MLLM and detector-based progress signals, with the MLLM score used alone when the detector abstains. The simulator's success predicate is used only for evaluation.

4 EXPERIMENTS

We evaluate on three public benchmarks (Figure 14), our R^2bench benchmark with planner comparisons and multi-step video demonstrations (Section 4.3), and the held-out V-ReasonBench (Section 4.4).

4.1 COMMON PROTOCOL

Backbone parameters remain frozen. "Training-free" refers to the harness; the backbones retain their prior training. Benchmark scores are reserved for evaluation. Below we specify the task interfaces, generation budgets and verification signals; Appendix G gives additional settings.

4.2 EVALUATION ACROSS THREE VISUAL REASONING BENCHMARKS

The three benchmarks use complementary criteria: rule-based visual correctness on VBVR-Bench (Wang et al., 2026a), VLM-judged task execution on RBench (Deng et al., 2026), and simulator goal success on LIBERO-Plus (Fei et al., 2026).

4.2.1 VBVR-BENCH: STRUCTURED VISUAL TASKS

Setup. The suite contains 500 samples: five instances of each of 100 task types. The task types comprise 50 in-domain (ID) and 50 out-of-domain (OOD) types across five cognitive categories, where OOD means absent from the million-clip VBVR training corpus. We render with VBVR-Wan2.2 (Video-Reason Team, 2026), frozen, using $T{=}49$ frames at a resolution of 512×512. Each episode runs at most 5 segments with a revision budget of $B{=}2$ per segment under a fixed seed. A gpt-5.4 planner reads (I_0, p), selects the tool and writes revision requests; a gpt-5.2 evaluator scores candidate segments.

Table 1: Results on VBVR-Bench. 🔥 denotes training-based methods; ❄ denotes training-free methods. †Fine-tuned on the VBVR Dataset. Best results in **bold**.

	Method	Overall	ID	OOD
	Human	0.974	0.960	0.988
Closed	Sora 2 (OpenAI, 2025)	0.546	0.569	0.523
	Seedance 2.0 (Team Seedance, 2026)	0.544	0.570	0.517
	Veo 3.1 (Google DeepMind, 2025)	0.480	0.531	0.429
	Runway Gen-4 Turbo (Runway, 2025)	0.403	0.392	0.414
	Kling 2.6 (Kuaishou Technology, 2025)	0.369	0.408	0.330
Open-source	LTX-2.3 (raw) (HaCohen et al., 2026)	0.313	0.329	0.297
	HunyuanVideo (raw) (Kong et al., 2024)	0.273	0.281	0.265
	CogVideoX 1.5 (raw) (Yang et al., 2025b)	0.273	0.302	0.244
	VBVR-LTX2.3†	0.516	0.580	0.453
	VBVR-Wan2.1†	0.592	0.724	0.461
	VBVR-Wan2.2† (Video-Reason Team, 2026)	0.685	0.760	0.610
	+ Demystifying (Wang et al., 2026b) ❄	0.715	0.772	0.658
	+ CollabVR (Kim et al., 2026) ❄	0.730	0.785	0.675
	+ VideoTPO (Chen et al., 2026) ❄	0.745	0.797	0.693
	VLM-as-Teacher (Cheng et al., 2026) 🔥	**0.781**	**0.803**	0.759
Ours	VBVR-Wan2.2† + **IROHA** ❄	**0.781**	0.777	**0.785**

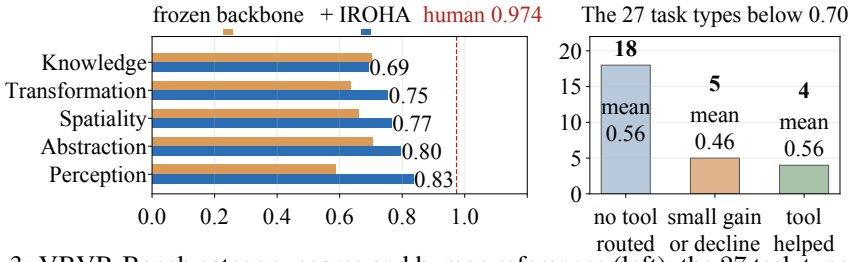


Figure 3: VBVR-Bench category scores and human references (left); the 27 task types below 0.70, grouped by observed tool use and score change (right). Perception ($0.588 \rightarrow 0.835$) has the smallest human gap; Knowledge ($0.703 \rightarrow 0.692$) the largest.

VBVR tool registry. Tool families map the planner's analysis to visual solutions (Figures 10 and 11), using geometric, search, sequence and image-editing routines. Structural preservation is evaluated separately in Appendix L. Appendices E and B detail algorithms and failures.

IROHA scores 0.781 with frozen VBVR-Wan2.2 weights (Table 1). Our paired baseline scores 0.650 at the same per-call settings: 49 frames, 512×512 and seed 1 (Table 6); IROHA adds tool calls and retries. The published checkpoint scores 0.685 under its own settings. VLM-as-Teacher also reports 0.781, using a step-distilled 5B backbone with per-instance LoRA updates; ours uses A14B (14B active parameters per denoising step) (Cheng et al., 2026; Team Wan et al., 2025).

The published fine-tuned baseline has an ID–OOD score gap of 0.150, while IROHA scores 0.777 on ID and 0.785 on OOD tasks. This split is defined relative to the VBVR fine-tuning corpus. The scheduler routes the planner's task description through the tools' declared signatures. Section 4.4 evaluates transfer to a separate benchmark with the registry fixed in advance.

Compared with the baseline using the same per-rollout generation settings, IROHA achieves a paired gain of $+0.131$, 95% CI $[+0.106, +0.157]$; intervals and paired tests are in Appendix O.

Results by task category. IROHA scores 0.781, compared with the human score of 0.974 (Figure 3). Perception has the largest improvement ($0.588 \rightarrow 0.835$) and the smallest gap to human performance. Knowledge has the smallest change ($0.703 \rightarrow 0.692$) and the largest gap. Of the 100 task types, 27 score below 0.70: eighteen lack an appropriate tool; five show small gains or regressions with tools; four improve but remain below 0.70 (Appendix J). Figure 4 and Table 27 show large gains.

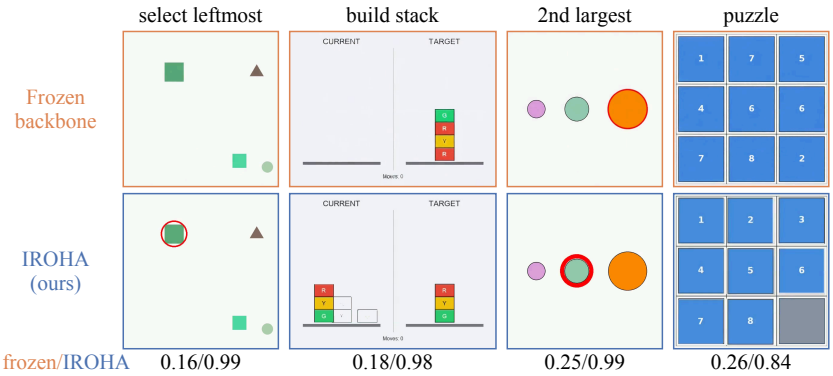


Figure 4: Final frames on four VBVR task types with large gains. Top: frozen-backbone errors. Bottom: tool-computed answers rendered with IROHA. Labels show per-task mean scores.

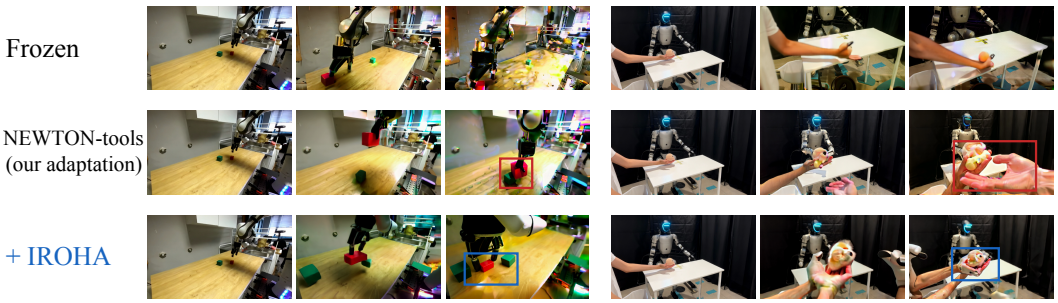


Figure 5: Two RBench cases, frames 0, 24 and 48 (0, 3 and 6 s), with frozen LingBot-Video, NEWTON-tools and IROHA. Left: place the red block left of the green one. Right: take and return the toy. **Blue: IROHA's gripper–block and gripper–palm contact. Red: NEWTON-tools holds the block above/right of the green reference; human hands support the toy throughout the shown frames.**

4.2.2 RBench: task execution in natural scenes

Setup and tools. For RBench, the planner rewrites the instruction from the first frame, the frozen backbone samples three seeds, and the verifier selects the best candidate on task completion and physical plausibility. The baseline uses one raw-prompt generation. The full configuration adds input grounding and output detector checks (Minderer et al., 2022; 2023). Appendix B.3 details the generation and grounding protocol. Appendix F.2 records a rejection-and-retry trace.

Results with frozen backbones. RBench has 250 image-instruction cases per backbone, with 50 in each of five categories. All configurations are scored against the original instruction. Table 2 compares full IROHA with *IROHA w/o tools*, which omits grounding and detector checks. Full IROHA raises the AVG from 0.408 to 0.492 on Wan2.2, from 0.561 to 0.710 on LingBot-Video (Ma et al., 2026), and from 0.531 to 0.637 on Cosmos3-Nano (NVIDIA, 2026). On Wan2.2, adding tools increases AVG from 0.481 to 0.492. Tool gains therefore depend on the backbone. Figure 5 compares two cases with NEWTON.

Comparison with NEWTON. Our NEWTON adaptation uses the released GitHub tools with the same gpt-5.4 planner, backbone, generation settings, budget, cases and scoring protocol, without the original Flow-GRPO-trained planner. Its AVG scores are $0.469/0.660/0.609$, giving full IROHA gains of $+0.023/+0.050/+0.028$. Category-level results are mixed: NEWTON scores lower on Wan2.2 long-horizon planning (0.504 versus 0.549) and higher on LingBot-Video spatial reasoning (0.786 versus 0.736). Table 9 reports paired intervals for harness configurations relative to raw-prompt best-of-three. Appendix C.1 examines the released verification interface.

Task accomplishment and rendering consistency. In the LingBot-Video tool-free diagnostic, pooled task accomplishment improves from 0.225 to 0.418, while rendering consistency changes from 0.343 to 0.348 (Table 3). All seven task-accomplishment criteria improve. Object consistency

Table 2: Results on RBench (0–1 scale; 50 target cases per task category). AVG is the macro-average of the official task scores across the five task categories. IROHA (ours) includes grounding and detector checks; IROHA w/o tools omits both. Harness rows use three candidates; frozen uses one. Multi-ent.: multi-entity. Bold marks within-backbone maxima. §NEWTON: inference-only adaptation, without Flow-GRPO.

Public backbone	Configuration	Common	Spatial	Visual	Long-hor.	Multi-ent.	AVG
Wan2.2_A14B	frozen	0.381	0.454	0.330	0.501	0.373	0.408
	+ NEWTON§	0.434	0.577	0.410	0.504	**0.418**	0.469
	IROHA w/o tools	0.439	0.539	**0.475**	0.539	0.411	0.481
	IROHA (ours)	**0.451**	**0.582**	0.468	**0.549**	0.411	**0.492**
LingBot-Video	frozen	0.578	0.643	0.505	0.634	0.444	0.561
	+ NEWTON§	0.661	0.786	0.560	0.709	0.584	0.660
	IROHA w/o tools	0.671	**0.811**	0.560	0.809	0.667	0.704
	IROHA (ours)	**0.678**	0.736	**0.593**	**0.857**	**0.687**	**0.710**
Cosmos3-Nano	frozen	0.550	0.670	0.394	0.576	0.466	0.531
	+ NEWTON§	0.605	**0.808**	0.462	0.659	0.509	0.609
	IROHA w/o tools	0.643	0.765	**0.499**	0.644	0.519	0.614
	IROHA (ours)	**0.663**	0.805	0.487	**0.689**	**0.539**	**0.637**

Table 3: LingBot-Video RBench diagnostic: frozen vs. tool-free harness, same judge; 0–1 criterion scores. Group means pool valid case–criterion entries (Appendix K).

	Task accomplishment	Rendering consistency
Frozen	0.225	0.343
Harness	0.418	0.348
Δ	+0.193	+0.005

Table 4: Completion on R^2bench (%) over 50 paired cases shared across planners. Gain and paired 95% CI are in percentage points. Full results: Appendix I.

Planner	One-shot → loop	Gain [95% CI]
Haiku 4.5	22 → 26	+4 [−4, +12]
Sonnet 5	46 → 90	+44 [+28, +60]
Gemini 2.5 Pro	60 → 90	+30 [+14, +46]
Gemini 3 Flash	70 → 98	+28 [+14, +42]
Gemini 3 Pro	72 → 100	+28 [+16, +40]
GPT-5-mini	66 → 88	+22 [+6, +38]
GPT-5.6-sol	44 → 70	+26 [+14, +40]

Table 5: LIBERO-Plus with frozen LingBot-VA. Simulator goal-success rate (%) on all instances, paired by instance; n is the count per dimension. The harness evaluates three candidates per control block; frozen executes the default. Δ is the change in percentage points.

	Lang.	View	Init.	Layout	Light	Texture	Noise	Overall
n	383	419	393	312	274	289	449	2519
frozen	94.78	42.24	90.08	84.62	80.66	71.63	55.01	72.77
+ harness	**95.56**	**47.02**	**93.13**	**85.90**	**82.85**	69.90	**57.46**	**74.79**
Δ	+0.78	+4.77	+3.05	+1.28	+2.19	−1.73	+2.45	+2.02

falls from 0.405 to 0.385 and manipulator consistency from 0.420 to 0.370. The diagnostic gains concentrate on task accomplishment; Appendix K gives the individual criteria.

The main comparison uses three candidates against one frozen rollout, so its gains include additional sampling. Table 9 controls rollout count. Selector–judge agreement can also affect scores: cross-family rescoring gives positive gains for the tool-free configuration under all three judges (Table 21).

4.2.3 LIBERO-PLUS: SIMULATED ROBOT MANIPULATION

Setup and tools. The planner rewrites the instruction, and the frozen policy samples three candidate trajectories per control block. Selection combines MLLM scores of task progress and motion stability with the detector's measurement of the reduction in the distance between the object and its target. Appendix K gives the scoring formula and selection rules.

LIBERO-Plus perturbs LIBERO (Liu et al., 2023) along seven dimensions and decides success with the simulator's goal predicate. The frozen backbone is LingBot-VA (Li et al., 2026a), which emits both future frames and joint actions; it is near saturation on unperturbed LIBERO (≈98.5%) and our



Figure 6: LIBERO-Plus 1136: put the labelled alphabet soup and tomato sauce in the basket. **Red: frozen leaves tomato sauce outside. Blue: IROHA places both targets inside.** See Appendix R.3 for this qualitative result.



Figure 7: Hanoi challenge: move the middle stack to the right peg. **Red: frozen lifts the whole stack; NEWTON deforms the red disk.** See Appendix R.2 for the comparison protocol. This qualitative comparison uses different rollout budgets and video assembly rules.

frozen baseline scores 72.8% on the perturbed suite. Candidate selection uses the model-predicted rollout; the simulator's success flag is reserved for evaluation.

Across 2,519 paired instances from 413 source traces, overall success increases from 72.77% to 74.79% (+2.02 percentage points; trace-clustered 95% CI $[+0.43, +4.45]$; Table 5). Scores improve in six of the seven dimensions, including viewpoint perturbations (42.2 to 47.0) and sensor noise (55.0 to 57.5), while background texture decreases by 1.73 points. Appendix K reports paired tests; Figure 6 shows one example. The gain costs three imagined rollouts and one joint MLLM assessment per control block.

4.3 R^2BENCH: CURATED MANIPULATION TASKS

Scores on VBVR-Bench, RBench and LIBERO-Plus reflect planning together with video generation or policy execution. We introduce the Real-world Reasoning Benchmark (R^2bench) to compare planners under explicit state-transition rules, measuring initial-state perception, move legality and episode completion. R^2bench comprises the Tower of Hanoi, stacking, sorting, rearrangement and assembly, each with legal moves and a goal state. A solver-verified sampler draws 20 solvable initial states per scenario; a text-to-image model renders them as real-world style images. Planner comparisons use paired subsets to check move legality and completion (Table 8; Figure 9). Perception is evaluated on 100 initial images. Planner completion is evaluated on 50 paired cases. Separate Hanoi video demonstrations examine state checks, retries, and backtracking (Table 28); Figure 7 compares three methods from the same initial scene.

Planner comparison. We compare planners on R^2bench under one-shot planning and legality-gated execution. Table 4 reports seven planners from three families. Six gain 22–44 percentage points; Haiku's estimated gain is 4 points (95% CI $[-4, +12]$). The legality-gated loop combines additional model calls with stepwise feedback for rule-checked task completion. Appendix I reports all eleven planners and repeat measurements; Appendix M reports costs on a separate case subset.

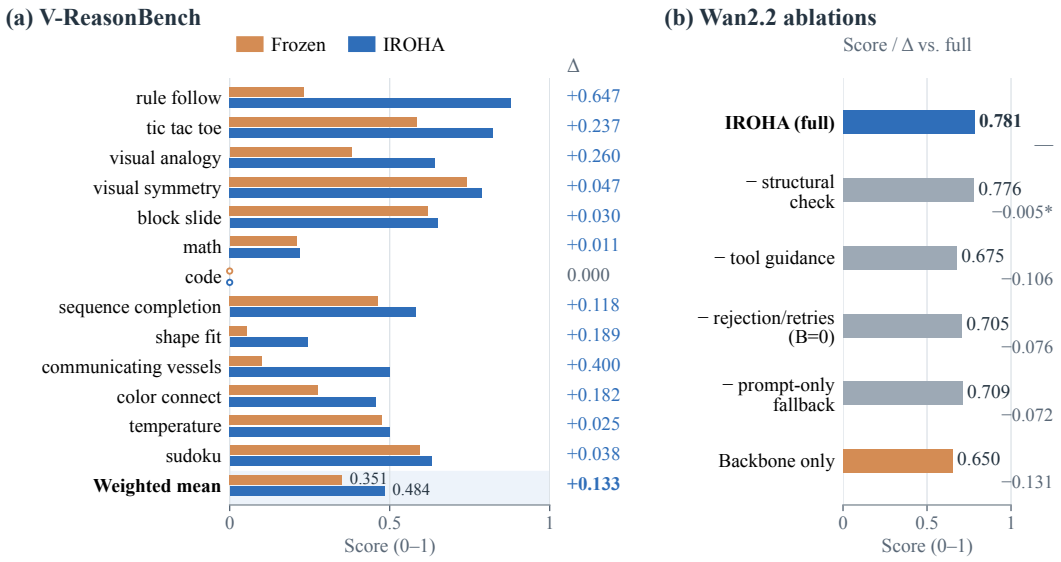


Figure 8: (a) V-ReasonBench: all 328 samples across thirteen families, including Sudoku; the sample-weighted mean rises from 0.351 to 0.484 (+0.133). Δ is the harness gain. (b) Wan2.2 ablations with generator, seed and sampler fixed; labels give scores and changes from the full system. Exact values: Tables 11 and 13. *Paired structural-check analysis: Appendix L.

4.4 A HELD-OUT BENCHMARK

After fixing the initial registry, we evaluated on V-ReasonBench (Luo et al., 2025), which contains 328 samples from 29 task types. The weighted mean is 0.351 → 0.484 (+0.133), with gains in twelve of thirteen families (Figure 8(a)). Appendix B.5 gives the scores and sample counts.

4.5 ABLATION STUDY

All variants use the same generator, seed, configuration and 500 samples (Figure 8(b); Appendix H). Each evaluated variant scores below the full system. The tool-guidance ablation replaces tool guidance with rewritten prompts and removes tool selection and verification against tool outputs; its score decreases by 0.106. With matching tools disabled on held-out types, planning, verification and revision over rewritten prompts score 0.634, compared with 0.530 for the backbone and 0.895 with matching tools (Appendix B.4).

Disabling rejection and retries based on verification (B=0) lowers the score by 0.076; removing the prompt-only fallback lowers it by 0.072. Under the same verifier, increasing independent candidates from K=1 to K=3 yields +0.026, with an interval excluding zero. This sampling gain is smaller than the full harness gain of +0.131 (Appendix H.2). Perception contributes 0.074 of the overall gain (Appendix B).

5 CONCLUSION

IROHA connects short-horizon generation in frozen video and world-action models to logical task requirements and long-horizon state analysis. Its harness translates executable solutions into visual guidance, grounds task state in generated frames, and uses outcome checks to guide selection or revision.

The method improves VBVR-Bench by 0.131, RBench scores by 0.084/0.149/0.106 across three backbones, and LIBERO-Plus success by 2.02 percentage points. In the tool-free LingBot-Video diagnostic, task accomplishment improves by 0.193 while pooled rendering consistency changes by 0.005. Multistep video demonstrations illustrate state checks and backtracking across successive moves. These findings support task completion as a distinct objective for generative reasoning. Future work can allocate checks and retries according to task dependencies and state uncertainty.

AI USE STATEMENT

Generative AI tools assisted with polishing the writing of the paper, drafting parts of it, generating synthetic dataset, and formatting references. The authors have reviewed all AI-assisted work. The authors are responsible for the final text, code, figures, analyses, and claims, including material prepared with AI assistance.

ETHICS STATEMENT

We acknowledge the ICLR Code of Ethics. This study uses public benchmarks, simulated manipulation, and synthetic scenes; no new identifying data was collected. The real-world style demonstrations use generated initial frames and videos and do not document physical robot executions. Synthetic visual content can misrepresent events if presented as recorded footage, so its origin should remain explicit when these materials are reused. Simulator success and automated video checks do not establish safety for deployment on physical robots. Use of third-party datasets and models remains subject to their respective licenses and terms. Evaluation limitations, including possible agreement between MLLM selectors and judges, are discussed in Appendix A.

REPRODUCIBILITY STATEMENT

Section 4 describes the benchmark protocols and evaluation settings. Appendices E, F, and G document the tool interfaces, harness algorithm, and VBVR generation and verification settings, including the seed and revision budget. Appendices B.3 and K give additional RBench and LIBERO-Plus comparisons and evaluation details. Appendices I and R describe the R^2bench planner protocols, data construction, and video demonstrations. Appendix O specifies the confidence intervals and their uncertainty scope. Some configurations depend on hosted model APIs; repeating calls may therefore yield different outputs.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. *arXiv preprint arXiv:2410.24164*, 2024a.

Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-Shot Robotic Manipulation with Pretrained Image-Editing Diffusion Models. In *International Conference on Learning Representations*, 2024b.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In *Proceedings of the Conference on Robot Learning*, volume 229, pp. 2165–2183, 2023.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim

Rocktäschel. Genie: Generative Interactive Environments. In *Proceedings of the International Conference on Machine Learning*, volume 235, pp. 4603–4623, 2024.

Zefan Cai, Haoyi Qiu, Tianyi Ma, Haozhe Zhao, Gengze Zhou, Tingting Liao, Xinyan Velocity Yu, Kung-Hsiang Huang, Ke Wan, Shawn Lin, Parisa Kordjamshidi, Minjia Zhang, Wen Xiao, Jiuxiang Gu, Nanyun Peng, and Junjie Hu. MMGR: Multi-Modal Generative Reasoning Benchmark and Evaluation. *arXiv preprint arXiv:2512.14691*, 2025.

Harold Haodong Chen, Disen Lan, Wen-Jie Shu, Qingyang Liu, Zihan Wang, Sirui Chen, Wenkai Cheng, Kanghao Chen, Hongfei Zhang, Zixin Zhang, Rongjin Guo, Yu Cheng, and Ying-Cong Chen. TiViBench: Benchmarking Think-in-Video Reasoning for Video Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11403–11413, 2026.

Junhao Cheng, Liang Hou, Tianxiong Zhong, Xin Tao, Pengfei Wan, Kun Gai, and Jing Liao. VLMs are Good Teachers for Video Reasoning via Adaptive Test-Time Optimization. *arXiv preprint arXiv:2606.02564*, 2026.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*, 2021.

Nicolae Cudlenco, Mihai Masala, and Marius Leordeanu. Authoring for Living Worlds: Tool-Constrained LLM Agents for Executable Multi-Actor Scenarios. *arXiv preprint arXiv:2604.10383*, 2026.

Yufan Deng, Zilin Pan, Hongyu Zhang, Xiaojie Li, Ruoqing Hu, Yufei Ding, Yiming Zou, Yan Zeng, and Daquan Zhou. Rethinking Video Generation Model for the Embodied World. *arXiv preprint arXiv:2601.15282*, 2026.

Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning Universal Policies via Text-Guided Video Generation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 9156–9172, 2023.

Yilun Du, Mengjiao Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Brian Ichter, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B. Tenenbaum, Leslie Kaelbling, Andy Zeng, and Jonathan Tompson. Video Language Planning. In *International Conference on Learning Representations*, 2024.

Hao Fei, Shengqiong Wu, Wei Ji, Hanwang Zhang, Meishan Zhang, Mong-Li Lee, and Wynne Hsu. Video-of-Thought: Step-by-Step Video Reasoning from Perception to Cognition. In *Proceedings of the International Conference on Machine Learning*, volume 235, pp. 13109–13125, 2024.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: A Progressive Robustness Benchmark for Visual-Language-Action Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38574–38583, 2026.

Yuxiang Feng, Juncheng Wang, Chao Xu, Yijie Qian, Huihan Wang, Wenlong Hou, Yang Liu, Baigui Sun, Yong Liu, and Shujun Wang. NEWTON: Agentic Planning for Physically Grounded Video Generation. *arXiv preprint arXiv:2605.18396*, 2026.

FrontisAI. Awesome self-improving agents, 2026. URL `https://github.com/FrontisAI/Awesome-Self-Improving-Agents`.

Google DeepMind. Veo: A text-to-video generation system. Technical report, Google DeepMind, 2025. URL `https://storage.googleapis.com/deepmind-media/veo/Veo-3-Tech-Report.pdf`.

Tanmay Gupta and Aniruddha Kembhavi. Visual Programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14953–14962, 2023.

Yoav HaCohen, Benny Brazowski, Nisan Chiprut, Yaki Bitterman, Andrew Kvochko, Avishai Berkowitz, Daniel Shalem, Daphna Lifschitz, Dudu Moshe, Eitan Porat, Eitan Richardson, Guy Shiran, Itay Chachy, Jonathan Chetboun, Michael Finkelson, Michael Kupchick, Nir Zabari, Nitzan Guetta, Noa Kotler, Ofir Bibi, Ori Gordon, Poriya Panet, Roi Benita, Shahar Armon, Victor Kulikov, Yaron Inger, Yonatan Shiftan, Zeev Melumian, and Zeev Farbman. LTX-2: Efficient Joint Audio-Visual Foundation Model. *arXiv preprint arXiv:2601.03233*, 2026.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. In *International Conference on Learning Representations*, 2020.

Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. doi: 10.1109/TSSC.1968.300136.

Xuming He, Zehao Fan, Hengjia Li, Fan Zhuo, Hankun Xu, Senlin Cheng, Di Weng, Haifeng Liu, Can Ye, and Boxi Wu. RULER-Bench: Probing Rule-based Reasoning Abilities of Next-level Video Generation Models for Vision Foundation Intelligence. *arXiv preprint arXiv:2512.02622*, 2025.

Lee Hsin-Ying, Hanwen Jiang, Yiqun Mei, Jing Shi, Ming-Hsuan Yang, and Zhixin Shu. MotiMotion: Motion-Controlled Video Generation with Visual Reasoning. In *Proceedings of the International Conference on Machine Learning*, 2026.

Lingxuan Huang, Sizhe He, Hengji Zhou, Liqiang Nie, Lianghao Xia, and Chao Huang. ViMax: Agentic Video Generation. *arXiv preprint arXiv:2606.07649*, 2026.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive Benchmark Suite for Video Generative Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21807–21818, 2024.

Ziqi Huang, Ning Yu, Gordon Chen, Haonan Qiu, Paul Debevec, and Ziwei Liu. VChain: Chain-of-Visual-Thought for Reasoning in Video Generation. *arXiv preprint arXiv:2510.05094*, 2025.

Karthik Inbasekar, Guy Rom, and Omer Shlomovits. WorldJen: An End-to-End Multi-Dimensional Benchmark for Generative Video Models. *arXiv preprint arXiv:2605.03475*, 2026.

Joowon Kim, Seungho Shin, Joonhyung Park, and Eunho Yang. CollabVR: Collaborative Video Reasoning with Vision-Language and Video Generation Models. *arXiv preprint arXiv:2605.08735*, 2026.

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 Conference on Robot Learning*, volume 270, pp. 2679–2713, 2025.

Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B. Tenenbaum. Learning to Act from Actionless Videos through Dense Correspondences. In *International Conference on Learning Representations*, 2024.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, Kathrina Wu, Qin Lin, Junkun Yuan, Yanxin Long, Aladdin Wang, Andong Wang, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Hongmei Wang, Jacob Song, Jiawang Bai, Jianbing Wu, Jinbao Xue, Joey Wang, Kai Wang, Mengyang Liu, Pengyu Li, Shuai Li, Weiyan Wang, Wenqing Yu, Xinchi Deng, Yang Li, Yi Chen, Yutao Cui, Yuanbo Peng, Zhentao Yu, Zhiyu He, Zhiyong Xu, Zixiang Zhou, Zunnan Xu, Yangyu Tao, Qinglin Lu, Songtao Liu, Dax Zhou, Hongfa Wang, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, and Caesar Zhong. HunyuanVideo: A Systematic Framework For Large Video Generative Models. *arXiv preprint arXiv:2412.03603*, 2024.

Kuaishou Technology. Kling AI launches video 2.6 model with "simultaneous audio-visual generation" capability, redefining AI video creation workflow, December 2025. URL https://ir.kuaishou.com/news-releases/news-release-details/kling-ai-launches-video-26-model-simultaneous-audio-visual.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, Yujun Shen, and Yinghao Xu. Causal World Modeling for Robot Control. *arXiv preprint arXiv:2601.21998*, 2026a.

Qirui Li, Jinkun Hao, Yibo Li, Ran Yi, Paul L. Rosin, and Yu-Kun Lai. PhysAgent: Reflective Agentic Physics Control for Physically Plausible Video Generation. *arXiv preprint arXiv:2607.16355*, 2026b.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's Verify Step by Step. In *International Conference on Learning Representations*, 2024.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, volume 36, pp. 44776–44791, 2023.

Yaofang Liu, Xiaodong Cun, Xuebo Liu, Xintao Wang, Yong Zhang, Haoxin Chen, Yang Liu, Tieyong Zeng, Raymond Chan, and Ying Shan. EvalCrafter: Benchmarking and Evaluating Large Video Generation Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22139–22149, 2024.

Yang Luo, Xuanlei Zhao, Baijiong Lin, Lingting Zhu, Liyao Tang, Yuqi Liu, Ying-Cong Chen, Shengju Qian, Xin Wang, and Yang You. V-ReasonBench: Toward Unified Reasoning Benchmark Suite for Video Generation Models. *arXiv preprint arXiv:2511.16668*, 2025.

Shuailei Ma, Jiaqi Liao, Xinyang Wang, Jingjing Wang, Chaoran Feng, Zijing Hu, Chong Bao, Zichen Xi, Yuqi Gan, Weisen Wang, Yanhong Zeng, Qin Zhao, Zifan Shi, Wei Wu, Hao Ouyang, Qiuyu Wang, Shangzhan Zhang, Jiahao Shao, Yipengjing Sun, Liangxiao Hu, Lunke Pan, Nan Xue, Kecheng Zheng, Yinghao Xu, Xing Zhu, Yujun Shen, and Ka Leong Cheng. Scaling Mixture-of-Experts Video Pretraining for Embodied Intelligence. *arXiv preprint arXiv:2607.07675*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative Refinement with Self-Feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594, 2023.

Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. Simple Open-Vocabulary Object Detection. In *Proceedings of the European Conference on Computer Vision*, volume 13670, pp. 728–755, 2022.

Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling Open-Vocabulary Object Detection. In *Advances in Neural Information Processing Systems*, volume 36, pp. 72983–73007, 2023.

Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models understand physical principles? *arXiv preprint arXiv:2501.09038*, 2025.

NVIDIA. Cosmos 3: Omnimodal World Models for Physical AI. *arXiv preprint arXiv:2606.02800*, 2026.

NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun

Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos World Foundation Model Platform for Physical AI. *arXiv preprint arXiv:2501.03575*, 2025.

OpenAI. Sora 2 is here, 2025. URL https://openai.com/index/sora-2/.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment Anything in Images and Videos. In *International Conference on Learning Representations*, 2025.

Runway. Creating with Gen-4 video, 2025. URL https://help.runwayml.com/hc/en-us/articles/37327109429011-Creating-with-Gen-4-Video.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language Models Can Teach Themselves to Use Tools. In *Advances in Neural Information Processing Systems*, volume 36, pp. 68539–68551, 2023.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652, 2023.

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual Inference via Python Execution for Reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11888–11898, 2023.

Team Seedance. Seedance 2.0: Advancing Video Generation for World Complexity. *arXiv preprint arXiv:2604.14148*, 2026.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and Advanced Large-Scale Video Generative Models. *arXiv preprint arXiv:2503.20314*, 2025.

Jingqi Tong, Yurong Mou, Hangcheng Li, Mingzhe Li, Yongzhuo Yang, Ming Zhang, Qiguang Chen, Tianyi Liang, Xiaomeng Hu, Yining Zheng, Xinchi Chen, Jun Zhao, Xuanjing Huang, and Xipeng Qiu. Thinking with Video: Video Generation as a Promising Multimodal Reasoning Paradigm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 41121–41129, 2026.

Video-Reason Team. VBVR-Wan2.2: Fine-tuning Wan2.2-I2V-A14B on the VBVR dataset, 2026. URL https://huggingface.co/Video-Reason/VBVR-Wan2.2.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An Open-Ended Embodied Agent with Large Language Models. *arXiv preprint arXiv:2305.16291*, 2023a.

Maijunxian Wang, Ruisi Wang, Juyi Lin, Ran Ji, Thaddäus Wiedemer, Qingying Gao, Dezhi Luo, Yaoyao Qian, Lianyu Huang, Zelong Hong, Jiahui Ge, Qianli Ma, Hang He, Yifan Zhou, Lingzi

Guo, Lantao Mei, Jiachen Li, Hanwen Xing, Tianqi Zhao, Fengyuan Yu, Weihang Xiao, Yizheng Jiao, Jianheng Hou, Danyang Zhang, Pengcheng Xu, Boyang Zhong, Zehong Zhao, Gaoyun Fang, John Kitaoka, Yile Xu, Hua Xu, Kenton Blacutt, Tin Nguyen, Siyuan Song, Haoran Sun, Shaoyue Wen, Linyang He, Runming Wang, Yanzhi Wang, Mengyue Yang, Ziqiao Ma, Raphaël Millière, Freda Shi, Nuno Vasconcelos, Daniel Khashabi, Alan Yuille, Yilun Du, Ziming Liu, Bo Li, Dahua Lin, Ziwei Liu, Vikash Kumar, Yijiang Li, Lei Yang, Zhongang Cai, and Hokin Deng. A Very Big Video Reasoning Suite. *arXiv preprint arXiv:2602.20159*, 2026a.

Ruisi Wang, Zhongang Cai, Fanyi Pu, Junxiang Xu, Wanqi Yin, Maijunxian Wang, Ran Ji, Chenyang Gu, Bo Li, Ziqi Huang, Hokin Deng, Dahua Lin, Ziwei Liu, and Lei Yang. Demystifying Video Reasoning. In *Proceedings of the European Conference on Computer Vision*, 2026b. arXiv:2603.16870.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations*, 2023b.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/TIP.2003.819861.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022.

Lilian Weng. Harness engineering for self-improvement, July 2026. URL `https://lilianweng.github.io/posts/2026-07-04-harness/`.

Thaddäus Wiedemer, Yuxuan Li, Paul Vicol, Shixiang Shane Gu, Nick Matarese, Kevin Swersky, Been Kim, Priyank Jaini, and Robert Geirhos. Video models are zero-shot learners and reasoners. *arXiv preprint arXiv:2509.20328*, 2025.

Kai Xu, Ellis Brown, Shrikar Madhu, Rob Fergus, He He, and Saining Xie. PaintBench: Deterministic Evaluation of Precise Visual Editing. *arXiv preprint arXiv:2606.00188*, 2026.

Cheng Yang, Haiyuan Wan, Yiran Peng, Xin Cheng, Zhaoyang Yu, Jiayi Zhang, Junchi Yu, Xinlei Yu, Xiawu Zheng, Dongzhan Zhou, and Chenglin Wu. Reasoning via Video: The First Evaluation of Video Models' Reasoning Abilities through Maze-Solving Tasks. *arXiv preprint arXiv:2511.15065*, 2025a.

Sherry Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning Interactive Real-World Simulators. In *International Conference on Learning Representations*, 2024.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. CogVideoX: Text-to-Video Diffusion Models with An Expert Transformer. In *International Conference on Learning Representations*, 2025b.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 11809–11822, 2023a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations*, 2023b.

Hangfan Zhang, Shao Zhang, Kangcong Li, Chen Zhang, Yang Chen, Yiqun Zhang, Lei Bai, and Shuyue Hu. Self-Harness: Harnesses That Improve Themselves. *arXiv preprint arXiv:2606.09498*, 2026a.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin Godel Machine: Open-Ended Evolution of Self-Improving Agents. *arXiv preprint arXiv:2505.22954*, 2025a.

Ke Zhang, Cihan Xiao, Jiacong Xu, Yiqun Mei, and Vishal M. Patel. Think Before You Diffuse: Infusing Physical Rules into Video Diffusion. *arXiv preprint arXiv:2505.21653*, 2025b.

Xiaotian Zhang, Jianhui Wei, Yuan Wang, Jie Tan, Yichen Li, Yan Zhang, Ziyi Chen, Daoan Zhang, Dezhi Yu, Wei Xu, Songtao Jiang, and Zuozhu Liu. How Far Are Video Models from True Multimodal Reasoning? *arXiv preprint arXiv:2604.19193*, 2026b.

Yixian Zhang, Huanming Zhang, Feng Gao, Xiao Li, Zhihao Liu, Yi Nie, Chunyang Zhu, Jiaxing Qiu, Yuchen Yan, Jiyuan Liu, Wenhao Tang, Jiaji Rao, Zhengru Fang, Changxu Wei, Yu Wang, Wenbo Ding, and Chao Yu. Harness VLA: Steering Frozen VLAs into Reliable Manipulation Primitives via Memory-Guided Agents. *arXiv preprint arXiv:2607.08448*, 2026c.

Yuyang Zhang, Wenyao Zhang, Zekun Qi, He Zhang, Haitao Lin, Jingbo Zhang, Yao Mu, Xiaokang Yang, Wenjun Zeng, and Xin Jin. ImageWAM: Do World Action Models Really Need Video Generation, or Just Image Editing? *arXiv preprint arXiv:2606.19531*, 2026d.

Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. RoboDreamer: Learning Compositional World Models for Robot Imagination. In *Proceedings of the International Conference on Machine Learning*, volume 235, pp. 61885–61896, 2024.

Tinghui Zhu, Sheng Zhang, James Y. Huang, Selena Song, Xiaofei Wen, Yuankai Li, Hoifung Poon, and Muhao Chen. Video Models Can Reason with Verifiable Rewards. *arXiv preprint arXiv:2605.15458*, 2026.

APPENDIX

A LIMITATIONS

Generalization. We evaluate generalization by disabling matching tools B.4 and testing on a separate benchmark obtained after the initial registry was fixed (Appendix B.5).

Evaluation. Selector–judge agreement may contribute to MLLM-judged gains. Cross-family RBench rescoring preserves positive tool-free mean gains with different magnitudes; it does not validate the full configuration independently. Rule-based VBVR scores, simulator success and discrete move legality provide other criteria, but endpoint checks do not certify motion physics.

Training and inference cost. Training-free refers to the harness; VBVR-Wan2.2 retains its prior fine-tuning on a million-clip corpus. Main comparisons add inference: VBVR retries, RBench best-of-three selection, and three LIBERO-Plus trajectories with a joint MLLM call per control block. Matched rollout counts do not match all tool or verifier computation. In the separate R^2bench planner study, token cost increases approximately 7–18 times and loop latency is 171–639 seconds per case, excluding video generation (Appendix M). These are planner-only costs. End-to-end latency is unmeasured; hosted APIs do not expose the implementation needed for reliable FLOP accounting. Appendix P states the measurement scope.

Tool reliability. Test-time synthesis runs without human intervention, and a cached program can encode the first instance's assumptions and fail on later instances despite passing compilation and type checks. Timeouts and fallback handle execution failure; they do not establish semantic correctness. These limitations affect reproducibility, robustness and scaling to new task types.

B ADDITIONAL RESULTS

B.1 PER-CATEGORY BREAKDOWN

Table 6 reports scores by cognitive axis for the VBVR-Wan2.2 baseline, the harness and proprietary systems. The harness improves four of the five axes. Perception has the largest increase (0.588 to 0.835); Knowledge decreases from 0.703 to 0.692. Veo 3.1's published category scores are separated into ID and OOD splits (Cheng et al., 2026); we retain its reported overall score and omit category entries for which that source does not provide an Overall-split score. Some Knowledge tasks require factual information beyond the geometric and symbolic operations provided by the registry.

Table 6: Per-category scores from the VBVR-Bench rule-based evaluator (Overall split). We compare against the strongest closed proprietary systems for context. The five axes hold 115 (Abs), 70 (Kno), 150 (Per), 65 (Spa) and 100 (Tra) of the 500 samples, so the Overall column is their sample-weighted mean. Veo 3.1's source reports separate ID/OOD category scores rather than Overall-split category scores (Cheng et al., 2026); "–" marks those unreported entries. [§] is our paired frozen-backbone run described in Section 4.2.1, using the same weights and per-call generation settings as IROHA. **Bold** marks the best score on each axis. The Overall column is the sample-weighted mean over all 500 samples; because the five axes contain unequal numbers of samples it need not equal the unweighted mean of the per-axis columns.

Method	Overall	Abs	Kno	Per	Spa	Tra
Sora 2 (OpenAI, 2025)	0.546	0.582	0.475	0.546	0.530	0.564
Seedance 2.0 (Team Seedance, 2026)	0.544	0.610	0.462	0.539	0.480	0.572
Veo 3.1 (Google DeepMind, 2025)	0.480	–	–	–	–	–
VBVR-Wan2.2 (Video-Reason Team, 2026)	0.685	0.739	0.686	0.633	0.696	0.691
VBVR-Wan2.2[§] (frozen, our run)	0.650	0.705	0.703	0.588	0.660	0.634
+ **IROHA**	**0.781**	**0.795**	0.692	**0.835**	**0.765**	**0.754**

Weighting each axis by its share of the 500 samples, Perception contributes 0.074 of the $+0.131$ aggregate gain, more than half. It is also the largest axis. Knowledge contributes -0.002 (Table 7). The overall improvement is therefore concentrated in Perception, although four axes improve.

Table 7: Cognitive-axis contributions to the $+0.131$ overall gain. Contribution is $(n_i/500)(\text{IROHA}_i - \text{frozen}_i)$ with the per-axis scores of Table 6; the column sums to the paired gain up to rounding.

Axis	n	frozen	IROHA	Contribution	Share
Perception	150	0.588	0.835	$+0.074$	57%
Transformation	100	0.634	0.754	$+0.024$	18%
Abstraction	115	0.705	0.795	$+0.021$	16%
Spatiality	65	0.660	0.765	$+0.014$	10%
Knowledge	70	0.703	0.692	-0.002	-1%
Total	500	0.650	0.781	$+0.131$	100%

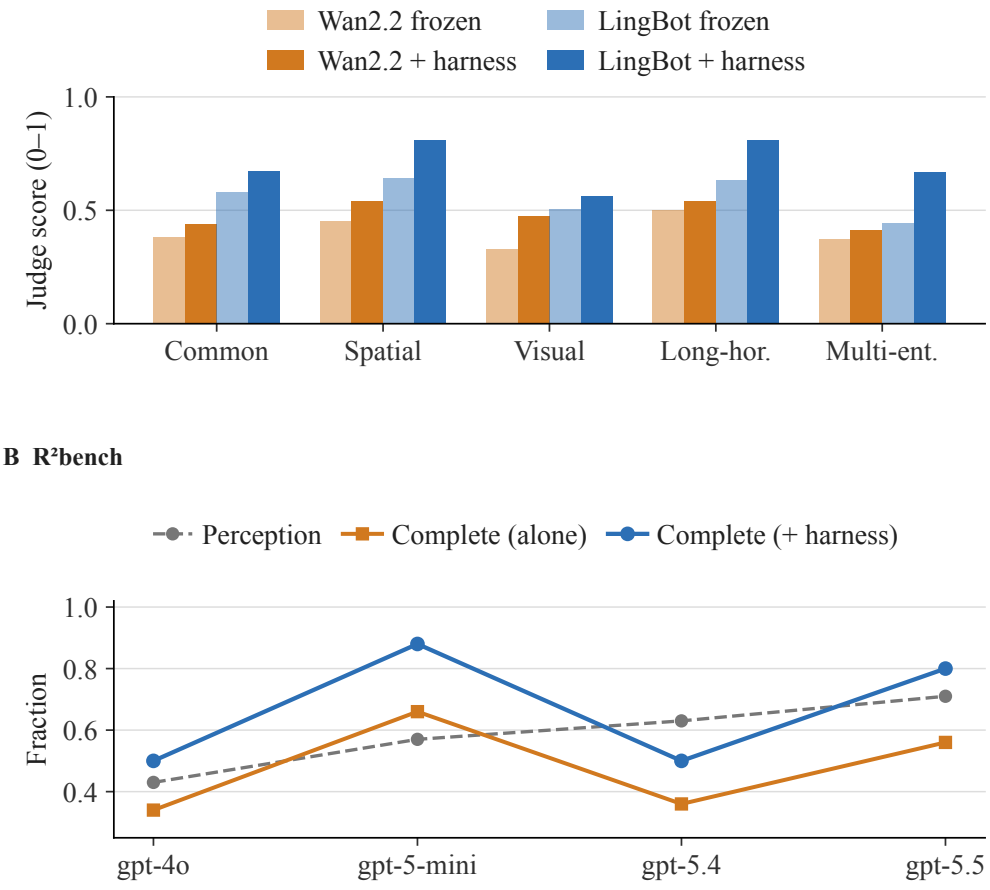


Figure 9: Robot-manipulation studies. Top: RBench judge scores (0–1); the same tool-free loop improves both a general and an embodied-pretrained frozen backbone on all five categories. Bottom: on R^2bench, one-shot completion is not monotonic in planner tier, whereas harnessed completion is uniformly higher; perception rises monotonically (Table 8).

B.2 RBench and R^2bench result visualizations

Figure 9 compares RBench category scores for two frozen backbones and summarizes planner performance on R^2bench. Table 8 reports initial-state perception, move legality and episode completion for four planners. Perception uses all 20 initial states per scenario; legality and completion use the same 10 cases per scenario under one-shot planning and the harness loop.

Table 8: R^2bench (5 scenarios, 20 initial-state images each), fixed renderer, varying planner MLLM. *Percept.*: initial states read correctly off the frame; *Compl.*: proposed moves that are legal; *Complete*: episodes solved. "alone" is the one-shot MLLM; "+h" adds the harness's legality-gated retry loop. All scores are rule-based; perception on all 20 cases per scenario; compliance and completion paired on the same 10 cases per scenario under both protocols.

Planner MLLM	**Percept.** ↑	**Compl. (alone/+h)** ↑	**Complete (alone)** ↑	**Complete (+h)** ↑
gpt-4o (weak)	0.43	0.66 / 0.80	0.34	0.50
gpt-5-mini	0.57	0.89 / 0.99	0.66	0.88
gpt-5.4	0.63	0.82 / 0.95	0.36	0.50
gpt-5.5 (strong)	0.71	0.87 / 0.97	0.56	0.80

B.3 Additional Analysis on RBench

The gains in Table 2 combine prompt rewriting with selection among three generated candidates. Table 9 compares these operations at one and three rollouts per case.

Generation and grounding. For RBench's natural-scene instructions, the registry provides object grounding through an open-vocabulary detector (Minderer et al., 2022; 2023). The planner reads the first frame and rewrites the instruction into a single-shot action description naming the manipulator, object, motion and endpoint. The frozen backbone samples three seeds under that description, and the verifier scores each candidate's final frame from 0 to 10 on task completion and physical plausibility, keeping the best. The baseline is one generation from the raw prompt. The full RBench configuration adds grounding: the detector localizes the instruction's objects in I_0 and supplies their boxes to the planner. Output checks assess object permanence and segment-identity stability to detect identity drift, teleportation and disappearance.

Design. We hold the frozen backbone, case list, seed pool and evaluation judge fixed. Configuration `#2` generates three candidates from the raw instruction and selects one with the verifier. The `#4`−`#2` contrast evaluates the planner's rewrite at three rollouts, conditional on matched verifier inputs and selection rules. The rollout budget is matched; `#4` adds a planner call and `#5` also adds detector processing. Configuration `#3` uses the rewrite with one rollout. Configuration `#1` averages the scores of the same three candidates that `#2` selects from, without selecting an output. This estimates unselected candidate quality: it has the same conditional expectation as uniform selection from those candidates, with a different variance. Its comparison with `#0` also includes variation in the generated videos.

Protocol and uncertainty. All six configurations target the same $5 \times 50 = 250$ cases per backbone in one judge run. Every configuration is evaluated against the benchmark's original instruction. Target counts are nominal: recorded rescoring runs exclude failed judge outputs, with 47–50 valid scores per category for the frozen and harness configurations. The reported 95% intervals use 20,000 paired case-bootstrap resamples over valid paired scores. They describe case-level uncertainty within this run and do not include variation from regenerating the videos or repeating the judge calls. Table 2 reports the category gains to baselines.

Candidate selection. Relative to one raw-prompt rollout, raw-prompt best-of-three (`#2`−`#0`) improves the score by $+0.020$, $+0.058$ and $+0.048$ on Wan2.2, LingBot-Video and Cosmos3-Nano. This recovers approximately 27%, 41% and 58% of the observed tool-free harness gain over the single-rollout baseline. The unselected three-candidate means (`#1`) differ from `#0` by $+0.005$, $+0.005$ and -0.003. Without selection, the expected score of an independent candidate is unchanged by the number of other candidates sampled.

Prompt rewriting. With three rollouts in both configurations, the tool-free harness increment over raw-prompt best-of-three (`#4`−`#2`) is $+0.053$ $[+0.013, +0.095]$, $+0.085$ $[+0.055, +0.135]$ and $+0.035$ $[+0.005, +0.073]$. All three reported intervals exclude zero. At one rollout, rewriting (`#3`−`#0`) adds $+0.028$, $+0.078$ and $+0.030$. On LingBot-Video, $+0.078$ is about half of the *total* tool-free harness gain of $+0.143$ over `#0`; the rewriting increment at three rollouts is $+0.085$. The one- and three-rollout contrasts use different candidate budgets.

Table 9: **RBench score with rollout counts.** K is the number of generated candidates. Configuration #2 is the raw-prompt best-of-three reference. Configuration #4 is the tool-free ablation; #5 is full IROHA, including input grounding and output detector checks. For #4 and #5, Δ_{net} and its 95% CI compare the row with #2; the preceding delta column compares with the single-rollout baseline #0. All rows target 250 cases per backbone in one judge run; CIs use valid paired case scores (20,000 bootstrap resamples). Configuration #1 averages the three candidate scores without selecting an output. Matched K does not include the additional planner or detector computation.

		Budget			Paired contrast		
Config.	Conditioning	K	select	Score	vs. #0	Δ_{net}	95% CI
Wan2.2-I2V-A14B (frozen 0.408)							
#0	raw instruction	1	–	0.408	–	–	
#1	raw, mean of 3	3	none	0.413	+0.005	–	–
#2	raw + best-of-3	3	verif.	0.428	+0.020	–	*(reference)*
#3	rewrite $p^\star$ only	1	–	0.436	+0.028	–	
#4	IROHA w/o tools	3	verif.	0.481	+0.073	+0.053	[+0.013, +0.095]
#5	IROHA (ours)	3	v+det	0.492	+0.084	+0.064	[+0.030, +0.093]
LingBot-Video-1.3B (frozen 0.561)							
#0	raw instruction	1	–	0.561	–	–	
#1	raw, mean of 3	3	none	0.566	+0.005	–	–
#2	raw + best-of-3	3	verif.	0.619	+0.058	–	*(reference)*
#3	rewrite $p^\star$ only	1	–	0.639	+0.078	–	
#4	IROHA w/o tools	3	verif.	0.704	+0.143	+0.085	[+0.055, +0.135]
#5	IROHA (ours)	3	v+det	0.710	+0.149	+0.091	[+0.063, +0.143]
Cosmos3-Nano (frozen 0.531)							
#0	raw instruction	1	–	0.531	–	–	
#1	raw, mean of 3	3	none	0.528	−0.003	–	–
#2	raw + best-of-3	3	verif.	0.579	+0.048	–	*(reference)*
#3	rewrite $p^\star$ only	1	–	0.561	+0.030	–	
#4	IROHA w/o tools	3	verif.	0.614	+0.083	+0.035	[+0.005, +0.073]
#5	IROHA (ours)	3	v+det	0.637	+0.106	+0.058	[+0.023, +0.095]

Table 10: Tool-removal probe on h=6 held-out OOD task types with their tools disabled. In the "Tool removed" setting, the scheduler uses prompt-only conditioning. Mean over all 30 held-out samples (5 per task type); the tool-removed row has a bootstrap 95% CI of $[0.543, 0.725]$.

Setting on held-out OOD task types	**Overall**
Full harness (matching tool available)	0.895
Harness, matching tool removed	0.634
Raw backbone (no harness)	0.530

Grounded tools. Adding input grounding and output detector checks (#5−#4) changes the AVG by +0.011, +0.006 and +0.023 on Wan2.2, LingBot-Video and Cosmos3-Nano, respectively. These point differences measure the two additions jointly. Table 9 reports intervals against the raw best-of-three reference (#2); no paired interval is available for the direct #5−#4 contrast. Input grounding and output checking are not varied separately in this experiment.

B.4 Evaluation with matching tools disabled

The OOD result in Table 1 uses a fixed tool registry and measures performance on tasks absent from the backbone's fine-tuning corpus. To test performance without a matching tool, we hold out h=6 registry-covered OOD task types, disable every tool that targets them, and rerun all $6 \times 5 = 30$ samples. The scheduler then conditions the frozen generator on the rewritten prompt $p^\star$ alone. Table 10 compares the harness with the matching tool available, the prompt-only fallback and the raw backbone.

Planning, verification and revision with prompt-only conditioning improve the mean from 0.530 to 0.634 on these task types. This recovers approximately one quarter of the improvement observed with a matching tool (0.895). Table 10 gives the tool-removed mean's 95% interval; the three point estimates describe the role of tool availability on this subset.

B.5 A HELD-OUT BENCHMARK WITH A FIXED INITIAL REGISTRY

The probe in Appendix B.4 disables tools within the suite used to develop the registry. We additionally evaluate a separate benchmark obtained after the registry was fixed.

The benchmark. V-ReasonBench (Luo et al., 2025) contributes 328 samples over 29 task types in 13 families, each family with its own released checker. We fixed the initial tool registry before obtaining V-ReasonBench. During evaluation, the scheduler first searches the existing tools for a matching handler. If no handler matches the task type, the planner synthesizes a new program and caches it for subsequent samples of that type (Appendix E). Generation uses the budget of the reported VBVR row (49 frames, 512×512, 30 steps, seed 1).

Table 11: V-ReasonBench results. Figure 8(a) plots these results.

Task family	**frozen**	**+ IROHA**	Δ
rule follow	0.233	0.880	+0.647
tic tac toe	0.585	0.822	+0.237
visual analogy	0.381	0.641	+0.260
visual symmetry	0.742	0.789	+0.047
block slide	0.619	0.649	+0.030
math	0.209	0.220	+0.011
code	0.000	0.000	0.000
sequence completion	0.463	0.581	+0.118
shape fit	0.054	0.243	+0.189
communicating vessels	0.100	0.500	+0.400
color connect	0.275	0.457	+0.182
temperature	0.475	0.500	+0.025
sudoku	0.593	0.631	+0.038
Weighted mean (328)	0.351	0.484	+0.133

Results by family. Twelve family means improve. The largest gains are rule follow ($+0.647$), communicating vessels ($+0.400$) and visual analogy ($+0.260$).

Aggregate results and sample counts. Visual analogy, block slide, communicating vessels, color connect and temperature each have 10 samples; sequence completion has 20, rule follow 25, shape fit 28, visual symmetry and tic tac toe 30 each, code 45, and math and sudoku 50 each. These counts sum to 328. With n_f samples in family f, the mean is $\sum_f n_f s_f / 328$. Using the displayed family scores gives 0.350631 and 0.484067, a gain of 0.133436 before rounding. Color connect and temperature scores are converted to $[0, 1]$. Figure 8(a) plots all thirteen families and their sample-weighted mean.

Synthesized-tool failures. In the observed failures, a program encoded assumptions specific to the displayed instance and was reused for other instances of the same task type. These programs compiled and returned the expected output type despite their instance-specific assumptions.

Evaluation protocol and coverage. The released benchmark uses pass@5. We run the harness once per sample with fixed seed 1. Each run uses the retry and segment limits described in Appendix F. The frozen configuration also uses seed 1, and we report the mean score of one run per sample. All thirteen families are included in Table 11. The test-time synthesis procedure is described in Appendix E.

C DETAILED COMPARISON WITH CLOSELY RELATED METHODS

Inside the generator. Video models exhibit zero-shot visual understanding and reasoning (Wiedemer et al., 2025). VBVR-Wan2.2 (Video-Reason Team, 2026) fine-tunes Wan2.2-I2V-A14B on the million-clip VBVR Dataset and is the strongest end-to-end fine-tuned generator on VBVR-Bench; VideoRLVR (Zhu et al., 2026) trains against verifiable rewards and Demystifying Video Reasoning (Wang et al., 2026b) ensembles latent trajectories. We adopt VBVR-Bench for its large leaderboard and its controlled in-/out-of-domain split.

External modules around a frozen generator. This line spans MLLM chain-of-thought that conditions the generator (Huang et al., 2025; Zhang et al., 2025b; Fei et al., 2024; Tong et al., 2026), preference-guided test-time selection (VideoTPO, introduced with TiViBench (Chen et al., 2026)), and VLM-as-Teacher (Cheng et al., 2026), which uses a vision-language judge's feedback to optimize a per-instance LoRA adapter on the generator; Appendix N contrasts that design with ours in detail.

Video-based planning and robot control. UniPi (Du et al., 2023) represents policies as text-conditioned video plans, and Video Language Planning (Du et al., 2024) searches over video and language predictions. AVDC (Ko et al., 2024) derives actions from generated videos through dense correspondences; SuSIE (Black et al., 2024b) proposes image subgoals for a goal-conditioned controller. RoboDreamer (Zhou et al., 2024) composes video predictions from language primitives. Dreamer (Hafner et al., 2020) learns behavior through latent imagination, while Genie (Bruce et al., 2024) learns interactive environments from videos. Vision-language-action models such as RT-2 (Brohan et al., 2023), OpenVLA (Kim et al., 2025) and π_0 (Black et al., 2024a) map visual observations and language instructions to robot actions.

Visual programming and agentic loops. Reflexion (Shinn et al., 2023) and Self-Refine (Madaan et al., 2023) revise outputs using textual feedback; verifier-guided best-of-N selects among sampled solutions (Cobbe et al., 2021; Lightman et al., 2024). Visual programming agents (Gupta & Kembhavi, 2023; Surís et al., 2023) compose visual modules into executable programs; VISPROG also performs language-guided image editing. Our video tasks require the generated visual outcome to satisfy the instruction, so we evaluate the rendered result. NEWTON (Feng et al., 2026) and PhysAgent (Li et al., 2026b) use planning and verification around a generator or simulator, with an emphasis on physical plausibility; the original NEWTON method also trains its planner with reinforcement learning. Our evaluation includes structured visual correctness, instruction-following video generation and simulated manipulation. Harness VLA (Zhang et al., 2026c) uses analytic primitives for non-contact motion, frozen VLA skills for contact, and memory-guided retries. Our LIBERO-Plus study evaluates predicted visual progress before selecting an action trajectory from a frozen world-action model. This supports comparison of model-proposed outcomes without specifying an analytic controller for each candidate; it also inherits prediction errors and adds rollout cost. IROHA adds no training or test-time parameter updates to its pretrained backbone; Table 1 also includes other methods with frozen parameters.

Self-improving harnesses. Self-Harness (Zhang et al., 2026a) has an agent rewrite its own harness for coding and application benchmarks through weakness mining, harness proposal, and regression-tested acceptance. Our harness is fixed during evaluation: matching searches existing tool capabilities, with no benchmark-score input. The held-out synthesis study separately caches newly generated programs.

Concurrent video-generation agents. Authoring for Living Worlds (Cudlenco et al., 2026) builds executable event graphs through tool-constrained planning and renders them with a deterministic 3D engine. ViMax (Huang et al., 2026) coordinates agents for narrative planning and visual continuity across scenes. Both are relevant to external generation control, but their output interfaces and success criteria differ from our frozen-I2V task evaluations. MotiMotion (Hsin-Ying et al., 2026) uses a training-free visual reasoner to refine trajectories and infer secondary motion, paired with a trained motion-conditioned generator.

C.1 Verification in the Public NEWTON Implementation

We inspected the public OpenNewton implementation at commit `072bda2` (Feng et al., 2026). Its verifier compares each generated video with a baseline video generated from the raw scenario text on semantic adherence and physical correctness, including action order. The loop stops when this relative score reaches 5 on a $[-10, 10]$ scale, using improvement over the baseline as its stopping criterion. The returned issue list is retained only when the candidate scores below the baseline, although semantic and physical notes remain available.

The implementation includes simulation references, first/last keyframes, conditioning pre-checks and multi-round feedback (Feng et al., 2026). Its stored state tracks generation attempts and conditioning; the inspected control path has no explicit predicate for the legality of each discrete move. In our

real-world style Hanoi demonstration, recovered post-frame states are checked for legal moves before extending the sequence. Appendix R.2 reports these discrete checks and motion-fidelity observations separately.

The original NEWTON paper trains its planner with Flow-GRPO (Feng et al., 2026), while training code and trained planner have not been released.

Relation to the RBench adaptation. Table 2 evaluates our separately adapted NEWTON-tools configuration under the matched settings described in Section 4.2.2. Figure 5 shows the three-way visual comparison for RBench cases `0017` (spatial relationship) and `0027` (multi-entity collaboration).

D Extended Related Work: Reasoning Benchmarks

The benchmarks we summarize in the main text span perceptual quality, physical commonsense, and discrete reasoning correctness. VBench (Huang et al., 2024) and EvalCrafter (Liu et al., 2024) score perceptual axes of generated video; Physics-IQ (Motamed et al., 2025) probes physical commonsense. A recent wave of suites (Wang et al., 2026a; Luo et al., 2025; He et al., 2025; Chen et al., 2026; Zhang et al., 2026b; Inbasekar et al., 2026) constructs hierarchies of reasoning tasks evaluated against rule-based or carefully designed proxies rather than a learned judge, and maze-style probes (Yang et al., 2025a) narrow the evaluation to a single verifiable problem. In the still-image setting, PaintBench (Xu et al., 2026) makes the same case for judge-free scoring of precise single-answer edits and reports that even the strongest native image editor reaches only 17.1% on its tasks; it excludes agentic and programmatic editors that delegate the edit to an external engine, an approach also used by our tools. These benchmarks report substantial errors on tasks requiring precise visual outcomes. We adopt VBVR-Bench as our main evaluation because it has the largest leaderboard of published systems and the most controlled in-domain and out-of-domain split.

Causal interactions and generative reasoning. MotiBench (Hsin-Ying et al., 2026) tests interaction-centric scenes with motion-triggered events, including secondary effects beyond a requested trajectory. MMGR (Cai et al., 2025) covers abstract reasoning, embodied navigation and physical commonsense, with checks on both answers and intermediate reasoning steps. These suites could test behavior beyond the spatial grounding and endpoint checks emphasized in our natural-scene study.

E Tool Registry and Test-Time Synthesis

Table 12 lists the ten tools implementing the eight families of Figure 11, and the technique each uses.

Table 12: The ten tools in the registry. *Key technique* names the geometric construction or image-processing primitive each uses to turn the planner's scene analysis into a rendered solution.

Tool	Key technique
Marking	sort / filter inventory $\rightarrow$ stroke / dot / outline
Grid edits	list ops (insert / substitute / edit / delete) + grid render
Construction	object bbox $\rightarrow$ target placements (stack / shelf / concentric)
Navigation	grid metrics from start/goal bboxes; path cells $\rightarrow$ per-step agent overlay
Perception	UI state machine, counting overlay, sequence highlight
Geometry	midpoint, intersection, tangent, color-graph connect
Transformation	rotation, mirroring, scaling; symmetry completion
Analytic physics	Snell, projectile, domino chain, fluid density
Generic puzzle	tile clustering $\rightarrow$ grid coordinates; move sequence render
Selection	self-consistent bbox proposal + clustering

Offline registry. The ten functional tools in Table 12 are implemented across 17 modules. The tools are developed before evaluation and execute automatically during evaluation. At inference, the scheduler searches existing tools using the planner's analysis.

Tool interface and self-check. Tools compute a symbolic solution and render a candidate clip or guidance image. A lightweight self-check evaluates the output against the planner's analysis. If it

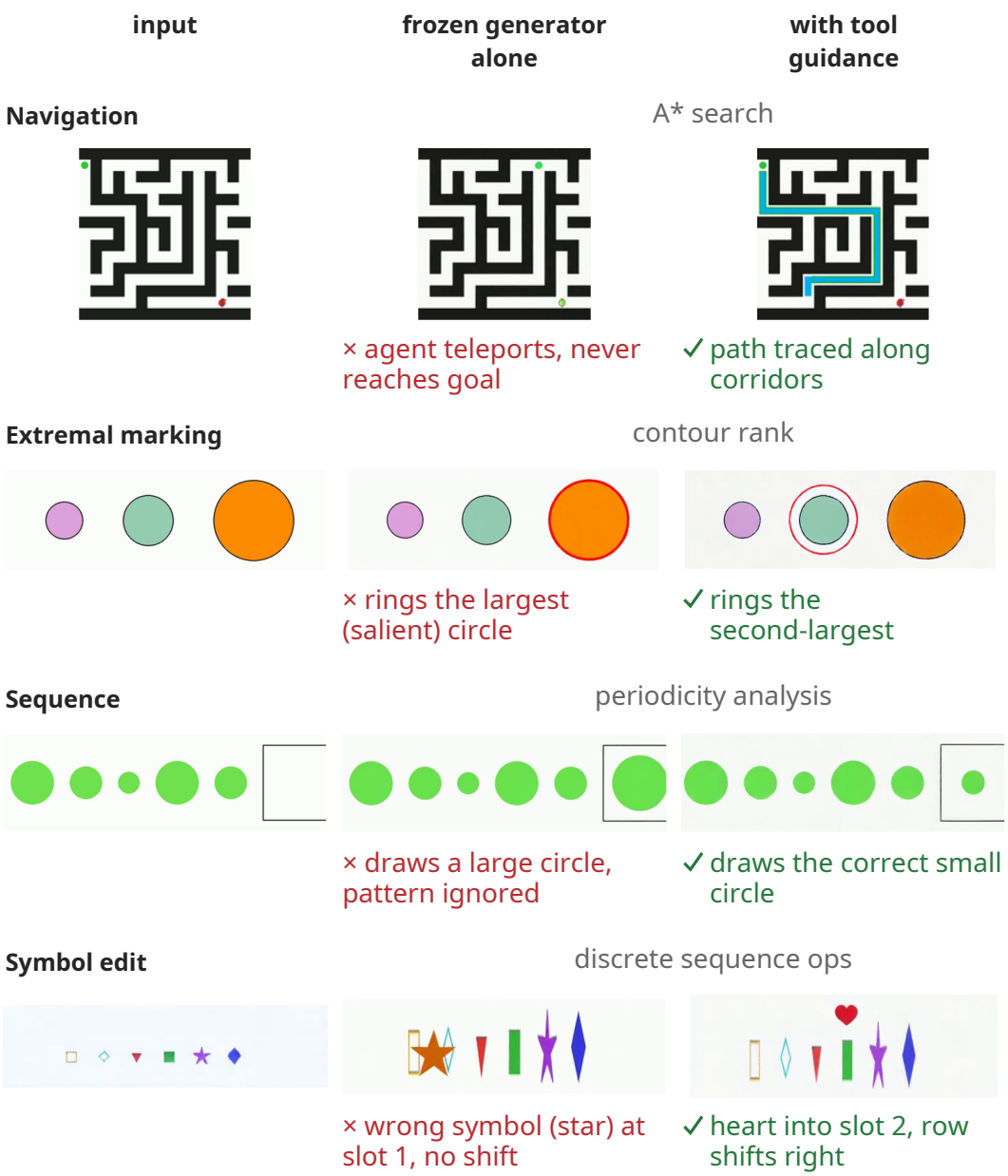


Figure 10: Tool-guided generation on four task families from Figure 11. Each example shows the input, the frozen generator alone and the generator with tool guidance. Panels are unedited frames from the scored VBVR-Bench runs in Section 4.2.1; the middle column is the raw-backbone configuration and the right column is the harness configuration. The baseline examples show teleportation in navigation, marking of the largest circle when the second-largest is requested, failure to follow a size sequence, and insertion of the wrong symbol without the required shifts of neighbouring symbols.

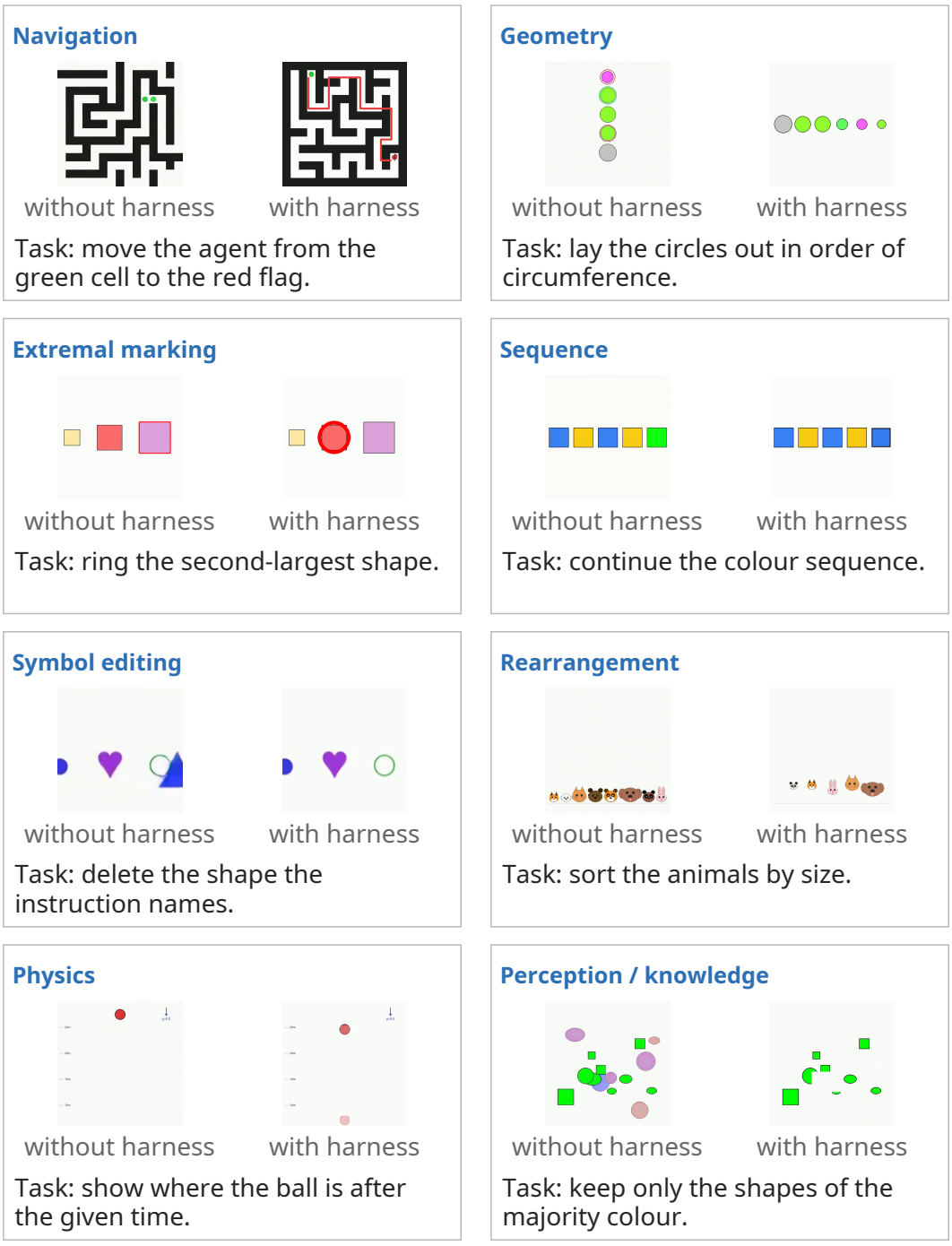


Figure 11: The VBVR-Bench tool registry: eight task families with a shared interface, each consuming the planner's analysis, computing its sub-problem's solution in code, and emitting pixel-level guidance. Every cell pairs the frozen backbone with the same backbone inside the harness on one instance of that family; tasks that require temporal evaluation are shown at the time point where the two clips differ most, with identical cropping. The ten tools implementing these eight families and their per-family techniques are in Appendix E.

confidently judges the answer wrong (confidence ≥ 0.7), the scheduler falls back to the generator. Lower confidence alone does not trigger that rule. Whenever the generator is invoked, it receives a single starting image and text. The natural-scene selection tool uses multiple box proposals; other registry tools are deterministic for fixed inputs.

Test-time synthesis on V-ReasonBench. This separate procedure uses gpt-5.4 for planning and synthesis and gpt-5.2 for scoring, with no human intervention during a run. When a task type is first encountered without a handler, the planner constructs a synthesis prompt from the task text and its initial-frame analysis with the task text and its initial-frame analysis: category, answer, scene description, key elements, object inventory and task type, truncated to $4{,}000$ characters. The template asks for a program that takes the initial frame and analysis as input and returns a 49-frame video. The program must preserve the initial frame and render the inferred answer in subsequent frames. It explicitly requires the program to solve the task when the analysis lacks an answer.

Execution has a 20-second `SIGALRM` timeout. Compilation, execution, return-type or frame-conversion failure discards the program output and invokes generator fallback. Source is cached on disk by the benchmark's `task_folder`, passed as the `task_type` cache key, and reused across that type's samples. Synthesis uses the initial frame, task instruction and planner analysis.

Samples are sorted by image filename and split into four concurrent contiguous shards, each processed in ascending order. The shared cache persists across task types, restarts and runs. Cached programs that fail compilation or execution are reused on later samples; failed synthesis is retried on a subsequent sample.

F Harness Algorithm and Execution Trace

We first specify the VBVR-Bench algorithm, then expand the teaser in Figure 1 with the proposed execution protocol.

F.1 VBVR-Bench algorithm

Algorithm 1 VBVR-Bench candidate generation and feedback revision.

Require: input frame I_0, instruction p, frozen I2V generator $\mathcal{G}_\theta$, registry $\mathcal{O}$, revision budget B
Ensure: a candidate clip C for the current segment

$(a, p^\star) \leftarrow \textsc{Planner}(I_0, p)$; $r \leftarrow 0$
loop
  $o \leftarrow \textsc{Match}(a, \mathcal{O})$
  $c \leftarrow \textsc{RunTool}(o, I_0, a)$, or $\varnothing$ if unavailable/failed
  if $c \neq \varnothing$ and self-check judges the answer wrong with confidence ≥ 0.7 **then**
    $c \leftarrow \varnothing$
  end if
  if $c \neq \varnothing$ **then**
    $I \leftarrow$ tool-provided starting frame; $p_g \leftarrow p^\star$
  else
    $I \leftarrow I_0$; $p_g \leftarrow p$ if $r = 0$, otherwise $p^\star$
  end if
  $C \leftarrow \mathcal{G}_\theta(I, p_g; \text{seed} = 1)$ ▷ image condition
  $(v, q, e, \iota) \leftarrow \textsc{Verify}(C, a)$
  if $v \geq 6.0$ and $q \geq 0.50$ and $e =$ `True` **then**
    return C
  end if
  if $r = B$ **then**
    return C ▷ revision budget exhausted
  end if
  $r \leftarrow r + 1$; $(a, p^\star) \leftarrow \textsc{Planner}(I_0, p, C, \iota)$ ▷ rewrite prompt
end loop

Algorithm 1 makes the tool/fallback interface explicit. e is the judge's endpoint-correctness decision relative to the planner's answer. The scalar score and proxy are defined in Appendix G. At $B = 0$,

the verifier scores the first generated candidate and the scheduler returns it for the current segment. With no applicable tool, the initial generation uses the original frame and task prompt. A rejected generation returns to the planner for a changed prompt even though the seed stays fixed.

The episode scheduler permits at most $S = 5$ segments, each with at most $B = 2$ revisions. It compares successive segment end frames and stops after two consecutive normalized-correlation values reach 0.95 (Appendix G). Appendix L defines the pixel-difference masks used in the structural-preservation analysis.

F.2 RBench execution trace with rejection and retry

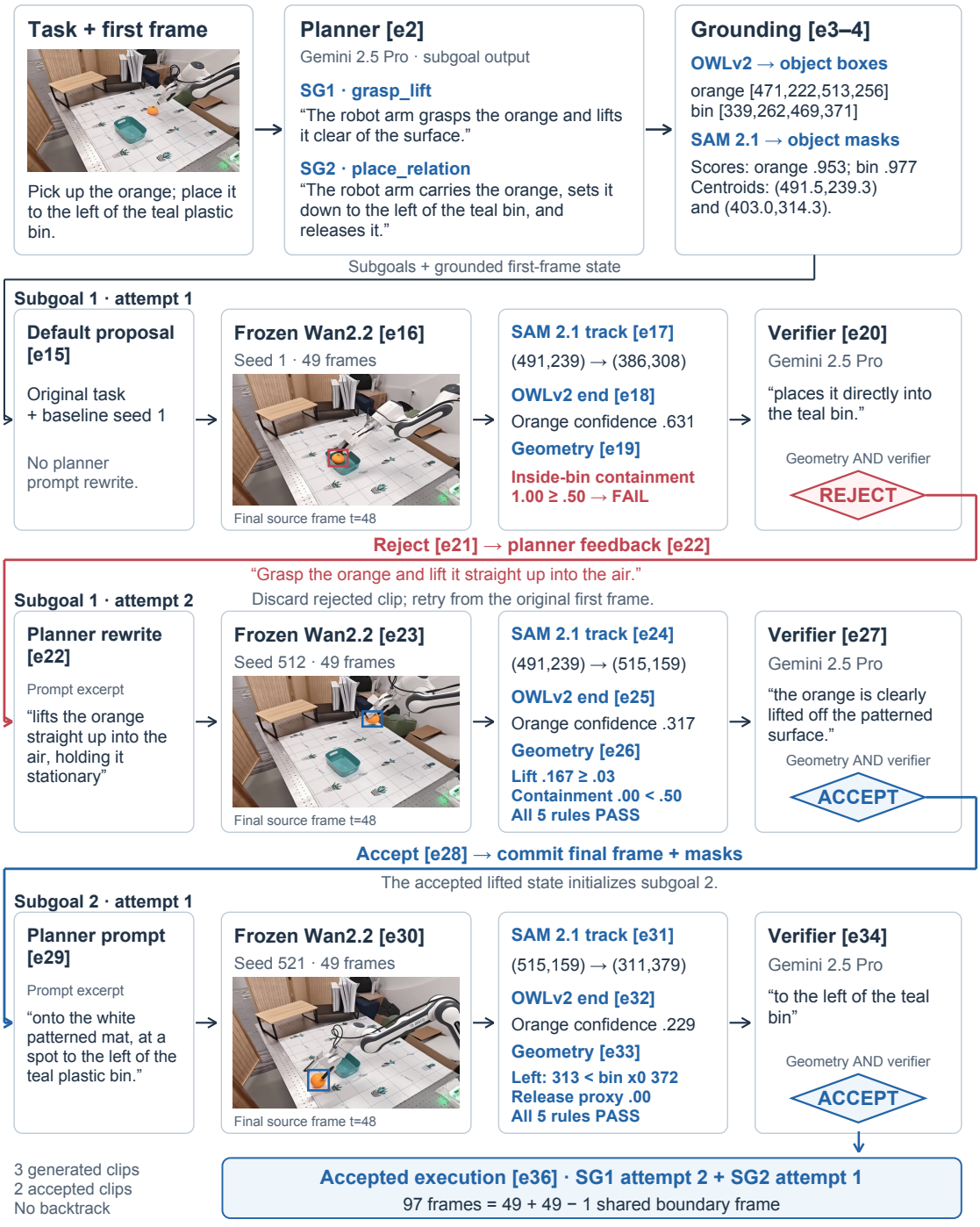


Figure 12: RBench execution workflow, `default_first_s500`. **Red: rejection sends failed checks and verifier feedback to the planner for retry. Blue: acceptance commits the lifted state, then the left-placement segment.** The first proposal uses the original task prompt. Event IDs refer to the recorded calls; only accepted clips enter the final video.

Setup and scope. Figure 12 expands the teaser in Figure 1: pick up an orange and place it to the left of a teal bin (`spatial_relationship_0023`). This qualitative run executes two successive subgoals with retries.

Plan, tools and decisions. The planner decomposes the instruction into grasp-and-lift and left-placement subgoals. Initial OWLv2 (Minderer et al., 2023) boxes prompt SAM 2.1 (Ravi et al., 2025) masks. For each generated attempt, SAM 2.1 propagates the object and bin masks; OWLv2 detects the manipulator and orange in the last frame; geometric checks test persistence, mask-area stability, bin motion and the subgoal-specific condition. The verifier receives the tracked frames and the check results. This demonstration accepts an attempt only when the geometric checks and verifier both pass.

The first attempt uses the original task prompt and baseline seed 1, without a planner rewrite. It places the orange inside the bin: the object-box containment ratio is 1.00, exceeding the 0.50 threshold, and the verifier returns `reject`. The planner receives the failed rule, verifier evidence and suggestion, then rewrites the prompt to lift the orange vertically and hold it above its original position. The retry from the same subgoal start (seed 512) passes with normalized lift 0.167 and bin containment 0.00. Its final frame and masks initialize the second subgoal, whose new prompt specifies the mat to the left of the bin. This attempt (seed 521) passes the left-relation check ($x_{\text{orange}} = 313 < 372$, the bin's left boundary) and the verifier accepts it. The two accepted segments form a 97-frame video; no backtracking is triggered. The logged release check is a bounding-box overlap proxy; the final frame still shows the gripper close to the orange.

Recorded resources. The three generation calls total 887.1 seconds. Six gemini-2.5-pro planning and verification calls use $24{,}870$ tokens and 145.8 seconds, excluding two additional baseline checks.

G Pipeline Hyperparameters

The settings in this section apply to the VBVR-Bench configuration.

Backbone. We load VBVR-Wan2.2 with the released settings, r=32 and strength 0.8, and keep all backbone parameters fixed. Generation runs at $T = 49$ frames, 16 fps, 512×512 resolution and 30 inference steps, with negative prompt "camera moving".

Judge and endpoint decision. The MLLM receives $K = 5$ frames at normalized temporal positions $\{0, 0.25, 0.5, 0.75, 1\}$ and produces $N = 3$ judgments on a 0–10 scale. We take their median m. The judge also outputs `correct_endpoint`, comparing the final frame with the answer inferred by the planner. Each judgment with a false endpoint decision has its score capped at 3. The endpoint flag is determined by majority vote. Acceptance requires a correct endpoint and the score and proxy thresholds below.

Programmatic proxy. The proxy is

$$q = 0.30F + 0.25D + 0.20Q + 0.25A, \qquad q \in [0, 1]. \tag{1}$$

F is first-frame fidelity: single-window SSIM (Wang et al., 2004) against the input frame, mapped piecewise to $[0, 1]$. D measures temporal smoothness from frame differences. Q measures sharpness and noise. A is task-dependent: new red-mark pixels for selection, marking and geometry types, and last-frame fidelity with smoothness otherwise.

For T frames, let X_t contain RGB values in $[0, 1]$, Y_t their grayscale luminance, and R the grayscale input reference. Let $s(Y, R)$ denote single-window SSIM with constants 0.01^2 and 0.03^2. The fidelity mapping is $F = \phi(s(Y_0, R))$, where

$$\phi(s) = \begin{cases} \max(0, 0.5s/0.70), & s < 0.70, \\ 0.5 + 0.3(s - 0.70)/0.15, & 0.70 \le s < 0.85, \\ 0.8 + 0.2(s - 0.85)/0.10, & 0.85 \le s < 0.95, \\ 1, & s \ge 0.95. \end{cases}$$

For temporal smoothness, let δ_t be the mean absolute RGB difference between adjacent frames, averaged over pixels and channels:

$$\delta_t = \text{mean}\,|X_{t+1} - X_t|, \qquad D = 0.6\left[1 - \min\left(1, \frac{\text{std}(\delta)}{0.1}\right)\right] + 0.4\left[1 - \min\left(1, \frac{\max_t \delta_t}{0.3}\right)\right].$$

The standard deviation uses the population definition.

For image quality, sample frames at indices $0, k, 2k, \ldots < T$, with $k = \max(1, \lfloor T/10 \rfloor)$. On each sampled frame, use grayscale values $G_t = 255Y_t$, the four-neighbor Laplacian L, and the 3×3 mean filter H, both with replicated edge padding. Define

$$u_t = \min\left(1, \frac{\text{Var}(LG_t)}{1000}\right), \qquad n_t = \text{mean}\,|G_t - HG_t|,$$

$$Q = \text{mean}_t\left[0.6u_t + 0.4\left(1 - \min\left(1, \frac{n_t}{30}\right)\right)\right].$$

Here $(LG)(x, y) = G(x-1, y) + G(x+1, y) + G(x, y-1) + G(x, y+1) - 4G(x, y)$.

For selection, marking and geometry, count a final-frame pixel as newly red when its 0–255 RGB values satisfy $R > 140$, $G < 100$, $B < 100$ and $R - G > 60$, and the corresponding input pixel falls outside $R > 140$, $G < 100$, $B < 100$. With N_{red} such pixels, $A = \min(1, N_{\text{red}}/50)$. For other task types, $A = 0.5D + 0.5\phi(s(Y_{T-1}, R))$. Each of F, D, Q, A lies in $[0, 1]$.

Acceptance and self-check. The blended score is $v = 0.7m + 0.3(10q)$. A video generated with tool guidance or by prompt-only fallback is accepted if $v \geq 6.0$, $q \geq 0.50$, and `correct_endpoint=True`. Before generation, the tool self-check discards the guidance when it judges the answer wrong with confidence at least 0.7.

Seed and revision policy. VBVR generation uses seed 1 throughout; no seed search is performed. Rejection sends the candidate and feedback to the planner, which rewrites the prompt and revises the analysis as needed. A retry therefore changes text conditioning despite using the same seed. With no matching tool, the initial fallback uses the original image I_0 and instruction p; subsequent attempts use feedback-revised text. When $B = 2$ revisions are exhausted, the last candidate is retained so every sample receives a score.

Segment cap and stopping. At most five segments are attempted; 8 of 500 episodes reach this cap. Let E_j denote the last frame of segment j. The stopping statistic is $u_j = (1 + \text{corr}(E_{j-1}, E_j))/2$, a correlation mapped to $[0, 1]$. A counter starts at zero, increases when $u_j \geq 0.95$ and resets otherwise; stopping requires two consecutive such comparisons. No comparison is made before two segment end frames exist.

H Additional Ablations

Figure 8(b) visualizes the component-ablation scores in Table 13.

Table 13: Wan2.2 ablations with generator, seed and sampler fixed; Δ compares with the full system. *Paired structural-check analysis: Appendix L.

Configuration	Overall	Δ
IROHA (full)	**0.781**	—
− structural check	0.776	−0.005*
− tool guidance	0.675	−0.106
− rejection and retries (B=0)	0.705	−0.076
− prompt-only fallback	0.709	−0.072
Backbone only	0.650	−0.131

H.1 How the answer is rendered

A tool produces the answer in symbolic form, and that answer has to be rendered into a video before the scorer can read it. This appendix evaluates the contribution of the rendering method. Holding the

planner and the tools fixed, we render the same computed answers in different possible ways. The alternatives include: the tool's answer committed as a still image edit with no motion at all, and the input frame held unchanged.

Table 14: Rendering alternatives on the 500 VBVR-Bench cases. The planner and tools are fixed for the first three configurations. The alternatives use a still image edit or hold the input frame unchanged; the frozen-backbone baseline is also shown.

Rendering configuration	**Overall**
Full harness with structural check (IROHA)	**0.781**
Still image edit, no motion	0.439
Input frame held, nothing rendered	0.175
Frozen backbone, no harness	0.650

The full harness scores 0.342 above the still-image-edit variant; this difference includes its generation and selection paths.

Evaluation of final-frame edits. VBVR-Bench includes structured canvas tasks whose answers can be specified as changes to the final frame. For tasks such as marking a shape, a program can compute the required edit before video generation. The unchanged input frame scores 0.175 and sequential still image edit scores 0.439, compared with 0.781 for the harness. We combine several edited images into a sequence for video evaluation. Video generation changes both temporal rendering and frame content in this comparison.

Comparison with published systems. The proprietary video systems in Table 1 score 0.546, 0.544 and 0.480. These scores exceed the still-image-edit score of 0.439 in Table 14, although the systems use different configurations.

Related evidence. ImageWAM (Zhang et al., 2026d) reports competitive image-editing results on manipulation benchmarks, and PaintBench (Xu et al., 2026) evaluates precise visual edits with deterministic checks. Table 14 adds an evaluation of static edits under the VBVR-Bench protocol.

H.2 Best-of-K candidate selection

The harness uses more generator calls per episode than the single-rollout baseline. To measure the effect of additional candidates on VBVR-Bench, we sample the frozen backbone K times per episode from the same seed pool and let the harness verifier select an output. No tool or revision is used. Table 15 compares the three sampling configurations on all 500 samples.

Table 15: Best-of-K on the frozen backbone, selected by the harness's verifier, over all 500 samples scored by all three sampling configurations. The lower block is a paired bootstrap over those samples.

Setting	**Overall**	95% **interval**
$K = 1$	0.650	—
$K = 2$	0.671	—
$K = 3$	0.676	—
K=2 against K=1	+0.021	[+0.009, +0.062]
K=3 against K=1	+0.026	[+0.015, +0.068]

Both reported intervals exclude zero. Increasing the candidate budget from one to three improves the score by +0.026 under this protocol. The full harness gain is +0.131, with an interval of [+0.106, +0.157]. The two comparisons share the verifier; the full harness additionally combines planner analysis, tool guidance and revision. Thus the latter contrast evaluates these components together.

H.3 Verification loop

Retry budget. Disabling rejection and retries based on verification reduces the reported VBVR-Bench score by 0.076 (Table 13). Table 16 varies the retry budget B while holding the other settings, including the segment cap, fixed. The score increases from 0.705 at B=0 to 0.781 at B=2 and remains 0.781 at B=3. At B=0, the verifier scores the first rollout and the scheduler returns it for

the current segment, as in Table 13. Under these settings, a third retry produces no further measured improvement.

Table 16: Retry budget B at the reported configuration, with `max_segments` held fixed at 5. B=0 accepts the first rollout; the reported configuration is B=2.

Retry budget	$B = 0$	$B = 1$	$B = 2$	$B = 3$
Overall	0.705	0.755	0.781	0.781
Improvement over to $B = 0$	-	0.050	0.076	0.076
95% CI	-	[0.025, 0.073]	[0.048, 0.103]	[0.048, 0.103]

H.4 Action-level comparison of verifier signals

We compare selection signals on the 70-task development sample (10 tasks per perturbation dimension), holding the frozen policy, candidate set (N=3) and episode budget fixed. Table 17 shows three dimensions. The MLLM judge improves viewpoint success from 40% to 60% and decreases lighting success from 90% to 60%. Detector-based selection leaves the language and lighting scores unchanged and decreases viewpoint success from 40% to 30%. With the blended signal (0.8 MLLM + 0.2 min–max-normalized detector, margin-gated), viewpoint success is 80% and lighting success is 90%, with 86.7% overall across the three displayed dimensions. This blend is used in Table 5. The development sample and full-suite results cover different task distributions. Across all 70 paired development tasks, there are 5 frozen-only and 6 harness-only successes. The main evaluation uses all 2,519 instances (Section 4.2.3).

Table 17: Verifier-signal comparison on three dimensions of the 70-task development sample (10 tasks per dimension), same frozen policy and budgets throughout. Success rate (%).

Selection signal	**View**	**Lang.**	**Light**	**Overall**
none (frozen policy)	40.0	100.0	90.0	76.7
MLLM judge	60.0	90.0	60.0	70.0
open-vocabulary detector	30.0	100.0	90.0	73.3
blended (used in Table 5)	**80.0**	90.0	90.0	**86.7**

I Planner Capability on R^2bench Across Model Families

We vary the planner on the five R^2bench scenarios with fixed move-legality and goal-completion rules (Appendix R.1; Table 18 and Figure 13). The comparison measures planner variation within this fixed task suite. One-shot completion does not increase monotonically with model tier. Appendix M reports the cost study.

For each planner, p is an exact binomial test on its discordant pairs, and the interval is a paired bootstrap over cases. Because cases recur across planners, we report a two-way clustered bootstrap for the pooled result rather than a pooled exact test.

Across eleven planners, 162 one-shot failures become harness successes and 26 cases change in the opposite direction. The pooled completion gain is $+0.248$, with a two-way clustered bootstrap interval of $[+0.131, +0.375]$ (resampling planners and cases, 20k draws). A mixed-effects estimate, with planner as a random intercept and case as a fixed effect, is $+0.248$, $[+0.205, +0.291]$. Nine per-planner intervals in Table 18 are strictly above zero. The gpt-5.4 interval displays a $+0.00$ lower endpoint, with exact paired $p = 0.118$; the claude-haiku-4.5 interval includes zero.

claude-haiku-4.5 gains $+0.04$ (95% CI $[-0.04, +0.12]$, $p = 0.625$), with a 0.44 stall rate; gpt-4o gains $+0.16$, with a 0.30 stall rate. Both have reported perception accuracy 0.43, but their traces differ after rejection: gpt-4o often proposes a different move, whereas claude-haiku-4.5 repeats an illegal move until its budget is exhausted. In two additional runs of claude-haiku-4.5, the stall rates were 0.56 and 0.40, compared with 0.44 in the original run. Completion ranged from 0.20 to 0.32, against 0.22 one-shot in this repeat analysis. The standard deviation across three runs was 0.060, exceeding the reported $+0.04$ gain.

The other nine planners stall on at most 0.06 of episodes, including claude-sonnet-5 at 0.06, and gain between $+0.14$ and $+0.44$.

Table 18: R^2bench across three model families. *Alone* emits a full plan per case; *+harness* is asked for one move at a time, legality-checked and retried. *Stall* is the fraction of harnessed episodes that exhaust the retry budget without a legal move. *claude-opus-5 is evaluated on the 48 pairs scored under both protocols; two single-shot cases could not be repeated.

	Planner	alone	+harness	fix/break	p	95% CI	stall
GPT	gpt-4o	0.34	**0.50** (+0.16)	9/1	0.021	[+0.04, +0.28]	0.30
	gpt-5-mini	0.66	**0.88** (+0.22)	16/5	0.027	[+0.06, +0.38]	0.00
	gpt-5.4	0.36	**0.50** (+0.14)	11/4	0.118	[+0.00, +0.28]	0.00
	gpt-5.5	0.56	**0.80** (+0.24)	17/5	0.017	[+0.06, +0.42]	0.00
	gpt-5.6-sol	0.44	**0.70** (+0.26)	14/1	0.001	[+0.14, +0.40]	0.00
Gemini	gemini-2.5-pro	0.60	**0.90** (+0.30)	18/3	0.001	[+0.14, +0.46]	0.00
	gemini-3-flash	0.70	**0.98** (+0.28)	15/1	0.001	[+0.14, +0.42]	0.00
	gemini-3-pro	0.72	**1.00** (+0.28)	14/0	<0.001	[+0.16, +0.40]	0.00
Claude	claude-haiku-4.5	0.22	0.26 (+0.04)	3/1	0.625	[−0.04, +0.12]	**0.44**
	claude-sonnet-5	0.46	**0.90** (+0.44)	24/2	<0.001	[+0.28, +0.60]	0.06
	claude-opus-5*	0.50	**0.88** (+0.38)	21/3	<0.001	[+0.21, +0.54]	0.04
	all eleven, clustered	0.51	**0.75**	162/26	—	[+0.13, +0.38]	

Repeat measurements for claude-sonnet-5 and claude-opus-5 use 50 and 48 paired cases, respectively. Sonnet scores 0.90 and 0.86 in two harness runs against 0.46 one-shot; Opus scores 0.88 and 0.90 against 0.50. Their run-to-run standard deviations are 0.028 and 0.014, compared with mean gains of +0.42 and +0.39.

J Failure Analysis: the Task Types Below 0.70

Table 19 lists the task types discussed in Section 4.2.1, grouped by observed tool use and score change. Tool use is the fraction of episodes invoking a tool in the run records; score change is measured relative to the frozen backbone.

Bookshelf decreases by 0.280. Its scorer compares the full frame with a reference using mean absolute difference and two score bands; redrawing the shelf can change pixels enough to cross a score-band boundary. Domino chain decreases by 0.180, and 40% of its episodes invoke the physics tool. That tool models independent falls and omits the gap required by the task.

Eighteen task types invoke no tool. Their mean harness score is 0.561, compared with 0.548 for the frozen baseline. These episodes use prompt rewriting without matched tool guidance. Appendix B.4 evaluates the same fallback mechanism on a separate six-type subset.

K Additional RBench and LIBERO-Plus Analyses

Figure 14 summarizes results across the three public benchmarks. This appendix gives category-level results and evaluation details for Sections 4.2.2 and 4.2.3.

RBench category results. Nine of the fifteen tool-free comparisons and ten of the fifteen full-configuration comparisons with the frozen baseline have bootstrap intervals excluding zero. Across the two harness configurations and three backbones, this occurs in three of six comparisons for multi-entity collaboration and four of six for each other category. By backbone, the counts are nine of ten for Cosmos3-Nano, seven of ten for LingBot-Video and three of ten for Wan2.2. Their frozen means are 0.531, 0.561 and 0.408, respectively.

Adding grounding and output detector checks to the tool-free harness raises LingBot-Video's long-horizon score from 0.809 to 0.857 and its multi-entity score from 0.667 to 0.687, while its spatial score falls from 0.811 to 0.736. Appendix B.3 gives the backbone-level mean differences and defines the paired comparisons.

RBench sub-criteria. Table 20 expands the LingBot-Video tool-free diagnostic in Table 3. The judge uses five sub-criteria per category, with some criteria shared across categories; the table groups

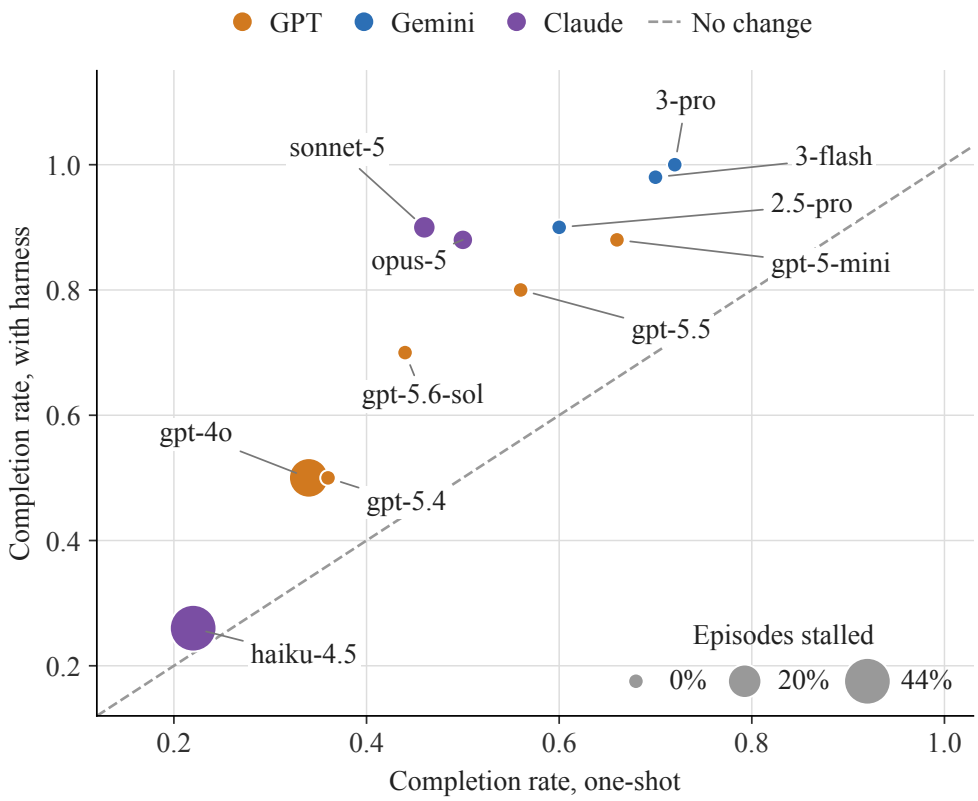


Figure 13: Planner completion on R^2bench under the one-shot and harness protocols on paired cases. All eleven harnessed point estimates exceed the one-shot estimates; nine paired confidence intervals in Table 18 are strictly above zero. Each point represents one planner. Point area indicates the fraction of harnessed episodes that exhaust the retry budget without a legal move. The point nearest the diagonal is claude-haiku-4.5, with a 44% stall rate in the reported run; two independent reruns also show high stall rates.

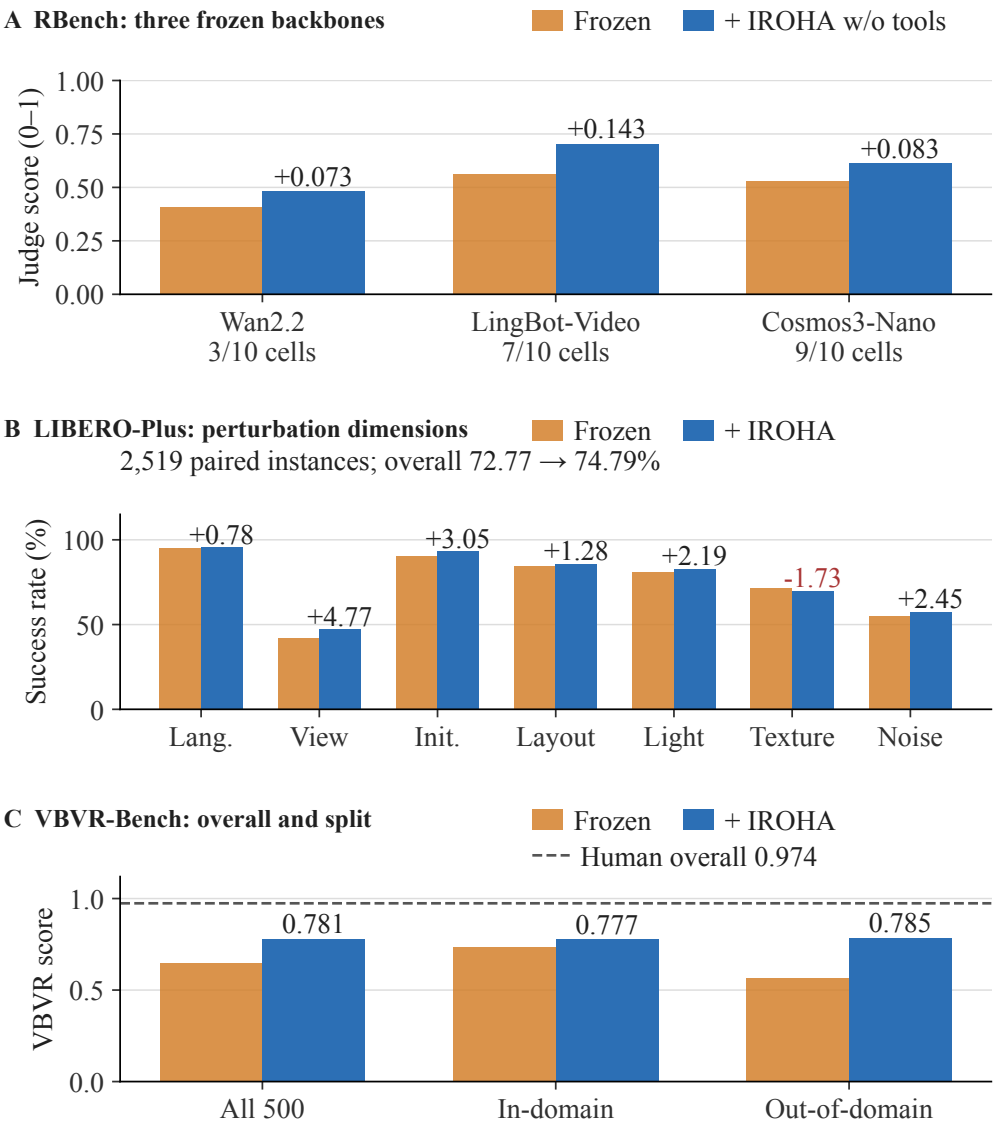
them by task accomplishment and rendering consistency. Each criterion mean pools valid case–criterion entries across the categories in which it is assessed. The pooled rows aggregate these entries within each group.

Over the same cases the tool-free harness improves 162 and worsens 49 (Wilcoxon $p = 1.6 \times 10^{-14}$).

Changing the judge. We rescore the same frozen-baseline and tool-free-harness videos with three judges from different model families (Table 21), changing only the evaluation judge.

All nine point gains are positive. gemini-2.5-pro gives the smallest gains on all three backbones. LingBot-Video has the largest gain under gpt-5.5 ($+0.143$) and the smallest under gemini-2.5-pro ($+0.020$), so the backbone ordering depends on the judge. claude-sonnet-5's gain is close to gpt-5.5's on Wan2.2 ($+0.080$ versus $+0.073$) and about half as large on LingBot-Video. Possible shared verifier–judge preferences are discussed in Appendix A.

LIBERO-Plus candidate scoring. For LIBERO-Plus, the planner, generator, and verifier operate per control block. The planner repairs the perturbed instruction into an imperative naming the object and the goal, retaining rewrites of three to forty words. The frozen policy samples three candidate trajectories per control block, and two signals score them. Frames from the candidate trajectories are tiled into a single image, and the MLLM rates each trajectory from 0 to 10 on task progress and motion stability. The detector identifies the object and its target from the instruction and measures



RBench: CIs excluding zero, across 5 categories × 2 harness configurations.

Figure 14: Results on three benchmarks, each evaluated with its own scorer. Top: RBench judge scores (0–1) for three frozen backbones and the tool-free harness. Counts of intervals excluding zero combine the tool-free and full configurations, with ten category-level comparisons per backbone. Middle: LIBERO-Plus success rates for the frozen policy and full harness by perturbation dimension. Bottom: VBVR-Bench scores for the frozen backbone and full harness, overall and by split, against the human score. RBench AVG scores and LIBERO-Plus success rates are in Tables 9 and 5, respectively. Panel (c) shows the published baseline; the paired comparison in the text uses our matched baseline.

Table 19: Summary of the 27 task types scoring below 0.70, grouped by observed tool use and score change. Frozen is the same backbone without the harness. *Routed* is the fraction of that type's episodes on which a tool ran.

	Task type	frozen	+IROHA	routed
no matched tool	grid highest cost	0.448	0.360	0%
	key door matching	0.516	0.466	0%
	high density liquid	0.475	0.475	0%
	animal matching	0.433	0.490	0%
	pigment colour mixing	0.495	0.495	0%
	symbol insert	0.509	0.509	0%
	clock	0.514	0.510	0%
	shape outline then move	0.636	0.525	0%
	separate object spinning	0.573	0.557	0%
	separate objects no spin	0.679	0.562	0%
	LEGO construction assembly	0.577	0.577	0%
	construct concentric ring	0.629	0.629	0%
	color addition	0.643	0.635	0%
	symbol deletion	0.640	0.640	0%
	multiple keys for one door	0.512	0.654	0%
	domino chain branch path prediction	0.736	0.659	0%
	grid shortest path	0.668	0.668	0%
	object subtraction	0.690	0.690	0%
small gains or declines	bookshelf	**0.580**	0.300	100%
	domino chain gap analysis	**0.669**	0.489	40%
	ball bounces given time	0.436	0.440	40%
	2d object rotation	0.499	0.520	100%
	draw next sized shape	0.530	0.534	100%
larger gains	mark tangent point of circles	0.190	**0.402**	100%
	gravity physics	0.346	**0.484**	100%
	highlight horizontal lines	0.438	**0.682**	100%
	symbol substitute	0.400	**0.693**	100%

Table 20: RBench criterion-level diagnostics, for frozen LingBot-Video and the tool-free harness, without input grounding or output detector checks, scored in one judge run.

	Criterion	frozen	+ harness
Task accomplishment	event completion ratio	0.315	**0.713**
	visual reasoning accuracy	0.308	**0.600**
	spatial relation accuracy	0.300	**0.530**
	task completion	0.163	**0.395**
	action coordination	0.053	**0.270**
	manipulation feasibility	0.338	**0.510**
	action execution	0.203	**0.260**
Rendering consistency	entity consistency	0.345	**0.488**
	anomaly check	0.218	**0.260**
	object consistency	**0.405**	0.385
	manipulator consistency	**0.420**	0.370
	Task accomplishment, pooled	0.225	**0.418**
	Rendering consistency, pooled	0.343	0.348

the reduction in the distance between the object and its target:

$$\Delta d_i = \|\mathbf{o}_0 - \mathbf{g}_{i,T}\|_2 - \|\mathbf{o}_{i,T} - \mathbf{g}_{i,T}\|_2, \qquad s_i^{\text{det}} = p_i + 2\Delta d_i.$$

Here $\mathbf{o}_0$ and $\mathbf{o}_{i,T}$ are the object's bounding-box centers in the shared initial frame and candidate i's final frame; $\mathbf{g}_{i,T}$ is the target's final-frame center, used as the reference for both distances. Coordinates are normalized by the agent-view image dimensions. The presence term p_i is the fraction of the two objects detected in the final frame, and Δd_i is set to zero when a required detection is unavailable. The scores are normalized and combined with weights of 0.8 for the MLLM and 0.2 for the detector. A candidate replaces the policy's default only when its score exceeds the default by 0.05 on the

Table 21: RBench comparison on the same frozen-baseline and tool-free-harness videos, over thirty cells of 50 videos. Δ_{raw} retains each judge's paired harness gain in the corresponding score.

	gpt-5.5		gemini-2.5-pro		claude-sonnet-5	
Public backbone	frozen	Δ_{raw}	frozen	Δ_{raw}	frozen	Δ_{raw}
Wan2.2_A14B	0.408	+0.073	0.255	+0.040	0.333	+0.080
LingBot-Video	0.561	+0.143	0.055	+0.020	0.108	+0.075
Cosmos3-Nano	0.531	+0.083	0.155	+0.033	0.208	+0.050

normalized score scale. When the instruction names fewer than two objects, the detector abstains and selection uses only the MLLM score.

Trace-level uncertainty. The instance-level McNemar test ($p = 0.021$; Table 22) treats the 2,519 rows as independent, although multiple perturbations share an underlying episode. We therefore keep the instance pairing and resample 413 trace clusters: 393 singletons and 20 larger clusters of 25–143 instances. For each of 20k draws, we sample 413 clusters with replacement and compute the difference over their pooled instances (Table 22).

Table 22: LIBERO-Plus uncertainty under instance-level and trace-level analysis. Both analyses use the same paired outcomes and report the same mean difference.

Unit	**Units**	**Mean**	95% **interval or** p
Instance (McNemar)	2519	+2.02	259/208, $p = 0.021$
Trace	413	+2.02	[+0.43, +4.45]

The random-selection comparisons use 367 paired trace clusters: the random-frozen interval is $[+1.29, +4.59]$ and the harness-random interval is $[+1.38, +3.31]$ percentage points. On the 1,737 common cases, mean success rates are 74.67% for frozen, 76.34% for random selection and 77.37% for the harness. Random selection samples one of three candidate trajectories uniformly at each control block, with a separate random generator for each task–episode pair seeded by $(1, \text{task_id}, \text{episode_id})$. The subset contains 347 singleton clusters and 20 larger clusters; the main analysis includes 46 additional singleton clusters whose cases lack random-selection results, giving 413 clusters in total.

L Structural Preservation and Candidate Selection

We evaluate structural preservation, MLLM-based selection and oracle selection on 500 samples. The restoration masks below are computed from pixel differences, independently of planner- or tool-supplied regions.

Restoration metric and masks. For input frame I_0 and candidate frame I_t, define

$$d_t(x, y) = \max_{c \in \{R,G,B\}} |I_t(x, y, c) - I_0(x, y, c)|, \qquad M_t(x, y) = \mathbf{1}[d_t(x, y) > \tau]. \tag{2}$$

Pixel values are on the 0–255 scale; using BGR instead of RGB leaves this maximum unchanged. Apply a 5×5 morphological closing to M_t, retain 8-connected components with area at least 120 pixels, then dilate with a 3×3 kernel to obtain $\widetilde{M}_t$. The restored frame is $\widetilde{I}_t = \widetilde{M}_t I_t + (1 - \widetilde{M}_t) I_0$. This operation is applied to every frame. The values $5, 10, 20, 25, 40, 60$ in Table 23 are thresholds τ in maximum-channel absolute pixel difference. These global-threshold variants are reported separately from the structural-check row.

The structural check counts salient regions across the full final frame: external contours with HSV saturation $S > 50$ and area $100 < a < 0.3HW$, where H and W are the image dimensions. The preservation score is the final-frame count divided by the input-frame count, clipped to $[0, 1]$. Starting from the committed answer, the check considers restored variants with $\tau = 25$ and 40, constructed using the masks above. A candidate replaces the current choice only when its preservation score exceeds the current score by more than 10^{-6}. If neither variant passes this check, the committed answer is retained unchanged.

The candidates include the committed answer, restored variants and the frozen backbone's rollout. Restored variants are computed from existing frames, and the backbone rollout is generated before this comparison. The selection analysis therefore requires no additional video generation.

Table 23: Choosing among configurations of the same episode, every row scored on all 500 samples. Δ is paired against the committed answer, with a 95% bootstrap interval over samples (8k resamples). The oracle rows use the benchmark scores to select the best candidate and give upper bounds for selection from this candidate set.

	Score	Δ	95% **CI**
Committed answer	0.776	—	
Restore everywhere, by edit threshold			
5	0.775	-0.001	$[-0.009, +0.008]$
10	0.778	$+0.002$	$[-0.007, +0.011]$
20	0.780	$+0.005$	$[-0.004, +0.014]$
25	0.773	-0.003	$[-0.013, +0.007]$
40	0.770	-0.006	$[-0.018, +0.006]$
60	0.764	-0.012	$[-0.025, +0.001]$
Frozen backbone rollout	0.650	-0.126	$[-0.151, -0.101]$
MLLM verifier picks per sample	0.760	-0.016	$[-0.029, -0.003]$
MLLM verifier picks per task type	0.759	-0.017	$[-0.031, -0.003]$
Structural check	**0.781**	$+0.005$	$[+0.001, +0.012]$
Best available, per task type	0.805	$+0.029$	
Best available, per sample	0.816	$+0.040$	

The structural check improves the score by 0.005, with an interval excluding zero. The six global restoration thresholds yield intervals including zero (Table 23); coarse restoration can erase intended edits. MLLM selection scores 0.016 below the committed answer.

In observed examples, candidates share the answer annotation but differ in other scene regions. The verifier assigns scores that agree to three decimal places despite benchmark-score differences of approximately half a point. Side-by-side presentation and position swaps also produce similar verifier scores on the examined families.

Oracle selection reaches 0.805 per task type and 0.816 per sample, compared with 0.650 for the frozen baseline. The reported 0.781 system captures approximately 85% and 79% of those respective gains over the baseline. The remaining score differences are 0.024 and 0.035 within the evaluated candidate set.

Two observed failure modes limit restoration. In the family with the largest gap to oracle selection, three of five instances lose scene structure and score zero; restoring the scene improves their outputs and scores. Other episodes preserve the scene but blur the tool's annotation during generation until the scorer cannot read it. Restoring input-frame pixels cannot reconstruct an annotation absent from the input. Detecting annotation loss can identify these failures, but improving them requires a suitable alternative candidate.

M BUDGETED COMPARISON ON R^2BENCH

We measure tokens, latency and solve rate on R^2bench for four planners under one-shot and legality-gated execution. The latter requests one move per call and retries illegal proposals. Figure 15 converts the measured costs and completion rates to expected solved cases per $100 of tokens. Table 24 also projects solved cases per hour of planner/API wall time. Both protocols use the same case list within each measured planner comparison.

The legality-gated protocol improves completion for all four planners: $+0.17$ for gpt-5-mini, $+0.30$ for gpt-5.2, $+0.13$ for gpt-5.4 and $+0.21$ for gpt-5.5. Improved and worsened paired cases number $10/5$, $12/3$, $8/4$ and $9/3$, respectively. Each planner covers 29–30 cases, with pairing within planners. Planner token cost per case increases approximately seven- to eighteen-fold, excluding video generation. The loop completes a larger fraction of attempted cases while solving fewer cases under the evaluated fixed token and time budgets. Table 24 reports each planner's one-shot and loop results together.

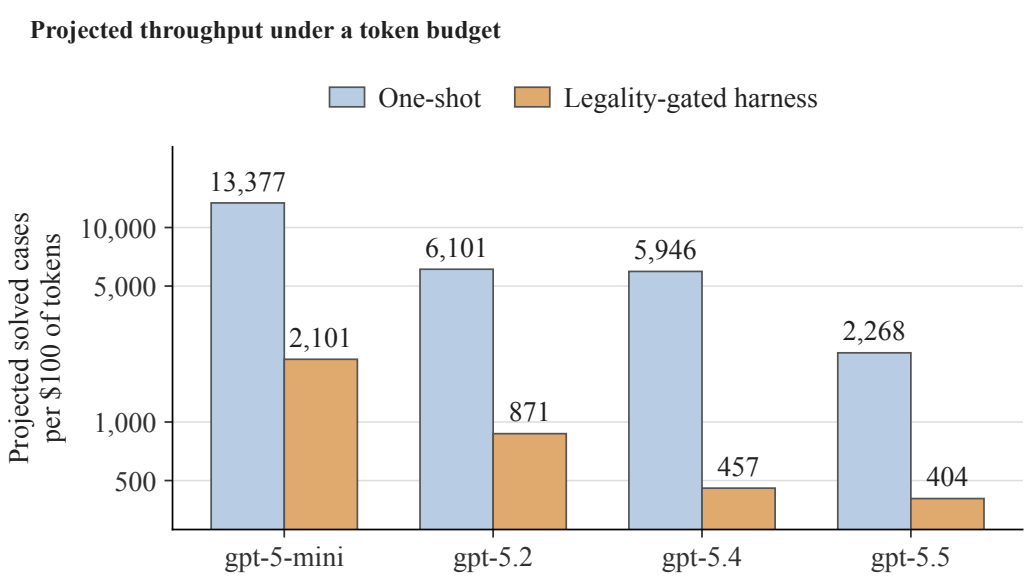


Figure 15: Projected solved R^2bench cases per $100 in planner tokens, computed from the completion rates and per-case token costs in Table 24. Costs include planner retries at the stated list prices and exclude video generation.

Table 24: Budgeted comparison on R^2bench. The measured GPT rows use 29, 30, 30 and 29 paired cases for gpt-5-mini, gpt-5.2, gpt-5.4 and gpt-5.5, respectively. Every measured column uses cases completed under both modes, so the solve rates, token counts and costs describe the same instances. Costs use published list prices per million tokens (gpt-5-mini $0.25/2.00$, gpt-5.2 $0.60/4.00$, gpt-5.4 $1.25/10.00$, gpt-5.5 $2.00/16.00$ for input/output) applied to token counts, which include reasoning tokens where the API bills them. *per $100* and *per planner h* project expected solved cases under the stated token and time budgets, using the original unrounded measurements; displayed costs and latencies are rounded. Time is wall-clock seconds per case for the planner protocol, excluding video generation.

Planner	Mode	Solved	Tok. in/out	$/case	Time/case (s)	per $100	per planner h
gpt-5-mini	one-shot	0.655	1.2k / 2.3k	0.005	36	13,377	66.0
	loop	**0.828**	14.0k / 17.9k	0.039	506	2,101	5.9
gpt-5.2	one-shot	0.500	1.5k / 1.8k	0.008	49	6,101	36.4
	loop	**0.800**	20.4k / 19.9k	0.092	639	871	4.5
gpt-5.4	one-shot	0.367	4.0k / 0.1k	0.006	8	5,946	158.1
	loop	**0.500**	74.8k / 1.6k	0.109	171	457	10.5
gpt-5.5	one-shot	0.655	5.3k / 1.1k	0.029	38	2,268	62.6
	loop	**0.862**	46.3k / 7.6k	0.214	255	404	12.2

N Relation to Test-Time Optimization of the Generator (VLM-as-Teacher)

VLM-as-Teacher (Cheng et al., 2026) uses a vision-language judge to evaluate goal and process constraints and guide test-time optimization of a video generator. Table 1 reports its published VBVR-Bench score alongside ours.

Use of tool outputs. VLM-as-Teacher converts judge feedback into a differentiable reward and optimizes a per-instance LoRA adapter. In our VBVR-Bench configuration, executable tools compute task-specific constraints, using methods such as A^* (Hart et al., 1968), closed-form geometry and periodicity analysis, and provide candidate clips or single-image guidance to the frozen generator.

Parameter updates. VLM-as-Teacher performs approximately sixteen online optimization steps per sample. Our harness adds no training or test-time parameter updates. Its inference-time state includes the plan, tool selection and generated candidates.

Verification decisions. Both methods use MLLM evaluations of task requirements. VLM-as-Teacher uses them to optimize the adapter. The VBVR-Bench harness uses acceptance and rejection decisions to retain a rendered segment or request a revision of its analysis or generation setup.

The published comparison uses a step-distilled VBVR-Wan2.2-5B backbone for VLM-as-Teacher and A14B weights for our main experiment, with different adaptation protocols. Table 1 reports these results under their respective settings. Combining per-instance adaptation with executable tool guidance remains an untested extension.

O Per-Row Confidence Intervals and Significance

Table 25 reports the paired comparison with our frozen-backbone run; the text below states how its intervals are computed. Stochastic inputs include the generator seed and the verifier's $N{=}3$ sampling, so we report the reference run as the point estimate with a sample-level bootstrap CI (20k resamples): IROHA scores 0.781 $[0.759, 0.802]$; the paired gain is $+0.131$ $[+0.106, +0.157]$, with paired t and Wilcoxon tests both $p < 10^{-20}$. The 173 samples with tied scores average 0.787, above the suite mean. On the 327 samples with different scores, IROHA averages 0.777 against the backbone's 0.577; the ratio of improved to worsened cases is 2.6 to 1. Because the 500 samples comprise 100 task types with 5 correlated instances each, and the five instances of a type share a generator, we also report a task-type *clustered* bootstrap (100 clusters, 20k resamples): IROHA 0.781 $[0.747, 0.814]$, paired gain $+0.131$ $[+0.086, +0.178]$; the conclusion is unchanged under clustering. We report 95% confidence intervals computed by a 20k-resample bootstrap over per-sample scores from the VBVR-Bench rule-based evaluator, together with paired tests for each Δ over the VBVR-Wan2.2 baseline. Pairing is on the sample level: the benchmark samples are the random draw, and both runs score the same samples, so paired tests account for two configurations evaluated on the same benchmark draw.

Table 25: Statistical analysis of IROHA against the frozen-backbone baseline (paired on the samples both runs scored). t is the paired t-test, W-p the Wilcoxon signed-rank p-value, d the paired Cohen's d. The comparison uses 500 samples scored by both runs. The IROHA gain is significant by every test.

Comparison	Δ mean [95% CI]	t (p)	W-p	d	win/lose/tie
IROHA vs. baseline	$+0.131$ $[+0.106, +0.157]$	10.05 (9.5×10^{-22})	5.3×10^{-21}	0.45	235/92/173

Uncertainty scope. These intervals quantify variation across benchmark cases for the recorded reference runs. They do not quantify variation across independent end-to-end repetitions of the complete system.

P Training Requirements and Inference Cost

Table 26 summarizes backbone training and test-time parameter updates for the compared methods.

IROHA adds no backbone fine-tuning or per-sample parameter update. Inference includes planner, verifier and generator calls. The separate planner study in Appendix M reports paired completion, cost and latency measurements. Table 13 evaluates the verifier's score contribution on VBVR-Bench; these studies use different tasks and cost units.

What the cost measurements include. Appendix M measures planner/API wall time and token costs; its per-hour values omit video generation, detector execution and other pipeline stages. The RBench demonstration separately records 887.1 seconds for three generation calls and 145.8 seconds for six planning/verifier calls (Appendix F.2); these are generation and planning/verifier time totals for that trace. Hosted planner and verifier APIs do not expose their execution graphs, batching or hardware, so token counts and latency cannot determine their FLOPs.

Table 26: Backbone training and test-time parameter updates for the compared configurations. "Backbone train" describes prior backbone training; "Test-time updates" describes per-sample parameter optimization. * Both IROHA on VBVR-Bench and VLM-as-Teacher use backbones fine-tuned on the VBVR Dataset. The NEWTON-tools row refers to our RBench adaptation; it uses a gpt-5.4 planner and adds no training.

Method	Backbone train	Test-time updates
VBVR-Wan2.2 (Video-Reason Team, 2026)	1M-clip FT	none
VLM-as-Teacher (Cheng et al., 2026)	inherited*	∼16 LoRA steps
NEWTON tools (RBench adaptation) (Feng et al., 2026)	pretrained	none
IROHA (VBVR-Bench)	inherited*	none
IROHA (RBench full)	pretrained	none

Q Largest Per-Task Improvements

Table 27 reports gains on spatial selection and construction tasks. Each row averages the five instances of one VBVR task type.

Table 27: Largest per-task improvements of IROHA over the frozen Wan2.2 backbone, on representative VBVR task types; the harness identifies the failure mode and dispatches to a tool, and the frozen generator produces the episode from that guidance. Each score is the mean over the five instances of the task type.

Task type	Frozen backbone	IROHA	Δ
select leftmost shape	0.16	0.99	+0.83
construction stack	0.18	0.98	+0.80
mark second-largest shape	0.25	0.99	+0.73
select longest polygon side	0.23	0.95	+0.72
identify nearest-to-square rectangle	0.29	0.97	+0.68

R Real-world style Demonstrations and Execution Records

This appendix describes R2bench and the recorded execution traces in synthetically generated scenes.

R.1 Construction of R^2bench

The five scenario templates specify symbolic states, legal moves and goal conditions for the Tower of Hanoi, stacking, sorting, rearrangement and assembly. A solver-verified sampler draws 20 solvable initial states per scenario, and a text-to-image model renders each state as a real-world style scene (Figure 16). The symbolic state provides the reference for initial-state perception, while the transition and goal rules score move legality and episode completion. Table 8 reports these measures under the one-shot and harness protocols.

R.2 Robot arm completing the Tower of Hanoi

Given a real-world-style initial frame showing a robot arm and a reasoning task such as an unfinished Tower of Hanoi puzzle, the agent is tasked with generating a video that completes the puzzle. Consider the Tower of Hanoi challenge as an example: a legal move cannot place a larger disk on a smaller disk. The plan length increases with disk count, and each rendered move is checked before generating the next. We instantiate the planner, task-specific tools and verifier for tasks like this.

Multistep execution. Table 28 summarizes multistep execution with state checks and backtracking, including generated attempts, verifier outcomes and retained segments. Figure 7 qualitatively compares frozen-backbone, NEWTON and IROHA outputs for one initial scene.

Showcase protocol. Figure 7 uses the four-disk scene, with the stack initially on the middle peg and the goal on the right. NEWTON here uses the NEWTON-tools adaptation in Table 2, with the

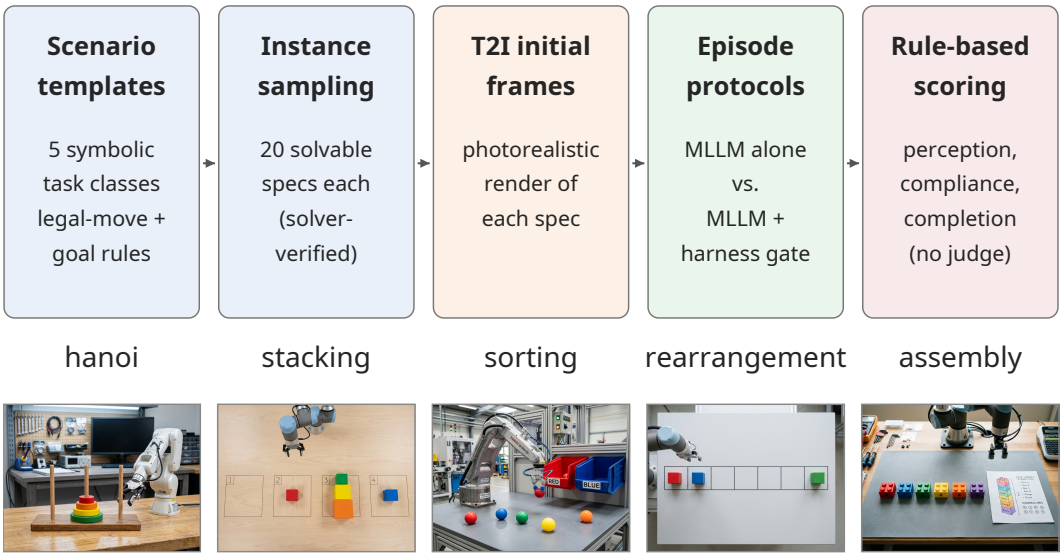


Figure 16: Construction of R^2bench (Appendix R.1). Five symbolic scenario templates define legal-move and goal rules; a solver-verified sampler draws 20 solvable instances per scenario; a text-to-image model renders each instance into a real-world style initial frame (bottom row shows one generated initial frame per scenario); episodes run under two protocols (planner alone vs. planner inside the harness); and scoring is entirely rule-based.

same gpt-5.4 planner and released tools. The frozen Wan2.2-I2V-A14B clip contains 49 frames; the NEWTON and IROHA videos contain 637 and 721 frames, respectively. All three are 640×480 at 16 frames per second. The NEWTON video includes all attempts, and the IROHA video contains 15 retained segments. Every panel uses the same spatial crop to show the pegs and gripper. The IROHA run is recorded as reaching the goal by the state checks. The verifier checks whether the post-frame is reachable by a single legal move and whether that move matches the intended action.

Table 28: Execution results for episodes in R^2bench. The planner and verifier both use gpt-5.4, and the frozen generator is Wan2.2-A14B. A *committed segment* is one rendered move that survived to the final clip; accepts exceed committed segments because a backtrack discards already-accepted segments and redoes them; *raw rollouts* counts every clip the frozen generator produced, including sampling-stage discards and verifier rejections. The outcome counts partition all 1,575 raw rollouts.

Quantity	**Value**
Episodes attempted	100
Episodes reaching the goal state	90
Raw generator rollouts	1575
Discarded during sampling, before main verification	725
Verifier accepts	715
Verifier rejects: not a single-move successor	95
Verifier rejects: legal but unintended move	40
Committed segments (subset of accepts)	580
Accepted segments later removed by backtracking	135

R.3 ADDITIONAL LIBERO-PLUS QUALITATIVE RESULTS

These examples pair frozen-policy failures with successful harness runs from the evaluation in Table 5. The frozen episodes reach the 796-frame recording limit without success; the harness episodes terminate with simulator success. Frames show recorded simulator observations. Each original camera view is 128×128. Frame indices refer to each episode's own timeline.

Figure 6 shows instance 1136. Its successful episode contains 263 frames, and the harness chooses a non-default candidate in 10 of 17 control blocks. At block 2, the raw MLLM scores are 0.1 for default candidate 0 and 0.9 for selected candidate 1. These are candidate-selection scores, while episode success is determined by the simulator. The official task definition maps instance 1136 to placing the alphabet soup and tomato sauce in the basket. Both targets are labelled in Figure 6.

Figures 17 and 18 show the other four selected cases. The non-default selection counts are $12/20$ for instance 1835, $6/24$ for 680, and $7/17$ each for 1664 and 833. These counts describe the recorded control decisions. Wrist views are used where they show the objects more clearly than the third-person camera.

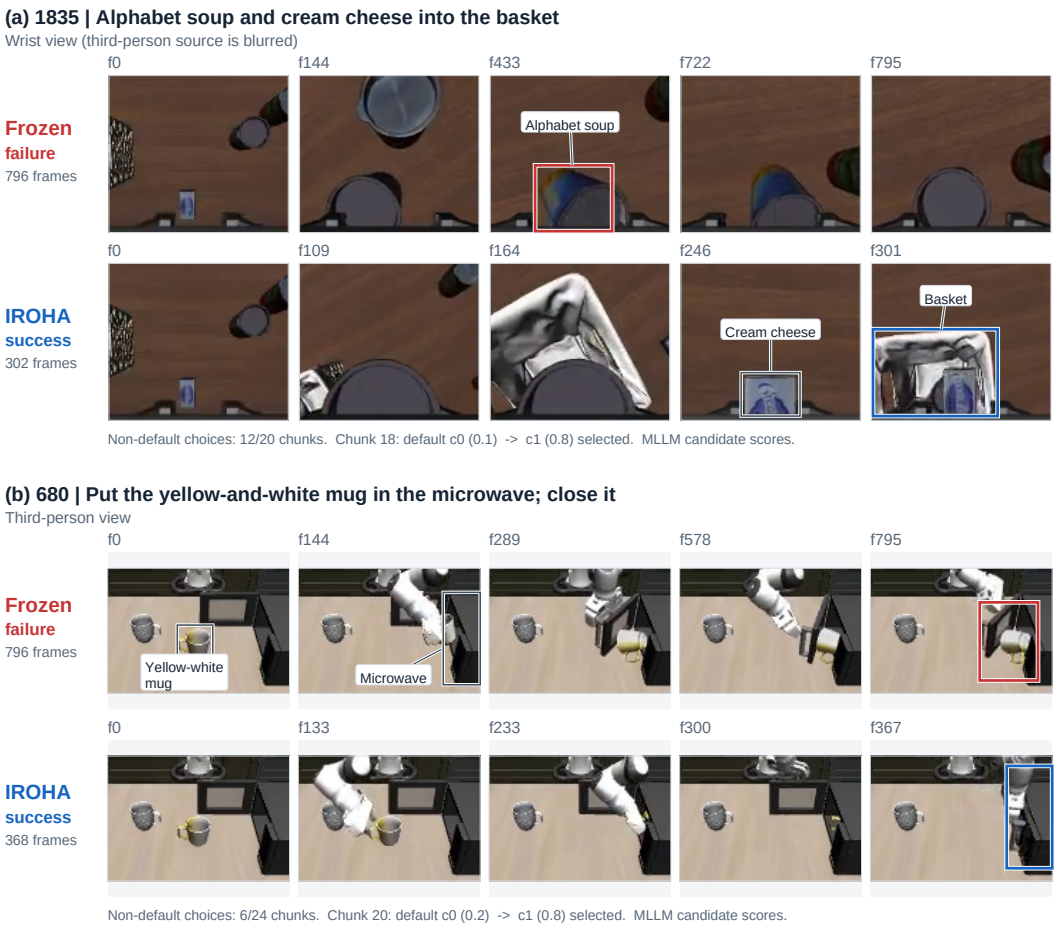


Figure 17: Additional LIBERO-Plus results: (a) instance 1835, put alphabet soup and cream cheese in the basket; (b) instance 680, put the mug in the microwave and close the door. Labels locate the task objects and their target containers. **Red: the frozen episode leaves a target outside the basket in (a), and the mug outside the open microwave in (b). Blue: the successful episodes show the target placement in the basket and the closed microwave, respectively.** Scores beneath the panels are logged MLLM candidate scores at selected control blocks.

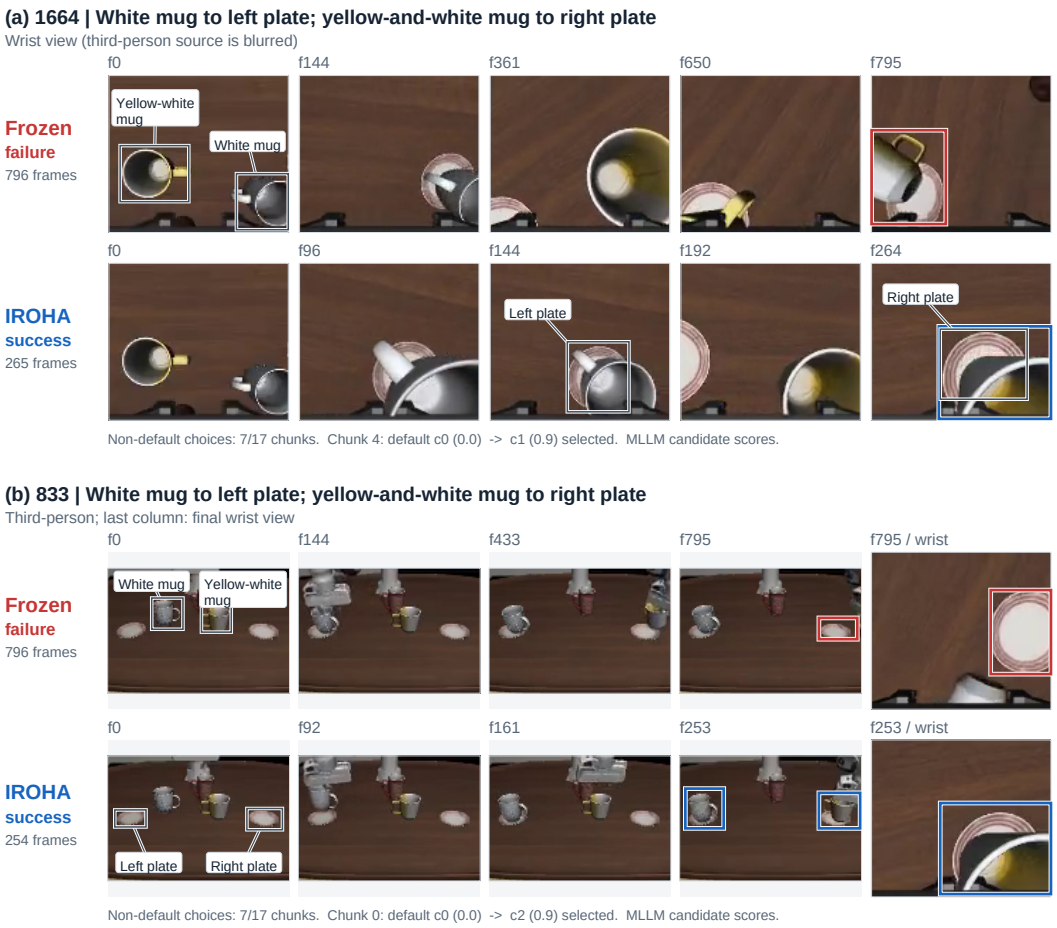


Figure 18: Two-mug placement runs, instances 1664 and 833: place the white mug on the left plate and the yellow-and-white mug on the right plate. Labels identify both mugs and their target plates. **Red: the frozen episodes end with an incomplete mug placement. Blue: the successful episodes show the mugs at their target plates.** The panels use the recorded camera views and each episode's own frame indices.