WORLDGUIDE: GOAL-DIRECTED VIDEO WORLD MODEL FOR PROCEDURAL TASK EXECUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Video generators and video-based world models can synthesize plausible visual trajectories, but long-horizon procedural tasks require generation to adapt to what has actually been produced. A model must determine the next action from its generated state, execute that action, and recognize when the task is complete. Open-loop generation cannot adapt to execution outcomes, while existing closed-loop systems often rely on pretrained executors or indirect verification. This leaves a gap between deciding an action and successfully realizing it. We formulate procedural video generation as *closed-loop task execution in visual world space* and introduce **WorldGuide**. Given only an initial image and a task goal, WorldGuide predicts an atomic action, generates its corresponding video clip, and uses the generated result to select the next action or terminate. The Planner and Executor are trained on the same step-level procedural demonstrations: the Planner learns to predict the next atomic action or task completion from visual progress, while the Executor is directly trained to realize the predicted actions. Hierarchical visual memory maintains state across long-horizon execution with bounded history token cost. Due to the lack of step-level action-video supervision for joint planner-executor training, we introduce **WorldGuide Bench**: approximately 59K step-annotated videos across 245 tasks and 27 procedural categories. WorldGuide achieves a **33.33%** Task Success on WorldGuide Bench, compared with **29.90%** for the strong recent video model MiniMax-H3, even though MiniMax-H3 receives reference action plans, and achieves **47.69%** on Video-CraftBench compared with **32.73%** for MiniMax-H3 under goal-only conditioning. These results demonstrate the importance of coupling planning with learned execution for goal-directed procedural video generation.

1 INTRODUCTION

Video-based world models predict how visual states evolve under actions, goals, or other control signals. Direct prompt-to-video models (OpenAI, 2025b; Google DeepMind, 2025; Wu et al., 2025; Kong et al., 2024; Wan et al., 2025; Yang et al., 2025) and autoregressive approaches (Yuan et al., 2025; Deng et al., 2024) can synthesize plausible visual trajectories, but visual plausibility does not ensure task completion. Open-loop generation fixes the prompt or action sequence before rollout, without observing progress, revising subsequent actions, or determining when the goal is reached. As a result, an incorrect or incomplete step cannot be corrected and generation may continue beyond task completion (Fig. 1a). This becomes particularly problematic for long-horizon procedures such as origami, assembly, and cooking, where errors in one step can affect everything that follows.

Closed-loop approaches introduce feedback, but a gap remains between planning an action and reliably executing it. ORCA (He et al., 2026), CollabVR (Kim et al., 2026), and NovaPlan (Fu et al., 2026a) pair a pretrained vision-language planner with a frozen video executor. A correct plan can therefore still fail when the executor cannot realize the requested atomic action (Fig. 1b). SPIRAL (Yang et al., 2026) uses a separately trained critic to evaluate generated outcomes and provides its judgment, rather than directly supervising atomic action execution. BERNINI (Team et al., 2026) plans in latent semantic space before rendering, but does not predict successive procedural actions from generated progress or determine task completion. Goal-directed generation therefore requires a planner grounded in the evolving visual state and an executor trained to realize the actions it predicts. This motivates our formulation of *closed-loop task execution in visual world space*.

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

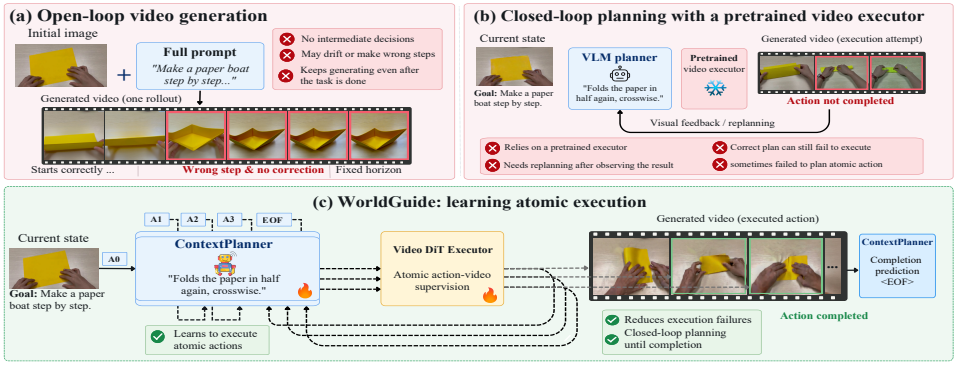

Figure 1: **From open-loop synthesis to learned procedural execution.** (a) Open-loop generation follows a fixed prompt with no intermediate decisions, so it can drift into a wrong step and keep generating after the task is done. (b) A planner can inspect outcomes and replan, but with a frozen pretrained executor, a correct instruction can still fail to execute. (c) WorldGuide trains the ContextPlanner and Executor on the same demonstrations, the Executor conditioned on the planner's action embeddings, so generated outcomes guide the next action and the decision to stop.

This gap raises five challenges that a closed-loop procedural video model must address. First, if the executor is frozen or only indirectly supervised through a reward model, it cannot be trusted to correctly realize an arbitrary planner-proposed action; *the executor itself needs to be trained on the atomic actions it is asked to perform*. Second, a planner conditioned only on its previously requested actions cannot determine whether those actions were successfully executed, partially completed, or resulted in visual drift. Therefore, *planning must be grounded in the generated visual state, not action history alone*. Third, *task completion must also be predicted from visual progress rather than determined by a fixed generation length or external retry budget.* Fourth, because every generated clip affects subsequent decisions, so *the long-horizon execution requires persistent visual context without unbounded memory growth*. Fifth, existing instructional-video datasets do not provide demonstrations that pair atomic actions with their visual consequences and explicit completion signals. Therefore, *closed-loop procedural training requires dedicated step-level action–video supervision*.

To address these challenges, we introduce **WorldGuide**, a video world model for closed-loop procedural task execution (Fig. 1c). First, we construct **WorldGuide Bench** (§ 3.5), comprising approximately 59K procedural videos across 245 tasks and 27 categories, segmented into temporally grounded atomic action clips with explicit completion signals. Second, we train the **ContextPlanner** (§ 3.2) on this data to predict the next atomic action or a completion token from the generated visual state and action history (§ 3.1). Third, we train the **Executor** (§ 3.3) on the same demonstrations, conditioned on the frozen ContextPlanner's action embeddings, so it learns to render the same atomic actions the planner is trained to propose. Fourth, to support long-horizon state continuity, the Executor maintains a **hierarchical, compressed visual memory** (§ 3.4), representing older history at progressively coarser spatial resolution to bound the token cost as the rollout grows. Our experiments validate these design choices. Closed-loop execution improves Task Success by **21.62%** over the open-loop variant (§ 4.2); visual feedback improves Task Success by **18.61%** (§ 4.3) and Plan Success by **5.34%** while reducing Repeat/Skip by **4.46%** (§ 4.4); while memory provides gains of **5.77%** on WorldGuide Bench and **17.91%** on Video-CraftBench (§ 4.5–§ 4.6). Our proposed approach achieves the highest Aesthetic Quality and Overall Consistency on WorldGuide Bench (§ 4.7).

2 Related Work

2.1 Video Generation and World Models

Modern video generators (Yang et al., 2025; Wan et al., 2025; Wu et al., 2025) synthesize strong visual trajectories, but operate open-loop, without selecting the next semantic action from generated progress or recognizing task completion. Long-context methods such as FramePack (Zhang et al., 2026a) and interactive world generators (Mao et al., 2025b;a) improve temporal continuity via compressed visual history, but memory alone does not determine how to advance a procedure. World models extend visual prediction toward decision making: VideoWorld 2 (Ren et al., 2026) learns policies

through autoregressive latent dynamics, hierarchical world models (Zhang et al., 2026b) plan over latent subgoals, vision-language-action models (Zitkovich et al., 2023; Kim et al., 2024) predict robot controls, and Video Language Planning (Du et al., 2024) combines policies, value functions, and video dynamics through tree search. None of these approaches select actions from visual progress, execute them through directly supervised video generation, and decide completion within a single closed loop. WorldGuide addresses this by coupling visual memory with progress-conditioned action selection, atomic-action execution, and learned termination, all grounded in generated visual states.

2.2 Planner–Executor Video Generation

Planner–executor systems move video generation toward closed-loop interaction, but couple planning and execution differently. BERNINI (Team et al., 2026) plans latent visual semantics before rendering, without recurrent procedural decisions from generated progress. TempAct (Wang et al., 2026) jointly trains planning and execution but follows prescribed temporal spans without visual-state replanning. PhysAgent (Li et al., 2026b) refines physical programs via scene reconstruction and physics simulation before synthesis. ORCA (He et al., 2026) and CollabVR (Kim et al., 2026) add visual feedback through verification and retries, but keep pretrained executors untrained with their planners. SPIRAL (Yang et al., 2026) trains planning and generation jointly, but still relies on a separate critic and reinforcement learning for execution refinement. None learn the full procedural loop: selecting an action from generated progress, executing it under direct supervision, and deciding completion. WorldGuide instead learns next-action selection, atomic-action execution, and completion prediction from the same demonstrations, coupling planning with learned execution in a recurrent closed loop.

3 Method

3.1 Overview and Problem Formulation

Given an initial image x_0 and task goal g, **WorldGuide** performs a goal-directed video world-model through a recurrent planner-executor loop (Fig. 2). At step t, the ContextPlanner π_θ predicts a decision $z_t \in \mathcal{A} \cup \{\texttt{DONE}\}$ from the goal g, visual context x_t, and action history h_t, where $\mathcal{A}$ is the atomic-action space. If $z_t = \texttt{DONE}$, the rollout terminates; otherwise $a_t = z_t$, and the Executor f_ϕ generates clip v_t conditioned on a_t, the current state x_t, and visual memory M_t:

$$\begin{gathered} z_t \sim \pi_\theta(\cdot \mid g,\, x_t,\, h_t)\,, \qquad a_t = z_t \text{ if } z_t \neq \texttt{DONE}, \qquad v_t = f_\phi(a_t,\, x_t,\, M_t)\,, \\ x_{t+1} = \mathrm{obs}\big(v_{\max(0,t-K+1):t}\big)\,, \qquad h_{t+1} = \mathrm{tail}_K([h_t, a_t])\,, \qquad M_{t+1} = \mathrm{update}(M_t, v_t) \end{gathered} \tag{1}$$

with x_0 the input image and h_0 empty. Here $v_{i:j}$ denotes clips i through j, obs builds the planner's visual context from them, and x_t, h_t retain the latest $K{=}3$ steps (tail_K truncates the history), while update refreshes the Executor's memory. The rollout ends when the planner predicts DONE or the step budget is reached, and the generated clips are concatenated into the final video (Appendix D.3).

The ContextPlanner and Executor are initialized from Qwen2.5-VL-7B and HunyuanVideo-1.5. We train the ContextPlanner first, freeze it, and use it to embed action captions for Executor fine-tuning (Appendix A); at inference, action prediction and termination depend on the Executor's generated results. The Executor also adapts an existing hierarchical memory design (§ 3.4) for visual continuity.

3.2 ContextPlanner

The planner predicts the next atomic action a_t from the task goal g, visual context x_t, and action history $h_t = (a_i)_{i=\max(0,t-K)}^{t-1}$. For $t > 0$, $x_t = \mathrm{obs}(v_{\max(0,t-K):t-1})$ contains the corresponding generated clips. Thus, K controls both histories: $K = 1$ supplies one clip and its action, and $K = 3$ supplies the last three clips and actions. We use $K = 3$. These aligned histories are distinct from the Executor's compressed latent memory M_t, which the planner does not consume. The output is a single natural-language instruction or a completion token; the visual history provides evidence of recent progress and helps avoid repeated steps. We train it by supervised fine-tuning of Qwen2.5-VL-7B on demonstration histories and their next-action or completion targets, using token-level cross-entropy masked to response tokens. No architectural modification or auxiliary objective is used.

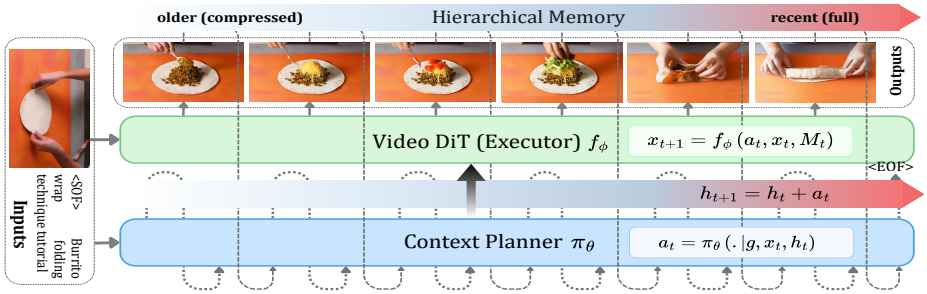

Figure 2: **Overview of WorldGuide.** Given the inputs and bounded clip–action history ($K = 3$), the ContextPlanner predicts the next atomic action a_t or a completion token. The Executor renders the action from the current visual state x_t and hierarchical memory M_t, which progressively compresses older latent frames. The generated clip updates the planner context and executor memory, and the loop continues until completion is predicted or the step budget is reached.

3.3 Executor

The Executor is a HunyuanVideo-1.5 (Wu et al., 2025) diffusion transformer conditioned on action embeddings from the frozen ContextPlanner and on visual context. During training, it receives the ground-truth action caption paired with the target clip; at inference, it receives the ContextPlanner's predicted action through the same embedding pathway.

Each training sample provides a target video latent $z \in \mathbb{R}^{C\times L\times H\times W}$, where L is the clip's latent-frame length, with text and vision conditioning features $c_{\text{text}}, c_{\text{vision}}$. The text condition c_{text} includes the ground-truth action embeddings from the frozen ContextPlanner. Under a flow-matching parameterization, the noisy latent at noise level σ is $\tilde{z} = (1-\sigma)z + \sigma\epsilon$ with $\epsilon \sim \mathcal{N}(0, I)$, and the model predicts the residual velocity $\hat{y}_\phi = f_\phi(\tilde{z}, \sigma, c)$ toward the target $(\epsilon - z)$:

$$\mathcal{L}_{\text{exec}} = \mathbb{E}_{z,\epsilon,\sigma}\left[\frac{\|m \odot (\hat{y}_\phi - (\epsilon - z))\|_2^2}{\max(\sum m, 1)}\right], \tag{2}$$

where m restricts supervision to valid regions. This loss updates only Executor parameters ϕ; ContextPlanner parameters θ remain frozen. Reference-image conditioning enters as a latent tensor $z_{\text{cond}} \in \mathbb{R}^{C\times L\times H\times W}$ stacked channel-wise with $\tilde{z}$, accompanied by a validity mask m_{cond}, so that $c = (z_{\text{cond}}, m_{\text{cond}}, c_{\text{text}}, c_{\text{vision}})$. Without memory, only the reference frame is populated and the remaining slots are masked out. With memory, history latents also populate the conditioning tensor and are embedded into context tokens (Sec. 3.4; Appendix C.2).

3.4 Hierarchical Memory Compression

We adopt YUME's spatial history-compression schedule, following Yume-1.5 and FramePack (Mao et al., 2025a; Zhang et al., 2026a). Recent latent frames receive finer spatial embeddings than older ones, bounding the Executor's history-token count.

Memory bank. Let $T \le 1365$ denote the capacity for non-anchor history latent frames. The compressor additionally handles the oldest history latent frame as an anchor, giving a total history length $F \le T + 1 \le 1366$:

$$M_t = \Pi_T\left(\left[z^{(0)} \,\|\, \cdots \,\|\, z^{(t-1)}\right]\right) \in \mathbb{R}^{C\times F\times H\times W} \tag{3}$$

Here, Π_T forms the bounded history tensor from preceding clips, and H, W are latent spatial dimensions. F counts history latent slots supplied to compression, including the anchor and temporal padding; the current reference is separate. These are

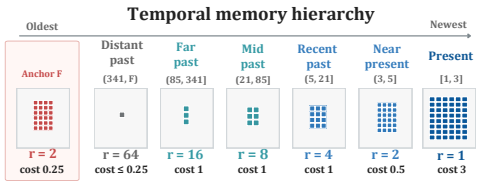

Figure 3: Longest-history branch ($343 \le F \le 1366$). Age 1 is newest; age F is the anchor. Rates r denote spatial reduction relative to base patch embedding, and costs are nominal latent-frame equivalents before spatial padding, with the separately embedded reference adding cost 1 at $r = 1$. Shorter branches appear in Appendix C.3.

Table 1: **Procedural video datasets and their supervision for planning and generation.** Sizes retain their original units: videos (vid.), segments (seg.), clips. Tasks/Dom.: tasks/domains; Step-Clip: temporally localized step annotations; Atomic Act.: single-action granularity; Coverage Target: annotation protocol targets all visible task-relevant steps, excluding background intervals, rather than a verified coverage rate; Task Goal: explicit task-level goal; Completion: explicit task-completion supervision. Gen., Eval., and Recog. denote generation, evaluation, and recognition.

Dataset	Size	Tasks/Dom.	Step-Clip	Atomic Act.	Coverage Target	Task Goal	Completion	Type	Public
EgoPlan-IT	50K QA	–/1	✗	✓	✗	✓	✗	Planning QA	✓
Ego4D Goal-Step	48K seg.	86/1	✓	✗	✗	✓	✗	Localization	✓
COIN	11.8K vid.	180/12	Fixed-vocab	Partial	✗	✓	✗	Localization	✓
CrossTask (primary)	2.75K vid.	18/4	✓	Partial	✗	✓	✗	Weak localization	✓
YouCook2	2K vid.	89/1	✓	✗	✓	✓	✗	Segmentation	✓
HT-Step	19.7K vid.	433/1	✓	Partial	✗	✓	✗	Grounding	✓
Assembly101	4.3K vid.	101/1	✓	✓	✓	✗	✗	Recognition	✓
CaptainCook4D	384 vid.	24/1	✓	Partial	✓	✓	✗	Error/localization	✓
SemComp-Data	1.27K	21/6	✗	✗	✗	✓	✗	Eval-only	Partial
ActVideoGen	118K clips	–/Multi	✓	Partial	Partial	✓	✗	Gen. training	✗
EgoForge / X-Ego	15K clips	–/Multi	✗	✓	✗	✓	✗	Gen. training + eval	✗
Ego-Exo4D Keystep	27.6K seg.	17/3	✓	Partial	✗	✓	✗	Recognition	✓
WorldGuide (ours)	**59K vid.**	**245/27**	✓	✓	✓	✓	✓	**Gen. training + eval**	**Planned**

latent-frame counts after VAE encoding, not video frames. The schedule capacity is an upper limit, not the length used at every step; Appendix C.1 specifies how the conditioning path limits F.

Multi-scale patch embedding. With DiT spatial patch size $p = 2$, E_r is a 3-D convolution with kernel and stride $(1, pr, pr)$ for $r \in \{1, 2, 4, 8, 16\}$. It maps C latent channels to d features. We initialize E_1 from the pretrained patch embedding and larger kernels by trilinear upsampling. The coarsest embedder is $E_{64} = E_{16} \circ \Phi$, where Φ is a channel-preserving convolution with kernel and stride $(1, 4, 4)$. Its effective spatial stride is $128 = 64p$. All embedders preserve temporal length.

Partition and token budget. For nonempty history, the selected branch splits it into segments S_j with spatial rates r_j, the anchor at rate r_a; Fig. 3 shows the longest branch. Ignoring spatial padding,

$$h_j = \mathrm{flatten}\big(E_{r_j}(M_t[:, S_j])\big) \in \mathbb{R}^{n_j \times d}, n_j \quad = |S_j| \frac{HW}{p^2 r_j^2}. \tag{4}$$

The anchor, chronological history segments, and current reference form

$$\mathcal{M} = [\, h_a \,;\, h_1 \,;\, \cdots \,;\, h_J \,;\, \mathrm{flatten}(E_1(z_{\mathrm{ref}}))\,], \tag{5}$$

where h_a embeds the oldest history latent frame. Across all supported branches,

$$\frac{p^2}{HW}|\mathcal{M}| = \frac{1}{r_a^2} + \sum_j \frac{|S_j|}{r_j^2} + 1 \le 8 < 9, \qquad T \le 1365. \tag{6}$$

One full-resolution latent-frame equivalent is HW/p^2 tokens. This bound measures token cost, including the anchor and reference, not a reduction of history to eight temporal latent frames; spatial padding changes the actual count (Appendix C.3).

3.5 Dataset for WorldGuide Training

Closed-loop procedural execution requires supervision for three connected decisions: which action follows observed progress, how that action changes the visual state, and when the task is complete. Existing instructional datasets provide narration, action labels, or localized step descriptions (Miech et al., 2019; Zhou et al., 2018; Tang et al., 2019; Zhukov et al., 2019; Damen et al., 2020), but need additional processing to support this training formulation. We construct **WorldGuide Bench** to combine ordered atomic-action clips, task goals, and explicit completion

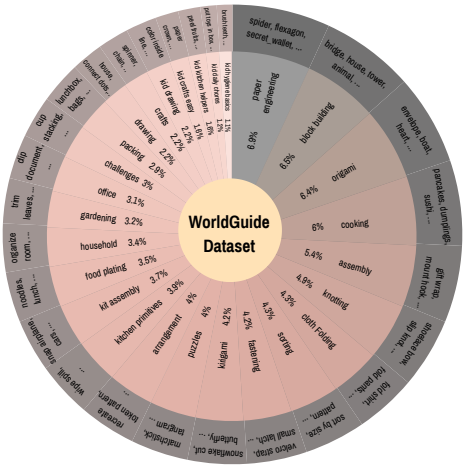

Figure 4: Task taxonomy of WorldGuide Bench. The inner ring shows procedural categories and their dataset shares; the outer ring lists representative tasks within each category.

Table 2: **WorldGuide Bench evaluation.** Unstarred baselines use reference actions; $*$ denotes methods with their own planning-execution loop. WorldGuide predicts actions from the task goal and generated history. **Aesth.**: Aesthetic Quality; **Imag.**: Imaging Quality; **Dynamic**: Dynamic Degree; **Smooth.**: Motion Smoothness; **Consist.**: Overall Consistency; **Align.**: Alignment; **Plan Acc.**: Plan Accuracy; **Task Succ.**: Task Success Rate. Higher is better except Repeat/Skip (Appendix E).

Model	Video Quality Check					Video Planning Check				
	Aesth.	Imag.	Dynamic	Smooth.	Consist.	Align.	Plan Acc.	Order	Repeat/Skip ↓	Task Succ.
Cosmos-Predict2.5-2B	38.77	62.11	82.52	96.32	17.46	63.75	38.05	84.08	3.98	11.94
Open-Sora 2.0-11B	33.42	44.91	60.19	95.06	17.42	62.43	31.27	79.56	6.53	8.54
HunyuanVideo-1.5-8.3B	42.43	65.57	88.84	96.00	18.91	62.28	43.53	80.00	3.00	12.00
Wan2.2-14B	41.46	58.87	16.02	93.67	17.52	67.33	4.37	21.65	2.06	2.58
CogVideoX-5B	38.76	54.99	64.08	96.58	18.09	68.30	34.33	75.74	3.96	9.90
Helios-14B	39.32	56.06	72.33	95.90	18.60	63.99	25.71	76.47	1.51	8.04
UniVideo-13B	45.75	61.92	48.00	97.09	18.05	65.63	10.77	52.00	4.00	4.00
FlashMotion-34B	41.69	67.26	90.29	95.46	17.93	62.92	24.10	77.39	2.51	5.53
MAGI-1-24B	39.14	63.08	2.30	98.31	17.00	69.33	17.70	61.63	2.33	3.49
RoboMaster-5B	32.60	17.68	4.37	97.09	6.66	37.13	4.29	5.50	0.50	4.00
SpMem-5B	35.79	47.69	96.73	98.94	20.72	60.92	33.74	78.95	2.94	10.53
Astra-1.3B	32.00	43.43	97.67	95.73	15.03	65.40	27.23	60.71	0.64	11.90
Yume-1.5-5B	40.61	65.72	87.86	96.90	17.39	60.79	19.72	74.26	2.39	2.97
SkyReels-V3-14B	42.64	58.78	14.56	**99.66**	17.03	72.54	11.23	48.52	1.01	3.90
MiniMax-H3-33B[1]	41.61	56.60	80.58	97.99	19.84	70.04	57.99	86.79	4.06	29.90
LTX-2.5-22B	43.85	57.92	85.92	99.29	15.63	70.10	16.14	49.76	1.46	7.32
HY-WorldPlay-8.3B	39.25	64.56	85.95	96.30	15.28	55.47	15.67	73.46	3.07	1.26
PhysAgent*	42.76	65.09	52.68	98.39	16.72	70.61	18.20	56.56	1.98	8.42
TempAct-1.3B*	44.68	**71.06**	**100.00**	97.81	15.23	65.09	9.11	29.68	0.50	3.98
Bernini-14B*	45.12	63.13	93.20	98.23	11.91	53.48	5.30	22.06	**0.00**	1.96
WorldGuide-8.3B (ours)	**49.36**	64.77	98.06	98.96	**31.13**	**73.85**	**62.85**	**92.42**	6.06	**33.33**

signals within each demonstration (Table 1). This shared supervision supports both ContextPlanner decisions and Executor training.

The dataset contains **58,679 videos** spanning **245 tasks** across **27 procedural categories** (Fig. 4). We use Gemini 2.5 Flash (Gemini Team, Google, 2025) to filter web instructional videos and annotate temporally grounded atomic actions and completion. We used total **980 videos** in test dataset. Construction details, annotation examples, and training-sample preparation are provided in Appendix B, and a human audit of the annotations in Appendix B.6.

4 Experiments

Open-loop video generation follows instructions and a duration chosen before rollout, without checking intermediate task progress or deciding when the goal has been reached. Planner–executor systems can introduce feedback, but some use pretrained components without procedural training, while others plan the action sequence before rendering. WorldGuide trains both the ContextPlanner and Executor on procedural demonstrations to select and carry out atomic actions toward a long-horizon goal. Generated outcomes then guide the next action and the decision to stop.

This formulation raises five empirical questions. *First*, does closed-loop execution improve procedural task completion? *Second*, does generated visual feedback improve replanning? *Third*, can WorldGuide recognize completion and stop? *Fourth*, how do planning and execution failures interact? *Fifth*, does improving procedural correctness come at the cost of visual quality?

4.1 Experimental Setup

We fine-tune Qwen2.5-VL-7B-Instruct as the ContextPlanner and HunyuanVideo-1.5 as the Executor, on WorldGuide Bench. The planner uses $K{=}3$ recent clip-action pairs. We evaluate 980 test samples from WorldGuide Bench and all 294 Video-CraftBench samples (Ren et al., 2026). Training and inference settings are given in Appendix D. On WorldGuide Bench, starred baselines (Bernini, PhysAgent, and TempAct) use their own planning-execution loops. WorldGuide receives only the initial image and task goal. On Video-CraftBench, all models receive the initial image and goal without reference actions. Appendix E.1 details the protocols and ablations.

Metrics. We assess visual quality and video-text consistency using VBench (Zheng et al., 2025a), and measure similarity between generated and reference videos using **Paired Alignment**. We

[1] https://huggingface.co/MiniMaxAI/MiniMax-H3

Table 3: **Video-CraftBench: goal-conditioned generation.** All models receive the initial image and task goal without reference actions. $*$ denotes methods with their own planning–execution loops. Higher is better except Repeat/Skip.

Model	Video Quality Check					Video Planning Check				
	Aesth.	Imag.	Dynamic	Smooth.	Consist.	Align.	Plan Acc.	Order	Repeat/Skip ↓	Task Succ.
Cosmos-Predict2.5-2B	35.83	63.66	95.92	98.61	19.49	63.54	5.66	32.42	5.46	2.05
Open-Sora 2.0	33.86	30.26	47.62	99.04	21.61	52.97	22.03	82.96	2.05	14.32
HunyuanVideo-1.5	45.30	67.62	99.66	98.91	21.91	63.94	14.66	64.97	3.06	24.15
Wan2.2	**46.05**	72.47	**100.00**	98.31	22.28	71.98	2.98	18.22	4.11	2.40
CogVideoX	40.60	63.49	89.46	98.61	21.52	**75.13**	15.04	69.73	8.16	2.38
Helios	36.75	49.98	92.18	99.20	19.13	54.76	13.83	80.56	1.36	6.46
UniVideo	45.05	51.71	62.46	97.52	20.99	60.41	9.77	50.55	4.03	12.09
FlashMotion	43.78	67.29	99.32	98.83	20.73	59.19	7.32	38.89	1.70	3.74
MAGI-1	36.72	**75.40**	0.00	**99.61**	16.40	71.67	6.62	75.00	8.50	0.68
RoboMaster	35.98	71.30	**100.00**	99.26	18.59	65.75	3.36	15.58	1.37	0.68
SpMem	36.17	62.39	94.90	99.18	20.11	65.49	11.72	55.11	6.46	3.74
Astra	27.80	38.23	95.58	94.29	7.94	55.56	3.11	47.96	6.80	0.00
Yume-1.5	39.99	62.40	99.66	99.26	23.13	67.17	5.36	29.58	3.46	5.88
SkyReels-V3	42.01	72.14	41.84	99.59	19.74	64.50	8.29	58.80	1.74	1.04
MiniMax-H3[1]	40.65	69.65	**100.00**	99.14	23.18	68.84	28.64	88.55	2.16	32.73
LTX-2.5	41.60	68.30	85.71	99.46	21.37	66.22	6.59	46.08	11.68	3.44
HY-WorldPlay	35.03	63.84	62.24	99.03	13.66	66.17	2.77	38.74	1.37	0.00
PhysAgent*	39.08	67.69	88.10	97.69	18.86	68.61	16.62	83.13	1.40	1.05
TempAct*	42.43	72.22	**100.00**	98.32	**23.54**	64.71	7.57	51.26	**0.34**	24.47
Bernini*	41.14	61.84	98.26	98.15	16.47	58.36	5.56	29.51	1.77	7.77
WorldGuide (ours)	40.19	64.63	99.65	98.86	22.09	71.80	**57.42**	**94.45**	17.08	**47.69**

use Gemini to assess task execution in generated videos. **Task Success Rate** measures final goal achievement, and **Plan Accuracy** measures completion of required sub-goals. **Order Score** evaluates prerequisite order, while **Repeat/Skip** penalizes unnecessary repetition. Finally, GPT-5.2 evaluates the predicted action sequences. Its **Plan Success** score indicates whether a sequence would achieve the goal if executed correctly. Full metric definitions and evaluation prompts are provided in Appendix E.

4.2 Does Closed-Loop Execution Improve Procedural Task Completion?

Open-loop generation commits to an action sequence and rollout length before execution, preventing subsequent decisions from responding to incomplete or incorrect actions. WorldGuide instead predicts each action-consequences and termination from generated progress. Compared with its open-loop variant, this formulation raises Task Success from **11.71%** to **33.33%** and Plan Accuracy from **22.87** to **62.85** (Table 4). WorldGuide also outperforms MiniMax-H3 on WorldGuide Bench despite its access to reference actions (29.90% Task Success; Table 2), and on Video-CraftBench under goal-only conditioning (**47.69%** versus 32.73%; Table 3). A human study confirms this ranking (Appendix F).

However, closing the loop alone does not guarantee task completion: the planner must select appropriate atomic actions, and the executor must realize their consequences. As shown in Table 2 and 3, WorldGuide consistently outperforms other closed-loop planner–executor methods, including PhysAgent, TempAct, and Bernini, demonstrating that procedural training and grounding each decision in generated outcomes are critical for reliable completion.

4.3 Does Generated Visual Feedback Improve Replanning?

A closed-loop planner is only useful if it observes what was actually executed. To isolate this effect, we remove generated clips from the planner input, keeping only the task goal and previous actions; the Executor is unchanged. Restoring visual feedback raises Task Success from **14.72%** to **33.33%** and Plan Accuracy from **29.29** to **62.85** (Table 4). This shows action history alone is insufficient: knowing what the planner *requested* does not reveal whether the action was completed, partially realized, or drifted visually. Observing the generated outcome lets WorldGuide replan from the state that exists, so later decisions can compensate for execution errors and stay aligned with task progress.

4.4 Can WorldGuide Recognize Completion and Stop?

Completing a task requires knowing not only what to do next, but when to stop. The same ContextPlanner handles both, predicting either the next atomic action or a completion token from the rollout history. In autonomous execution, using generated visual feedback improves Plan Success from **86.41%** to **91.75%** and reduces Repeat/Skip from **20.19** to **15.73** (Table 5). This shows that

Table 4: **WorldGuide ablations.** Removing step-wise replanning (*open-loop*), planner visual feedback (*w/o visual*), or Executor memory (*w/o Mem*) lowers Task Success; *oracle ContextPlanner* substitutes reference actions to isolate planning errors. Protocols follow Tables 2 and 3 (Appendix E.1).

Variant	Video Quality Check					Video Planning Check				
	Aesth.	Imag.	Dynamic	Smooth.	Consist.	Align.	Plan Acc.	Order	Repeat/Skip ↓	Task Succ.
WorldGuide Bench										
WorldGuide (open-loop)	36.76	59.38	73.30	98.42	18.86	67.40	22.87	53.81	0.49	11.71
WorldGuide (w/o visual)	36.77	58.74	83.92	98.25	18.67	68.11	29.29	68.70	**0.08**	14.72
WorldGuide (w/o Mem)	38.70	55.35	88.35	98.14	19.70	64.74	55.36	91.12	4.39	27.56
WorldGuide (oracle ContextPlanner)	**52.44**	**66.72**	97.68	**99.49**	**33.37**	**77.14**	**65.42**	**94.28**	2.04	**38.82**
WorldGuide (ours)	49.36	64.77	**98.06**	98.96	31.13	73.85	62.85	92.42	6.06	33.33
Video-CraftBench										
WorldGuide (w/o Mem)	38.82	63.81	99.63	98.64	20.21	**73.77**	39.62	92.51	**4.10**	29.78
WorldGuide (ours)	**40.19**	**64.63**	**99.65**	**98.86**	**22.09**	71.80	**57.42**	**94.45**	17.08	**47.69**



Figure 5: Qualitative comparison for *"step-by-step fried rice tutorial"*. MiniMax-H3, LTX-2.5, HunyuanVideo, and Cosmos-Predict2.5 receive reference action sequences; Bernini and TempAct use their own planning–execution loops. WorldGuide predicts actions from the task goal and generated progress. Frames are uniformly sampled from each trajectory.

visual observations help the planner distinguish genuine task progress from merely knowing which actions were previously requested, leading to more complete and less redundant procedures.

With ground-truth histories, the planner reaches **79.66** Plan Accuracy at $K = 3$, versus **64.03** in autonomous rollout, where every decision depends on its generated outcomes. Despite this harder setting, Plan Success still reaches **91.75%**, showing WorldGuide can construct and terminate valid procedures from generated progress. Plan Success measures whether the predicted plan would achieve the goal if executed correctly, evaluated separately from visual stopping accuracy.

Table 5: **ContextPlanner evaluation on WorldGuide Bench.** Comparison of ground-truth-history planning against autonomous rollout with generated history; visual feedback improves plan quality during autonomous execution. FT: supervised fine-tuning; K: history length.

Setting	FT	K	Plan Acc. ↑	Order ↑	Repeat/Skip ↓	Success ↑
Qwen2.5-VL baseline	❄	3	1.04	0.00	8.82	0.00
ContextPlanner	🔥	1	56.48	42.35	21.16	39.17
ContextPlanner	🔥	2	75.83	71.28	15.56	44.66
ContextPlanner	🔥	3	79.66	72.56	14.71	47.04
WorldGuide (w/o visual)	🔥	3	63.84	77.46	20.19	86.41
WorldGuide planner-executor	🔥	3	64.03	79.45	15.73	91.75

4.5 How Do Planning and Execution Failures Interact?

Planning and execution errors are tightly coupled in closed-loop generation. Replacing the learned planner with oracle reference actions improves Task Success by **+5.49 %** (Table 4), using the same trained Executor and memory, showing stronger planning can push task completion further.

Execution quality also affects what the planner sees next. With the ContextPlanner and Executor checkpoint unchanged, enabling memory improves Task Success by **+5.77%** on WorldGuide Bench

and **+17.91%** on Video-CraftBench, and video-judged Plan Accuracy by **+7.49%** (Table 4). Since each generated state feeds the next planning step, execution errors can corrupt later evidence, underscoring how tightly planning and execution depend on each other in closed-loop task completion.

4.6 Does Procedural Improvement Compromise Visual Quality?

WorldGuide primarily targets reliable procedural completion, so its gains do not uniformly extend to every low-level video-quality metric. On WorldGuide Bench, it achieves the highest Aesthetic Quality (**49.36**) and Overall Consistency (**31.13**), while baselines like TempAct and SkyReels score higher on individual metrics such as Imaging Quality, Dynamic Degree, or Motion Smoothness (Table 2), consistent with differences in backbone scale, pretraining data, and optimization for visual fidelity.

More importantly, WorldGuide substantially improves Task Success, reaching **33.33%** on WorldGuide Bench and **47.69%** on Video-CraftBench. On Video-CraftBench, this comes with modest drops in some visual-quality metrics relative to MiniMax-H3, including Imaging Quality (**64.63** vs. **69.65**) and Overall Consistency (**22.09** vs. **23.18**). WorldGuide thus prioritizes task execution over visual fidelity in challenging cases, and improving this balance remains an important direction.

4.7 Qualitative Results

In Fig. 5, WorldGuide adds aromatics, vegetables, and rice before seasoning and mixing. Its text actions identify which ingredients to add and in what order. MiniMax-H3 shows seasoned rice early, then adds white rice again and changes cookware. TempAct and LTX-2.5 largely remain at ingredient handling, while Bernini loses scene consistency. These examples show why plausible individual actions are insufficient for a coherent procedure. Additional tasks appear in Appendix G.1.

5 Discussion & Limitations

WorldGuide shows that closed-loop procedural video generation substantially improves task completion, though the remaining gap points to room for more robust long-horizon execution. Failures typically arise from a premature or redundant planner action, or an Executor clip that does not fully realize the intended state transition; since each generated state feeds the next decision, such errors can propagate. The gains from visual feedback and memory show that better state tracking directly improves both planning and execution.

A promising direction is to strengthen this coupling further with richer state-validity signals and explicit recovery when a step is incomplete or inconsistent. The Executor already conditions on the ContextPlanner's action embeddings; joint optimization of both modules is a natural next step. Bounded visual memory compresses older states to control inference cost, and more selective retention of task-relevant information could improve long-horizon consistency while keeping this efficiency. These directions extend the current closed-loop formulation rather than requiring a different one.

6 Conclusion

We introduced **WorldGuide**, a video world model that turns procedural video generation into closed-loop task execution. The central idea is that a procedure should unfold through decisions grounded in the generated visual state: what to do next and when to stop depend on what has actually been accomplished. WorldGuide implements this principle by repeatedly selecting a semantic action, rendering its consequence, and using the resulting visual state to guide subsequent execution. Compressed visual memory supports state continuity across successive steps of long-horizon, goal-directed execution. We also introduced **WorldGuide Bench**, comprising approximately 59K annotated videos across 245 tasks and 27 procedural categories. Among the evaluated video generators and video world models, WorldGuide achieves the highest Task Success on both WorldGuide Bench and Video-CraftBench. Despite remaining execution and stopping errors, these results support a shift from open-loop future synthesis to closed-loop task execution. Our work takes a step towards how video world models can use their generated consequences to decide what to do next and when to stop, guiding step-by-step execution toward the final goal.

AI USE STATEMENT

We used Gemini 2.5 Flash to filter and annotate the source videos of WorldGuide Bench (Appendix B.2). These automatic annotations are validated by an independent human audit, in which three annotators rate a task-balanced sample against the original video footage and report per-criterion agreement (Appendix B.6, Table 9). For evaluation, Gemini 3.5 Flash and GPT-5.2 serve as automatic judges for video quality and predicted plans, respectively (Appendix E); these automatic judgments are further corroborated by a separate human study of model outputs (Appendix F). All prompts used for annotation and evaluation are provided in Figs. 6, 7, and 8. We also used generative AI tools to assist with writing, including editing, restructuring, and improving the clarity of the manuscript. We did not use generative AI tools to generate research ideas or experimental results. All AI-assisted text was reviewed and revised by the authors, and we take full responsibility for the final content of this paper, including all text, claims, and artifacts.

ETHICS STATEMENT

WorldGuide Bench is built from publicly available YouTube instructional videos of everyday tasks such as origami, cooking, and assembly, collected under permissive (Creative Commons) licenses that permit reuse for research purposes. The videos are used only for non-commercial research, and we release only source identifiers, timestamps, and step annotations, not the original videos, respecting the rights of content owners. Our annotations describe task actions and completion signals rather than the identity or personal attributes of demonstrators, although some videos may incidentally contain identifiable individuals performing the task. Our human evaluation reports only aggregate ratings and collects no personal information about raters (Appendix B.6, Appendix F). WorldGuide is intended for applications such as instructional guidance and procedural understanding (Appendix A). As with any video generation technology, it could be misused to create misleading content, and we encourage its use consistent with the ICLR Code of Ethics.

REPRODUCIBILITY STATEMENT

Section 3 specifies the WorldGuide rollout, ContextPlanner, Executor, and hierarchical memory, and Appendix C gives the full memory conditioning, multi-scale embedding, and token budget. Training configurations and hyperparameters for both modules are listed in Appendix D (Tables 11 and 12), together with the rollout and sampling settings (Appendix D.3). WorldGuide Bench is described in Sec. 3.5 and Appendix B, including collection, the annotation prompt (Fig. 6), the training-sample format, and the evaluation split. Evaluation protocols, baselines, ablation variants, metric definitions, and complete judge prompts are provided in Appendix E. We will release the WorldGuide Bench annotations and splits, together with our training code and checkpoints, upon publication.

REFERENCES

Triantafyllos Afouras, Effrosyni Mavroudi, Tushar Nagarajan, Huiyu Wang, and Lorenzo Torresani. Ht-step: Aligning instructional articles with how-to videos. *Advances in Neural Information Processing Systems*, 36:50310–50326, 2023.

Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, et al. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL https://arxiv.org/abs/2502.13923.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):4125–4141, 2020.

Haoge Deng, Ting Pan, Haiwen Diao, Zhengxiong Luo, Yufeng Cui, Huchuan Lu, Shiguang Shan, Yonggang Qi, and Xinlong Wang. Autoregressive video generation without vector quantization. *arXiv preprint arXiv:2412.14169*, 2024.

Yilun Du, Sherry Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, Leslie Kaelbling, et al. Video language planning. In *International Conference on Learning Representations*, volume 2024, pp. 31138–31155, 2024.

Jiahui Fu, Junyu Nan, Lingfeng Sun, Hongyu Li, Jianing Qian, Jennifer L Barry, Kris Kitani, and George Konidaris. Novaplan: Zero-shot long-horizon manipulation via closed-loop video language planning. *arXiv preprint arXiv:2602.20119*, 2026a.

Xiao Fu, Xintao Wang, Xian Liu, Jianhong Bai, Runsen Xu, Pengfei Wan, Di Zhang, and Dahua Lin. Learning video generation for robotic manipulation with collaborative trajectory control. In *International Conference on Learning Representations*, volume 2026, pp. 144128–144142, 2026b.

Gemini Team, Google. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Google DeepMind. Veo. `https://deepmind.google/models/veo/`, 2025. Accessed May 2, 2026.

Google DeepMind. Gemini 3.5 flash. Model Card, 2026. URL `https://deepmind.google/models/model-cards/gemini-3-5-flash/`. Published May 19, 2026.

Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19383–19400, 2024.

Yoav HaCohen, Benny Brazowski, Nisan Chiprut, Yaki Bitterman, Andrew Kvochko, Avishai Berkowitz, Daniel Shalem, Daphna Lifschitz, Dudu Moshe, Eitan Porat, et al. Ltx-2: Efficient joint audio-visual foundation model. *arXiv preprint arXiv:2601.03233*, 2026.

Xuanhua He, Tianyu Yang, Ke Cao, Ruiqi Wu, Cheng Meng, Yong Zhang, Zhuoliang Kang, Xiaoming Wei, and Qifeng Chen. Active intelligence in video avatars via closed-loop world modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27239–27248, 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 Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Debang Li, Zhengcong Fei, Tuanhui Li, Yikun Dou, Zheng Chen, Jiangping Yang, Mingyuan Fan, Jingtao Xu, Jiahua Wang, Baoxuan Gu, et al. Skyreels-v3 technique report. *arXiv preprint arXiv:2601.17323*, 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.

Quanhao Li, Zhen Xing, Rui Wang, Haidong Cao, Qi Dai, Daoguo Dong, and Zuxuan Wu. Flashmotion: Few-step controllable video generation with trajectory guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8986–8996, 2026c.

Xiaofeng Mao, Zhen Li, Chuanhao Li, Xiaojie Xu, Kaining Ying, Tong He, Jiangmiao Pang, Yu Qiao, and Kaipeng Zhang. Yume-1.5: A text-controlled interactive world generation model. *arXiv preprint arXiv:2512.22096*, 2025a.

Xiaofeng Mao, Shaoheng Lin, Zhen Li, Chuanhao Li, Wenshuo Peng, Tong He, Jiangmiao Pang, Mingmin Chi, Yu Qiao, and Kaipeng Zhang. Yume: An interactive world generation model. *arXiv preprint arXiv:2507.17744*, 2025b.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2630–2640. IEEE, 2019.

OpenAI. Gpt-5.2 system card. OpenAI, December 2025a. URL `https://openai.com/index/gpt-5-system-card-update-gpt-5-2/`. Published December 11, 2025.

OpenAI. Sora 2: Advancing video generation models. `https://openai.com/index/sora-2/`, 2025b. OpenAI research release, published September 30, 2025. Accessed November 11, 2025.

Rohith Peddi, Shivvrat Arya, Bharath Challa, Likhitha Pallapothula, Akshay Vyas, Bhavya Gouripeddi, Qifan Zhang, Jikai Wang, Vasundhara Komaragiri, Eric Ragan, et al. Captaincook4d: A dataset for understanding errors in procedural activities. *Advances in Neural Information Processing Systems*, 37:135626–135679, 2024.

Lu Qiu, Yi Chen, Yuying Ge, Yixiao Ge, Ying Shan, and Xihui Liu. Egoplan-bench2: A benchmark for multimodal large language model planning in real-world scenarios. *International Journal of Computer Vision*, 134(5):222, 2026.

Zhongwei Ren, Yunchao Wei, Xiao Yu, Guixun Luo, Yao Zhao, Bingyi Kang, Jiashi Feng, and Xiaojie Jin. Videoworld 2: Learning transferable knowledge from real-world videos. *arXiv preprint arXiv:2602.10102*, 2026.

Fadime Sener, Dibyadip Chatterjee, Daniel Shelepov, Kun He, Dipika Singhania, Robert Wang, and Angela Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21064–21074. IEEE, 2022.

Yifan Shen, Jiateng Liu, Xinzhuo Li, Yuanzhe Liu, Bingxuan Li, Houze Yang, Wenqi Jia, Yijiang Li, Tianjiao Yu, James Matthew Rehg, et al. Egoforge: Goal-directed egocentric world simulator. *arXiv preprint arXiv:2603.20169*, 2026.

Yale Song, Eugene Byrne, Tushar Nagarajan, Huiyu Wang, Miguel Martin, and Lorenzo Torresani. Ego4d goal-step: Toward hierarchical understanding of procedural activities. *Advances in neural information processing systems*, 36:38863–38886, 2023.

Wenqiang Sun, Haiyu Zhang, Haoyuan Wang, Junta Wu, Zehan Wang, Zhenwei Wang, Yunhong Wang, Jun Zhang, Tengfei Wang, and Chunchao Guo. Worldplay: Towards long-term geometric consistency for real-time interactive world modeling. *arXiv preprint arXiv:2512.14614*, 2025.

Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. Coin: A large-scale dataset for comprehensive instructional video analysis. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1207–1216. IEEE, 2019.

Bernini Team, Chenchen Liu, Junyi Chen, Lei Li, Lu Chi, Mingzhen Sun, Zhuoying Li, Yi Fu, Ruoyu Guo, Yiheng Wu, et al. Bernini: Latent semantic planning for video diffusion. *arXiv preprint arXiv:2605.22344*, 2026.

Hansi Teng, Hongyu Jia, Lei Sun, Lingzhi Li, Maolin Li, Mingqiu Tang, Shuai Han, Tianning Zhang, WQ Zhang, Weifeng Luo, et al. Magi-1: Autoregressive video generation at scale. *arXiv preprint arXiv:2505.13211*, 2025.

Xiaoyu Tian, Haotian Wang, Shuaiting Chen, Hao Zhou, Kaichi Yu, Yudian Zhang, Jade Ouyang, Junxi Yin, Jiong Chen, Baoyan Guo, et al. Astra: Automated synthesis of agentic trajectories and reinforcement arenas. *arXiv preprint arXiv:2601.21558*, 2026.

Keyu Tu, Zhuowei Chen, Mengqi Huang, Yuxin Wang, Jiahao Zhu, Zhendong Mao, and Yongdong Zhang. Semcomp-bench: Benchmarking semantic task completion in video generation. *arXiv preprint arXiv:2608.17426*, 2026.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Jing Wang, Xiangxin Zhou, Jiajun Liang, Kaiqi Liu, Wanyuan Pang, Zhenyu Xie, Tianyu Pang, and Xiaodan Liang. Tempact: Advancing temporal plausibility in autoregressive video generation via planner-executor rl. *arXiv preprint arXiv:2606.28016*, 2026.

Cong Wei, Quande Liu, Zixuan Ye, Qiulin Wang, Xintao Wang, Pengfei Wan, Kun Gai, and Wenhu Chen. Univideo: Unified understanding, generation, and editing for videos. In *International Conference on Learning Representations*, volume 2026, pp. 113905–113933, 2026.

Bing Wu, Chang Zou, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Jack Peng, Jianbing Wu, Jiangfeng Xiong, Jie Jiang, et al. Hunyuanvideo 1.5 technical report. *arXiv preprint arXiv:2511.18870*, 2025.

Tong Wu, Shuai Yang, Ryan Po, Yinghao Xu, Ziwei Liu, Dahua Lin, and Gordon Wetzstein. Video world models with long-term spatial memory. *Advances in Neural Information Processing Systems*, 38:49371–49393, 2026.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. Byt5: Towards a token-free future with pre-trained byte-to-byte models, 2022. URL https://arxiv.org/abs/2105.13626.

Yu Yang, Yue Liao, Jianbiao Mei, Baisen Wang, Xuemeng Yang, Licheng Wen, Jiangning Zhang, Xiangtai Li, Liang Lv, Hanlin Chen, et al. Spiral: Self-evolving action-conditioned video generation via reflective planning agents. *arXiv preprint arXiv:2603.08403*, 2026.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, volume 2025, pp. 83048–83077, 2025.

Hangjie Yuan, Weihua Chen, Jun Cen, Hu Yu, Jingyun Liang, Shuning Chang, Zhihui Lin, Tao Feng, Pengwei Liu, Jiazheng Xing, et al. Lumos-1: On autoregressive video generation from a unified model perspective. *arXiv preprint arXiv:2507.08801*, 2025.

Shenghai Yuan, Yuanyang Yin, Zongjian Li, Xinwei Huang, Xiao Yang, and Li Yuan. Helios: Real real-time long video generation model. *arXiv preprint arXiv:2603.04379*, 2026.

Lvmin Zhang, Shengqu Cai, Muyang Li, Gordon Wetzstein, and Maneesh Agrawala. Frame context packing and drift prevention in next-frame-prediction video diffusion models. *Advances in Neural Information Processing Systems*, 38:30546–30566, 2026a.

Wancong Zhang, Basile Terver, Artem Zholus, Soham Chitnis, Harsh Sutaria, Mido Assran, Randall Balestriero, Amir Bar, Adrien Bardes, Yann LeCun, et al. Hierarchical planning with latent world models. *arXiv preprint arXiv:2604.03208*, 2026b.

Dian Zheng, Ziqi Huang, Hongbo Liu, Kai Zou, Yinan He, Fan Zhang, Lulu Gu, Yuanhan Zhang, Jingwen He, Wei-Shi Zheng, et al. Vbench-2.0: Advancing video generation benchmark suite for intrinsic faithfulness. *arXiv preprint arXiv:2503.21755*, 2025a.

Zangwei Zheng, Xiangyu Peng, Yuxuan Lou, Chenhui Shen, Tom Young, Xinying Guo, Binluo Wang, Hang Xu, Hongxin Liu, Mingyan Jiang, et al. Open-sora 2.0: Training a commercial-level video generation model in $200k. *arXiv preprint arXiv:2503.09642*, 2025b.

Luowei Zhou, Chenliang Xu, and Jason Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3532–3540. IEEE, 2019.

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

Supplementary Contents

A. Design Rationale . . . 16

B. WorldGuide Bench . . . 16

B.1. Collection and Filtering . . . 17
B.2. Temporal Action Annotation . . . 17
B.3. Training Samples . . . 17
B.4. Statistics and Evaluation Split . . . 17
B.5. Annotation Example . . . 17
B.6. Human Audit of Annotations . . . 20

C. Hierarchical Memory . . . 20

C.1. History and Reference Conditioning . . . 21
C.2. Multi-Scale Embedding and Injection . . . 21
C.3. Compression Schedule and Token Budget . . . 21

D. Training and Inference . . . 22

D.1. ContextPlanner Training . . . 22
D.2. Executor Training . . . 22
D.3. Rollout and Termination . . . 23

E. Evaluation Protocol and Metrics . . . 24

E.1. Benchmarks, Baselines, and Ablations . . . 24
E.2. Video-Judge Metrics . . . 25
E.3. Planner Evaluation . . . 26
E.4. Paired Alignment . . . 26
E.5. VBench Video-Quality Metrics . . . 27

F. Human Evaluation of Generated Videos . . . 28

G. Additional Results . . . 29

G.1. WorldGuide Bench Qualitative Comparisons . . . 29
G.2. Video-CraftBench Results . . . 37
G.3. Statistical Significance of Task Success . . . 37

H. Limitations and Failure Analysis . . . 45

Interactive qualitative results: We provide anonymous interactive webpages for inspecting generated videos and dataset annotations. The video comparison page shows complete rollouts of WorldGuide and the baselines side by side: **https://worldguide-comparison.pages.dev**. The annotation page shows additional WorldGuide Bench videos with their step-level annotations: **https://worldguide-annotation.pages.dev**. These pages are supplementary qualitative material; all quantitative results and conclusions are reported in the paper.

A Design Rationale

This section explains three design choices of WorldGuide: the choice of backbones, the sequential training of the two modules, and the language-level interface between planning and execution.

One model for planning and action encoding. HunyuanVideo-1.5 (Wu et al., 2025) encodes its text prompts with Qwen2.5-VL-7B-Instruct (Bai et al., 2025). We therefore initialize the ContextPlanner from this same model, so that a single network both predicts the next action and encodes it for the Executor. This choice has three advantages. First, *the Executor receives action embeddings from the model it was pretrained with, so fine-tuning adapts an existing interface rather than learning a new one*. Second, *WorldGuide adds no separate planning network: the ContextPlanner takes the place of the Executor's text encoder, keeping the complete system at the size of the original HunyuanVideo-1.5 pipeline*. Third, *the action predicted by the planner at inference reaches the Executor through exactly the embedding pathway used during training.*

Choice of Executor. HunyuanVideo-1.5 has 8.3B parameters, fewer than many of the evaluated generators, yet it is among the strongest at procedural execution. Given reference actions on WorldGuide Bench, it achieves the highest Plan Accuracy (43.53) and Task Success (12.00%) of all baselines except the 33B MiniMax-H3, ahead of larger models such as Wan2.2-14B, LTX-2.5-22B, and FlashMotion-34B (Table 2). Building on it keeps WorldGuide compact while starting from a strong executor.

Sequential training. Both modules are trained on the same procedural demonstrations, but sequentially rather than end-to-end. The ContextPlanner is first fine-tuned for next-action and completion prediction; it is then frozen while the Executor is fine-tuned on action captions embedded by it (Appendix D). Freezing is deliberate. If the Executor's flow-matching loss also updated the ContextPlanner, its parameters would be optimized to produce embeddings that ease video generation, with no signal that preserves its language outputs. Its predicted actions could then drift away from faithful, readable instructions. Freezing the ContextPlanner keeps the interface between planning and execution in natural language.

Why a language-level interface matters. WorldGuide represents each predicted action as a natural-language instruction and uses it to generate the corresponding video clip. This language interface has three main benefits:

- **Interpretable guidance.** Each generated clip is paired with a clear instruction describing the intended action. This makes the generated procedure easier to understand and follow.
- **Diagnosable failures.** Planning and execution errors can be analyzed separately. Since actions are represented as text, we can replace predicted actions with reference actions while keeping the Executor unchanged. We use this setting for the oracle ContextPlanner experiment in Table 4.
- **Independent evaluation.** The predicted action sequence can be evaluated directly as text, independently of the quality of the generated videos (Table 5).

In this work, we keep the language interface fixed during Executor training. Jointly optimizing the Planner and Executor while preserving clear language outputs is an interesting direction for future work (Sec. 5).

B WorldGuide Bench

WorldGuide Bench supplies the three signals on which the closed loop is trained: which atomic action comes next, how that action changes the visual state, and when the task is complete. This section describes how videos are collected (Appendix B.1), annotated (Appendix B.2), converted into training samples (Appendix B.3), and split (Appendix B.4), followed by an annotation example (Appendix B.5) and a human audit (Appendix B.6).

Table 1 compares WorldGuide Bench with existing procedural video datasets. These target planning question answering (EgoPlan-IT (Qiu et al., 2026)), step localization and grounding (Ego4D Goal-Step (Song et al., 2023), COIN (Tang et al., 2019), CrossTask (Zhukov et al., 2019), HT-Step (Afouras et al., 2023)), segmentation, recognition, and error detection (YouCook2 (Zhou et al., 2018), Assembly101 (Sener et al., 2022), CaptainCook4D (Peddi et al., 2024), Ego-Exo4D Keystep (Grauman et al., 2024)), or video generation and its evaluation (SemComp-Data (Tu et al., 2026), ActVideoGen (Yang et al., 2026), EgoForge (Shen et al., 2026)). None of them provides explicit task-completion supervision, which WorldGuide Bench adds to step-level atomic-action clips and task goals.

B.1 Collection and Filtering

We collect public instructional videos of at most seven minutes from YouTube, covering 245 tasks in 27 procedural categories such as origami, cooking, construction, assembly and repair, sorting, and knotting. Each task is retrieved with task-specific queries expanded with short-form and multilingual variants in 15 languages, so every task has many demonstrations with different objects, scenes, and demonstrators. Files that fail integrity or frame-decoding checks are discarded.

Gemini 2.5 Flash (Gemini Team, Google, 2025) rates each video with a Video Planning Score from 0 to 5 for topical relevance and demonstration completeness (Fig. 6). We keep only videos scored 5, retaining 58,679 of 133,360 retrieved records (44.0%; Table 7). Annotation quality is verified by a human audit (Appendix B.6).

All source videos are publicly available and used for non-commercial research. We will release source identifiers, timestamps, and step annotations rather than the videos themselves.

B.2 Temporal Action Annotation

Gemini converts each video into an ordered sequence of atomic physical actions. Each action is assigned a start and end timestamp, with a target duration of 1–5 seconds (Fig. 6). Actions longer than 5 seconds are split into multiple consecutive steps. The final step is marked as task completion. For videos longer than 5 minutes, we run annotation at $1.5\times$ playback speed for efficiency and rescale the predicted timestamps back to the original video timeline. As a result, annotated steps can span 1.5–7.5 seconds in the original video. All clips are extracted from the original-speed videos, and empty or invalid clips are removed.

B.3 Training Samples

Each annotated step becomes one training record (Table 6): the task context, the action caption, the target clip cut at the step's timestamps, the step index and normalized progress, the preceding clips and captions, and a completion flag set on the final step. A one-second preview from the start of the source video provides the initial visual state. From these records, the ContextPlanner learns to predict the next caption, or the completion token once the task is complete, from the goal and up to $K{=}3$ preceding clip–caption pairs; the Executor learns to generate each clip from its caption and preceding visual context. Both modules are thus supervised by the same demonstrations, each with its own objective (Appendix A).

B.4 Statistics and Evaluation Split

Table 7 reports the number of videos after each filtering stage. We hold out four videos per task, giving 980 test videos and 57,699 training videos; all clips of a held-out video are excluded from the training of both modules. The split therefore evaluates unseen demonstrations of known tasks.

B.5 Annotation Example

Table 8 shows a complete annotation example from an origami video. Each video in WorldGuide Bench follows the same format: a task description, a short description of the work subject, and an ordered sequence of atomic actions with their start and end times. During preprocessing, we split the video into one clip for each annotated step. We then organize the clips, action captions, step order, and completion flag into the step-level records described in Appendix B.3. The example in Table 8

Gemini Dataset-Captioning Prompt

```
You are an expert temporal video captioning model.
Task: Watch the video and convert it into a timeline of
actionable steps. Temporal alignment between each step and its
timestamp is very important.
Main task/topic: {query}.
Description: {description}.
Video Planning Score:
- 0: Video does not match the topic.
- 1-2: Poor quality, missing steps, or hard to follow.
- 3-4: Good quality, most steps visible.
- 5: Excellent, complete step-by-step tutorial.
Timeline Rules:
- Output only the steps necessary to complete the task. Please
align the tasks with the particular timeline.
- Each step must be between 1 and 5 seconds. (Adjusted for
long-video stability).
- If an action takes longer than 5s, split it into consecutive
steps.
- Use precise verbs: Fold, Aligns, Presses, Flips, Rotates.
- No descriptions of background, hands, or music.
Important:
- If Score = 0, output "Video Planning Score: 0" and stop.
- If Score > 0, output the Score followed by the timeline.
Strict output format:
Video Planning Score: <0-5>
Work subject: <one-line title describing the specific {task} of
{category} shown in the video>
Time: <start>-<end> sec, Step <N>: <one clear physical action>
```

Figure 6: Per-video prompt supplied to Gemini 2.5 Flash for dataset captioning. The video, together with the retrieval query, task, category, and description, is passed alongside this prompt, prefixed by a fixed system instruction: *"You are a professional technical writer. Your task is to provide a non-repetitive, high-accuracy timeline of physical actions from video."* Only videos assigned a Video Planning Score of 5 are retained.

Table 6: **WorldGuide Bench sample format.** Each record contains an annotated action, its corresponding video clip, and the preceding procedural context. The completion flag identifies the final step.

Information	Contents
Task context	Category, task label, and task goal.
Action annotation	Step index, action caption, and start/end timestamps in the original video.
Target clip	Video segment corresponding to the annotated action.
Preceding history	Previous clips and their action captions. The ContextPlanner uses up to $K = 3$ previous steps, while the Executor uses visual memory.
Progress	Step index normalized by the total number of annotated steps.
Completion	Binary flag indicating whether the current action is the final step.

Table 7: **WorldGuide Bench construction statistics.** Counts are reported after each filtering stage. Only videos receiving a Gemini Video Planning Score of 5/5 are retained. The video-disjoint test split holds out four videos per task.

Quantity	Count
Top-level categories	27
Task topics	245
Merged raw video metadata records	133,360
Validated (decodable) video records	132,911
Videos with a non-null Gemini annotation	119,798
Videos with planning score = 5 (retained)	58,679
Training split videos	57,699
Test split videos (held out)	980
videos per task	4
tasks / categories	245 / 27

Table 8: **Complete annotation of one WorldGuide Bench video.** Task: origami, *airplane_dart*. Work subject: folding a paper airplane (dart style). Each row is one clip; Time gives its start and end in the source video (mm:ss).

Step	Time	Action caption
1	00:04–00:05	Folds the bottom right corner of the paper to the top left.
2	00:05–00:06	Unfolds the paper.
3	00:06–00:07	Folds the bottom left corner of the paper to the top right.
4	00:07–00:08	Unfolds the paper.
5	00:08–00:09	Flips the paper over.
6	00:09–00:10	Folds the bottom edge of the paper upwards.
7	00:10–00:11	Unfolds the paper.
8	00:11–00:12	Flips the paper over.
9	00:12–00:13	Collapses the side edges inwards to form a triangle at the bottom.
10	00:13–00:14	Presses the folded triangle flat.
11	00:14–00:15	Folds the left corner of the triangle towards the center.
12	00:15–00:16	Folds the right corner of the triangle towards the center.
13	00:16–00:17	Folds the bottom tip of the triangle upwards.
14	00:17–00:18	Presses the fold firmly.
15	00:18–00:19	Folds the entire paper in half lengthwise.
16	00:19–00:20	Flips the paper over.
17	00:20–00:21	Folds down one wing, aligning it with the bottom edge.
18	00:21–00:22	Presses the wing fold.
19	00:22–00:23	Flips the paper over.
20	00:23–00:24	Folds down the second wing, aligning it with the bottom edge.
21	00:24–00:25	Presses the wing fold. `<|Task Completed|>`

contains 21 aligned clip–caption pairs. The final step is marked with the completion flag `<|Task Completed|>`. More annotation examples are available at **https://worldguide-annotation.pages.dev**.

This example shows three important properties of our annotations. *First*, each caption describes a single physical action. This includes fine-grained intermediate actions, such as folding and then unfolding the paper in Steps 1–4. *Second*, the same action can appear at different stages of a task. For example, *"Unfolds the paper"* appears at Steps 2, 4, and 7, while *"Flips the paper over"* appears at Steps 5, 8, 16, and 19. Similarly, *"Presses the wing fold"* appears at Steps 18 and 21. Therefore, the action text alone is not enough to determine the current task progress. The ContextPlanner also observes the generated clips to understand the current visual state (Sec. 4.3). *Third*, the completion flag on the final step provides supervision for learning when to stop.

For each step, the Executor learns to generate the corresponding video clip from the action caption and previous visual context. The ContextPlanner learns to predict the next action from up to $K=3$ previous clip-action pairs. For example, Steps 15–17 are used as context when predicting Step 18.

Criterion	Level	Annot. 1	Annot. 2	Annot. 3	Majority
Caption correctness	Step	82.4	82.6	81.9	**83.0**
Action atomicity	Step	82.2	81.5	82.3	**82.9**
Task-label correctness	Video	82.9	82.4	83.2	**83.7**
Essential-action coverage	Video	83.2	84.4	83.9	**84.5**
Temporal order	Video	82.1	81.0	81.6	**81.6**
Demonstration completeness	Video	81.4	81.7	82.6	**82.9**

Table 9: **Human audit of WorldGuide Bench annotations** (% judged correct) on 245 task-balanced videos with 3,168 action steps. *Majority* aggregates the three annotators by item-level vote.

B.6 Human Audit of Annotations

Since WorldGuide Bench is automatically annotated, we conduct a human audit to evaluate the annotation quality. Three independent annotators review 245 videos, with one video sampled from each task. These videos contain a total of 3,168 annotated action steps. For each video, the annotators watch the original video together with its task label and ordered action captions. Each annotator evaluates the annotations independently.

Evaluation criteria. We evaluate the annotations using six criteria. Each criterion is answered based only on the visual evidence in the video.

- **Caption correctness** (per step): Does the caption correctly describe the visible action, the involved objects, and the result of the action?
- **Action atomicity** (per step): Does the caption describe one clear action rather than multiple actions?
- **Task-label correctness** (per video): Does the task label correctly describe the task performed in the video?
- **Essential-action coverage** (per video): Do the captions cover all actions needed to complete the task?
- **Temporal order** (per video): Are the captions ordered according to when the actions occur in the video?
- **Demonstration completeness** (per video): Does the final visible state show that the task has been completed?

Scoring. For each item, annotators choose *Correct* (1), *Incorrect* (0), or *Cannot determine* (U). The U option is used when the video does not provide enough visual evidence to make a decision, and these ratings are excluded from the score. The score is the percentage of rated items marked as Correct. For step-level criteria, scores are first averaged within each video and then across videos, so that every task contributes equally. We also report a majority-vote score. An item receives a final Correct or Incorrect label when at least two of the three annotators agree.

Results. Table 9 summarizes the human audit results. The majority-vote scores range from 81.6% to 84.5% across all six criteria. Caption correctness reaches 83.0%, task-label correctness reaches 83.7%, and essential-action coverage reaches 84.5%. The scores from the three annotators differ by at most 1.3 percentage points for each criterion, showing similar judgments across annotators. Overall, the audit indicates that the automatic annotation pipeline provides reliable supervision for action captions, temporal order, and task completion.

C Hierarchical Memory

The Executor uses the spatial history compression strategy from YUME (Mao et al., 2025a; Zhang et al., 2026a). Recent latent frames are kept at higher spatial resolution, while older frames are

progressively compressed to lower resolutions. This keeps the number of memory tokens bounded as the generated video becomes longer.

C.1 History and Reference Conditioning

Terminology. The history contains $F \leq T + 1 \leq 1366$ latent frames. It includes up to $T \leq 1365$ recent frames and the oldest frame, which we call the *anchor*. We also use a separate *reference* slot containing the latest generated frame. The reference may contain the same frame as the newest history entry, but it uses separate token positions.

We use L to denote the latent length of the clip being generated. All frame counts in this section refer to latent frames after VAE encoding. For N continuously encoded video frames, the VAE produces $\lfloor (N-1)/4 \rfloor + 1$ latent frames.

Supplied history. To construct the history, we concatenate the latent frames from previous clips and keep the most recent frames that fit within the available history length. If the history is shorter than the required length, we left-pad it; these padded positions add tokens but contain no visual information. Since one conditioning slot is reserved for the reference frame, the history length satisfies

$$F \leq \min(L - 1, T + 1).$$

The history is updated after each generated clip. If there is no previous history, the memory branch is skipped.

Conditioning input. The input to the DiT is

$$x_{\mathrm{DiT}} = [\, \tilde{z}\,;\, z_{\mathrm{cond}}\,;\, m_{\mathrm{cond}}\,] \in \mathbb{R}^{(2C+1)\times L\times H\times W}. \tag{7}$$

Here, $\tilde{z}$ is the noisy video latent, z_{cond} contains the visual conditioning, and m_{cond} indicates which conditioning positions are valid. Without memory, only the reference slot in z_{cond} is filled. With memory, z_{cond} contains the history followed by the reference frame. The same history latents are also spatially compressed into memory tokens, as described in Appendix C.2.

C.2 Multi-Scale Embedding and Injection

Multi-scale embedders. We use different spatial resolutions to represent the visual history. With patch size $p = 2$, we define embedders E_r for compression rates $r \in \{1, 2, 4, 8, 16\}$. Each E_r is a 3-D convolution with kernel size and stride $(1, pr, pr)$, which maps C input channels to d-dimensional features. E_1 uses the pretrained patch-embedding weights. For larger compression rates, we initialize the convolution kernels by trilinearly interpolating the pretrained weights.

For the coarsest rate, $r = 64$, we first apply a channel-preserving 3-D convolution with kernel size and stride $(1, 4, 4)$, followed by E_{16}. This gives an effective spatial stride of 128. The temporal kernel size and stride are always one. Therefore, the compression only reduces the number of spatial tokens while preserving all temporal latent frames.

Positional encoding and memory injection. Each compressed history segment receives 3-D rotary positional embeddings based on its spatial grid. Temporal positions continue across consecutive history segments, preserving their temporal order. These positions correspond to latent-frame positions rather than timestamps in the original video.

The resulting memory tokens $\mathcal{M}$ are added to the Executor's context sequence together with the visual and text conditioning:

$$\tau = [\, \mathcal{M}\,;\, c_{\text{vision}}\,;\, c_{\text{text}}\,]. \tag{8}$$

This adds visual-history information as conditioning tokens without increasing the number of video tokens being denoised or changing the training objective in Eq. 2. When no history is available, the memory branch is skipped and the Executor uses only the reference-image conditioning.

C.3 Compression Schedule and Token Budget

The schedule is selected by the history length F. Table 10 gives the longest schedule ($343 \leq F \leq 1366$), with ages running from the newest frame (1) to the anchor (F).

Table 10: Nominal token cost for $343 \leq F \leq 1366$ history latent frames. One full-resolution latent-frame equivalent is HW/p^2 tokens; the reference is counted separately.

Segment	Latent frames	Rate	Cost upper bound
Present	3	1	3.00
Near present	2	2	0.50
Recent past	16	4	1.00
Mid past	64	8	1.00
Far past	256	16	1.00
Distant past	$F - 342 \leq 1024$	64	0.25
Oldest anchor	1	2	0.25
History subtotal	F	–	7.00
Reference	1	1	1.00
Total	$F + 1$	–	**8.00**

Shorter histories. For $7 \leq F \leq 342$, the newest three frames use rate one, the next two use rate two, and successively older tiers use rates four, eight, and sixteen as needed. The oldest anchor uses rate one for $F \leq 86$ and rate two thereafter. For $F \leq 6$, both endpoints use rate one and the intervening frames use rate two. For $F \leq 2$, the empty intermediate slice reuses the newest latent, as in YUME. Using nominal area ratios before spatial padding, the bounds including anchor and reference are

$$\frac{p^2}{HW}|\mathcal{M}|_{\text{nominal}} \leq \begin{cases} 4, & 1 \leq F \leq 6, \\ 6.5, & 7 \leq F \leq 22, \\ 7.5, & 23 \leq F \leq 86, \\ 7.75, & 87 \leq F \leq 342, \\ 8, & 343 \leq F \leq 1366. \end{cases} \tag{9}$$

For every history length, the history, anchor, and reference therefore cost at most eight full-resolution latent-frame equivalents before spatial padding. This bounds the number of context tokens, not the number of retained temporal frames.

Spatial padding. With padding, a rate-r embedder produces $\lceil H/(pr) \rceil \lceil W/(pr) \rceil$ tokens per latent frame, the coarsest embedder $\lceil H/128 \rceil \lceil W/128 \rceil$, and the reference $\lceil H/p \rceil \lceil W/p \rceil$. These integer counts can exceed the nominal bound when H and W are not divisible by the strides, but for a fixed history capacity and resolution the token count remains bounded independently of rollout length.

D Training and Inference

The two modules are trained sequentially on the WorldGuide Bench training split (Appendix B.4): first the ContextPlanner, then the Executor with the ContextPlanner frozen (Appendix A).

D.1 ContextPlanner Training

We fine-tune all parameters of Qwen2.5-VL-7B-Instruct (Bai et al., 2025) to predict the next action or the completion token from the task goal, the current visual state, and up to $K{=}3$ preceding clip–action pairs. The cross-entropy loss is applied only to response tokens (Table 11).

D.2 Executor Training

We fine-tune HunyuanVideo-1.5 (Wu et al., 2025) for action-conditioned image-to-video generation with the masked flow-matching objective of Eq. 2, using precomputed VAE latents. Each action caption is embedded by the frozen ContextPlanner, and the multi-scale memory embedders are trained together with the Executor (Table 12). The memory ablation evaluates this final checkpoint with and without memory (Appendix E.1).

Table 11: ContextPlanner training configuration. Full-parameter fine-tuning of Qwen2.5-VL-7B-Instruct for next-step action prediction from goal, visual state, and a history of $K=3$ clip–action pairs.

Component	Configuration
Base model	Qwen2.5-VL-7B-Instruct
Training objective	Autoregressive next-step prediction
Trainable parameters	Full-parameter fine-tuning
Precision	BF16
Distributed training	DeepSpeed ZeRO-3
Hardware	8 nodes $\times$ 8 AMD Instinct MI210 (64 GB)
Training time	7 days
Per-GPU batch size	2
Gradient accumulation	8
Effective global batch size	1024
Optimizer	AdamW
Learning rate	2×10^{-5}
Weight decay	0.01
LR schedule	Cosine
Warmup steps	100
Epochs	2
Maximum sequence length	4096
History length K	3 clip–action pairs

Table 12: Executor training configuration. HunyuanVideo-1.5 DiT fine-tuned for language-conditioned image-to-video generation on step-level clips under the masked flow-matching objective (Eq. 2).

Component	Configuration
Base model	HunyuanVideo-1.5 DiT
Training objective	Masked MSE flow matching
Input mode	Image-to-video
Target resolution	480×832
Target FPS	24
Text conditioning	Action-caption embeddings from the frozen ContextPlanner, and ByT5 (Xue et al., 2022) features
Visual conditioning	Image-conditioned latent branch and vision states
Hardware	8 nodes $\times$ 8 AMD Instinct MI210 (64 GB)
Training time	42 days
Learning rate	1×10^{-5}
Weight decay	1×10^{-4}
Gradient clipping	1.0
Precision	Mixed BF16; DiT backbone in FP32
Diffusion scheduler	FlowMatchEulerDiscreteScheduler
Timestep sampling	Logit-normal
Logit-normal parameters	$\mu = 0.0$, $\sigma = 1.0$
CFG dropout rate	0.1
Distributed training	FSDP
Sequence parallelism	Disabled (size 1)
Per-GPU batch size	1
Gradient accumulation	2
Effective global batch size	128 video segments

D.3 Rollout and Termination

At the first step, the ContextPlanner receives the task goal and the initial image; afterwards it keeps the latest $K=3$ generated clip–action pairs. When it emits `<|Task Completed|>`, the rollout stops immediately and no further clip is generated. Otherwise, the Executor renders the predicted action and its memory is refreshed with the new clip (Appendix C.1). Rollouts are capped at 80 steps, above the longest reference procedure in WorldGuide Bench (62 steps). The planner is never given

the reference step count, and a rollout that reaches the cap is scored as generated rather than marked successful. The generated clips are concatenated into the final video. Because each clip is generated separately and the memory token count is bounded (Appendix C.3), the per-step cost does not grow with rollout length.

Sampling uses 10 denoising steps, guidance scale 7.5, and flow shift 5.0. Training clips are sampled at 24 FPS, and generated videos are played back at 16 FPS.

E Evaluation Protocol and Metrics

E.1 Benchmarks, Baselines, and Ablations

Baselines. We compare WorldGuide with recent video generation models (Wu et al., 2025; Wan et al., 2025; Yuan et al., 2026; Wei et al., 2026; Li et al., 2026c; Zheng et al., 2025b; Yang et al., 2025; Li et al., 2026a), video world models (Ali et al., 2025; Sun et al., 2025; Mao et al., 2025a; Fu et al., 2026b; Wu et al., 2026; Tian et al., 2026; Teng et al., 2025; HaCohen et al., 2026), and three planner–executor methods: Bernini (Team et al., 2026), PhysAgent (Li et al., 2026b), and TempAct (Wang et al., 2026). These planner–executor methods use their own planning–execution loops and are marked with $*$ in the result tables.

WorldGuide Bench. We evaluate one rollout for each of the 980 held-out videos (Table 2). Unstarred baselines are given the complete ground-truth action plan. They generate the procedure step by step, following the reference actions in order. For each step, the model receives one action caption and continues generation from the last generated frame. Each step is generated for approximately 2 seconds (48 frames at 24 FPS or 32 frames at 16 FPS, depending on the lengths supported by each model). Therefore, these baselines know the full action sequence before generation begins.

In contrast, the starred planner–executor baselines and WorldGuide receive only the initial image and task goal. They must determine how to complete the task using their own planning process. No model receives future ground-truth video frames.

Video-CraftBench. We evaluate all 294 samples from Video-CraftBench_v0.1 (Ren et al., 2026), which contains block assembly and paper folding tasks.[2] In this benchmark, all models receive only the initial image and task goal, without a reference action plan (Table 3). Since no reference plan is provided, the judge evaluates whether the generated video completes the sub-goals needed to achieve the given task. Different valid procedures and step granularities are accepted.

Ablation variants. Table 4 evaluates four variants using the same evaluation protocol.

- **Open-loop:** The model predicts the full action sequence once from the initial image and task goal, then executes all actions without replanning. The action sequence and video length are therefore fixed before generation starts.
- **W/o visual:** The model still predicts actions step by step, but generated video clips are removed from the ContextPlanner input. The Executor remains unchanged.
- **W/o Mem:** We use the same final Executor checkpoint but disable its memory branch. Each new clip is conditioned only on the reference image.
- **Oracle ContextPlanner:** We replace the predicted actions with the ground-truth actions and provide them to the same Executor and memory in their reference order. Generation stops after the final reference action. The model does not receive any future ground-truth video frames.

Judges. We use Gemini 3.5 Flash (Google DeepMind, 2026) to evaluate the generated videos in Tables 2, 3, and 4. We use GPT-5.2 (OpenAI, 2025a) to evaluate the predicted text plans in Table 5. Although the two judges share some metric names, they evaluate different aspects of the output, as

[2] https://huggingface.co/maverickrzw/Video-CraftBench_v0.1

Table 13: Metric meanings on WorldGuide Bench. Video scores assess rendered execution; text scores assess the predicted procedure. Video-CraftBench uses goal-based video judging without a reference plan.

	Video judge	**Text judge**
Input	Rollout, goal, reference plan	Predicted plan, goal, reference plan
Plan Accuracy	Visibly achieved sub-goals	Semantic correctness and coverage
Order Score	Executed prerequisite order	Planned prerequisite order
Repeat/Skip	Unnecessary executions; omissions reduce Plan Accuracy	Redundant actions and missing sub-goals
Success	Final visible task completion	Plan would succeed if executed correctly

Gemini Task-Evaluation Prompt Structure

Role and objective: Evaluate visible procedural execution against the task goal, using the ground-truth plan as a reference for required sub-goals and dependencies.

Inputs: complete generated video; task description; ordered ground-truth steps $1, \ldots, N$.

Rules: watch the complete video; accept semantically equivalent actions, different step granularities, and valid reorderings of independent steps; credit a sub-goal only when its required effect is visibly achieved; complete every required field and return only valid JSON.

Metric instructions:

- **Plan Accuracy:** determine whether each reference step's required sub-goal is visibly achieved, including by equivalent actions; report $100 \times$ (achieved sub-goals$/N$).
- **Order Score:** among executed steps, report $100 \times$ (steps respecting prerequisites/executed steps); use zero when no step is executed.
- **Unnecessary Repeat:** count repeated executions that add no task progress; report $100 \times$ (unnecessary repetitions$/N$). Omissions reduce Plan Accuracy; valid changes in granularity are not repetitions.
- **Task Success:** return 1 only if the final visible state shows that the complete task is achieved; otherwise return 0.

Required JSON structure: per-step execution/correctness and reasons; observed step sequence and ordering violations; repeated steps and extra counts; binary task success, final-state description, supporting and opposing evidence; and a scalar summary of all four metrics.

Figure 7: Prompt structure for procedural video evaluation with the Gemini 3.5 Flash judge (temperature 0.0), used to produce the video-judge columns of Table 2. The judge receives the generated rollout, task description, and ground-truth plan with the `<|Task Completed|>` token stripped, and scores Plan Accuracy, Order Score, Repeat/Skip, and Task Success from visible execution alone; the ground-truth video is never shown.

summarized in Table 13. Their scores should therefore not be compared directly. Judge responses that cannot be parsed are excluded, and each score is averaged over the remaining valid judgments. We further validate the video evaluation with the independent human study in Appendix F, where WorldGuide is ranked first, followed by MiniMax-H3.

E.2 Video-Judge Metrics

The video judge (`gemini-3.5-flash`, temperature 0) receives the complete generated video, the task description, and, on WorldGuide Bench, the ordered reference plan with the completion token removed; it never sees the ground-truth video (Fig. 7). From its structured JSON output, we compute four metrics for video i with N_i reference steps. Equivalent actions, different step granularities, and valid reorderings of independent steps receive credit.

Plan Accuracy (↑). With $c_{ij} = 1$ if the sub-goal of step j is visibly achieved and 0 otherwise,

$$\mathrm{PlanAcc}_i = \frac{100}{N_i} \sum_{j=1}^{N_i} c_{ij}. \tag{10}$$

Missing, partial, or ambiguous effects receive no credit.

Order Score (↑). Of the N_i^{executed} observed steps, N_i^{ordered} respect their prerequisites:

$$\mathrm{Order}_i = 100 \, \frac{N_i^{\mathrm{ordered}}}{N_i^{\mathrm{executed}}}, \tag{11}$$

and $\mathrm{Order}_i = 0$ if no step is executed. Because a short but correctly ordered execution can score highly, Order is read together with Plan Accuracy.

Repeat/Skip (↓). With R_i repeated executions that add no task progress,

$$\mathrm{Repeat}_i = 100 \, \frac{R_i}{N_i}. \tag{12}$$

For the video judge, omitted steps are penalized by Plan Accuracy rather than by this term.

Task Success Rate (↑). $\mathrm{Success}_i = 1$ only if the final visible state achieves the complete task. Over N_{eval} valid judgments,

$$\mathrm{TSR} = \frac{100}{N_{\mathrm{eval}}} \sum_{i=1}^{N_{\mathrm{eval}}} \mathrm{Success}_i. \tag{13}$$

Plan Accuracy, Order, and Repeat/Skip are averaged over videos in the same way.

E.3 Planner Evaluation

Table 5 evaluates the action sequences predicted by the planner as text. The upper block evaluates Qwen2.5-VL and the ContextPlanner with different history lengths ($K \in \{1, 2, 3\}$). At each step, the planner predicts the next action from the task goal and the ground-truth clip–action history. The lower block evaluates the full WorldGuide loop, where the model runs autonomously using its own generated clips and predicted actions. The *w/o visual* variant follows the same loop but removes generated clips from the planner input.

We use GPT-5.2 (OpenAI, 2025a) to evaluate the predicted action sequences (Fig. 8). The judge receives the task description, predicted action sequence, and reference plan, but does not see the video. The reference plan defines the required sub-goals and their dependencies, but it is not treated as the only correct solution. The judge accepts equivalent actions, different step granularities, and valid changes in action order. We report four metrics. **Plan Accuracy** measures whether the predicted actions are semantically correct and cover the required sub-goals. **Order Score** measures whether prerequisite actions appear in the correct order. **Repeat/Skip** measures redundant actions and missing sub-goals. **Plan Success** measures whether the full predicted plan would achieve the task goal if all actions were executed correctly. Plan Success evaluates the action sequence itself, not whether the generated video stops at the correct visual state. Visual stopping is evaluated separately by the video judge. Stopping before the task is complete leads to failure in Task Success, while continuing after completion typically introduces unnecessary repeated actions.

E.4 Paired Alignment

Alignment compares each generated video with its paired ground-truth video using deterministic appearance and motion descriptors, independently of the LLM judges. Both videos are sampled at $B = 16$ uniformly spaced relative positions.

Appearance similarity (AS_i) is the mean cosine similarity between ℓ_2-normalized 16-bin RGB histograms at corresponding positions.

GPT-5.2 Planner-Evaluation Prompt Structure

Role and objective: You are an expert evaluator of procedural task plans. Determine whether the predicted action sequence provides a correct and complete procedure for accomplishing the given task.

Inputs: task description; predicted plan; ordered ground-truth reference plan.

Reference-plan interpretation: The ground-truth plan describes one valid procedure and is provided to clarify the task, required sub-goals, and action dependencies. Do not require exact step-by-step matching. Accept semantically equivalent actions, different step granularities, and alternative valid procedures that achieve the same task.

Evaluation rules:

- Evaluate the complete predicted plan before assigning scores.
- Judge semantic and procedural correctness, not lexical similarity.
- Do not penalize valid reordering of independent actions.
- A reference action is not considered skipped if its required effect is achieved through another valid action.
- Penalize missing prerequisites, invalid ordering, redundant actions, and actions that do not contribute to task completion.

Metrics:

- **Plan Accuracy:** correctness and completeness of the predicted procedure with respect to the task objective.
- **Order Score:** correctness of procedural dependencies and prerequisite ordering.
- **Repeat/Skip:** unnecessary repeated actions and uncovered task-critical sub-goals.
- **Plan Success:** whether executing the complete predicted plan correctly would accomplish the task.

Figure 8: Prompt structure for text-only planner evaluation with a GPT-5.2 judge, used to produce Table 5. The judge receives the task description, predicted plan, and ground-truth plan, with no video, and scores the procedural validity of the predicted text. The ground truth serves as a reference rather than a unique target, so semantically equivalent actions, orderings, and granularities are accepted.

State similarity (SS_i) compares the start, middle, and end frames. Each descriptor concatenates 8-bin per-channel RGB histograms with a 16-bin gradient-magnitude histogram of a 32×32 grayscale frame, and the cosine similarities are weighted toward the final state:

$$\mathrm{SS}_i = 0.20\, s_i^{\mathrm{start}} + 0.30\, s_i^{\mathrm{middle}} + 0.50\, s_i^{\mathrm{final}}. \tag{14}$$

Temporal alignment (TA_i) compares motion-intensity curves, the normalized mean absolute grayscale difference between adjacent samples. If both curves vary, $\mathrm{TA}_i = (\rho_i + 1)/2$ with Pearson correlation ρ_i; if both are constant, $\mathrm{TA}_i = 1$; if only one is constant, $\mathrm{TA}_i = \max(0,\, 1 - 20|\mu_i^{\mathrm{gen}} - \mu_i^{\mathrm{gt}}|)$ with mean intensities μ_i. The result is clipped to $[0, 1]$.

The final score is

$$\mathrm{Alignment}_i = 100\,(0.40\,\mathrm{AS}_i + 0.30\,\mathrm{SS}_i + 0.30\,\mathrm{TA}_i) \in [0, 100]. \tag{15}$$

Because it compares low-level appearance and motion, Alignment complements rather than replaces the task metrics.

E.5 VBench Video-Quality Metrics

We use the official VBench implementation (Zheng et al., 2025a) for the five dimensions in Table 14, reported on a 0–100 scale (higher is better). They measure perceptual and temporal quality, not procedural correctness.

Table 14: The five VBench dimensions reported in Tables 2 and 3, with the predictor used by the official implementation. All are perceptual or temporal quality measures; none assesses procedural correctness.

Dimension	Predictor	Measures
Aesthetic Quality (AQ)	LAION CLIP aesthetic	Mean frame-level aesthetic score
Imaging Quality (IQ)	MUSIQ	Technical image quality; penalizes blur, noise, and distortion
Dynamic Degree (DD)	RAFT optical flow	Percentage of videos classified as containing sufficient motion
Motion Smoothness (MS)	Frame interpolation	Agreement between observed and interpolated intermediate frames
Overall Consistency (OC)	ViCLIP	Video–text semantic consistency with the task description

Table 15: **Human evaluation of procedural video generation.** Task-completion scores on a 1–5 scale (higher is better), averaged over 72 ratings per model (18 tasks $\times$ 4 raters). CI: 95% bootstrap interval over tasks. Δ: WorldGuide's lead, with a two-sided Wilcoxon signed-rank test over the 18 per-task mean scores. Model labels follow Table 2.

Model	Mean $\pm$ SD	95% CI	Δ	p
WorldGuide	**2.750** $\pm$ 0.44	[2.67, 2.83]	–	–
MiniMax-H3	2.542 $\pm$ 0.50	[2.46, 2.62]	+0.21	0.022
LTX-2.5	2.472 $\pm$ 0.50	[2.36, 2.57]	+0.28	0.002
SkyReels	1.917 $\pm$ 0.28	[1.86, 1.97]	+0.83	$<$0.001
Bernini*	1.583 $\pm$ 0.50	[1.49, 1.68]	+1.17	$<$0.001
TempAct*	1.444 $\pm$ 0.50	[1.33, 1.54]	+1.31	$<$0.001
Cosmos-Predict2.5	1.000 $\pm$ 0.00	[1.00, 1.00]	+1.75	$<$0.001
HunyuanVideo-1.5	1.000 $\pm$ 0.00	[1.00, 1.00]	+1.75	$<$0.001
PhysAgent*	1.000 $\pm$ 0.00	[1.00, 1.00]	+1.75	$<$0.001

F Human Evaluation of Generated Videos

Setup. We conduct a human evaluation with four evaluators, nine models, and 18 task cases, resulting in 162 evaluated videos and 648 individual ratings. For each video, the evaluator first reads the task description and then watches the complete generated video. The evaluator assigns a task-completion score from 1 to 5.

Evaluators are asked to consider three aspects when assigning the score: (1) whether the necessary actions are visibly performed in a sensible order, (2) whether the essential steps are completed, and (3) whether the final task goal is visibly achieved. A higher score indicates more successful task completion.

Results. Table 15 reports each model's mean score with its standard deviation and a 95% confidence interval from 10,000 bootstrap resamples of the 18 tasks. WorldGuide achieves the highest score (2.750) and is ranked first by each of the four raters individually. Its lead is statistically significant over every compared model, including MiniMax-H3 ($p = 0.022$) and LTX-2.5 ($p = 0.002$), and remains significant after Holm correction for the eight comparisons. The raters agree strongly: Kendall's $W = 0.96$ for the model ranking and Krippendorff's $\alpha = 0.74$ (interval) over individual ratings. This study therefore provides an independent, statistically reliable validation of the automatic task-completion results.

G Additional Results

G.1 WorldGuide Bench Qualitative Comparisons

Figs. 9–15 compare WorldGuide (Ours) with 12 baselines, including Bernini, PhysAgent, and TempAct. The latter three use their own planning–execution loops; the other baselines receive reference action sequences. WorldGuide predicts actions from the goal and generated history. Frames in each row follow temporal order and illustrate action progression and visual continuity; complete rollouts are used for task scoring. Figure 11 extends the main fried-rice comparison (Fig. 5) with additional baselines. Complete generated videos for all compared methods can be viewed side by side at **https://worldguide-comparison.pages.dev**.

The comparisons show distinct limitations. In block construction, TempAct changes the supplied pieces and Bernini introduces floating blocks (Fig. 9); in paper folding, PhysAgent remains near a simple fold (Fig. 14). WorldGuide shows successive assembly and preparation steps, but also exhibits late texture drift in paper folding and an intermediate change in the sushi mold's contents (Figs. 14 and 15).

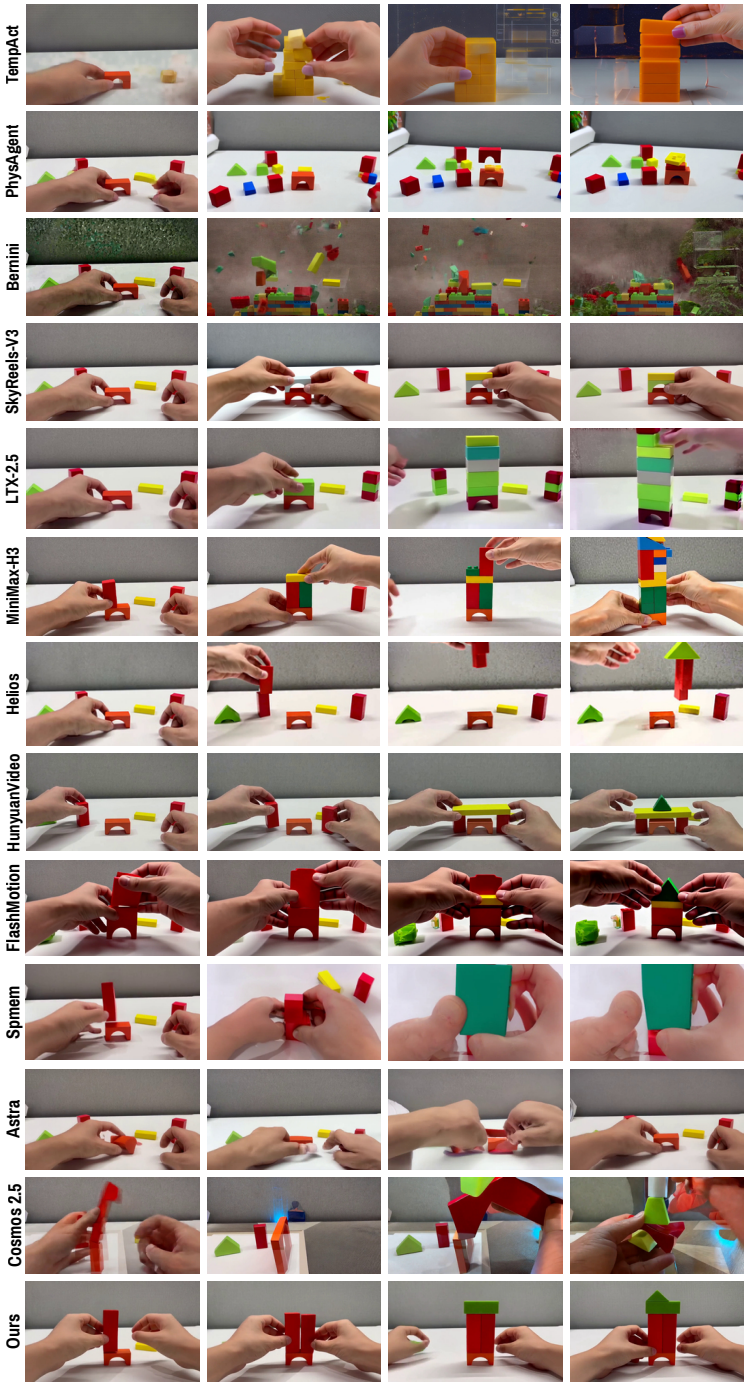

Figure 9: Building a tower from colorful wooden blocks. WorldGuide progressively adds upright pieces and a top to the arch base. TempAct substitutes different blocks, Bernini shows floating pieces and scene changes, and PhysAgent retains several small stacks.



Figure 10: Burrito assembly and folding. WorldGuide adds fillings before folding the tortilla. TempAct folds a tortilla without visible fillings, PhysAgent shifts to stacking round objects, and Bernini replaces the split-screen demonstration with a different scene.



Figure 11: Expanded fried-rice comparison with additional baselines beyond Fig. 5. WorldGuide shows ingredient additions and mixing in a pan; PhysAgent and TempAct focus on ingredient handling, while Bernini develops scene and object distortions.

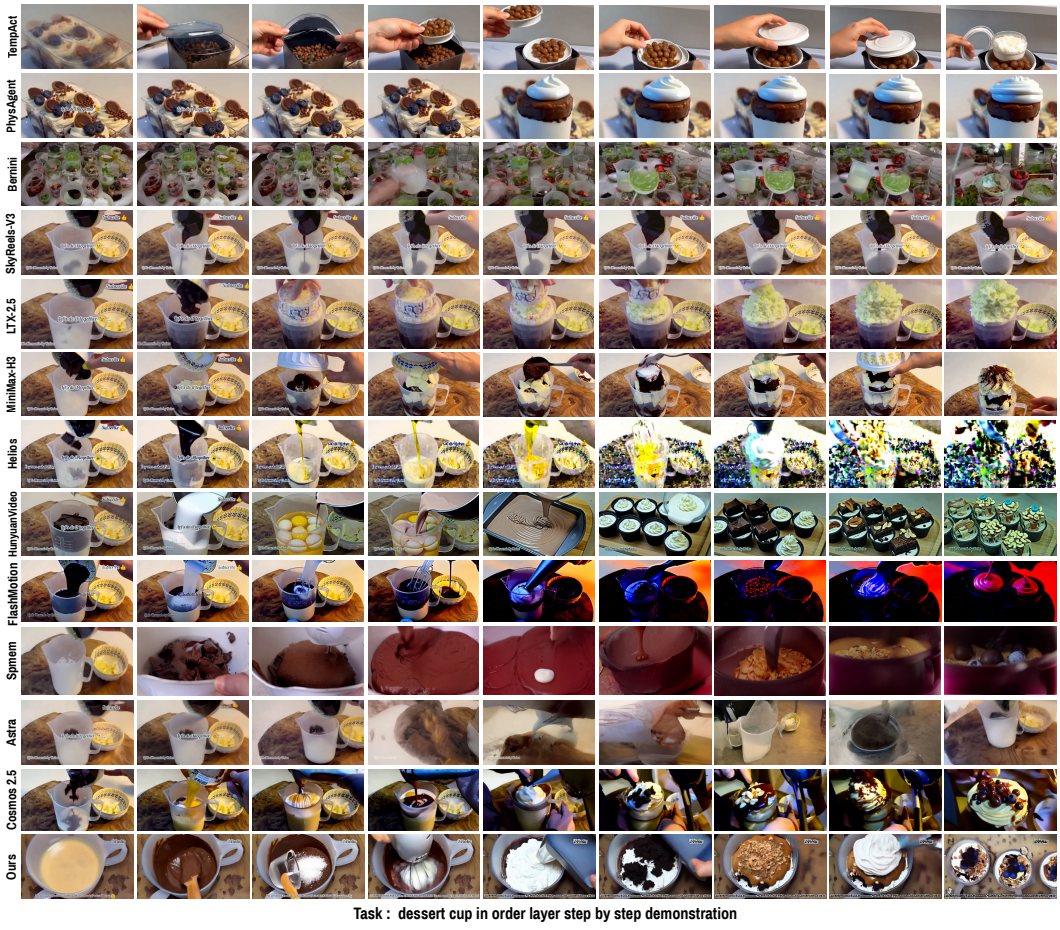

Figure 12: Layered dessert preparation. WorldGuide shows mixing, cream addition, and toppings. HunyuanVideo-1.5 and Cosmos-Predict2.5 also show preparation stages, while Helios develops severe distortion and SpMem changes the vessel and contents.



Figure 13: LEGO crocodile construction. WorldGuide progressively adds green pieces to an orange block structure. TempAct introduces large molded toy parts, while Bernini loses the tabletop arrangement.

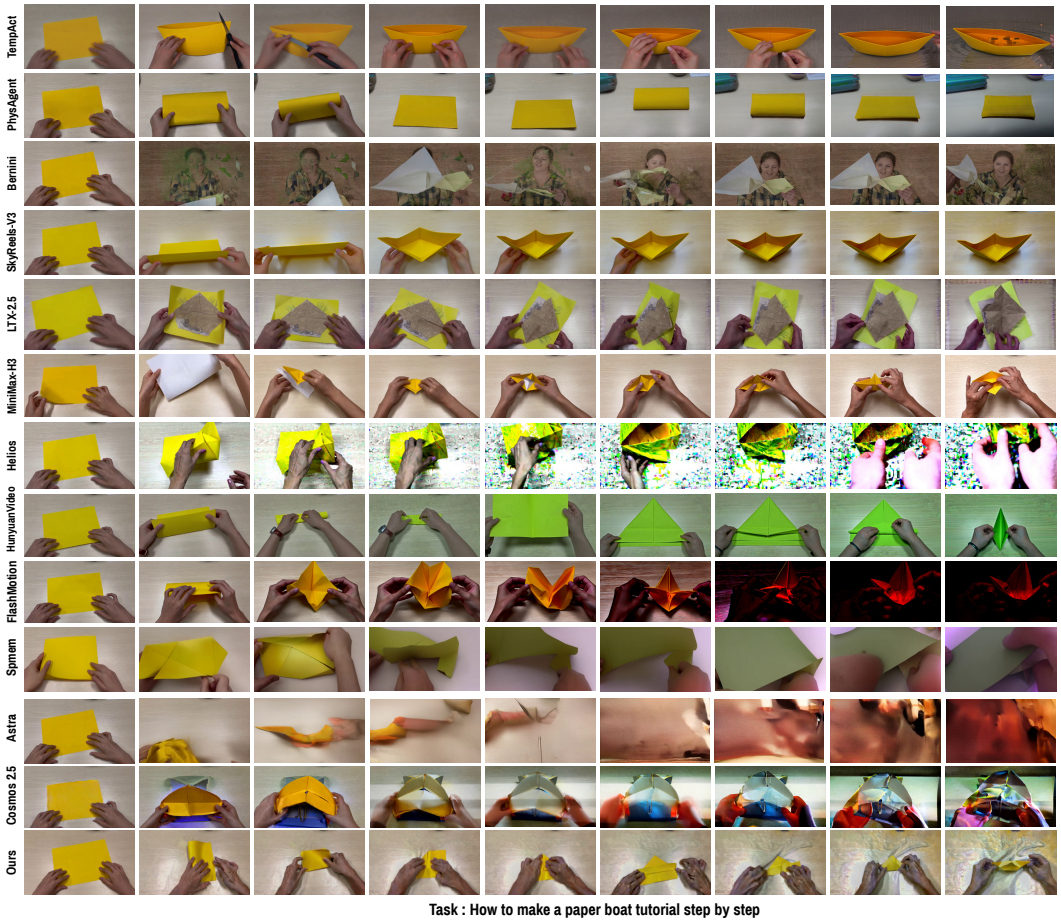

Figure 14: Paper-boat folding. WorldGuide shows successive folds but develops late hand and background distortion. TempAct produces a boat-shaped object while introducing scissors and changing the paper's appearance; PhysAgent remains near a simple rectangular fold.

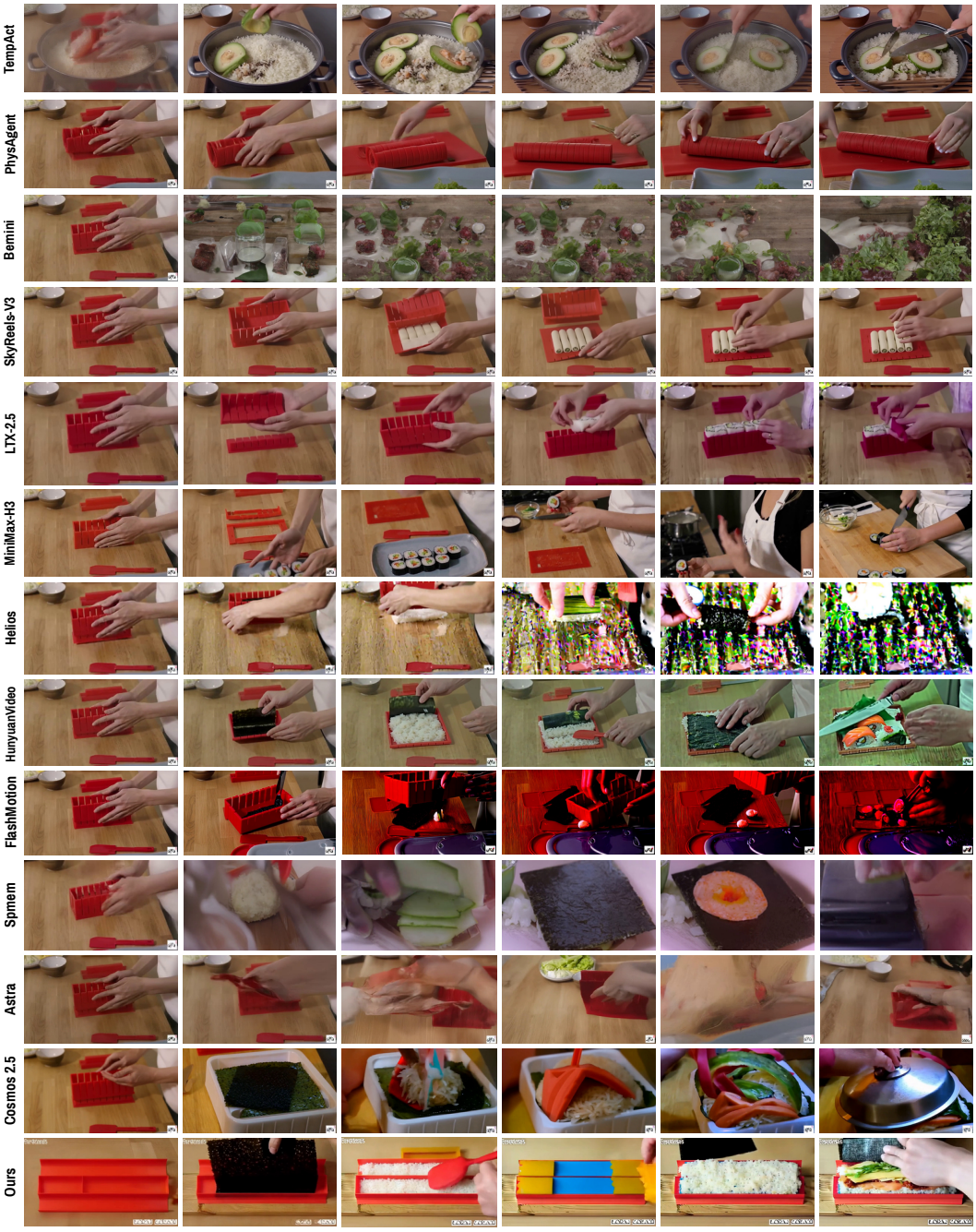

Figure 15: Sushi preparation. WorldGuide places seaweed, rice, and fillings into the red mold, but one intermediate sample shows colored block-like contents. TempAct switches to rice and avocado in a pan, while PhysAgent continues handling the red mold without visible ingredient addition.

Table 16: **Statistical significance of Task Success.** Wilson 95% intervals and two-sided two-proportion z-tests against WorldGuide. MiniMax-H3 receives the complete ground-truth action plan on WorldGuide Bench and only the task goal on Video-CraftBench.

Benchmark	Model	Task Succ. (%)	95% CI	p vs. WorldGuide
WorldGuide Bench	**WorldGuide**	**33.33**	[30.4, 36.3]	–
	MiniMax-H3 (reference actions)	29.90	[27.1, 32.8]	0.10
	WorldGuide (w/o Mem)	27.56	[24.9, 30.4]	0.006
	WorldGuide (w/o visual)	14.72	[12.6, 17.1]	<0.001
	WorldGuide (open-loop)	11.71	[9.8, 13.9]	<0.001
	WorldGuide (oracle ContextPlanner)	38.82	[35.8, 41.9]	0.011
Video-CraftBench	**WorldGuide**	**47.69**	[42.0, 53.4]	–
	MiniMax-H3	32.73	[27.6, 38.3]	<0.001
	TempAct*	24.47	[19.9, 29.7]	<0.001
	WorldGuide (w/o Mem)	29.78	[24.8, 35.2]	<0.001

G.2 Video-CraftBench Results

Table 3 reports the goal-conditioned results under the protocol of Appendix E.1, and Table 4 the corresponding memory ablation.

Figs. 16–22 compare the same 13 models on assembly and folding, with all models receiving an initial image and the task goal, without reference action captions. In the block-horse examples, WorldGuide adds pieces to the initial arch, while TempAct introduces sculpted horse components and Bernini changes the scene (Figs. 16 and 17). WorldGuide and MiniMax-H3 also form supported stacks in Fig. 18, while PhysAgent mostly rearranges loose pieces.

Folding remains challenging. TempAct introduces model-aircraft features in the airplane examples (Figs. 19 and 20). In Fig. 22, it cuts a star-edged shape, while MiniMax-H3 produces a crown-like object despite the airplane goal. WorldGuide shows successive folds, but its late paper appearance and scene texture deteriorate in Figs. 21 and 22.

G.3 Statistical Significance of Task Success

Each benchmark is evaluated with one rollout per test sample, so we quantify uncertainty over the test samples. Table 16 reports Wilson 95% intervals for Task Success (n=980 on WorldGuide Bench and n=294 on Video-CraftBench) and two-sided two-proportion z-tests against WorldGuide.

Ablations. Every design choice has a significant effect: removing closed-loop replanning, planner visual feedback, or memory lowers Task Success with $p \leq 0.006$, and the oracle ContextPlanner raises it ($p = 0.011$), confirming that better planning further improves completion.

Comparison with MiniMax-H3. On Video-CraftBench, where every model receives only the initial image and task goal, WorldGuide's lead of 14.96 points is highly significant ($z = 3.70$, $p < 0.001$). On WorldGuide Bench, the difference from MiniMax-H3 lies within the intervals ($p = 0.10$), but the two settings are not like-for-like: MiniMax-H3, a 33B model, receives the complete ground-truth action plan, whereas WorldGuide selects every action and the stopping point by itself. Matching a larger reference-conditioned model without reference actions is therefore the relevant result on this benchmark. Because both methods are evaluated on the same test samples, the unpaired z-test is conservative.

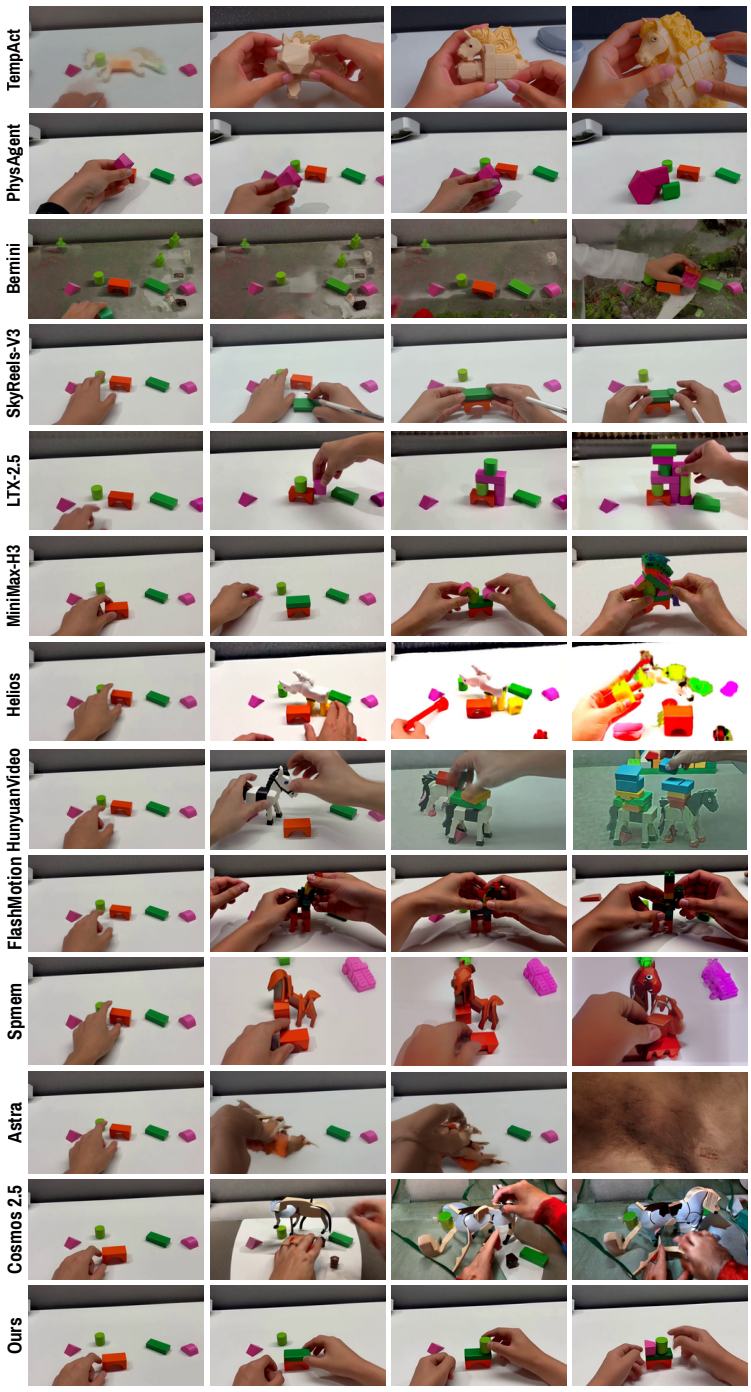

Figure 16: Video-CraftBench: building a horse from colored blocks. WorldGuide stacks pieces on the orange arch. TempAct introduces a sculpted horse, Bernini changes the scene, and PhysAgent groups loose blocks.

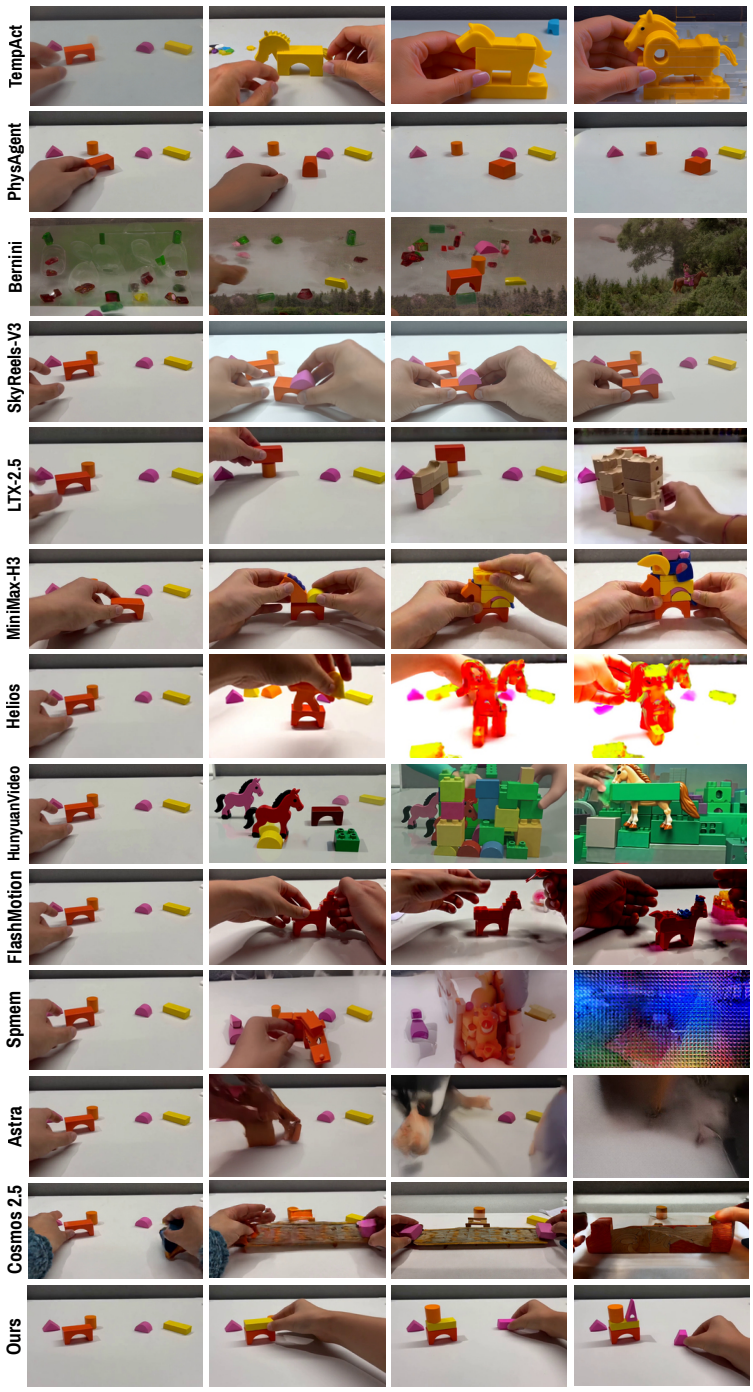

Figure 17: Video-CraftBench: a second block-horse example. WorldGuide adds a yellow bar and orange cylinder to the starting arch. MiniMax-H3 introduces shaped toy pieces, while LTX-2.5 substitutes different blocks.

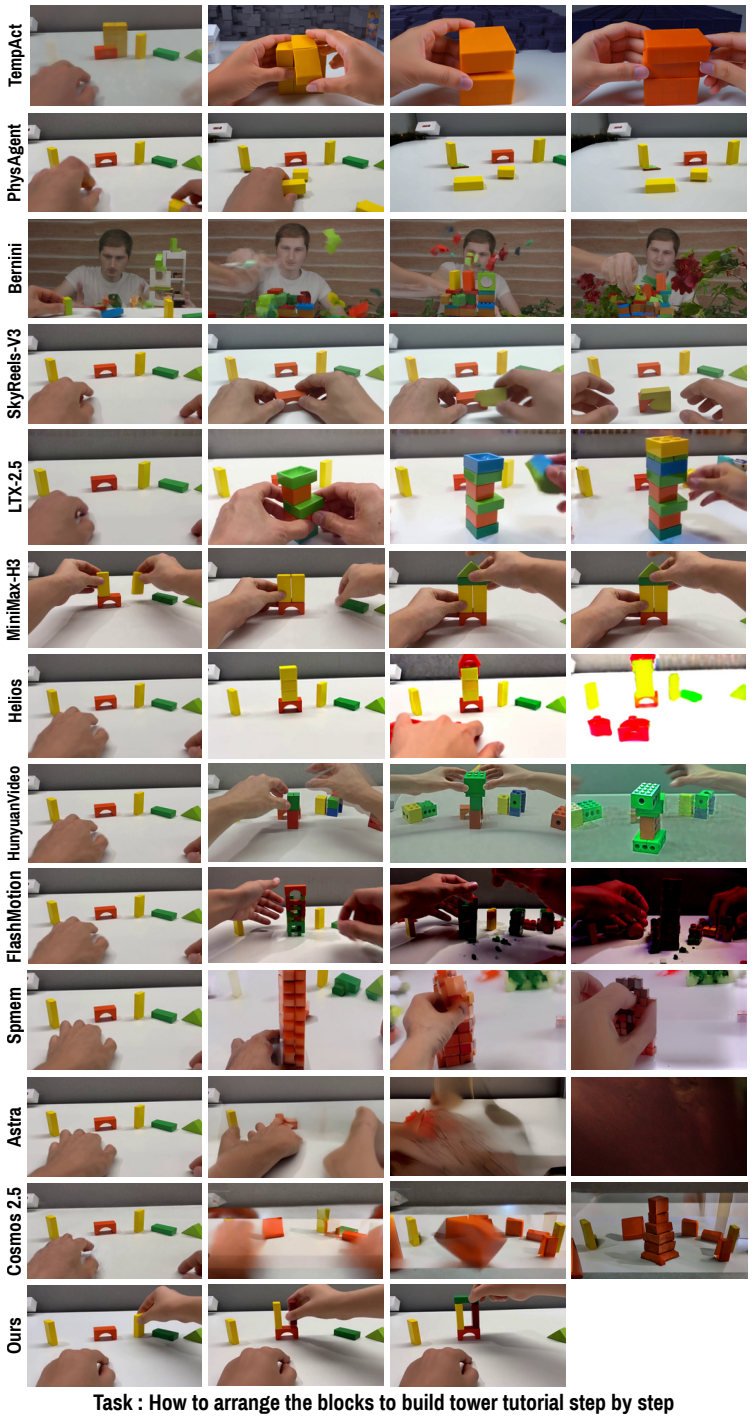

Figure 18: Video-CraftBench: building a block tower. WorldGuide and MiniMax-H3 form stacks supported by the initial arch. TempAct substitutes larger orange blocks, while PhysAgent mostly rearranges loose pieces. The blank cell contains no displayed frame.

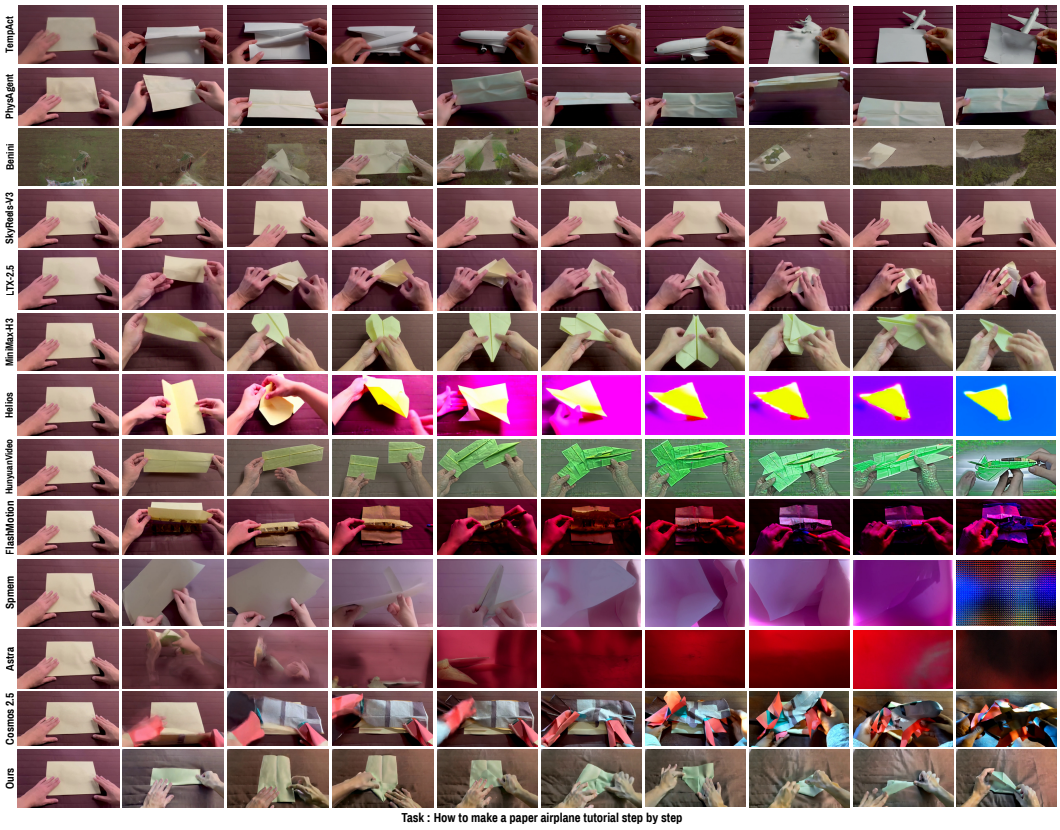

Figure 19: Video-CraftBench: folding a paper airplane. WorldGuide shows successive folds with late hand and paper distortion. MiniMax-H3 produces a pointed folded form, whereas TempAct introduces model-aircraft features such as engines and landing gear.

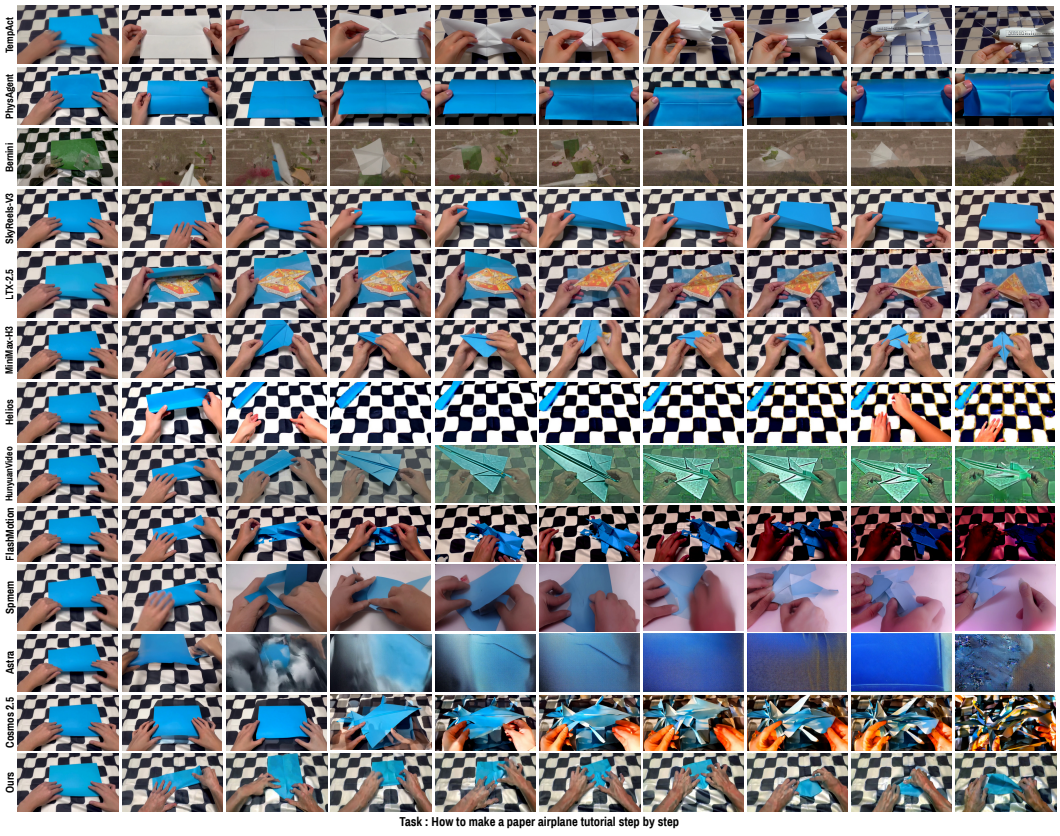

Figure 20: Video-CraftBench: folding a blue paper airplane. WorldGuide progresses through several folds, with visible late texture distortion. PhysAgent repeatedly handles a broad fold, while TempAct changes the blue sheet into a white model aircraft.



Figure 21: Video-CraftBench: folding an orange paper boat. WorldGuide shows successive folds with late changes in paper color and scene texture. TempAct substitutes a white boat-shaped object, while PhysAgent remains near a rectangular fold.

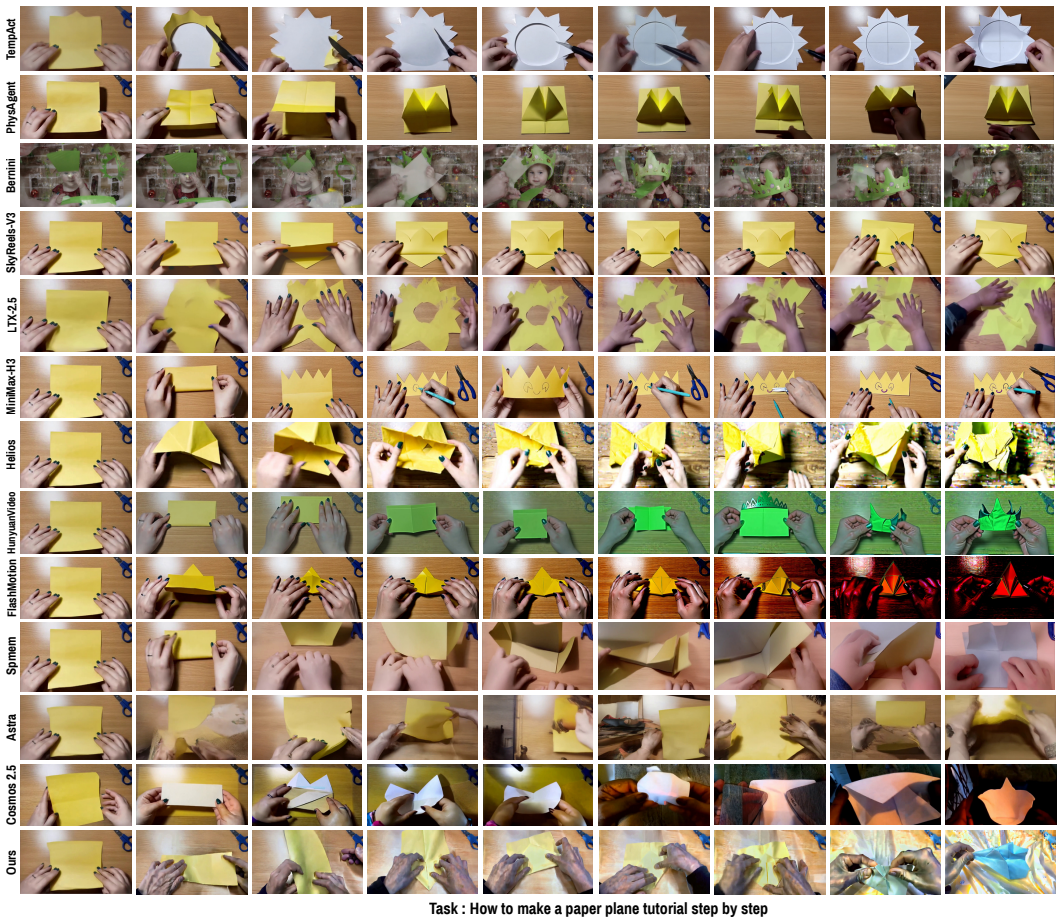

Figure 22: Video-CraftBench: folding a yellow paper plane. WorldGuide develops pointed folds but later changes the paper's color and surrounding texture. MiniMax-H3 produces a crown-like object and TempAct cuts a star-edged shape, both departing from the airplane goal.

H LIMITATIONS AND FAILURE ANALYSIS

Errors can compound across steps. The ContextPlanner may repeat an action, skip a prerequisite, or stop too early; the Executor may distort geometry or appearance even for a correct instruction. Because each generated clip becomes the next planning observation, replanning can respond to such errors but cannot undo an incorrect transition. The folding examples in Figs. 21 and 22 show correct action progression alongside late visual degradation.

The two modules are trained sequentially, without a joint task-completion objective (Appendix A). The bounded memory compresses older frames, which can discard details needed later, and enabling memory also increases repetition (Table 4). Outcome verification, explicit recovery from failed steps, and joint training are natural next steps.