SPIRAL: Self-Evolving Action-Conditioned Video Generation via Reflective Planning Agents

Yu Yang[1,2,*] **Yue Liao**[2,*] **Jianbiao Mei**[1,*] **Baisen Wang**[3,*] **Jiangning Zhang**[1]
Xiangtai Li[4] **Liang Lv**[5] **Hanlin Chen**[2] **Yong Liu**[1,†] **Shuicheng Yan**[2] **Gim Hee Lee**[2]
[1] Zhejiang University [2] National University of Singapore
[3] The High School Affiliated to Renmin University of China
[4] Nanyang Technological University [5] Wuhan University
https://yuyang-cloud.github.io/spiral/

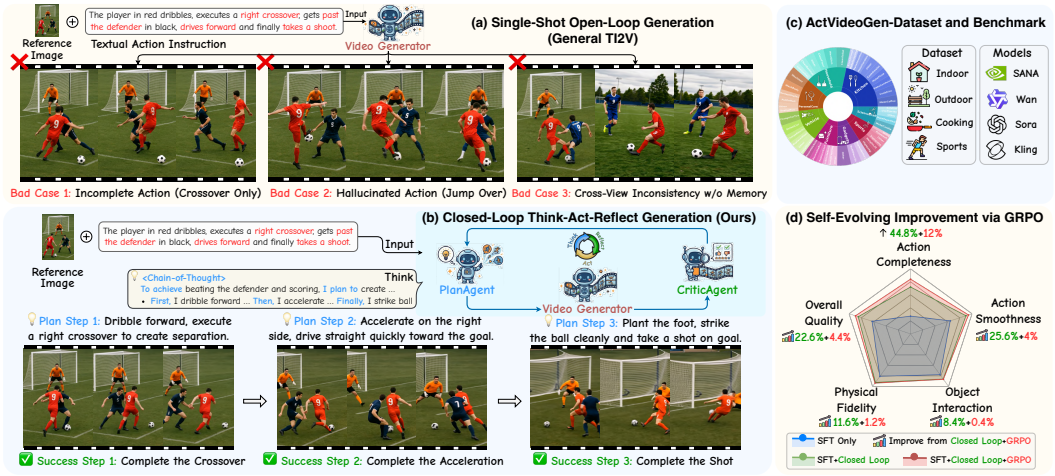


Figure 1: **Long-Horizon Action-Conditioned Video Generation**: *Challenges and Solution.* (a) General TI2V is single-shot and open-loop, often causing incomplete actions and hallucinated motions. (b) We propose a *closed-loop think-act-reflect* framework for iterative planning, generation, and verification. (c) We introduce the ActVideoGen-Dataset and Benchmark for task-specific experiments. (d) Our closed-loop design enables self-evolving, continually improving video generation quality.

Abstract

Long-horizon action-conditioned video generation aims to synthesize temporally coherent videos that follow complex action instructions over extended horizons, requiring procedural ordering, persistent action execution, and scene consistency beyond conventional TI2V's short-term fidelity. Existing single-shot video generation models typically operate in an open-loop manner, leading to incomplete action execution, hallucinated motions, and temporal drift. To address this, we propose SPIRAL, a closed-loop framework that performs *S*equential *P*lanning and *I*terative *R*eflection for *A*ction-conditioned *L*ong-horizon video generation. Specifically, SPIRAL instantiates a think-act-reflect process: a PlanAgent decomposes high-level goals into sub-actions, which condition a VideoGenerator to synthesize each segment alongside a ContextMemory for visual persistence, while a CriticAgent evaluates intermediate video segments to provide corrective feedback for iterative refinement. This closed-loop design further supports self-evolution by utilizing PlanAgent-proposed actions and CriticAgent-derived rewards for GRPO-based post-training under curriculum learning to enhance the video generator's

* equal contribution
† corresponding author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

long-horizon consistency. Moreover, we introduce ActVideoGen-Dataset for task-specific training, and establish ActVideoGen-Bench as a dedicated evaluation suite for measuring action fidelity and temporal coherence. Experiments across multiple TI2V backbones alongside the self-evolving strategy show consistent gains on ActVideoGen-Bench and VBench, demonstrating the effectiveness of SPIRAL.

1 Introduction

Recent advances in video generation [1–4] have enabled increasingly realistic and coherent visual synthesis across diverse scenes and motions, paving the way for controllable motion simulation [5–9] and dynamic modeling [10–13]. Beyond short-clip generation, a pivotal frontier is *long-horizon action-conditioned video generation*. Unlike conventional text-and-image-to-video (TI2V) tasks that focus on globally plausible content and short-term motion, this setting demands sustained action execution, motion causality, and consistent object-tool interaction over extended temporal horizons, making it a formidable challenge.

Despite their success in short-term synthesis, current TI2V models struggle to sustain such complex long-term execution. As illustrated in Fig. 1(a), this paradigm shift exposes several critical challenges: *incomplete action execution*, where multi-stage actions are truncated or only partially completed; *hallucinated motions*, where generated motions deviate from the intended instructions without planning and correction; and *long-horizon temporal incoherence*, where the absence of memory or state tracking causes object drift and scene inconsistency. These failures ultimately stem from the ***single-shot open-loop*** nature of general TI2V generation, where the lack of task decomposition, intermediate verification, and corrective feedback allows errors to compound over time.

To address these challenges, we propose SPIRAL, an agentic framework for long-horizon action-conditioned video generation as shown in Fig. 1(b). Rather than relying solely on a single-shot generator, SPIRAL serves as a closed-loop harness over the base video generator, augmenting it with explicit planning, verification, memory, and feedback. Specifically, a *PlanAgent* decomposes a high-level goal into step-wise sub-actions, a *VideoGenerator* synthesizes each segment conditioned on the current sub-action and historical context, a *CriticAgent* evaluates intermediate videos and provides corrective feedback, and a *Long-Horizon Memory* module preserves visual and semantic context. Together, these components instantiate a ***closed-loop think-act-reflect*** mechanism that improves action completeness through sequential planning, reduces hallucinations through iterative reflection, and mitigates long-horizon drift through memory-aware generation.

Furthermore, SPIRAL extends its utility beyond an inference-time harness to a ***trainable self-evolving mechanism***. Its closed-loop design naturally yields structured action plans and critic-derived rewards, enabling GRPO-based post-training to enhance the video generator. Concretely, the PlanAgent provides progressively challenging action instructions, and the video generator synthesizes candidate video segments. The CriticAgent then evaluates the action-video pairs in terms of action fidelity and temporal consistency, providing critic-based rewards that directly optimize the generator for improvement. By integrating a *curriculum learning* strategy that incrementally scales action complexity and temporal horizons, it drives the generator to seamlessly evolve from executing short atomic actions to mastering long procedural behaviors, leading to persistent performance gains.

Beyond the framework design, long-horizon action-conditioned video generation also requires task-specific supervision and evaluation protocols. We further introduce ***ActVideoGen-Dataset*** (Fig. 1(c)), comprising 24,616 tasks and 118,156 step-level annotations. Each sample is organized around a goal-level description, a structured CoT rationale, step-wise actions, and aligned video segments, providing supervised adaptation for both the PlanAgent and the video generator. Furthermore, we present ***ActVideoGen-Bench***, an evaluation suite tailored to this setting with hierarchical difficulty levels and multi-dimensional metrics, emphasizing action quality and long-term consistency.

Extensive experiments demonstrate that integrating SPIRAL across diverse TI2V backbones consistently improves long-horizon action fidelity and temporal consistency. As depicted in Fig. 1(d), compared to the open-loop baseline [14], SPIRAL's closed-loop framework improves overall action fidelity by 22.6%, while GRPO-based self-evolution yields an additional 4.4% gain. These findings confirm that SPIRAL serves not only as an effective inference-time harness, but also as a practical self-improving framework for long-horizon action-conditioned video generation.

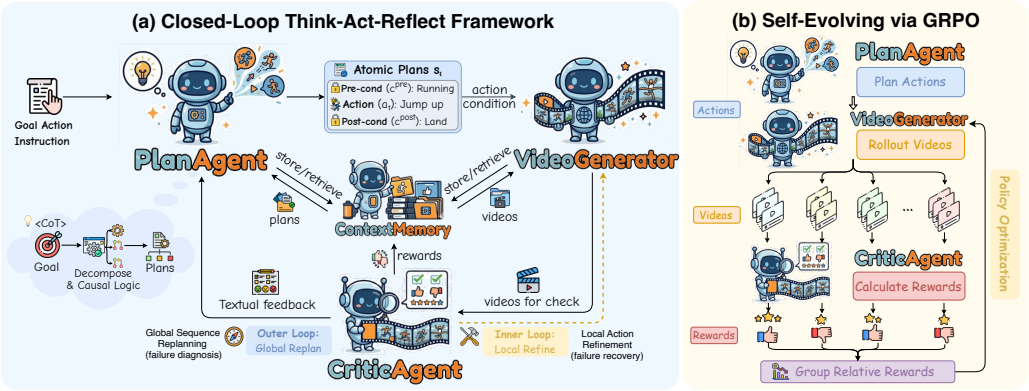


Figure 2: **SPIRAL Overview.** (a) **Closed-Loop Framework**: PlanAgent decomposes abstract goals into atomic plans for action-conditioned video generation, CriticAgent evaluates videos and triggers dual-level (inner/outer) refinement feedback (Fig. 11). (b) **Self-Evolving via GRPO**: guided by PlanAgent, VideoGenerator produces rollouts and optimized via CriticAgent rewards (Fig. 12).

Our key contributions are summarized as follows:

- We propose SPIRAL, a closed-loop, agentic framework for long-horizon action-conditioned video generation that augments a video generator with planning, reflective feedback, and context memory.
- We introduce a GRPO-based evolving strategy, leveraging planning and verification signals to continuously optimize the video generator for self-improvement beyond inference-time correction.
- We present ActVideoGen-Dataset and ActVideoGen-Bench, a task-specific dataset and benchmark designed for training and rigorously evaluating long-horizon action-conditioned video generation.

2 Related Work

Recent visual and video generation research is evolving from single-shot synthesis toward agentic, closed-loop paradigms driven by chain-of-thought reasoning and iterative planning [15–17]. Existing frameworks leverage structured workflows [18, 19], tool execution [20], and "thinking-while-generating" designs [21–23] to enhance controllability, grounded verification, and self-correction. While these works address complex visual trajectories, they often operate in a single-shot manner without progressive self-improvement. SPIRAL bridges this gap by coupling a closed-loop think-act-reflect architecture with GRPO-based self-evolution, converting agentic feedback into sustained action grounding over long temporal horizons. Additional related works are detailed in Appendix A.

3 Method

3.1 Overview

Problem Formulation. We formulate long-horizon, action-conditioned video generation as a Markov Decision Process (MDP) defined by the tuple $\mathcal{T} = \langle g, \mathcal{S}, \mathcal{V} \rangle$. Given a global goal g, the system determines structured action steps $\mathcal{S} = \{s_1, \ldots, s_T\}$ that guide the synthesis of a coherent video trajectory $\mathcal{V} = \{v_1, \ldots, v_T\}$ to accomplish the specified goal. At each timestep t, the framework jointly generates the plan s_t and the video segment v_t conditioned on the historical context, maximizing the cumulative rewards reflecting action completion and goal achievement.

Overall Framework. To ensure long-term consistency and controllability, we propose a hierarchical, closed-loop framework in Fig. 2(a) that operates iteratively: **(1)** a **PlanAgent** (π_{plan}), serving as the high-level policy, employs CoT reasoning to decompose global goal g into structured atomic plans s_t; **(2)** a **VideoGenerator** (π_{gen}) functions as the execution policy, synthesizing video segments v_t conditioned on the plan s_t and context memory $\mathcal{M}_{t-1}$; **(3)** a **CriticAgent** ($\mathcal{C}$) evaluates the action-video alignment via reward $r_t = \mathcal{C}(v_t, s_t)$, triggering refinement feedback upon detecting failures; and **(4) ContextMemory** accumulates successful action-video transitions as $\mathcal{M}_t = \mathcal{M}_{t-1} \cup \{(s_t, v_t)\}$ to preserve global coherence. Collectively, this cycle instantiates a ***think-act-reflect*** loop (detailed in Fig. 11), advancing beyond single-shot generation toward a robust plan-generate-verify formulation.

3.2 PlanAgent: Structured Reasoning and Planning

Directly conditioning a video generator on an abstract goal g in a single-shot manner often leads to incomplete execution or hallucinations. To mitigate this, we employ a **PlanAgent** (parameterized by a VLM π_{plan}) to explicitly decompose high-level goals into executable action sequences.

Formally, given a goal g and context memory $\mathcal{M}$, the agent employs Chain-of-Thought (CoT) reasoning to decompose the instruction, establish causal logic, and synthesize structured plans:

$$\mathcal{S} = \{s_1, \ldots, s_T\}, \quad \mathcal{S} \sim \pi_{\text{plan}}(\cdot \mid g, \mathcal{M}) \tag{1}$$

Each atomic plan is defined as a tuple $s_t = (a_t, c_t^{pre}, c_t^{post})$, where a_t denotes the language action instruction, while c_t^{pre}, c_t^{post} represent physical pre- and post-conditions, respectively. For instance, for an action a_t = 'jump over a hurdle', the agent specifies c_t^{pre} = 'running' and c_t^{post} = 'landed'. Crucially, this CoT paradigm compels the agent to explicitly reason about causal dependencies, ensuring that planned actions are physically feasible, not merely semantically relevant.

We instantiate PlanAgent with Qwen3-VL-8B [24] and execute a hierarchical training to enhance its causal planning capacity: we first conduct *Instruction Tuning (IT)* on our curated ActVideoGen-Dataset (Sec. 4.1) to instill structured CoT-based planning capabilities, followed by *Direct Preference Optimization (DPO)* to enforce alignment with physical reality and temporal logic. This regimen empowers the agent to not only capture causal dependencies but also mitigate logical hallucinations in complex tasks. Implementation details are provided in Appendix B.1.

3.3 VideoGenerator: Action-Conditioned Video Generation

The **VideoGenerator** (instantiated as a video diffusion model π_{gen}) serves as the execution policy, translating each atomic plan s_t into a video segment. For long-term consistency, synthesizing the current video segment v_t is conditioned on both action instruction a_t and historical memory $\mathcal{M}_{t-1}$:

$$v_t \sim \pi_{\text{gen}}(\cdot \mid a_t, \text{Encoder}(\mathcal{M}_{t-1})) \tag{2}$$

Specifically, instruction a_t specifies the intended motion or interaction, while historical memory $\mathcal{M}_{t-1}$, stored as visual keyframes or latent KV-caches, provides contextual guidance. The generation process is executed in a streaming manner, with each video segment corresponding to a single action step, and context memory preserving motion continuity over extended horizons.

Crucially, our framework supports plug-and-play integration of arbitrary video generators. To endow short-clip generators with long-horizon synthesis capabilities, we adopt a *Streaming Long-Tuning* strategy by performing supervised fine-tuning on our ActVideoGen-Dataset, empowering diverse T2V and I2V backbones to faithfully execute step-wise action instructions over extended durations. Appendix B.2 details adapting SVI [25] for long-horizon action-conditioned video generation.

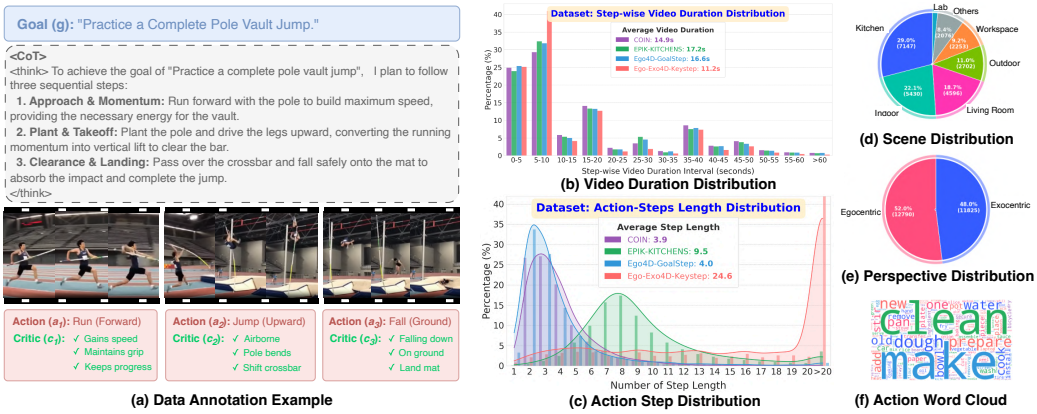
3.4 CriticAgent: Reward and Closed-Loop Feedback

Despite structured planning, open-loop generation remains susceptible to execution failures, such as missed actions, post-condition violations, and physical inconsistencies. To address this, we employ a **CriticAgent** (parameterized by a VLM $\mathcal{C}$) to enforce closed-loop verification and feedback.

Specifically, the atomic plan s_t generated by PlanAgent contains action instructions and physical pre- and post-conditions, which serve as critic anchors for CriticAgent to verify whether the generated video executes the intended action and satisfies the required conditions. Consequently, for each video segment v_t, CriticAgent assesses its alignment with plan s_t across five dimensions: *action adherence, object interaction, goal achievement, temporal coherence, and physical realism*, producing a scalar reward $r_t \in [0, 1]$ to quantify generation quality, alongside textual feedback f_t for diagnosis:

$$r_t, f_t = \mathcal{C}(v_t, s_t) \tag{3}$$

Leveraging these signals, we implement a dual-level feedback mechanism: **(1) Inner Loop (Local Refinement):** If r_t falls below a predefined threshold τ, indicating minor artifacts or incomplete action execution, the feedback f_t is used to refine the instruction a_t (e.g., emphasizing unmet post-conditions) for immediate regeneration; **(2) Outer Loop (Global Replanning):** If generation fails consecutively for K attempts, implying an infeasible plan (e.g., missing conditions or incorrect action ordering), the failure propagates to the PlanAgent to trigger the replanning of trajectory $\mathcal{S}$ from step t.



(a) Data Annotation Example (b) Video Duration Distribution (c) Action Step Distribution (d) Scene Distribution (e) Perspective Distribution (f) Action Word Cloud

Figure 3: **Overview of ActVideoGen-Dataset.** (a) An annotation example comprising Goal, CoT, and step-wise video-action-critic tuples. (b-f) Detailed dataset distributions across five key attributes.

To ensure robust verification, the CriticAgent undergoes a two-stage training regimen: first, we distill judgments from large VLMs (e.g., Gemini-3-Pro) via *Supervised Fine-Tuning (SFT)*; subsequently, we refine the model via *Pairwise Reward Modeling (RM)* using a Bradley-Terry objective to enhance discriminative accuracy. Detailed implementation protocols are provided in Appendix B.3.

3.5 Self-Evolving: Closed-Loop GRPO Optimization

While inference-time feedback mitigates immediate errors, we aim to internalize such corrections to permanently enhance VideoGenerator. To this end, we propose a **Self-Evolving** strategy based on GRPO, enabling closed-loop reinforcement learning driven by PlanAgent-proposed actions and CriticAgent-derived rewards to continuously optimize the VideoGenerator's performance.

Formally, initialized from supervised-finetuned π_{gen}, VideoGenerator policy π_θ undergoes iterative refinement leveraging stochastic actions from π_{plan} and rewards from $\mathcal{C}$ (Fig. 12). For action step s_t, we sample G video trajectories $\{v_{t,i}\}_{i=1}^{G}$ via previous policy $\pi_{\theta_{\text{old}}}$. CriticAgent evaluates each sample, assigning reward set $\{r_i\}_{i=1}^{G}$. We then compute advantage A_i using group-wise normalization:

$$A_i = \frac{r_i - \text{mean}(\{r_1, \ldots, r_G\})}{\text{std}(\{r_1, \ldots, r_G\}) + \delta} \tag{4}$$

The VideoGenerator parameters θ are updated by maximizing the GRPO objective:

$$\mathcal{J}(\theta) = \mathbb{E}_{\substack{s_t \sim \pi_{\text{plan}}(\cdot|g) \\ v_{t,i} \sim \pi_{\theta_{\text{old}}}(\cdot|s_t)}} \left[\frac{1}{G} \sum_{i=1}^{G} \min \left(\rho_{t,i} A_i, \text{clip}(\rho_{t,i}, 1-\epsilon, 1+\epsilon)\, A_i \right) - \beta\, D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{gen}}) \right] \tag{5}$$

where $\rho_{t,i} = \frac{\pi_\theta(v_{t,i}|s_t)}{\pi_{\theta_{\text{old}}}(v_{t,i}|s_t)}$ is the importance sampling ratio between π_θ and $\pi_{\theta_{\text{old}}}$, $D_{\text{KL}}(\cdot)$ denotes KL divergence, and ϵ, β are hyperparameters for policy clipping and regularization strength, respectively.

This closed-loop optimization effectively distills PlanAgent's reasoning and CriticAgent's verification into the generator's execution policy. Moreover, this framework naturally facilitates *Curriculum Learning*. By progressively scaling action complexity and temporal horizons, VideoGenerator evolves from generating atomic actions to executing complex, long-horizon procedural tasks, yielding sustained improvements. Additional implementation details are provided in Appendix B.5.

4 Dataset and Benchmark

4.1 ActVideoGen-Dataset Construction

To support task-specific training, we introduce **ActVideoGen-Dataset**, providing the structured $\langle g, \text{CoT}, \mathcal{S}, \mathcal{V} \rangle$ supervision. It integrates global reasoning with step-wise actions grounded in video segments, providing a comprehensive foundation for long-horizon planning and video generation.

Data Initialization. To ensure diverse real-world task coverage, we aggregate videos from 4 datasets: Ego4D [26, 27], Ego-Exo4D [28], COIN [29], and EPIC-KITCHENS [30]. These sources

encompass a broad range of procedural activities across egocentric and exocentric viewpoints. To eliminate redundant or irrelevant content, we use an automated shot-detection pipeline to filter out low-quality clips, retaining only segments that are strictly aligned with significant action boundaries for annotation.

Hierarchical Annotation Pipeline. We develop an automated pipeline to convert raw video clips into structured annotations $\langle g, \text{CoT}, \mathcal{S}, \mathcal{V} \rangle$, effectively bridging visual signals with explicit step-wise actions. The process consists of two strategic stages. *Stage 1: Step-wise Action Annotation.* This stage instantiates structured step tuple $s_t = (a_t, c_t^{pre}, c_t^{post})$ from video segments. Taking the video clip v_t as input, we leverage a VLM (e.g., GLM-4.5V [31]) to identify fine-grained action instruction a_t (specifying verb-object-tool interactions) and infer physical state transitions (c_t^{pre}, c_t^{post}). This yields an action sequence $\mathcal{S} = \{s_1, \ldots, s_T\}$ explicitly modeling physical dependencies. *Stage 2: CoT Planning Generation.* We employ an LLM (e.g., GPT-5.1) to orchestrate global reasoning logic based on goal g and the action sequence $\mathcal{S}$. By synthesizing a reasoning narrative CoT (e.g., `<think> ... </think>`), the model transforms disjoint steps into a coherent, logically grounded plan, providing intermediate supervision for long-horizon CoT-based planning.

Quality Verification and Human Alignment. To ensure dataset integrity, we implement a hierarchical pipeline integrating VLM-based filtering with human verification. Specifically, Qwen3-VL-235B [24] serves as the verifier, filtering out samples with suboptimal video-text alignment. Subsequent human verification on a random subset yields a 93% agreement rate between annotations and corresponding videos, confirming the annotations faithfully capture action semantics and causal dependencies. This rigorous validation ensures a reliable foundation for training planning agents and video generators. Comprehensive verification details are provided in Appendix C.

Dataset Statistics and Analysis. ActVideoGen-Dataset contains 24,616 tasks and 118,156 step-level action-video pairs. As shown in Fig. 3, the dataset is characterized by its diversity in scenes (indoor/outdoor), perspectives (ego/exocentric), and complexity (varying durations/steps). This scale and structure make ActVideoGen-Dataset ideal for training planning-oriented agents and action-conditioned video generation.

4.2 ActVideoGen-Bench Construction

To systematically evaluate long-horizon generation capacity of diverse T2V and I2V backbones and our closed-loop SPIRAL framework, we introduce **ActVideoGen-Bench**, a comprehensive evaluation suite featuring hierarchical difficulty levels and multi-dimensional metrics.

Eval Prompt Suite. We curate 300 evaluation prompts stratified into three difficulty levels to probe long-horizon generalization: *Simple* (1-3 action steps, $<$ 20s video duration), *Medium* (3-5 steps, 20-40s), and *Hard* ($>$ 5 steps, $>$ 40s). To ensure robust coverage, the suite spans diverse *scenes* (indoor, outdoor), *perspectives* (egocentric, exocentric), and *procedural activities* (e.g., sports, cooking).

Evaluation Metrics. We adopt a hybrid evaluation protocol, reporting *Static Quality* and *Dynamic Quality* via VBench [32] to assess general visual fidelity. However, given the focus on procedural action-conditioned video generation, such generic metrics are insufficient to determine whether intended actions are correctly executed. Consequently, the primary evaluation target is *Action Quality*. To this end, we introduce a multi-agent action evaluator utilizing GPT-5 as the reasoning engine, performing comprehensive assessments via multi-agent collaboration and cross-verified chain-of-query reasoning. Specifically, this system assesses Action Completeness, Action Smoothness, Object Interaction, and Physical Fidelity. Comprehensive details are provided in Appendix D.

5 Experiments

Implementation Details. (1) PlanAgent is built upon Qwen3-VL-8B and trained on ActVideoGen-Dataset with LoRA in two stages: Instruction Tuning (IT) to induce CoT reasoning and structured plan formatting, followed by Direct Preference Optimization (DPO) aligning generated plans with logical dependencies. **(2) CriticAgent** undergoes two-stage training: initially performing Supervised Fine-Tuning (SFT) by distilling judgments of a strong model (Gemini-3-Pro) on videos from diverse baselines (e.g., Sora, CogVideoX [33], SkyReels [34], VideoVerse [35]), and subsequently conducting Reward Modeling (RM) using the Bradley-Terry objective on the GAIA dataset [36] for action

Table 1: **PlanAgent Performance on EgoPlan-Bench.** Success rates (%) across In-Domain, Out-of-Domain, and All splits. Instr.: Instruction Tuning; Pref.: Preference Alignment.

Method	Base Model	Training Stage		EgoPlan-Bench		
		Instr. Tuning	Pref. Align.	In-Domain	Out-of-Domain	All
GPT-4V	-	-	-	38.40	36.90	37.98
GPT-5.1	-	-	-	55.08	54.37	54.78
SEED-LLaMA [37]	LLaMA2-Chat-13B	-	-	-	-	29.93
DeepVideo-R1 [38]	Qwen2.5-VL-7B	-	-	52.00	55.70	-
SEED-Bench-RL [39]	Qwen2-VL-7B	-	-	46.01	50.16	-
GRPO-CARE [40]	Qwen2.5-VL-7B	-	-	57.00	**57.00**	57.00
Video-LLaMA [41]	LLaMA2-Chat-7B			27.88	30.44	28.58
Video-LLaMA [41]		✓		52.14	40.52	48.94
Video-LLaMA [41]		✓	✓	54.65	44.42	51.83
PlanAgent (Ours)	Qwen3-VL-8B			36.60	35.72	35.81
PlanAgent + Mem.				44.68	43.31	43.63
PlanAgent + Mem.		✓		56.49	50.17	53.29
PlanAgent + Mem.		✓	✓	**62.46**	54.30	**58.72**

Table 2: **CriticAgent Performance on VideoGen-Reward Bench.** Accuracy (%) across three dimensions: Visual Quality (VQ), Motion Quality (MQ), Text Alignment (TA), as well as Overall Quality.

Method	Training Stage		VideoGen-Reward Bench							
	SFT	RM	VQ Acc.		MQ Acc.		TA Acc.		Overall Acc.	
			w/ Ties	w/o	w/ Ties	w/o	w/ Ties	w/o	w/ Ties	w/o
VideoPhy2 [42]	✓		-	-	-	-	37.04	22.14	30.75	26.41
AIGVE [43]	✓		38.05	30.80	-	-	30.76	11.66	37.09	37.08
LiFT-Critic [44]	✓		47.53	55.97	59.04	54.91	33.79	55.43	39.08	57.26
VideoScore [45]	✓		47.41	47.72	59.05	51.09	37.24	50.34	41.80	50.22
Q-Align [46]	✓		32.01	52.98	-	-	35.77	51.06	42.05	52.52
UnifiedReward [47]	✓	✓	41.27	39.42	-	-	40.11	36.58	53.31	58.83
Dover [48]	✓		39.34	**68.87**	-	-	38.01	55.65	54.27	68.58
VideoScore2 [49]		✓	34.67	65.87	-	-	48.70	65.92	54.53	65.59
Q-Save [50]	✓	✓	40.34	67.49	-	-	**50.52**	63.99	56.63	65.13
VisionReward [51]	✓	✓	47.43	59.03	59.03	60.98	46.56	61.15	56.77	67.59
CriticAgent (Ours)	✓		47.16	59.27	59.08	61.20	44.71	60.39	53.20	63.16
CriticAgent (Ours)	✓	✓	**49.79**	63.94	**59.98**	**62.42**	47.98	**66.52**	**57.31**	**68.86**

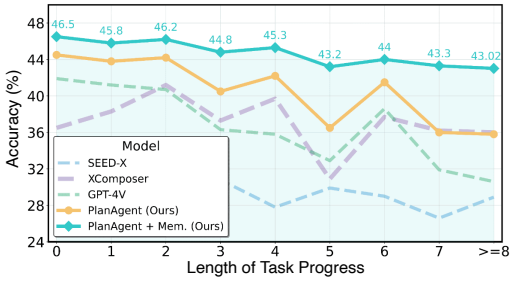


Figure 4: **PlanAgent Robustness to the Task Length.** Accuracy across varying horizons.

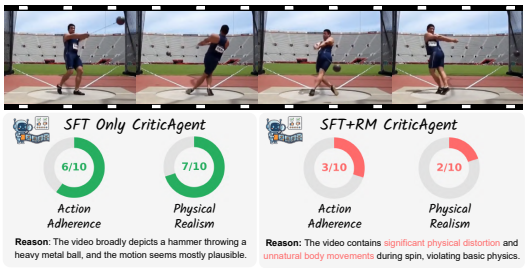


Figure 5: **CriticAgent Discriminative Capability.** Reward Modeling induces polarized scores.

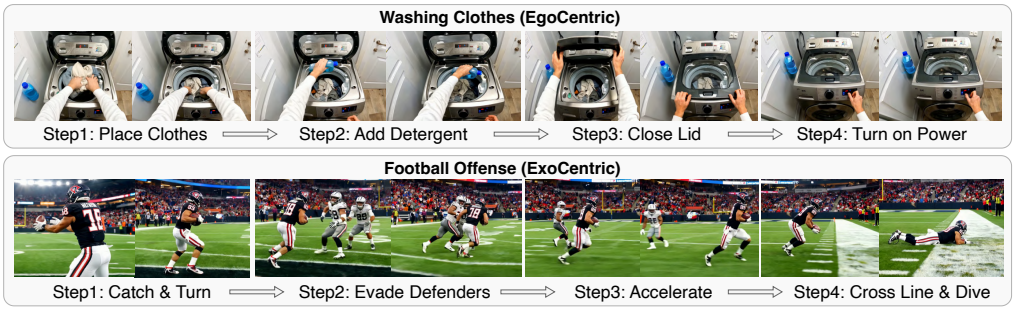


Figure 6: **Long-horizon Procedural Generation** results covering egocentric and exocentric human behavior generation, demonstrating coherent action execution across extended procedural steps.

quality preference learning. **(3) VideoGenerator** is instantiated by diverse T2V and I2V backbones, undergoing Streaming Long-Tuning on ActVideoGen-Dataset to instill action-following capabilities.

Evaluation Benchmarks. We evaluate our framework components across three benchmarks: **(1) EgoPlan-Bench** [52] assesses PlanAgent on long-horizon procedural planning, using task success rate as the primary metric; **(2) VideoGen-RewardBench** [53] evaluates CriticAgent as a video reward model, quantifying alignment with human judgments via preference accuracy; and **(3) ActVideoGen-Bench (Ours)** holistically evaluates the video quality of VideoGenerator in terms of visual fidelity, temporal coherence, and action accuracy under varying difficulty levels.

5.1 Evaluations on PlanAgent

Superiority in Long-Horizon Planning. Table 1 reports PlanAgent's performance on EgoPlan-Bench [52]. Across in-domain and out-of-domain splits, integrating context memory boosts accuracy from 35.81% to 43.63%, demonstrating that preserving historical state is vital for long-horizon consistency. While our method achieves higher in-domain accuracy than GRPO-CARE [40], GRPO-CARE generalizes better to OOD data. This likely occurs because GRPO fosters broader state exploration in the OOD domain, whereas our DPO strictly aligns with the training distribution.

Table 3: **Impact of SPIRAL on Long-Horizon Action Video Generation.** We integrate diverse T2V and I2V methods with SPIRAL, assessing static, dynamic, and action quality on ActVideoGen-Bench.

Method	Video Params		Static Quality		Dynamic Quality			Action Quality			
	Res.	FPS	Aesthetic Quality	Imaging Quality	Subject Consistency	Background Consistency	Motion Smoothness	Action Completeness	Action Smoothness	Object Interaction	Physical Fidelity
Text-to-Video (T2V)											
LongLive [14]	480P	16	0.549	0.735	0.937	0.946	0.990	0.273	0.646	0.776	0.808
LongLive [14] + SPIRAL (Ours)	480P	16	0.554	0.727	0.936	0.950	0.990	**0.721** +44.8%	**0.902** +25.6%	**0.860** +8.4%	**0.924** +11.6%
Wan2.1 [54]	480P	16	0.560	0.729	0.919	0.944	0.985	0.565	0.834	0.756	0.894
Wan2.1 [54] + SPIRAL (Ours)	480P	16	0.568	0.717	0.931	0.953	0.988	**0.875** +31.0%	**0.918** +8.4%	**0.926** +17.0%	**0.958** +6.4%
Sora [2]	720P	30	0.511	0.649	0.898	0.926	0.991	0.378	**0.820**	0.894	**0.852**
Sora [2] + SPIRAL (Ours)	720P	30	0.503	0.648	0.838	0.898	0.991	**0.540** +16.2%	0.714 -10.6%	**0.910** +1.6%	0.844 -0.8%
Seedance1.0 [55]	1080P	24	0.567	0.752	0.928	0.947	0.988	0.879	0.962	0.944	0.980
Seedance1.0 [55] + SPIRAL (Ours)	1080P	24	0.578	0.764	0.936	0.949	0.989	**0.956** +7.7%	**0.980** +1.8%	**0.962** +1.8%	**0.988** +0.8%
Image-to-Video (I2V)											
Wan2.1 [54]	480P	16	0.527	0.672	0.873	0.915	0.982	0.722	0.860	0.818	0.892
Wan2.1 [54] + SPIRAL (Ours)	480P	16	0.528	0.679	0.858	0.897	0.984	**0.827** +10.5	**0.934** +7.4%	**0.894** +7.6%	**0.970** +7.8%
Wan2.2 [54]	480P	24	0.492	0.642	0.892	0.925	0.974	0.709	0.916	0.818	0.934
Wan2.2 [54] + SPIRAL (Ours)	480P	24	0.469	0.640	0.861	0.884	0.975	**0.726** +1.7%	**0.918** +0.2%	**0.834** +1.6%	**0.942** +0.8%
SVI [25]	480P	16	0.505	0.636	0.838	0.903	0.977	0.233	0.746	0.816	0.782
SVI [25] + SPIRAL (Ours)	480P	16	0.509	0.649	0.846	0.902	0.984	**0.353** +12.0%	**0.754** +0.8%	**0.838** +2.2%	**0.808** +2.6%
Kling2.1 [56]	720P	24	0.542	0.706	0.904	0.932	0.989	0.042	0.702	0.704	0.848
Kling2.1 [56] + SPIRAL (Ours)	720P	24	0.550	0.709	0.910	0.919	0.991	**0.424** +38.2%	**0.896** +19.4%	**0.878** +17.4%	**0.878** +3.0%

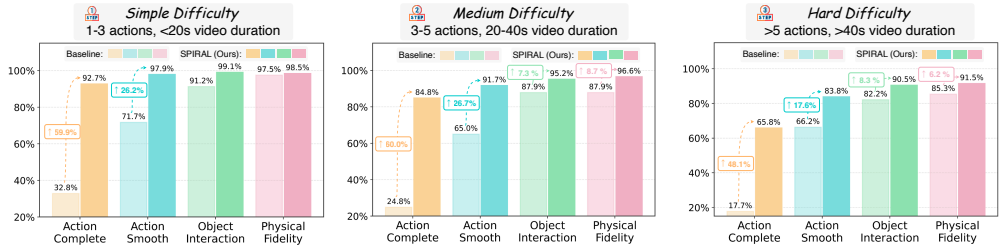


Figure 7: **Long-Horizon Video Generation Across Difficulties.** SPIRAL maintains high stability across all difficulty levels, mitigating the baseline's performance collapse on complex tasks.

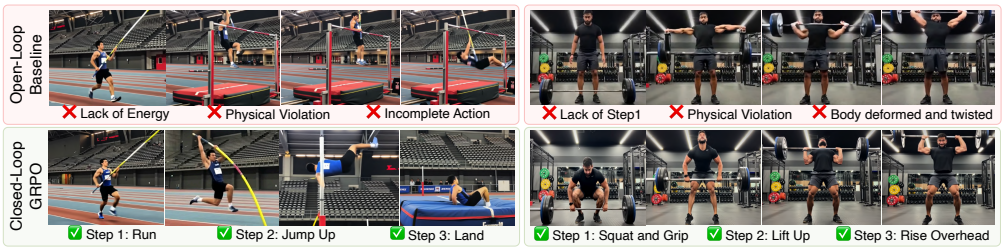


Figure 8: **Qualitative Comparison of Closed-Loop Self-Evolving.** Compared with the open-loop baseline, closed-loop GRPO produces more complete and physically plausible multi-step actions.

Impact of Hierarchical Training Stages. Ablating the training process highlights the complementary roles of both optimization phases. Instruction Tuning (IT) yields a 9.66% gain over the memory-augmented baseline, confirming the necessity of structured supervision for explicit CoT reasoning. Subsequently, Preference Alignment via DPO adds a 5.43% improvement, acting as a critical physical regularizer against logical inconsistencies and hallucinations. Ultimately, our two-stage training boosts overall accuracy to 58.72%, outperforming GRPO-CARE [40] by 3.94%.

Robustness To Extended Horizons. Figure 4 details accuracy across varying task lengths. Without memory, performance degrades sharply from 44.5% to 36.0%. Conversely, integrating the memory module maintains remarkable stability, sustaining 43.02% accuracy at extreme horizons. This empirically proves that preserving historical context is critical for preventing semantic drift.

5.2 Evaluations on CriticAgent

Superiority in Reward Assessment. Table 2 evaluates CriticAgent on the VideoGen-RewardBench [53]. It consistently outperforms previous methods in visual and motion quality, yielding a notable 5.37% gain in Text Alignment. This improvement stems from our reward modeling optimization on the GAIA dataset [36], which enhances sensitivity to text-action alignment. Ultimately, CriticAgent achieves an overall accuracy of 68.86%, validating our multi-dimensional scoring mechanism jointly captures visual fidelity and physical realism.

Impact of Training Stages. Ablating the training stages reveals that while SFT provides a robust initialization by distilling foundation model judgments, the subsequent Reward Modeling (RM)

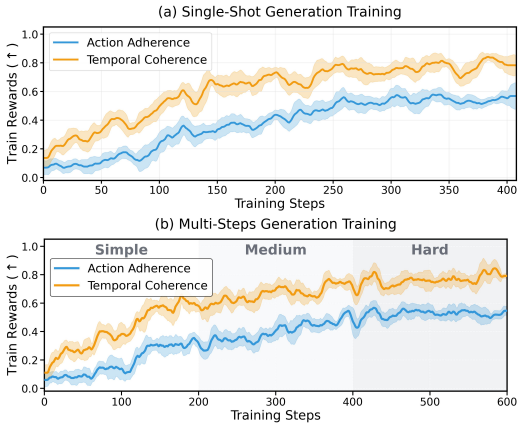


Figure 9: **GRPO Reward Convergence.** Curriculum learning (b) increases task complexity.

Table 4: Performance across Training Stages.

Training Stage	Action Compl.	Action Smooth.	Object Interact.	Physical Fidelity	Overall Quality
SFT Only	0.273	0.646	0.776	0.808	0.626
SFT + Closed Loop	0.721 +44.8%	0.902 +25.6%	0.860 +8.4%	0.924 +11.6%	0.852 +22.6%
SFT + CL + GRPO	**0.841** +56.8%	0.942 +29.6%	0.864 +8.8%	0.936 +12.8%	0.896 +27.0%

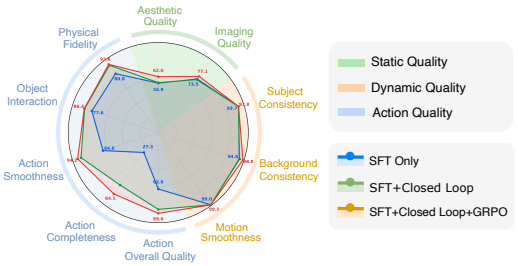


Figure 10: **Performance Gains via Closed Loop and GRPO.** Both stages improve action quality.

phase is critical for discrimination. Integrating preference learning delivers a 5.7% boost in overall performance, demonstrating its necessity for distinguishing execution failures, assessing text-video alignment, and ranking subtle action-quality differences.

Enhanced Discriminative Capability. Figure 5 compares score distributions before and after reward modeling (RM). Following SFT, scores for failed executions are diffuse, leaving ambiguous penalties for incomplete or misaligned actions. RM concentrates these scores near the minimum, improving discrimination of execution failures. This calibration sharpens reward signals for action adherence and physical plausibility, supporting subsequent GRPO-based optimization.

Table 5: **Blinded Human Evaluation.**

Method	Action Compl.	Action Smooth.	Object Interact.	Physical Fidelity	Overall Quality
Baseline	0.352	0.721	0.739	0.861	0.668
+ Closed-loop SPIRAL	0.766	0.831	**0.905**	0.879	0.845
+ GRPO-SPIRAL	**0.785**	**0.965**	0.826	**0.957**	**0.883**

5.3 Evaluations on Closed-Loop Video Generation

Superiority in Action Execution. As shown in Table 3, SPIRAL significantly enhances diverse T2V and I2V backbones. By leveraging the *PlanAgent* to decompose complex goals into executable actions and the *CriticAgent* to rectify execution errors, our closed-loop framework substantially improves overall Action Quality. Notably, action completeness achieves remarkable gains (e.g., +44.8% for LongLive), while other aspects also improve across most backbones, demonstrating that explicit planning and verification are crucial for reliable long-horizon generation.

Analysis of Generic Metrics. We observe minor fluctuations in generic visual and dynamic metrics, which are inherently biased toward short-term fidelity and limited motions. This bias inadvertently rewards open-loop baselines that fail to execute complete actions and collapse into short dynamics. In contrast, SPIRAL successfully executes full long-horizon tasks through staged atomic actions. Consequently, our framework achieves substantial gains in action completeness and procedural correctness, the true paramount metrics for long-horizon execution.

Qualitative Results. Fig. 6 visualizes closed-loop generation across egocentric interaction and exocentric human kinematics. As demonstrated, explicit planning and verification ensure coherent multi-step execution for both first-person manipulation and third-person motion control. Additional analyses of ultra-long procedural chains and open-loop comparisons are detailed in Appendix F.

Difficulty Stratification. Fig. 7 details performance across task difficulties (Simple, Medium, Hard). While baselines adequately handle simple tasks, they collapse on hard tasks with extended durations (e.g., >40s) and long-range action dependencies. Conversely, SPIRAL remains highly stable across all sequence lengths, confirming that explicit planning, memory preservation, and iterative verification are essential for reliable long-horizon action-conditioned video synthesis.

5.4 Evaluations on GRPO-based Self-Evolving

Curriculum-Driven Training Dynamics. Fig. 9 plots the GRPO reward curves for action adherence and temporal coherence. Driven by a curriculum learning regime (Fig. 9(b)) that progressively scales

Table 6: **Ablation on the Full Framework Components.**

Method	Agent		Feedback		Action Quality			
	Plan	Critic	Inner	Outer	Compl.	Smooth.	Interact.	Fidelity
Single-shot baseline	None				0.347	0.796	0.804	0.820
Prompt-only decomposition	Prompt				0.512	0.739	0.812	0.827
Trained PlanAgent only	Trained				0.628	0.754	0.838	0.840
Critic scoring only	Trained	✓			0.634	0.750	0.841	0.838
Local refinement only	Trained	✓	✓		0.715	0.816	**0.880**	0.848
Full SPIRAL	Trained	✓	✓	✓	**0.781**	**0.862**	0.872	**0.934**

Table 7: **Ablation on Memory Horizons.**

Memory	Visual Consistency		Action Quality		
	Subject	Background	Compl.	Smooth.	Fidelity
None (I_0)	0.712	0.745	0.654	0.783	0.825
Last Frame	0.835	0.852	0.712	0.815	0.868
Local Window	0.890	0.884	0.748	0.841	0.902
ContextMem. (Ours)	**0.938**	**0.941**	**0.781**	**0.862**	**0.934**

action complexity and temporal horizons, the generator transitions from executing atomic actions to mastering complex procedural tasks. The steady convergence demonstrates the efficacy of our critic-driven optimization, ensuring stable policy scaling even under long-horizon constraints.

Agent-Driven Optimization. By leveraging external reflective planning agents and critic-derived rewards, our self-evolving paradigm directly optimizes the generator's intrinsic weights (Fig. 10). As detailed in Table 4, GRPO optimization boosts action completeness (+12%) and smoothness (+4%) over the closed-loop baseline, proving that self-evolution effectively converts inference-time verification signals into permanent model enhancements.

Qualitative Analysis. Fig. 8 shows that the open-loop baseline suffers from incomplete motions and physical distortions, while closed-loop GRPO yields complete, temporally coherent sequences with reliable object interactions. Additional qualitative results are provided in Appendix F.4 and F.5.

Human Evaluation. Table 5 shows three blinded raters evaluated 150 generated videos from ActVideoGen-Bench. Results show that closed-loop SPIRAL improves human-rated action completeness and overall quality over the baseline. GRPO further improves overall quality (0.845 to 0.883), supporting gains beyond automatic scoring. The VLM-human correlations on action completeness are Pearson $r = 0.762$ and Spearman $\rho = 0.748$. Full results are provided in Appendix E.4.

5.5 Ablation Study

Ablation on PlanAgent. Table 6 separates decomposition from planner training. Prompt-only decomposition raises action completeness from the single-shot baseline's 0.347 to 0.512, while the trained PlanAgent further improves it to 0.628. This additional gain shows that task-specific training strengthens action planning beyond prompt-based decomposition.

Ablation on CriticAgent and Feedback. Adding critic scoring to the trained PlanAgent yields little change in action completeness (0.628 to 0.634), whereas local refinement raises it to 0.715 and improves action smoothness from 0.750 to 0.816. Adding global replanning completes the full SPIRAL framework, further improving completeness to 0.781 and physical fidelity from 0.848 to 0.934, with a small object interaction decrease (0.880 to 0.872). These results highlight the value of using critique for corrective feedback, with global replanning complementing local refinement.

Ablation on ContextMemory. Table 7 compares different memory horizons. Retaining only the last frame improves visual consistency over no history, while a local window brings further gains. ContextMemory achieves the highest subject/background consistency, and raises action completeness from 0.748 to 0.781 and physical fidelity from 0.902 to 0.934. These gains indicate that preserving broader context benefits both visual persistence and action execution.

6 Conclusion

We present SPIRAL, a closed-loop think-act-reflect framework designed to advance long-horizon action-conditioned video generation. SPIRAL seamlessly integrates a PlanAgent for task decomposition, a ContextMemory for visual persistence, and a CriticAgent for dual-level verification, effectively mitigating semantic drift and incomplete execution. Moving beyond inference-time guidance, we introduce a GRPO-based self-evolving strategy guided by curriculum learning, successfully converting agent-driven feedback into permanent parametric updates of the video generator. Furthermore, we contribute ActVideoGen-Dataset and ActVideoGen-Bench, establishing structured supervision and rigorous evaluation protocols for this domain. Extensive experiments on ActVideoGen-Bench demonstrate that SPIRAL consistently boosts action fidelity and temporal coherence across diverse T2V and I2V backbones, paving the way for robust and scalable long-horizon action video synthesis.

Acknowledgements and Disclosures

This work was supported by the National Key R&D Program of China (No. 2025YFF0511302). The authors declare no competing interests.

References

[1] 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.

[2] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. *OpenAI Blog*, 1(8):1, 2024.

[3] Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Daquan Zhou, Huan Ling, et al. Sana-video: Efficient video generation with block linear diffusion transformer. *arXiv preprint arXiv:2509.24695*, 2025.

[4] Team Seedance, De Chen, Liyang Chen, Xin Chen, Ying Chen, Zhuo Chen, Zhuowei Chen, Feng Cheng, Tianheng Cheng, Yufeng Cheng, et al. Seedance 2.0: Advancing video generation for world complexity. *arXiv preprint arXiv:2604.14148*, 2026.

[5] Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

[6] Xiaowei Chi, Chun-Kai Fan, Hengyuan Zhang, Xingqun Qi, Rongyu Zhang, Anthony Chen, Chi-Min Chan, Wei Xue, Qifeng Liu, Shanghang Zhang, et al. Empowering world models with reflection for embodied video prediction. In *Forty-second International Conference on Machine Learning*, 2025.

[7] Rick Akkerman, Haiwen Feng, Michael J Black, Dimitrios Tzionas, and Victoria Fernández Abrevaya. Interdyn: Controllable interactive dynamics with video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12467–12479, 2025.

[8] Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, et al. Genie envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025.

[9] Taiye Chen, Xun Hu, Zihan Ding, and Chi Jin. Learning world models for interactive video generation. *arXiv preprint arXiv:2505.21996*, 2025.

[10] Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Yusuf Aytar, Michael Rubinstein, Chen Sun, et al. Motion prompting: Controlling video generation with motion trajectories. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1–12, 2025.

[11] Zizhang Li, Hong-Xing Yu, Wei Liu, Yin Yang, Charles Herrmann, Gordon Wetzstein, and Jiajun Wu. Wonderplay: Dynamic 3d scene generation from a single image and actions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9080–9090, 2025.

[12] Jiahao Zhan, Zizhang Li, Hong-Xing Yu, and Jiajun Wu. Perpetualwonder: Long-horizon action-conditioned 4d scene generation. *arXiv preprint arXiv:2602.04876*, 2026.

[13] Wei Liu, Ziyu Chen, Zizhang Li, Yue Wang, Hong-Xing Yu, and Jiajun Wu. Realwonder: Real-time physical action-conditioned video generation. *arXiv preprint arXiv:2603.05449*, 2026.

[14] Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, et al. Longlive: Real-time interactive long video generation. *arXiv preprint arXiv:2509.22622*, 2025.

[15] Keming Wu, Zuhao Yang, Kaichen Zhang, Shizun Wang, Haowei Zhu, Sicong Leng, Zhongyu Yang, Qijie Wang, Sudong Wang, Ziting Wang, et al. Visual generation in the new era: An evolution from atomic mapping to agentic world modeling. *arXiv preprint arXiv:2604.28185*, 2026.

[16] Ziyu Guo, Renrui Zhang, Chengzhuo Tong, Zhizheng Zhao, Rui Huang, Haoquan Zhang, Manyuan Zhang, Jiaming Liu, Shanghang Zhang, Peng Gao, et al. Can we generate images with cot? let's verify and reinforce image generation step by step. *arXiv preprint arXiv:2501.13926*, 2025.

[17] Do Xuan Long, Yale Song, Min-Yen Kan, Tomas Pfister, and Long T Le. A^2RD: Agentic autoregressive diffusion for long video consistency. *arXiv preprint arXiv:2605.06924*, 2026.

[18] Zefeng He, Siyuan Huang, Xiaoye Qu, Yafu Li, Tong Zhu, Yu Cheng, and Yang Yang. Gems: Agent-native multimodal generation with memory and skills. *arXiv preprint arXiv:2603.28088*, 2026.

[19] Kaituo Feng, Manyuan Zhang, Shuang Chen, Yunlong Lin, Kaixuan Fan, Yilei Jiang, Hongyu Li, Dian Zheng, Chenyang Wang, and Xiangyu Yue. Gen-searcher: Reinforcing agentic search for image generation. *arXiv preprint arXiv:2603.28767*, 2026.

[20] Yunlong Lin, Zixu Lin, Kunjie Lin, Jinbin Bai, Panwang Pan, Chenxin Li, Haoyu Chen, Zhongdao Wang, Xinghao Ding, Wenbo Li, et al. Jarvisart: Liberating human artistic creativity via an intelligent photo retouching agent. *arXiv preprint arXiv:2506.17612*, 2025.

[21] Ziyu Guo, Renrui Zhang, Hongyu Li, Manyuan Zhang, Xinyan Chen, Sifan Wang, Yan Feng, Peng Pei, and Pheng-Ann Heng. Thinking-while-generating: Interleaving textual reasoning throughout visual generation. *arXiv preprint arXiv:2511.16671*, 2025.

[22] Achint Soni, Sreyas Venkataraman, Abhranil Chandra, Sebastian Fischmeister, Percy Liang, Bo Dai, and Sherry Yang. Videoagent: Self-improving video generation for embodied planning. In *NeurIPS 2025 Workshop on Bridging Language, Agent, and World Models for Reasoning and Planning*, 2025.

[23] Qinglin Zeng, Kaitong Cai, Ruiqi Chen, Qinhan Lv, and Keze Wang. Coagent: Collaborative planning and consistency agent for coherent video generation. *arXiv preprint arXiv:2512.22536*, 2025.

[24] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[25] Wuyang Li, Wentao Pan, Po-Chien Luan, Yang Gao, and Alexandre Alahi. Stable video infinity: Infinite-length video generation with error recycling. *arXiv preprint arXiv:2510.09212*, 2025.

[26] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18995–19012, 2022.

[27] 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.

[28] 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*, pages 19383–19400, 2024.

[29] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1207–1216, 2019.

[30] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European conference on computer vision (ECCV)*, pages 720–736, 2018.

[31] Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

[32] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21807–21818, 2024.

[33] 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. *arXiv preprint arXiv:2408.06072*, 2024.

[34] Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, et al. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025.

[35] Zeqing Wang, Xinyu Wei, Bairui Li, Zhen Guo, Jinrui Zhang, Hongyang Wei, Keze Wang, and Lei Zhang. Videoverse: How far is your t2v generator from a world model? *arXiv preprint arXiv:2510.08398*, 2025.

[36] Zijian Chen, Wei Sun, Yuan Tian, Jun Jia, Zicheng Zhang, Wang Jiarui, Ru Huang, Xiongkuo Min, Guangtao Zhai, and Wenjun Zhang. Gaia: Rethinking action quality assessment for ai-generated videos. *Advances in Neural Information Processing Systems*, 37:40111–40144, 2024.

[37] Yuying Ge, Sijie Zhao, Ziyun Zeng, Yixiao Ge, Chen Li, Xintao Wang, and Ying Shan. Making llama see and draw with seed tokenizer. *arXiv preprint arXiv:2310.01218*, 2023.

[38] Jinyoung Park, Jeehye Na, Jinyoung Kim, and Hyunwoo J Kim. Deepvideo-r1: Video reinforcement fine-tuning via difficulty-aware regressive grpo. *arXiv preprint arXiv:2506.07464*, 2025.

[39] Yi Chen, Yuying Ge, Rui Wang, Yixiao Ge, Lu Qiu, Ying Shan, and Xihui Liu. Exploring the effect of reinforcement learning on video understanding: Insights from seed-bench-r1. *arXiv preprint arXiv:2503.24376*, 2025.

[40] Yi Chen, Yuying Ge, Rui Wang, Yixiao Ge, Junhao Cheng, Ying Shan, and Xihui Liu. Grpo-care: Consistency-aware reinforcement learning for multimodal reasoning. *arXiv preprint arXiv:2506.16141*, 2025.

[41] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. In *Proceedings of the 2023 conference on empirical methods in natural language processing: system demonstrations*, pages 543–553, 2023.

[42] Hritik Bansal, Clark Peng, Yonatan Bitton, Roman Goldenberg, Aditya Grover, and Kai-Wei Chang. Videophy-2: A challenging action-centric physical commonsense evaluation in video generation. *arXiv preprint arXiv:2503.06800*, 2025.

[43] Xiao Liu and Jiawei Zhang. Aigve-macs: Unified multi-aspect commenting and scoring model for ai-generated video evaluation. *arXiv preprint arXiv:2507.01255*, 2025.

[44] Yibin Wang, Zhiyu Tan, Junyan Wang, Xiaomeng Yang, Cheng Jin, and Hao Li. Lift: Leveraging human feedback for text-to-video model alignment. *arXiv preprint arXiv:2412.04814*, 2024.

[45] Xuan He, Dongfu Jiang, Ge Zhang, Max Ku, Achint Soni, Sherman Siu, Haonan Chen, Abhranil Chandra, Ziyan Jiang, Aaran Arulraj, et al. Videoscore: Building automatic metrics to simulate fine-grained human feedback for video generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2105–2123, 2024.

[46] Haoning Wu, Zicheng Zhang, Weixia Zhang, Chaofeng Chen, Liang Liao, Chunyi Li, Yixuan Gao, Annan Wang, Erli Zhang, Wenxiu Sun, et al. Q-align: Teaching lmms for visual scoring via discrete text-defined levels. *arXiv preprint arXiv:2312.17090*, 2023.

[47] Yibin Wang, Yuhang Zang, Hao Li, Cheng Jin, and Jiaqi Wang. Unified reward model for multimodal understanding and generation. *arXiv preprint arXiv:2503.05236*, 2025.

[48] Haoning Wu, Erli Zhang, Liang Liao, Chaofeng Chen, Jingwen Hou, Annan Wang, Wenxiu Sun, Qiong Yan, and Weisi Lin. Exploring video quality assessment on user generated contents from aesthetic and technical perspectives. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 20144–20154, 2023.

[49] Xuan He, Dongfu Jiang, Ping Nie, Minghao Liu, Zhengxuan Jiang, Mingyi Su, Wentao Ma, Junru Lin, Chun Ye, Yi Lu, et al. Videoscore2: Think before you score in generative video evaluation. *arXiv preprint arXiv:2509.22799*, 2025.

[50] Xiele Wu, Zicheng Zhang, Mingtao Chen, Yixian Liu, Yiming Liu, Shushi Wang, Zhichao Hu, Yuhong Liu, Guangtao Zhai, and Xiaohong Liu. Q-save: Towards scoring and attribution for generated video evaluation. *arXiv preprint arXiv:2511.18825*, 2025.

[51] Jiazheng Xu, Yu Huang, Jiale Cheng, Yuanming Yang, Jiajun Xu, Yuan Wang, Wenbo Duan, Shen Yang, Qunlin Jin, Shurun Li, et al. Visionreward: Fine-grained multi-dimensional human preference learning for image and video generation. *arXiv preprint arXiv:2412.21059*, 2024.

[52] Yi Chen, Yuying Ge, Yixiao Ge, Mingyu Ding, Bohao Li, Rui Wang, Ruifeng Xu, Ying Shan, and Xihui Liu. Egoplan-bench: Benchmarking multimodal large language models for human-level planning. *arXiv preprint arXiv:2312.06722*, 2023.

[53] Jie Liu, Gongye Liu, Jiajun Liang, Ziyang Yuan, Xiaokun Liu, Mingwu Zheng, Xiele Wu, Qiulin Wang, Menghan Xia, Xintao Wang, et al. Improving video generation with human feedback. *arXiv preprint arXiv:2501.13918*, 2025.

[54] 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.

[55] Yu Gao, Haoyuan Guo, Tuyen Hoang, Weilin Huang, Lu Jiang, Fangyuan Kong, Huixia Li, Jiashi Li, Liang Li, Xiaojie Li, et al. Seedance 1.0: Exploring the boundaries of video generation models. *arXiv preprint arXiv:2506.09113*, 2025.

[56] Kling Team, Jialu Chen, Yuanzheng Ci, Xiangyu Du, Zipeng Feng, Kun Gai, Sainan Guo, Feng Han, Jingbin He, Kang He, et al. Kling-omni technical report. *arXiv preprint arXiv:2512.16776*, 2025.

[57] Wencan Jiang, Jiangning Zhang, Jianbiao Mei, Jinzhuo Liu, Yu Yang, Xiaobin Hu, Zhucun Xue, Yong Liu, and Dacheng Tao. Spike: An adaptive dual controller framework for cost-efficient long-horizon game agents. *arXiv preprint arXiv:2605.18636*, 2026.

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

[59] Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023.

[60] Jiaqi Liao, Zhengyuan Yang, Linjie Li, Dianqi Li, Kevin Lin, Yu Cheng, and Lijuan Wang. Imagegen-cot: Enhancing text-to-image in-context learning with chain-of-thought reasoning. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 17214–17223. IEEE, 2025.

[61] Le Zhuo, Liangbing Zhao, Sayak Paul, Yue Liao, Renrui Zhang, Yi Xin, Peng Gao, Mohamed Elhoseiny, and Hongsheng Li. From reflection to perfection: Scaling inference-time optimization for text-to-image diffusion models via reflection tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15329–15339, 2025.

[62] Hongyu Li, Manyuan Zhang, Dian Zheng, Ziyu Guo, Yimeng Jia, Kaituo Feng, Hao Yu, Yexin Liu, Yan Feng, Peng Pei, et al. Editthinker: Unlocking iterative reasoning for any image editor. *arXiv preprint arXiv:2512.05965*, 2025.

[63] Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*, 2025.

[64] Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-zero: Self-evolving reasoning llm from zero data. *arXiv preprint arXiv:2508.05004*, 2025.

[65] Qinsi Wang, Bo Liu, Tianyi Zhou, Jing Shi, Yueqian Lin, Yiran Chen, Hai Helen Li, Kun Wan, and Wentian Zhao. Vision-zero: Scalable vlm self-improvement via strategic gamified self-play. *arXiv preprint arXiv:2509.25541*, 2025.

[66] Rong Wu, Xiaoman Wang, Jianbiao Mei, Pinlong Cai, Daocheng Fu, Cheng Yang, Licheng Wen, Xuemeng Yang, Yufan Shen, Yuxin Wang, et al. Evolver: Self-evolving llm agents through an experience-driven lifecycle. *arXiv preprint arXiv:2510.16079*, 2025.

[67] Daocheng Fu, Jianbiao Mei, Licheng Wen, Xuemeng Yang, Cheng Yang, Rong Wu, Tao Hu, Siqi Li, Yufan Shen, Xinyu Cai, et al. Re-searcher: Robust agentic search with goal-oriented planning and self-reflection. *arXiv preprint arXiv:2509.26048*, 2025.

[68] Do Xuan Long, Xingchen Wan, Hootan Nakhost, Chen-Yu Lee, Tomas Pfister, and Sercan Ö Arık. Vista: A test-time self-improving video generation agent. *arXiv preprint arXiv:2510.15831*, 2025.

[69] Meng Chu, Xuan Billy Zhang, Kevin Qinghong Lin, Lingdong Kong, Jize Zhang, Teng Tu, Weijian Ma, Ziqi Huang, Senqiao Yang, Wei Huang, et al. Agentic world modeling: Foundations, capabilities, laws, and beyond. *arXiv preprint arXiv:2604.22748*, 2026.

[70] Yu Yang, Xuemeng Yang, Licheng Wen, Lingdong Kong, Xiaobin Hu, Dongyue Lu, Wei Chow, Xiyan Huang, Yuxiang Feng, Yue Liao, et al. Quo vadis, world modeling? *arXiv preprint arXiv:2608.02713*, 2026.

[71] Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos J Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024.

[72] Dongyue Lu, Guian Fang, Hur Lim, Yonggang Wu, Yanrong Wang, Hongshan Chen, Shengju Qian, Yue Liao, Wei Chow, Lingdong Kong, et al. Orbia: Do generated videos form a consistent 3d world? *arXiv preprint arXiv:2610.20693*, 2026.

[73] Jiahan Zhang, Muqing Jiang, Nanru Dai, Taiming Lu, Arda Uzunoglu, Shunchi Zhang, Yana Wei, Jiahao Wang, Vishal M Patel, Paul Pu Liang, et al. World-in-world: World models in a closed-loop world. *arXiv preprint arXiv:2510.18135*, 2025.

[74] Lingdong Kong, Wesley Yang, Jianbiao Mei, Youquan Liu, Ao Liang, Dekai Zhu, Dongyue Lu, Wei Yin, Xiaotao Hu, Mingkai Jia, et al. 3d and 4d world modeling: A survey. *arXiv preprint arXiv:2509.07996*, 2025.

[75] Ao Liang, Lingdong Kong, Tianyi Yan, Hongsi Liu, Wesley Yang, Ziqi Huang, Wei Yin, Jialong Zuo, Yixuan Hu, Dekai Zhu, et al. Worldlens: Full-spectrum evaluations of driving world models in real world. *arXiv preprint arXiv:2512.10958*, 2025.

[76] Yu Yang, Alan Liang, Jianbiao Mei, Yukai Ma, Yong Liu, and Gim Hee Lee. X-scene: Large-scale driving scene generation with high fidelity and flexible controllability. *arXiv preprint arXiv:2506.13558*, 2025.

[77] Ao Liang, Youquan Liu, Mingxiao Li, Wei Chow, Lingdong Kong, Yue Liao, Dongyue Lu, Yu Yang, Wei Yin, and Runnan Chen. Vital: From visual motion to precise action. *arXiv preprint arXiv:2610.20604*, 2026.

[78] Jianbiao Mei, Tao Hu, Xuemeng Yang, Licheng Wen, Yu Yang, Tiantian Wei, Yukai Ma, Min Dou, Botian Shi, and Yong Liu. Dreamforge: Motion-aware autoregressive video generation for multi-view driving scenes. *arXiv preprint arXiv:2409.04003*, 2024.

[79] Tong Wu, Shuai Yang, Ryan Po, Yinghao Xu, Ziwei Liu, Dahua Lin, and Gordon Wetzstein. Video world models with long-term spatial memory. *arXiv preprint arXiv:2506.05284*, 2025.

[80] Junshu Tang, Jiacheng Liu, Jiaqi Li, Longhuang Wu, Haoyu Yang, Penghao Zhao, Siruis Gong, Xiang Yuan, Shuai Shao, and Qinglin Lu. Hunyuan-gamecraft-2: Instruction-following interactive game world model. *arXiv preprint arXiv:2511.23429*, 2025.

[81] Roberto Henschel, Levon Khachatryan, Hayk Poghosyan, Daniil Hayrapetyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Streamingt2v: Consistent, dynamic, and extendable long video generation from text. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2568–2577, 2025.

[82] 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.

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

[84] MiniMax. Hailuo AI. https://hailuoai.video, 2024.

[85] Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. Dancegrpo: Unleashing grpo on visual generation. *arXiv preprint arXiv:2505.07818*, 2025.

[86] Hui Han, Siyuan Li, Jiaqi Chen, Yiwen Yuan, Yuling Wu, Yufan Deng, Chak Tou Leong, Hanwen Du, Junchen Fu, Youhua Li, et al. Video-bench: Human-aligned video generation benchmark. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18858–18868, 2025.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Both contributions and scope have been discussed in abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The detailed analysis of limitations has been discussed in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: This is an empirical study that excludes theory assumptions and proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All information needed to reproduce the experimental results have been disclosed. To ensure reproducibility, code and data are committed to be publicly available.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The detailed implementation procedures have been included in the appendix. To ensure reproducibility, code and data are committed to be publicly available.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: All training and test details have been discussed in either main body or appendix. To ensure reproducibility, code and data are committed to be publicly available.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Sufficient information about experiment settings have been discussed.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The details on computing resources have been discussed.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: This research follows the NeurIPS Code of Ethics properly.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The discussion on societal impacts has been included in the appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: The discussion on safeguards has been included in the appendix.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The acknowledgments on licenses have been included in the appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The discussions on new assets have been included in the appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs were employed in Section 4 to facilitate dataset annotation and validation, as well as to construct the evaluation benchmark.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.

SPIRAL: Self-Evolving Action-Conditioned Video Generation via Reflective Planning Agents

Appendix

Contents

A Additional Related Works **25**

A.1 Agentic Visual Generation 25

A.2 Agentic Video Generation 25

A.3 Self-Evolving Video Generation 25

A.4 Video World Models 25

B Additional Implementation Details of SPIRAL **26**

B.1 PlanAgent Implementation Details 26

B.2 VideoGenerator Implementation Details 28

B.3 CriticAgent Implementation Details 29

B.4 Inference-Time Closed-Loop Pipeline 30

B.5 Training-Time GRPO-based Self-Evolving 31

C Additional Details of ActVideoGen-Dataset **34**

C.1 Dataset Annotation and Verification 34

C.2 Dataset Examples and Statistics 34

D Additional Details of ActVideoGen-Bench **36**

D.1 Multi-Agent Evaluation Framework 36

D.2 Evaluation Metrics Definition 36

D.3 Case Studies of Evaluation 36

E Additional Quantitative Results **38**

E.1 VideoGenerator Generalization 38

E.2 Evaluator Stability 38

E.3 Inference-Time Cost 38

E.4 Human Evaluation and Evaluator Alignment 39

E.5 Robustness to Planner and Critic Errors 39

F Additional Qualitative Results **40**

F.1 End-to-End Pipeline Trace 40

F.2 Closed-Loop Feedback Refinement 40

F.3 Long-Horizon Procedure Generation 41

F.4 Comparison with Open-Loop Generation 41

F.5 GRPO-Based Self-Evolution Visualization 41

G Potential Societal Impact & Limitations **49**
G.1 Societal Impact 49
G.2 Known Limitations 49

H Public Resources Used **50**
H.1 Public Datasets Used 50
H.2 Public Implementations Used 50

A Additional Related Works

A.1 Agentic Visual Generation

Recent visual generation research is moving beyond one-shot generation toward agentic systems that treat generation as one step in a closed-loop decision process [15]. Existing systems span a spectrum from workflow-style pipelines to more autonomous visual agents [57]. For example, GEMS [18] introduces a planner–decomposer–verifier–refiner workflow equipped with memory and skill selection for multimodal generation, while Gen-Searcher [19] extends this paradigm with agentic web search and evidence collection for knowledge-intensive visual tasks. In the editing domain, JarvisArt [20] maps user intent into executable photo-retouching operations through reasoning and tool use, demonstrating that agentic generation can be embedded within professional visual software rather than merely producing pixels. Related embodied systems, such as CoT-VLA [58] and UniPi [59], further treat visual generation or prediction as part of a perception–action loop for downstream control. Overall, these works indicate that the central challenge of agentic visual generation is shifting from rendering fidelity alone toward reliable planning, grounded verification, memory preservation, and self-correction over long-horizon visual trajectories.

A.2 Agentic Video Generation

Recent work has explored incorporating chain-of-thought (CoT) reasoning and intermediate planning into video generation to improve consistency and controllability [16]. These approaches typically decompose high-level prompts into structured intermediate signals, and have evolved from pre-planning [60] or post-refinement [61] toward "thinking while generating" paradigms [21]. Representative examples include EditThinker [62], which frames image editing as iterative reasoning, and VideoAgent [22] and CoAgent [23], which cast video generation as a closed-loop, agentic process with planning, synthesis, and verification. In contrast to these works, we explicitly formulate a closed-loop framework with reflective planning agents and an explicit memory module, enabling sustained grounding of high-level actions over long temporal horizons.

A.3 Self-Evolving Video Generation

Recent years have witnessed growing interest in self-evolving paradigms across code generation [63], question answering [64, 65], and agentic search [66, 67]. A common theme is to leverage either internal signals or external feedback to progressively enhance solution quality. In video generation, early attempts have begun to extend these ideas to long-horizon generation and planning. VideoAgent [22] proposes a self-improving video planning framework that iteratively refines text-to-video generation via self-conditioned consistency, employs a VLM to guide plan selection, and leverages online environment feedback to mitigate hallucinations. VISTA [68] further advances this line of work by introducing a test-time, multi-agent iterative self-improvement framework that emulates human-like prompt refinement and jointly optimizes visual, audio, and contextual aspects of video generation through structured planning, critique, and targeted revision. In this work, we propose a self-evolution strategy using group relative policy optimization, enabling closed-loop reinforcement learning driven by critic-derived signals.

A.4 Video World Models

World models aim to learn the dynamics of the environment by predicting future observations conditioned on past observations and actions [69, 70]. Early studies focused on games and simulated environments [71, 72], with recent progress extending to robotics [73, 8], autonomous driving [74–77], and embodied settings. More recently, fueled by rapid progress in video foundation models [33, 1], video world models have advanced toward higher long-horizon generation quality and temporal consistency [78, 79], faster inference for interactive or online use [14, 3], richer and more diverse action controllability [80], and improved physical fidelity in motion, dynamics, and contact interactions [5]. Unlike prior works that primarily use text prompts to describe global scenes or short-term changes [81], our work focuses on persistent, object-grounded control, treating language as high-level semantic actions driving long-horizon behavior.

B Additional Implementation Details of SPIRAL

We provide comprehensive implementation details of our SPIRAL framework, elaborating on the PlanAgent (Sec. B.1), VideoGenerator (Sec. B.2), CriticAgent (Sec. B.3), inference-time closed-loop pipeline (Sec. B.4), and training-time GRPO-based self-evolving strategy (Sec. B.5).

B.1 PlanAgent Implementation Details

We detail the PlanAgent (π_{plan}), covering its architecture, the two-stage training protocol (SFT and DPO), preference data construction, and closed-loop inference logic.

Table 8: **Hyperparameter settings for PlanAgent training**. Parameters are differentiated by two training phases: Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO).

Parameter	SFT	DPO	Description
Optimization & Training Strategy			
Base model	Qwen3-VL-8B-Thinking		Backbone vision-language model
Learning rate	1.0×10^{-4}	5.0×10^{-6}	Peak learning rate for AdamW
LR scheduler	Cosine	Cosine	Decay schedule type
Warmup ratio	0.1	0.1	Ratio of warmup steps
Max epochs	3.0	3.0	Number of training epochs
Batch size/GPU	1	1	Per-device batch size
Gradient accumulation	8	8	Steps for effective batch construction
Precision	bf16	bf16	Mixed-precision training format
Max sequence length	8192	8192	Maximum context length in tokens
Model Architecture & LoRA Configuration			
LoRA rank (r)	16	16	Rank dimension for adaptation
LoRA alpha (α)	32	32	Scaling factor ($2 \times r$)
LoRA dropout	0.05	0.05	Dropout probability for LoRA layers
Target modules	all linear		Attention and MLP layers applied
DPO-Specific Configuration			
Beta (β)	-	0.1	KL penalty coefficient
Loss function	-	Sigmoid	Preference loss type

Architecture and Training Configuration. We instantiate the PlanAgent using Qwen3-VL-8B [24] as the backbone. To enable structured planning while preserving pre-trained multimodal capabilities, we apply Low-Rank Adaptation (LoRA) to all linear layers in the attention and feed-forward networks.

Training proceeds in two sequential stages. First, **Supervised Fine-Tuning (SFT)** teaches the model to produce explicit Chain-of-Thought (CoT) reasoning and to follow the required JSON schema. Second, **Direct Preference Optimization (DPO)** further aligns the model with physical plausibility and temporal consistency by contrasting valid plans against carefully constructed flawed alternatives. Both stages use the AdamW optimizer with a cosine scheduler and a global batch size of 8 via gradient accumulation. We set the context window to 8,192 tokens to accommodate interleaved visual history and long reasoning traces. Table 8 summarizes the hyperparameters.

DPO Preference Construction. A critical component of DPO training is constructing high-quality preference pairs (y_w, y_l) that expose hallucinations, invalid object interactions, and temporal inconsistencies. Unlike standard datasets that provide only ground-truth plans, we employ a teacher-student distillation pipeline to synthesize paired responses from ActVideoGen-Dataset:

- **Chosen Response (y_w).** We prompt a strong teacher (GPT-5.1) to generate a preferred response that first decomposes the goal through a rigorous CoT rationale inside *<think>* tags, and then outputs a valid JSON plan. The resulting plan must realize the global goal depicted in the ground-truth video.
- **Rejected Response (y_l).** We prompt the teacher to generate a hard negative response that appears plausible locally but fails to complete the goal because of injected planning errors. These errors include missing preconditions (e.g., pouring without opening), temporal disorder, invalid object interactions, or object hallucinations.

This model-in-the-loop synthesis makes the rejected responses subtle and contextually relevant, providing stronger learning signals than random negative sampling. Before DPO training, all generated pairs are filtered to ensure strict schema adherence.

Inference Logic. The PlanAgent supports two inference modes, corresponding to initial planning and feedback-driven correction.

- **Standard Planning Mode.** At the beginning of a task, the agent receives the goal and visual observations, including the current frame and uniformly sampled historical frames. We generate an action plan via temperature sampling, then parse the output to extract the reasoning trace and the structured step list $\mathcal{S}$. To simplify downstream execution, the agent is constrained to output only the <*think*>-block and the JSON object.
- **Closed-Loop Re-planning Mode.** When the CriticAgent detects a failure, the PlanAgent enters a re-planning state conditioned on a structured *Failure Analysis* context. This context contains the *execution history*, the *failed attempt*, and the *critical diagnosis*. The agent follows a diagnose-and-fix procedure: it rewrites the failed step according to the critique and revises subsequent steps to preserve logical continuity. The revised output starts from the failed step ID, enabling seamless replacement of the remaining plan.

System Prompts. We provide the exact system prompts used to steer the PlanAgent in both modes, designed to enforce the structured output format and specific reasoning requirements.

- **Standard Planning Prompt.** This prompt is used for the initial decomposition of the global goal into atomic actions.

System Prompt for PlanAgent (Standard Planning)

You are an **Action Planner** in a video generation system. Your role is to plan a sequence of executable steps that will guide a world model to produce a video achieving the given goal from the current visual state.

Input Data:
- **image**: ONE current reference image (the most recent frame of the scene).
- **history**: A few history images showing previous context.
- **GOAL**: A textual description of what needs to be achieved.

Instructions:
- Carefully observe the provided images to understand the current state.
- Reason step by step inside **<think>...</think>** to plan how to achieve the GOAL.
- Output a valid JSON with a single key `steps` (array), describing the planned sequence.
- **Constraint**: Each step must include specific fields: `sid`, `text`, `actions`, `pre`, and `post`.
- Use lemma forms for verbs and nouns in the `actions` field.

Expected Output Structure (JSON):

```
{
  "steps": [
    {
      "sid": 1,
      "action instruction": "Concise execution instruction (<= 36
          words)",
      "actions": [
        {"verb": "lemma", "objects": ["obj1"], "tool": "tool_name"}
      ],
      "pre": ["pre-condition1", "pre-condition2"],
      "post": ["post-condition1", "post-condition2"]
    },
    ...
  ]
}
```

- **Re-Planning Prompt.** This prompt is activated during the feedback loop to correct specific execution failures.

System Prompt for PlanAgent (Closed-Loop Re-planning)

You are an **Expert Plan Editor** for a procedural video generation system. Your goal is to recover from a failure by creating a **New, Corrected** sequence of future actions based on visual evidence and error diagnosis.

Input Context:
- **global_goal**: The final objective.
- **failed_attempt**: The specific step definition that failed execution.
- **critic_feedback**: Detailed natural language diagnosis of WHY the failure occurred.
- **remaining_steps**: The original draft for the future steps.

Re-planning Rules (Strict):
- **Diagnose and Fix**: Analyze the **critic_feedback**. You must discard the `failed_attempt` and replace it with valid steps. Fix the step's `text`, `actions`, and `post-conditions`.
- **Revise Future**: Review the `remaining_steps`. Ensure they logically follow your NEW corrected step or steps. You may modify, add, or delete steps to restore logical continuity.
- **Formatting**: The output sequence must start with the **Same SID** as the failed step.

Example Output (Re-planning):

```
<think>
The critic indicates the previous attempt failed because the object was
    not grasped firmly. I need to insert a grasp adjustment step
    before the lift action.
</think>
{
  "steps": [
    {
      "sid": 3,  // Same ID as the failed step
      "text": "Adjust grip to ensure a firm hold on the handle.",
      "actions": [{"verb": "adjust", "objects": ["grip"]}],
      "pre": ["hand near object"],
      "post": ["firm grip established"]
    },
    {
      "sid": 4,  // Revised subsequent step
      "text": "Lift the object vertically.",
      ...
    }
  ]
}
```

B.2 VideoGenerator Implementation Details

The VideoGenerator is designed as a **flexible, model-agnostic module** that supports plug-and-play integration with Text-to-Video (T2V) and Image-to-Video (I2V) backbones. Its goal is to execute each structured action instruction produced by the PlanAgent and generate the corresponding video segment. We adapt generic video generation backbones using the structured instruction-action pairs from ActVideoGen-Dataset through Supervised Fine-Tuning (SFT), enabling fine-grained action control and long-horizon instruction following.

Model Instantiation. To demonstrate the effectiveness of our SFT strategy, we present a concrete instantiation using Stable-Video-Infinity [25], which leverages the Wan2.1-I2V-14B [1] as the foundational backbone. To adapt this model for precise action control while preserving its massive pre-trained generative prior, we apply Low-Rank Adaptation (LoRA) to the query, key, value, output projection, and feed-forward network layers (q, k, v, o, ffn) of the DiT blocks. We set the LoRA rank $r = 64$ and alpha $\alpha = 128$. The model processes video data encoded by the Wan2.1 VAE and text instructions via the T5-XXL encoder. Training is conducted using DeepSpeed Stage 2 with BF16 mixed precision to optimize memory efficiency.

Multi-Resolution Training. Since ActVideoGen-Dataset contains videos with diverse aspect ratios, we adopt Multi-Resolution Training to improve generation quality across spatial formats. Training data is organized into resolution buckets, and the VideoGenerator is fine-tuned in three stages:

- **Stage 1: Base Adaptation.** We train on 480×480 square clips for 10 epochs with a learning rate of $1e-4$. This stage establishes basic alignment between action instructions and motions.
- **Stage 2: Rectangular Adaptation.** We increase the resolution to 640×480 and reduce the learning rate to $6e-5$ for 3 epochs. This stage adapts the generator to rectangular video layouts.

- **Stage 3: High-Resolution Adaptation.** We further fine-tune on 832×480 inputs with a learning rate of $4e-5$ for 3 epochs. This stage improves high-resolution generation capacity.

This multi-resolution setup exposes the VideoGenerator to diverse spatial layouts and instruction complexities, improving aspect-ratio robustness without losing the motion priors.

B.3 CriticAgent Implementation Details

We provide a comprehensive implementation for the CriticAgent, including its architecture, two-stage training protocol (SFT and RM), dataset construction strategies, and closed-loop inference logic.

Table 9: **Hyperparameter settings for CriticAgent training**. Parameters are differentiated by the two training phases: Supervised Fine-Tuning (SFT) and Pairwise Reward Modeling (RM).

Parameter	SFT	RM	Description
Optimization & Training Strategy			
Base model	Qwen3-VL-8B-Instruct		Backbone vision-language model
Learning rate	1.0×10^{-4}	5.0×10^{-6}	Peak learning rate for AdamW
LR scheduler	Cosine	Cosine	Decay schedule type
Warmup ratio	0.1	0.1	Ratio of warmup steps
Max epochs	3.0	1.0	Number of training epochs
Batch size/GPU	1	1	Per-device batch size
Gradient accumulation	8	8	Steps for effective batch construction
Precision	bf16	bf16	Mixed-precision training format
Max sequence length	8192	8192	Maximum context length in tokens
Model Architecture & LoRA Configuration			
LoRA rank (r)	16	16	Rank dimension for adaptation
LoRA alpha (α)	32	32	Scaling factor ($2 \times r$)
LoRA dropout	0.05	0.05	Dropout probability for LoRA layers
Target modules	all linear		Attention and MLP layers applied
RM-Specific Configuration			
Loss function	-	Bradley-Terry	Pairwise ranking loss objective
Margin	-	0.0	Margin for ranking separation

Architecture and Training Configuration. Analogous to the PlanAgent, the CriticAgent is initialized using the Qwen3-VL-8B [24] backbone. To enable fine-grained video evaluation capabilities while preserving generalization, we apply Low-Rank Adaptation (LoRA) across all linear layers.

The training regimen proceeds in two sequential stages. First, **Supervised Fine-Tuning (SFT)** conditions the model to act as a rigorous judge by internalizing explicit evaluation dimensions, such as action adherence, object interaction, temporal coherence, and physical realism. Second, **Pairwise Reward Modeling (RM)** improves discriminative precision through a Bradley-Terry objective, aligning the model's scores with human or oracle preference rankings. Both stages use the AdamW optimizer with a cosine learning rate scheduler; detailed hyperparameters are summarized in Table 9.

Reward Data Construction. We employ a hybrid data construction strategy to balance reasoning depth with fine-grained discriminative sensitivity.

- **SFT Data.** We implement a teacher-student distillation pipeline based on the VideoVerse benchmark [35]. Following its protocol, we synthesize a diverse video samples using state-of-the-art video generators, including CogVideoX-1.5-5B [33], SkyReels-V2-14B [34], HunyuanVideo [82], OpenSora2.0 [83], Wan2.1-14B [1], Wan2.2-A14B [1], Hailuo [84], Veo3, and Sora-2. We use Gemini-3-Pro as the oracle judge to generate CoT critiques and scalar scores over five dimensions: *Action Adherence*, *Object Interaction*, *Goal Achievement*, *Temporal Coherence*, and *Physical Realism*. These signals teach the CriticAgent a structured, evidence-based evaluation schema.
- **RM Data.** We leverage GAIA [36] to strengthen action-level sensitivity through pairwise reward modeling. Preference pairs (y_w, y_l) are constructed from two sources: *Quality Ranking* pairs, derived from human ratings of videos depicting the same action; and *Semantic Negative* pairs, formed by pairing a valid video with a semantically similar but incorrect action description (e.g., "sipping" vs. "drinking"). This contrastive setup encourages the CriticAgent to inspect fine-grained motion evidence rather than relying on coarse scene context.

Inference Logic. The CriticAgent provides actionable feedback inside the closed-loop pipeline through three operations:

- **Multi-Dimensional Scoring.** Given a generated segment v_t and its corresponding plan step s_t, the agent evaluates action-video alignment across predefined dimensions. The output is constrained to a JSON object with scalar scores $r \in [0, 1]$ and textual evidence justifying each judgment.
- **Thresholding and Feedback.** We use the primary metric, *Action Adherence*, to trigger correction. If its score falls below the success threshold $\tau = 0.7$, the CriticAgent produces a *revised action instruction* that explicitly addresses missing or violated conditions, such as ensuring that the hand clearly grasps the handle before lifting.
- **Streaming Evaluation.** For long-horizon tasks, evaluation is performed at the segment level. The agent inspects the clip corresponding to the current step v_t rather than the full accumulated video, keeping the critique focused on the immediate action dynamics specified by s_t.

System Prompts. We provide the exact system prompts used to steer the CriticAgent for evaluation, designed to enforce the structured output format and specific evidence-based reasoning requirements.

- **Multi-Dimensional Scoring Prompt.** This prompt performs a rigorous evaluation of the video across five key dimensions.

System Prompt for CriticAgent (Multi-Dimensional Scoring)

You are a meticulous **Video Critic**. Given a generated video and its task specification, your role is to evaluate the video across multiple dimensions using a structured, evidence-based methodology.

Input Data:
- **global_goal**: High-level textual description of the objective.
- **action_plan_list**: Ordered steps including `text`, `actions`, `pre`, and `post` conditions.
- **Video**: The AI-generated video frames.

Evaluation Dimensions:
- **A. Action Adherence**: Deduct points for missing or swapped steps.
- **B. Object Interaction**: Verify correct verb-tool-object usage.
- **C. Goal Achievement**: Check if `post` conditions are met.
- **D. Temporal Coherence**: Evaluate motion continuity and smoothness.
- **E. Visual & Physics Realism**: Assess lighting, gravity, and collisions.

Expected Output Structure (JSON):

```
{
  "scores": {
    "action_adherence": {
      "score": 0.85,
      "reason": "Concise reason..."
    },
    "object_interaction": {
      "reason": "General assessment...",
      "per_action": [
        {
          "verb": "pour", "tool": "kettle", "match": "yes",
          "score": 1.0, "reason": "Water flows naturally."
        }
      ]
    },
    "goal_achievement": {
      "reason": "General assessment...",
      "per_event": [
        {"event_id": 1, "score": 1.0, "reason": "Cup is full."}
      ]
    },
    "temporal_coherence": {"score": 0.9, "reason": "..."},
    "visual_physics_realism": {"score": 0.8, "reason": "..."}
  }
}
```

B.4 Inference-Time Closed-Loop Pipeline

Figure 11 illustrates the **inference-time closed-loop pipeline** used by SPIRAL. Given a global goal and a reference initial state, the system iteratively plans, generates, verifies, and corrects video segments until the goal is completed. The pipeline proceeds as follows:

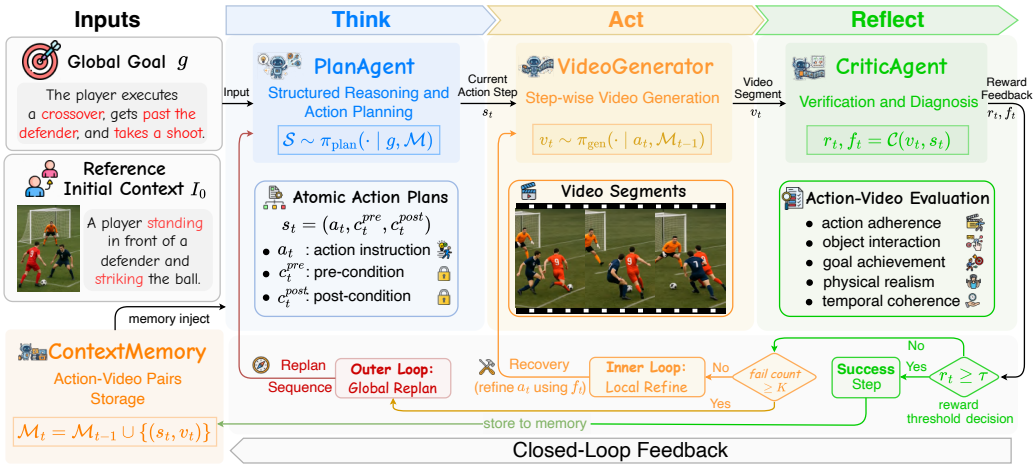


Figure 11: **Closed-Loop Think-Act-Reflect Pipeline.** PlanAgent decomposes the goal and reference context into atomic action plans, VideoGenerator executes each step into video segments, and CriticAgent verifies action-video alignment. Successful steps are stored in ContextMemory, while critic feedback triggers *local refinement* or *global replanning* for **closed-loop inference-time correction**.

- **Goal and Reference Initialization.** The system receives the textual goal g and the reference initial state, which define the target task and the starting scene configuration.
- **PlanAgent Planning.** The PlanAgent π_{plan} observes the goal and current context memory $\mathcal{M}_{t-1}$, then decomposes the task into structured atomic steps $\mathcal{S} = \{s_1, \ldots, s_T\}$. Each step specifies an action instruction together with pre- and post-conditions that make the plan executable.
- **VideoGenerator Execution.** For the current step s_t, the VideoGenerator π_{gen} synthesizes a video segment v_t conditioned on the action instruction and the accumulated visual context. This produces a local execution attempt for the planned action.
- **CriticAgent Verification.** The CriticAgent $\mathcal{C}$ evaluates whether v_t faithfully executes s_t, checking action adherence, object interaction, goal achievement, temporal coherence, and physical realism. It outputs a reward score r_t and textual feedback f_t that diagnoses any observed failure.
- **Feedback for Local Refinement.** If the score falls below the success threshold but the plan remains feasible, the feedback is used to locally refine the current action instruction. The VideoGenerator then regenerates the same step with a more explicit or corrected instruction, such as emphasizing an unmet precondition or post-condition.
- **Feedback for Global Replanning.** If repeated local refinement fails, the error is treated as a planning-level issue. The CriticAgent feedback is passed back to the PlanAgent, which replans from the failed step s_t onward while preserving the successful prefix stored in ContextMemory.

After a step is verified as successful, its action-video pair is appended to ContextMemory, $\mathcal{M}_t = \mathcal{M}_{t-1} \cup \{(s_t, v_t)\}$. The system then repeats the Think-Act-Reflect loop for subsequent steps, applying local refinement or global replanning whenever necessary, until the complete goal is achieved.

B.5 Training-Time GRPO-based Self-Evolving

Figure 12 summarizes the **training-time self-evolving procedure**. While inference-time feedback can correct individual failures, our goal is to internalize such corrections into the VideoGenerator itself. We therefore optimize the supervised VideoGenerator π_{gen} with Group Relative Policy Optimization (GRPO), using PlanAgent-generated action steps as tasks and CriticAgent scores as rewards.

The training loop follows a closed-loop think-act-reflect structure:

- **Plan Sampling:** PlanAgent samples a structured action sequence $\mathcal{S}$ from a training goal g.
- **Group Rollouts:** For each action step s_t, the current VideoGenerator policy samples a group of G candidate video segments $\{v_{t,i}\}_{i=1}^{G}$.
- **Critic Evaluation:** CriticAgent evaluates rewards $\{r_i\}_{i=1}^{G}$ based on action-video alignment.
- **GRPO Optimization:** GRPO normalizes rewards within the group, computes relative advantages, and updates the VideoGenerator while regularizing it toward the supervised initialization.

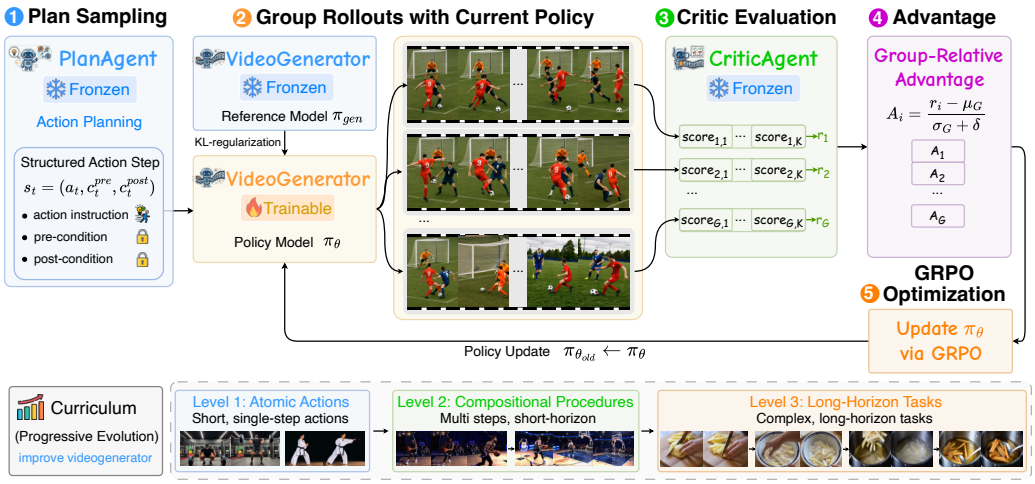


Figure 12: **Training-Time Self-Evolving via GRPO.** PlanAgent samples action steps, VideoGenerator produces group rollouts, and CriticAgent assigns rewards for GRPO-based policy optimization. A *curriculum* gradually increases task complexity from atomic actions to long-horizon procedures, enabling VideoGenerator to *internalize* planning and verification signals for **progressive-evolution**.

Below, we provide the mathematical details of this procedure. We first cast the denoising process inside VideoGenerator as a Markov Decision Process (MDP), then introduce stochastic exploration via SDE sampling, and finally define the GRPO objective used to update the VideoGenerator.

Video Generation as a Markov Decision Process. While the high-level framework operates on plan steps $t \in \{1, \dots, T\}$, the low-level generation of a single video segment v_t involves an iterative denoising process. We formulate this inner loop as a finite-horizon MDP defined by the tuple $\mathcal{M}_{\text{denoise}} = \langle \mathbf{S}, \mathbf{A}, P, R \rangle$:

- **State Space (S):** A state is defined as $\mathbf{x}_\tau = (\mathbf{c}_t, \tau, \mathbf{z}_\tau)$, where τ represents the current denoising timestep (from K down to 0), $\mathbf{z}_\tau$ denotes the noisy video latent, and $\mathbf{c}_t = (s_t, \text{Encoder}(\mathcal{M}_{t-1}))$ encodes the atomic plan s_t and memory context $\mathcal{M}_{t-1}$ at the t-th step.
- **Action Space (A):** The action $\mathbf{a}_\tau$ corresponds to the VideoGenerator's prediction at step τ. For a Rectified Flow model, $\mathbf{a}_\tau = \pi_\theta(\mathbf{x}_\tau)$ predicts the velocity field $\mathbf{u}_\tau$, transitioning the state from $\mathbf{z}_\tau$ to $\mathbf{z}_{\tau-1}$.
- **Transition Probability (P):** The transition $P(\mathbf{x}_{\tau-1} \mid \mathbf{x}_\tau, \mathbf{a}_\tau)$ is governed by the sampling solver (defined via SDEs below).
- **Reward Function (R):** We employ a sparse reward structure where signals are provided only at the terminal state $\tau = 0$ (the clean video v_t). The reward is determined by the CriticAgent $\mathcal{C}$:

$$R(\mathbf{x}_\tau) = \begin{cases} \mathcal{C}(v_t, s_t) & \text{if } \tau = 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Stochastic Exploration via SDEs. Standard sampling in Rectified Flow models follows an Ordinary Differential Equation (ODE): $\mathrm{d}\mathbf{z} = \mathbf{u}_t \mathrm{d}\tau$. However, GRPO requires stochastic exploration to generate diverse trajectories $\{v_{t,i}\}_{i=1}^G$ for relative advantage estimation. We therefore follow DanceGRPO [85] and adopt a reverse-time **Stochastic Differential Equation (SDE)** formulation.

Inspired by recent stochastic sampling theories, we introduce a diffusion term into the flow matching process. The reverse SDE for generation is given by:

$$\mathrm{d}\mathbf{z}_\tau = \underbrace{\left(\mathbf{u}_\tau(\mathbf{z}_\tau) - \frac{1}{2}\eta_\tau^2 \nabla_{\mathbf{z}} \log p_\tau(\mathbf{z}_\tau) \right)}_{\text{Drift Term}} \mathrm{d}\tau + \underbrace{\eta_\tau \mathrm{d}\mathbf{w}}_{\text{Diffusion Term}} \tag{7}$$

where $\mathbf{u}_\tau$ is the velocity predicted by π_θ, $\mathrm{d}\mathbf{w}$ denotes a standard Wiener process (Brownian motion), and η_τ is a time-dependent scalar controlling stochasticity. The score term $\nabla \log p_\tau(\mathbf{z}_\tau)$ guides the sample towards data distribution, derived from Gaussian assumption $p_\tau(\mathbf{z}_\tau) = \mathcal{N}(\mathbf{z}_\tau \mid \alpha_\tau \mathbf{x}, \sigma_\tau^2 \mathbf{I})$ as:

$$\nabla \log p_\tau(\mathbf{z}_\tau) = -\frac{\mathbf{z}_\tau - \alpha_\tau \mathbf{x}_{\text{pred}}}{\sigma_\tau^2} \tag{8}$$

Algorithm 1 Training-Time Self-Evolving via Closed-Loop GRPO

Require: Supervised VideoGenerator π_{gen}, PlanAgent π_{plan}, CriticAgent $\mathcal{C}$, Dataset of goals $\mathcal{D}_g$
Ensure: Evolved VideoGenerator π_θ initialized from π_{gen}

for iteration $= 1, \ldots, M$ **do**
  Sample global goal $g \sim \mathcal{D}_g$
  Think : Generate structured plan sequence $\mathcal{S} \sim \pi_{\text{plan}}(\cdot \mid g)$
  Initialize ContextMemory $\mathcal{M}_0 = \emptyset$
  Sync sampling policy $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$
  for step $t = 1, \ldots, T$ **do**
    // Iterate through plan steps
    Let current atomic plan $s_t \in \mathcal{S}$
    Act (Shared Noise): Sample initialization $\mathbf{z}_K \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    Generate G video candidates $\{v_{t,i}\}_{i=1}^G$ via $\pi_{\theta_{\text{old}}}$ using SDE solver (Eq. 7)
      $v_{t,i} \leftarrow \text{SolveSDE}(s_t, \mathcal{M}_{t-1}, \mathbf{z}_K)$ // Conditioned on shared noise
    Reflect : Evaluate alignment via CriticAgent (Eq. 3):
      $r_i \leftarrow \mathcal{C}(v_{t,i}, s_t)$ for $i = 1 \ldots G$
    Advantage : Compute A_i using group statistics (Eq. 4):
      $A_i = \frac{r_i - \text{mean}(\{r_1, \ldots, r_G\})}{\text{std}(\{r_1, \ldots, r_G\}) + \delta}$
    Update : Maximize objective $\mathcal{J}(\theta)$ (Eq. 5) via gradient ascent:
      $\mathcal{L}_{\text{surr}} = \frac{1}{G} \sum_{i=1}^G \min\left(\rho_{t,i} A_i, \text{clip}(\rho_{t,i}, 1-\epsilon, 1+\epsilon)\, A_i\right)$
      $\theta \leftarrow \theta + \eta \nabla_\theta \left(\mathcal{L}_{\text{surr}} - \beta D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{gen}})\right)$
    Select best segment $v_t^* = \text{argmax}_{v_{t,i}}(r_i)$
    Update ContextMemory $\mathcal{M}_t \leftarrow \mathcal{M}_{t-1} \cup \{(s_t, v_t^*)\}$
  end for
end for

By solving Eq. (7) numerically, we sample a group of diverse candidates conditioned on the same action step s_t, satisfying the exploration requirement for GRPO.

Shared Noise Initialization. A critical detail is the initialization of the starting noise $\mathbf{z}_K$. As demonstrated by [85], using independent noise for each group member leads to *reward hacking*, where the model overfits to specific noise artifacts. Therefore, we enforce a shared noise strategy:

$$\forall i \in \{1, \ldots, G\}: \quad \mathbf{z}_{K,i} = \mathbf{z}_{\text{shared}}, \quad \text{where } \mathbf{z}_{\text{shared}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{9}$$

This ensures that the variance in rewards r_i mainly reflects differences in stochastic sampling paths (Eq. (7)) rather than unrelated initial noise, stabilizing the gradient estimate.

Objective Function. We optimize the VideoGenerator policy π_θ using GRPO. For a plan step s_t, we sample a group of G trajectories $\{v_{t,i}\}_{i=1}^G$ using the old policy $\pi_{\theta_{\text{old}}}$. CriticAgent assigns rewards $\{r_i\}_{i=1}^G$, and the advantages are computed via group-wise normalization:

$$A_i = \frac{r_i - \mu_G}{\sigma_G + \delta}, \quad \text{where } \mu_G = \frac{1}{G}\sum_{j=1}^G r_j, \quad \sigma_G = \sqrt{\frac{1}{G}\sum_{j=1}^G (r_j - \mu_G)^2} \tag{10}$$

The final objective function maximizes the surrogate gain with importance sampling clipping and KL-regularization:

$$\mathcal{J}(\theta) = \mathbb{E}_{\substack{s_t \sim \pi_{\text{plan}}(\cdot|g) \\ v_{t,i} \sim \pi_{\theta_{\text{old}}}(\cdot|s_t)}} \left[\frac{1}{G} \sum_{i=1}^G \min\left(\rho_{t,i} A_i, \text{clip}(\rho_{t,i}, 1-\epsilon, 1+\epsilon)\, A_i\right) - \beta\, D_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{gen}}) \right] \tag{11}$$

where $\rho_{t,i} = \frac{\pi_\theta(v_{t,i}|s_t)}{\pi_{\theta_{\text{old}}}(v_{t,i}|s_t)}$ is the importance sampling ratio. The KL term $D_{\text{KL}}(\pi_\theta \| \pi_{\text{gen}})$ prevents the updated policy from drifting excessively from the supervised VideoGenerator initialization.

Full Training Algorithm. The complete closed-loop GRPO procedure is summarized in Algorithm 1. It integrates PlanAgent's structured reasoning and CriticAgent's verification signals to iteratively refine the VideoGenerator, turning feedback that was previously used only at inference time into a persistent training signal.

C Additional Details of ActVideoGen-Dataset

C.1 Dataset Annotation and Verification

Annotation process. As introduced in Sec. 4.1, we convert each procedural video into a structured record $\langle g, \text{CoT}, \mathcal{S}, \mathcal{V} \rangle$, where g is the global goal, $\mathcal{V} = \{v_t\}_{t=1}^{T}$ denotes ordered video segments, and $\mathcal{S} = \{s_t\}_{t=1}^{T}$ contains step-level annotations. The construction follows a two-stage pipeline:

- **Stage 1: Step-wise action annotation:** For datasets with temporal step boundaries, each segment v_t is clipped with short pre- and post-context and annotated by GLM-4.5V [31]. The model outputs a valid JSON object containing an atomic instruction, canonical verb-object-tool actions, 1-3 visual preconditions and postconditions, and the preserved time span. For goal-level clips without sub-step boundaries, we first infer ordered sub-steps and temporal spans from the continuous video, then normalize them to the same schema. This stage yields step tuples $s_t = (a_t, c_t^{pre}, c_t^{post})$ that encode both executable actions and physical state transitions.
- **Stage 2: CoT planning generation:** Given the refined goal and ordered step tuples, GPT-5.1 generates the global planning rationale. The prompt exposes the step instructions, actions, preconditions, and postconditions, requiring the model to explain step ordering, dependency relations, and the observable termination condition. Thus, each final annotation contains both grounded action-video pairs and a causal plan connecting them.

Veirification process. To ensure annotation quality, we validate the generated records through a two-step pipeline that combines model-based filtering with human verification:

- **Step 1: VLM-based filtering:** After annotation, we employ a VLM-based filtering pipeline to exclude low-quality samples. Qwen3-VL-235B [24] is used as an independent verifier rather than the original GLM-4.5V annotator. For each candidate step, the verifier checks whether the main action is visible within the specified time span, whether the target objects/tools are correctly involved, whether the postconditions are visually supported, and whether the text introduces hallucinated content. Low-confidence or clearly mismatched samples are discarded.
- **Step 2: Human verification:** After VLM filtering, we ask three human annotators to independently verify each action-video pair. Annotators were shown the video segment, step instruction, action tuple, and pre-/postconditions. A pair is counted as *fully aligned* only if the segment executes the annotated main action, uses the specified objects/tools when applicable, starts from states consistent with the preconditions, and reaches the stated postconditions without contradiction. Missing actions, wrong objects/tools, incorrect temporal localization, or unsupported postconditions are counted as misalignment, while minor wording differences that preserve the executable action and resulting state are allowed. We report pair-level alignment by *majority vote* among the three annotators; under this criterion, about 93% of the sampled pairs are fully aligned.

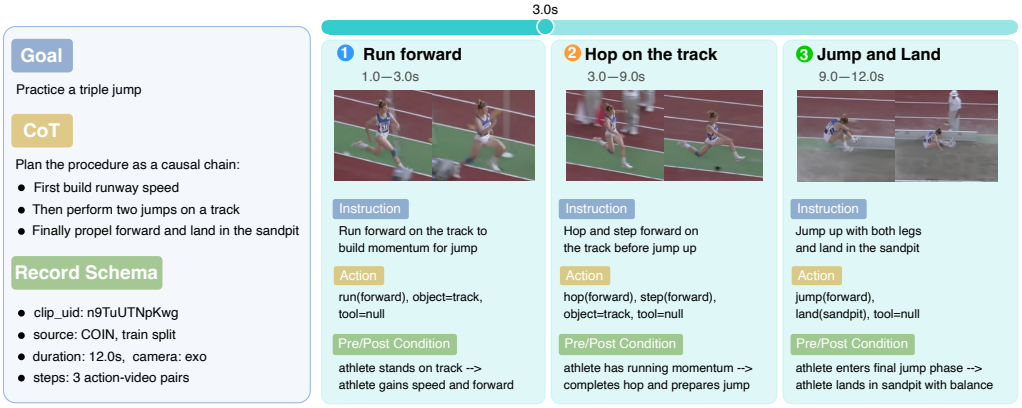
C.2 Dataset Examples and Statistics

Dataset example. Fig. 13 presents three representative annotation examples from ActVideoGen-Dataset, illustrating the breadth of procedural behaviors covered by our dataset:

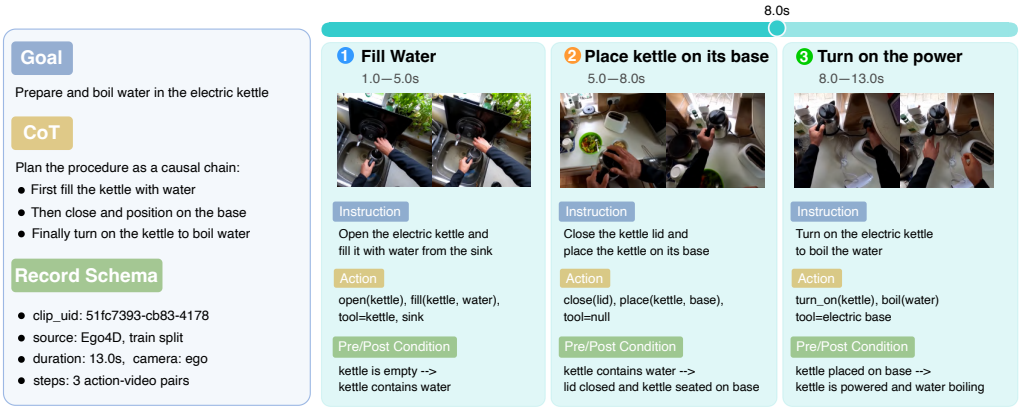
- **Exo-view human kinematics (Example 1):** A triple-jump sequence is decomposed into running, hop-step motion, and final landing.
- **Ego-view state transition (Example 2):** Boiling water with an electric kettle requires filling the kettle, placing it on the base, and turning it on.
- **Ego-view tool manipulation (Example 3):** Removing a bicycle wheel involves loosening the screw with a wrench, removing the screw, and detaching the wheel.

Despite their different visual perspectives and action types, all examples share the same structured annotation format. Each sample is organized around a global goal and a short CoT rationale, followed by temporally grounded action-video pairs. For each step, we provide an executable action instruction, canonical action-object-tool tuple, and pre-/post-conditions, making the annotation directly usable for both planning supervision and step-wise video generation.

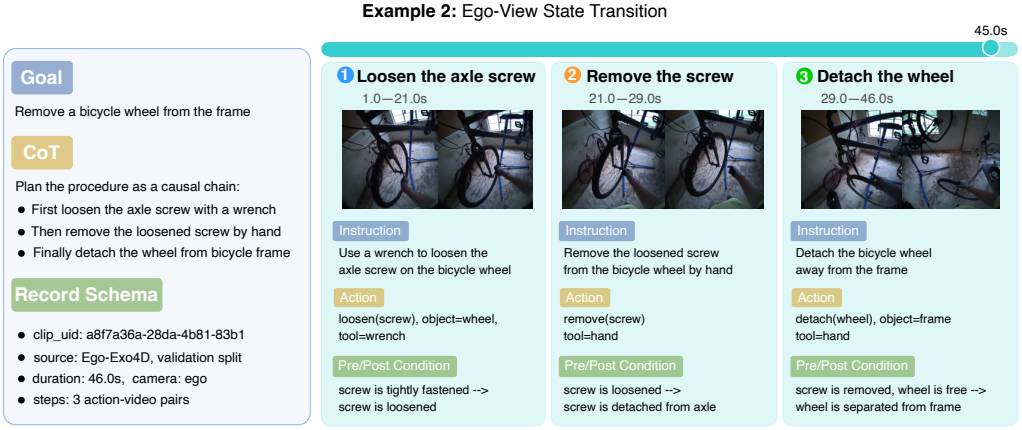
This format preserves the causal structure of procedural activities: earlier postconditions establish the visual and physical context needed by later actions.



Example 1: Exo-View Human Kinematics



Example 2: Ego-View State Transition



Example 3: Ego-View Tool Manipulation

Figure 13: **Detailed annotation example from ActVideoGen-Dataset.** A procedural task is represented by a global goal and CoT planning rationale, then decomposed into temporally grounded action-video pairs. Each step contains an action instruction, an object-tool tuple, and pre-/post-conditions, providing structured supervision for long-horizon action planning and video generation.

Dataset statistics. ActVideoGen-Dataset contains 24,616 procedural tasks and 118,156 step-level action-video pairs, with an average of approximately 4.8 steps per task. In terms of source composition, it aggregates four resources: Ego4D [26, 27], Ego-Exo4D [28], COIN [29], and EPIC-KITCHENS [30], covering both egocentric and exocentric perspectives. For complexity, the dataset spans short atomic procedures as well as longer multi-step activities, with varied task durations, step counts, and action transition patterns. For diversity, the collected tasks span a broad range of scenes and activity categories, including cooking, household manipulation, sports, and other real-world behaviors. Together, these statistics show that ActVideoGen-Dataset provides diverse and causal supervision for training and evaluating structured planning and action-conditioned video generation.

D Additional Details of ActVideoGen-Bench

D.1 Multi-Agent Evaluation Framework

As introduced in Sec. 4.2, ActVideoGen-Bench targets action quality beyond generic visual fidelity metrics. Since procedural videos require verifying ordered step execution, post-condition satisfaction, object interaction, and physical realism, single-turn QA evaluation is often insufficient. We therefore design a **multi-agent evaluation framework** based on the Chain-of-Query mechanism [86]. The framework couples a VLM-based host evaluator with two LLM assistant auditors: the host grounds the evaluation in video frames and action data, the auditors raise complementary verification queries, and the final scoring module parses structured outputs into video-level metrics.

The evaluation follows a four-stage pipeline:

- **Stage 1: Video grounding.** A video description agent (VLM) observes the sampled frames and produces a chronological description grounded in the required action steps and post-conditions.
- **Stage 2: Query generation.** Two assistant agents (LLMs) generate complementary verification queries. The action-completeness auditor checks action execution and post-condition satisfaction, while the multi-dimensional quality auditor focuses on object interaction, motion smoothness, and physical fidelity.
- **Stage 3: Evidence-based answering.** An answer agent (VLM) revisits the video frames with the synthesized chain of queries and provides evidence-based answers, including step-wise completion judgments and quality-related observations.
- **Stage 4: Final scoring.** A score agent (VLM) integrates the initial description, query-answer history, original prompt, and action decomposition data to output structured metrics.

This collaborative pipeline enables cross-verified evaluation of both semantic action completion and fine-grained visual quality, providing robust action quality evaluation.

D.2 Evaluation Metrics Definition

The evaluator produces four action-quality metrics, denoted as `AC`, `AS`, `OI`, and `PF`. These metrics serve as the action verification criteria introduced in Sec. 4.2, and support the video generation analyses in Sec. 5.3. The final score agent outputs them in a structured `<METRICS>` dictionary, and all reported scores are normalized to $[0, 1]$.

- **Action Completeness (AC).** Measures whether the generated video completes the required action sequence. A step is counted as complete only if the key action is visibly executed and all specified post-conditions are met. This is reported as the ratio X/Y, where X represents the number of successfully completed steps and Y is the total number of required steps.
- **Action Smoothness (AS).** Evaluates the temporal continuity of the generated actions. It accounts for transition quality between steps, motion continuity, inter-frame consistency, and the absence of abrupt cuts or jumps. The metric is evaluated on a 1–5 scale and normalized to $[0, 1]$ for reporting.
- **Object Interaction (OI).** Assesses the correctness and realism of interactions between the subject and any tools or objects. Key evaluation factors include contact quality, tool-use accuracy, object persistence, and the absence of clipping or floating artifacts. If no object interaction is involved, it is marked as `N/A`; otherwise, it is scored on a 1–5 scale and normalized to $[0, 1]$.
- **Physical Fidelity (PF).** Measures adherence to real-world physical constraints and natural motion patterns. It focuses on gravity, contact dynamics, and object deformation, penalizing artifacts like unnatural morphing or clipping. Scores are initially assigned on a 1–5 scale, ranging from severe physics violations to realistic body mechanics, and normalized to $[0, 1]$.

D.3 Case Studies of Evaluation

Fig. 14 shows two representative cases, covering successful and failed action execution. Each case includes the action decomposition, visual evidence, step-wise evaluation, and final judgment.

These cases demonstrate that our evaluator provides interpretable step-level diagnoses rather than only coarse video-level scores. It assigns high scores when the visual evidence satisfies the required post-conditions and penalizes concrete failures, such as missing steps or unrealistic motion, providing robust action-quality metrics for our ActVideoGen-Bench.

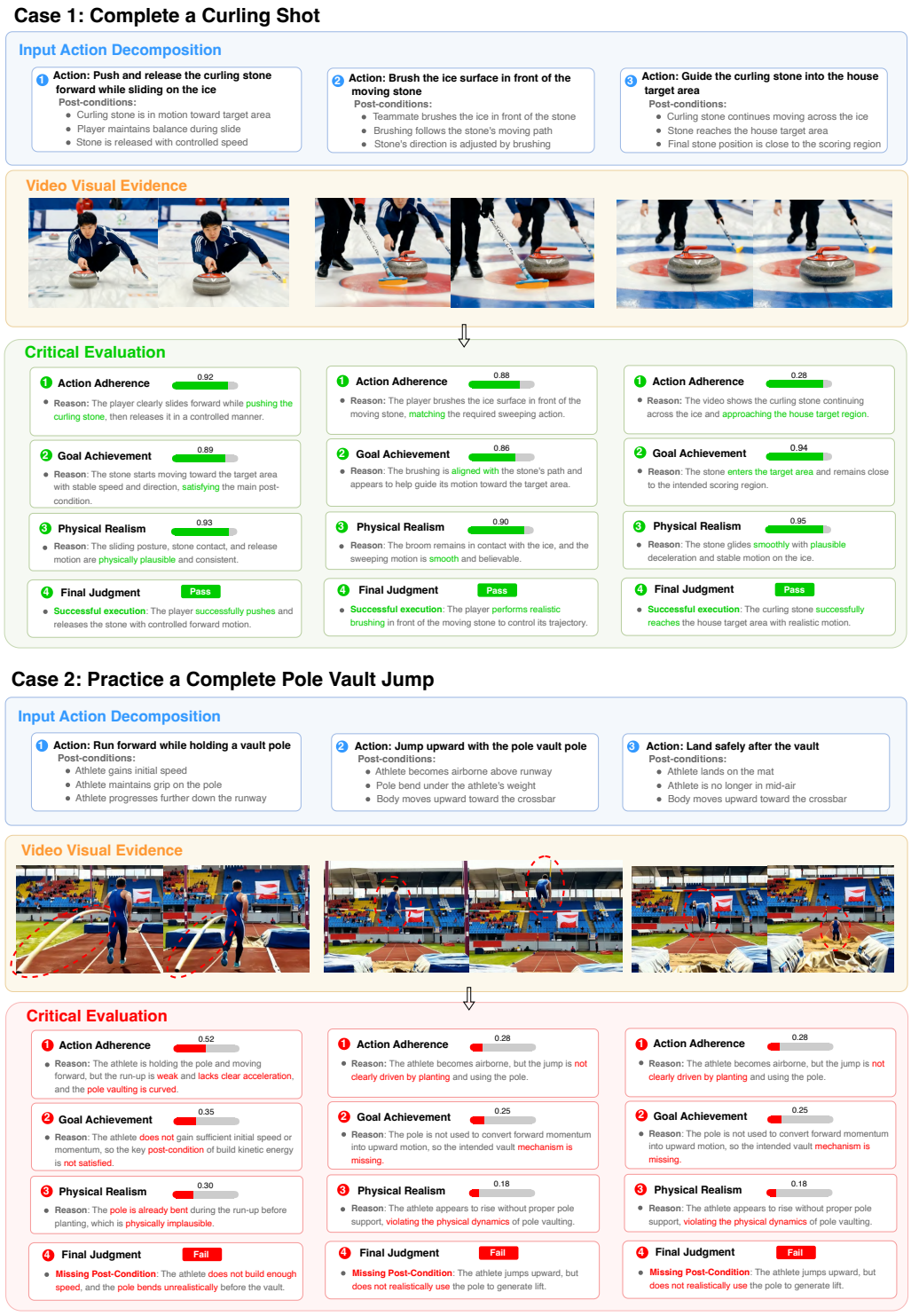


Figure 14: **Case studies of our multi-agent evaluator.** The evaluator grounds step-wise judgments in action decompositions and visual evidences, providing multi-dimensional assessments of action quality. Successful executions satisfy post-conditions, while failed actions reveal implausible motions.

E Additional Quantitative Results

This section further examines SPIRAL's generalization, efficiency, and reliability. We assess generator generalization after SFT, evaluator stability across independent sessions, inference-time quality-cost trade-offs, human evaluation and evaluator alignment, and robustness to planner and critic errors.

Table 10: **VideoGenerator Generalization before and after SFT.** We evaluate the pre- and post-SFT SVI models [25] on official SVI general prompts.

Method	Aesthetic Quality	Imaging Quality	Subject Consistency	Motion Smoothness	Dynamic Degree
Pre-SFT (Base)	63.84%	71.88%	98.13%	98.93%	17.61%
Post-SFT (Ours)	63.41%	73.07%	98.30%	99.05%	18.52%
Δ	-0.43%	+1.19%	+0.17%	+0.12%	+0.91%

Table 11: **Evaluator Consistency Across Independent Sessions.** We re-score 150 samples on action quality in two independent sessions.

Metric	Action Compl.	Action Smooth.	Object Interact.	Physical Fidelity	Overall Quality
Agreement	88.5%	76.2%	79.5%	82.0%	81.0%
Pearson r	0.812	0.685	0.764	0.792	0.753

Table 12: **Inference Cost and Generation Quality.** Module-level and end-to-end measurements on the same NVIDIA A100 GPUs, including runtime, memory, and action quality. The single-shot baseline generates 5s videos; SPIRAL generates 15s videos at different retry budgets.

Setting	Model	Inference Cost and Output			Action Quality				
		Avg. Time (s)	Memory (GB)	Generated Video Duration	Action Compl.	Action Smooth.	Object Interact.	Physical Fidelity	Overall Quality
Single-shot baseline	LongLive-2B [14]	8	28	1 step / 5s	0.273	0.646	0.776	0.808	0.626
PlanAgent	Qwen3-VL-8B [24]	4	19	–	–	–	–	–	–
VideoGenerator (per step)	LongLive-2B [14]	8	28	1 step / 5s	–	–	–	–	–
CriticAgent (per step)	Qwen3-VL-8B [24]	7	22	–	–	–	–	–	–
SPIRAL (steps=3, Retry=1)	Plan+Gen+Critic	49	69	3 steps / 15s	0.721	**0.902**	0.860	0.924	0.852
SPIRAL (steps=3, Retry=2)	Plan+Gen+Critic	64	69	3 steps / 15s	**0.786**	0.862	**0.872**	**0.930**	**0.867**

E.1 VideoGenerator Generalization

We examine whether SFT on our ActVideoGen-Dataset causes the VideoGenerator to overfit to planner-generated prompts. To test this, we compare the **base SVI model** [25] with our **SFT-tuned model** using the official SVI evaluation prompts on general scenarios.

As shown in Table 10, general video quality remains stable across all dimensions, with $|\Delta| < 1.2\%$. The post-SFT model slightly improves imaging quality and dynamic degree, while showing only a minor decrease in aesthetic quality. These results suggest that LoRA-based SFT preserves general generation capability while specializing the VideoGenerator for action-conditioned generation.

E.2 Evaluator Stability

As detailed in Appendix D, we use a **multi-agent evaluation framework** for action quality assessment. This design reduces single-inference variance by requiring multiple agents to ground their judgments in shared visual evidence. To test evaluator stability, we re-score 150 samples across two independent sessions and report agreement rates and Pearson correlations.

Table 11 shows that agreement rates range from 76.2% to 88.5%, while Pearson correlations range from 0.685 to 0.812. These results indicate that the multi-agent evaluator is reasonably stable for ActVideoGen-Bench, especially for fine-grained action-conditioned video assessment.

E.3 Inference-Time Cost

End-to-End Inference Cost. Table 12 reports module-level and end-to-end costs on the same NVIDIA A100 GPUs. LongLive-2B generates one 5s segment in 8s. SPIRAL generates three steps totaling 15s in 49s (Retry=1) or 64s (Retry=2), using 69GB in both cases. The additional runtime covers both longer outputs and planning/verification; the step and retry budgets provide explicit control over inference cost.

Quality-Cost Trade-off. For the same 15s output duration, Retry=2 adds 15s of runtime and raises action completeness from 0.721 to 0.786, object interaction from 0.860 to 0.872, and physical fidelity from 0.924 to 0.930. Overall quality rises from 0.852 to 0.867 without increasing GPU memory. Smoothness decreases from 0.902 to 0.862, revealing a completion-smoothness trade-off: additional correction favors action fulfillment, while Retry=1 offers lower latency and smoother execution.

Table 13: **Comparison of Human and Automatic Evaluation Results.** Scores from three blinded human raters and the GPT-5-based VLM evaluator on 150 videos from ActVideoGen-Bench, covering the baseline, closed-loop SPIRAL, and GRPO-SPIRAL. Higher is better for all metrics.

Method	Action Compl.		Action Smooth.		Object Interact.		Physical Fidelity		Overall Quality	
	VLM	Human	VLM	Human	VLM	Human	VLM	Human	VLM	Human
Baseline	0.273	0.352	0.646	0.721	0.776	0.739	0.808	0.861	0.626	0.668
+ Closed-loop SPIRAL	0.721	0.766	0.902	0.831	0.860	**0.905**	0.924	0.879	0.852	0.845
+ GRPO-SPIRAL	**0.841**	**0.785**	**0.942**	**0.965**	**0.864**	0.826	**0.936**	**0.957**	**0.896**	**0.883**

Table 14: **VLM–Human Alignment.** Agreement and correlations on 150 video samples.

Metric	Action Compl.	Action Smooth.	Object Interact.	Physical Fidelity	Overall Quality
Agreement	80.7%	74.0%	75.3%	79.8%	78.0%
Pearson r	0.762	0.694	0.641	0.708	0.746
Spearman ρ	0.748	0.635	0.626	0.704	0.725

Table 15: **Robustness Analysis.** Effects of planner errors, critic errors, and disabled feedback.

Setting	Action Compl.	Action Smooth.	Object Interact.	Physical Fidelity
Full SPIRAL	**0.781**	**0.862**	**0.872**	**0.934**
PlanAgent-Noise	0.747	0.829	0.870	0.905
CriticAgent-Noise	0.719	0.834	0.853	0.914
Feedback disabled	0.628	0.754	0.838	0.840

E.4 Human Evaluation and Evaluator Alignment

Evaluation Setup. Three blinded raters evaluated 150 videos using the action-quality criteria in Appendix D. Figure 15 illustrates the scoring interface, which presents videos, action descriptions, post-conditions, and metric-specific guidance. Table 13 extends Table 5 by pairing their judgments with automatic scores for the baseline, closed-loop SPIRAL, and GRPO-SPIRAL.

Human Evaluation Results. Closed-loop SPIRAL raises human-rated action completeness from 0.352 to 0.766 and overall quality from 0.668 to 0.845. GRPO further improves overall quality to 0.883, with clear gains in smoothness (0.831 to 0.965) and physical fidelity (0.879 to 0.957). Both evaluators rank the methods identically on overall quality, supporting benefits beyond automatic scores. Human-rated object interaction decreases from 0.905 to 0.826 after GRPO, indicating that its gains are not uniform across criteria.

Alignment with Human Judgments. Table 14 reports sample-level VLM–human consistency. Action completeness achieves 80.7% agreement, Pearson $r = 0.762$, and Spearman $\rho = 0.748$; overall quality also shows positive score and rank correlations ($r = 0.746$, $\rho = 0.725$). These results support alignment with human judgments, with lower correlations for object interaction.

Human Validation of GRPO Improvements. GRPO uses rewards from the frozen Qwen3-VL-8B CriticAgent; the GPT-5-based evaluator is not used for optimization. Shared-noise initialization controls starting-noise variation, while KL regularization limits drift from the reference policy (Appendix B.5). Together with the human-rated gains, this separation supports improvements beyond the training critic's scores, although it does not fully rule out reward bias.

E.5 Robustness to Planner and Critic Errors

Perturbation Settings. Using the setting of Table 6, *PlanAgent-Noise* randomly drops or swaps plan actions, and *CriticAgent-Noise* flips a subset of success/failure judgments. Each perturbs only the named agent and retains both feedback loops. *Feedback disabled* (CriticAgent-Invalid) uses unperturbed planning with local refinement and global replanning disabled.

Robustness Results. In Table 15, planner and critic noise reduce action completeness from 0.781 to 0.747 and 0.719: absolute drops of 0.034 and 0.062. Both perturbed settings outperform disabled feedback on all four metrics, suggesting that imperfect feedback remains useful for recovery. The larger drop under critic noise highlights the importance of reliable verification for accepting or revising generated segments.

Role of Closed-Loop Feedback. Disabling feedback reduces completeness by 0.153 and physical fidelity by 0.094, matching trained-PlanAgent-only generation in Table 6. This larger degradation supports the value of local refinement and global replanning beyond trained planning alone. The aggregate results suggest resilience to the tested agent errors, while their propagation within individual trajectories remains uncharacterized.

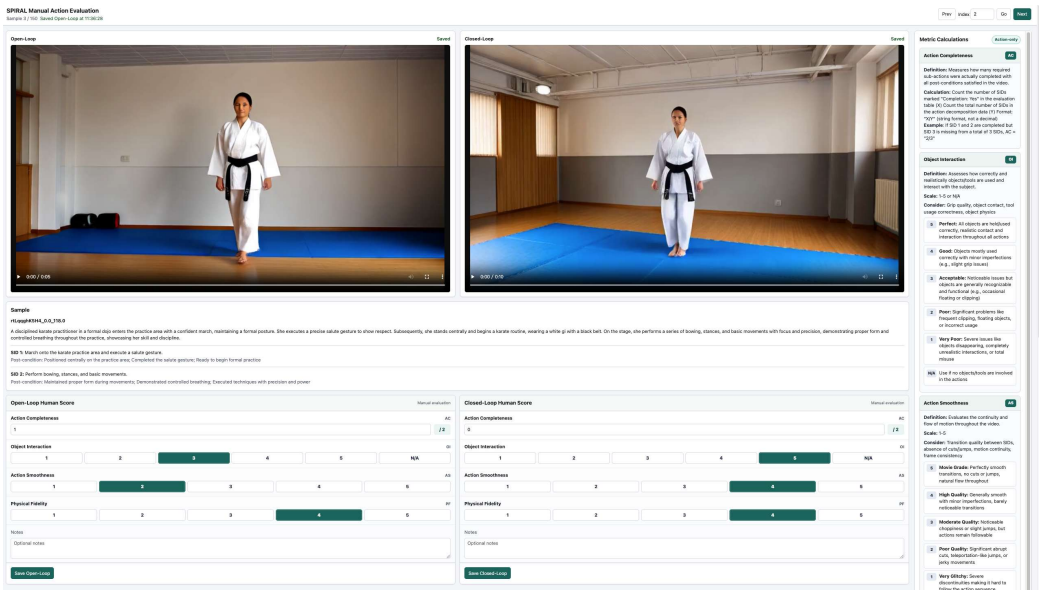


Figure 15: **Human Evaluation Interface.** The interface presents videos, action descriptions, and post-conditions, with scoring controls for action completeness, object interaction, action smoothness, and physical fidelity. The right panel provides metric definitions and rating guidelines.

F Additional Qualitative Results

Complementing the quantitative results in Sec. 5, we visualize SPIRAL's end-to-end pipeline (Sec. F.1), closed-loop error correction (Sec. F.2), and long-horizon and ultra-long procedural generation across diverse actions (Sec. F.3). We also compare single-shot and closed-loop generation (Sec. F.4) and illustrate the effects of GRPO-based self-evolution (Sec. F.5).

F.1 End-to-End Pipeline Trace

Figure 16 illustrates a complete execution trace of SPIRAL for a long-horizon user goal, covering the full process from task decomposition to final video composition.

- **PlanAgent:** Starting from the high-level input instruction, PlanAgent decomposes the goal into a sequence of executable action steps with explicit preconditions and postconditions, providing a structured procedural plan for long-horizon generation.
- **VideoGenerator:** Given each planned step and the current visual context, VideoGenerator synthesizes a local video segment that executes the corresponding action while preserving continuity with previously verified segments.
- **CriticAgent:** After each generation step, CriticAgent evaluates whether the video segment satisfies the planned action and returns a pass/fail judgment with corrective feedback when failures occur.

This highlights the interpretability of the **plan-generate-critic loop**. Instead of producing a long video in one pass, SPIRAL exposes intermediate plans, per-step generations, critic judgments, and accumulated visual states. The verified segments are composed into a coherent, long-horizon video, providing a transparent mechanism for reliable procedural generation.

F.2 Closed-Loop Feedback Refinement

Figure 17 visualizes the closed-loop feedback refinement process on a long-horizon soccer procedure, where the goal is decomposed into dribbling forward, bypassing defender, and shooting into the goal.

- **Failure Detection:** After each generated step, CriticAgent evaluates action adherence, goal achievement, and physical realism. In Step 2, the first generation loses ball control while bypassing the defender, violating the planned post-condition and causing the segment to fail.

- **Local Feedback Refinement:** Based on the critic feedback, SPIRAL explicitly refines the current instruction with the missing post-condition. The regenerated Step 2 corrects the failure while maintaining temporal continuity, allowing safely continue to the final shooting step.

Overall, this example shows how the closed-loop mechanism prevents local errors from accumulating across a long action chain. By detecting failures immediately and refining only the problematic step, SPIRAL preserves the successful prefix and composes a coherent final multi-action video.

F.3 Long-Horizon Procedure Generation

Figure 18 and Figure 19 present long-horizon procedural generation results across diverse action types and extended task lengths.

- **Diverse Action Modes:** Figure 18 shows two representative procedural generation modes. Third-person human kinematics focuses on full-body motion control, such as sports behaviors, while egocentric behavior generation emphasizes first-person action execution and interaction intent.
- **Ultra-Long Procedural Chains:** Figure 19 further evaluates SPIRAL on ultra long-horizon kitchen procedures (>30s video duration). SPIRAL demonstrates strong preservation in scene layout, object states, and action dependencies over a substantially longer horizon.

Overall, these results demonstrate the importance of explicit step decomposition for long-horizon action-conditioned generation. By restricting each segment to a focused atomic action and maintaining context across steps, SPIRAL can translate abstract procedural goals into temporally ordered videos while preserving identity, viewpoint consistency, and procedural continuity.

F.4 Comparison with Open-Loop Generation

We present qualitative comparisons in Fig. 20 and Fig. 21 across long-horizon procedural tasks to demonstrate the advantage of closed-loop generation.

- **Open-Loop Baseline:** The baseline must generate the entire procedure from a single prompt, requiring the model to implicitly infer all intermediate states and action dependencies at once. As shown in the comparisons, this often leads to missing actions, mixed actions, incorrect step ordering, and physically implausible transitions, especially when the target procedure contains a long chain of dependent actions.
- **Closed-Loop SPIRAL:** SPIRAL explicitly decomposes high-level goals into executable action steps and verifies the generated result after each step. This planning-and-verification loop prevents local failures from propagating, preserves procedural continuity, and enables more complete long-chain action generation with coherent temporal ordering and higher procedural fidelity.

Overall, these comparisons show that long-horizon procedural generation benefits from explicit closed-loop control. By replacing single-shot generation with step-wise planning and verification, SPIRAL improves both local action correctness and global procedural consistency.

F.5 GRPO-Based Self-Evolution Visualization

We further provide the comparison in Fig. 22 to demonstrate the effect of GRPO-based self-evolving on long-horizon action generation.

- **Without Self-Evolving:** The generator can still suffer from incomplete actions, unstable motion, inconsistent object interactions, and physically implausible intermediate states. These failures indicate that supervised tuning alone may not fully internalize the action-following and verification signals required for reliable long-horizon execution.
- **GRPO-Based Self-Evolving:** By optimizing the generator with critic-derived rewards, GRPO encourages more complete action execution, smoother temporal transitions, and higher physical plausibility. As a result, the generator better follows procedural instructions and demonstrates stronger intrinsic long-horizon generation capability.

Overall, GRPO-based self-evolving complements inference-time correction by improving the generator itself. The resulting model better internalizes critical feedback, leading to more reliable action execution and smoother long-horizon procedural videos.

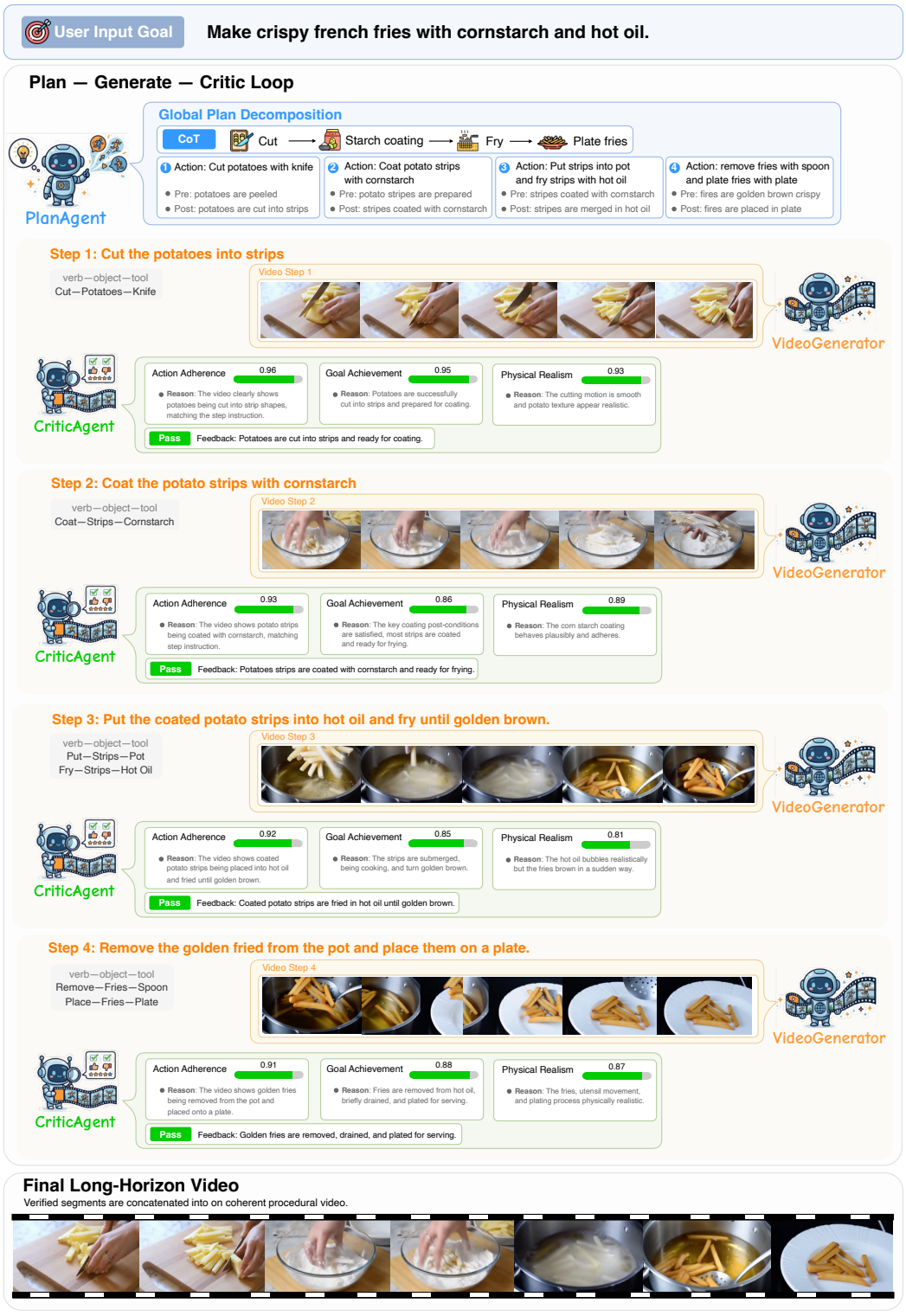


Figure 16: **End-to-end pipeline of SPIRAL.** Given a user goal, PlanAgent first decomposes the task into step-wise actions with explicit pre-/post-conditions. At each step, VideoGenerator synthesizes a video segment, while CriticAgent evaluates action-video alignment and returns judgment with feedback. Verified segments are sequentially accumulated to compose the final long-horizon video.



Figure 17: **Closed-loop feedback refinement.** For the soccer procedure, CriticAgent detects that the Step 2 first generation bypasses the defender but loses ball control, violating the planned post-condition. SPIRAL uses this feedback to locally refine the action instruction, regenerate a successful crossover while preserving possession, and continue to final shooting step without propagating error.

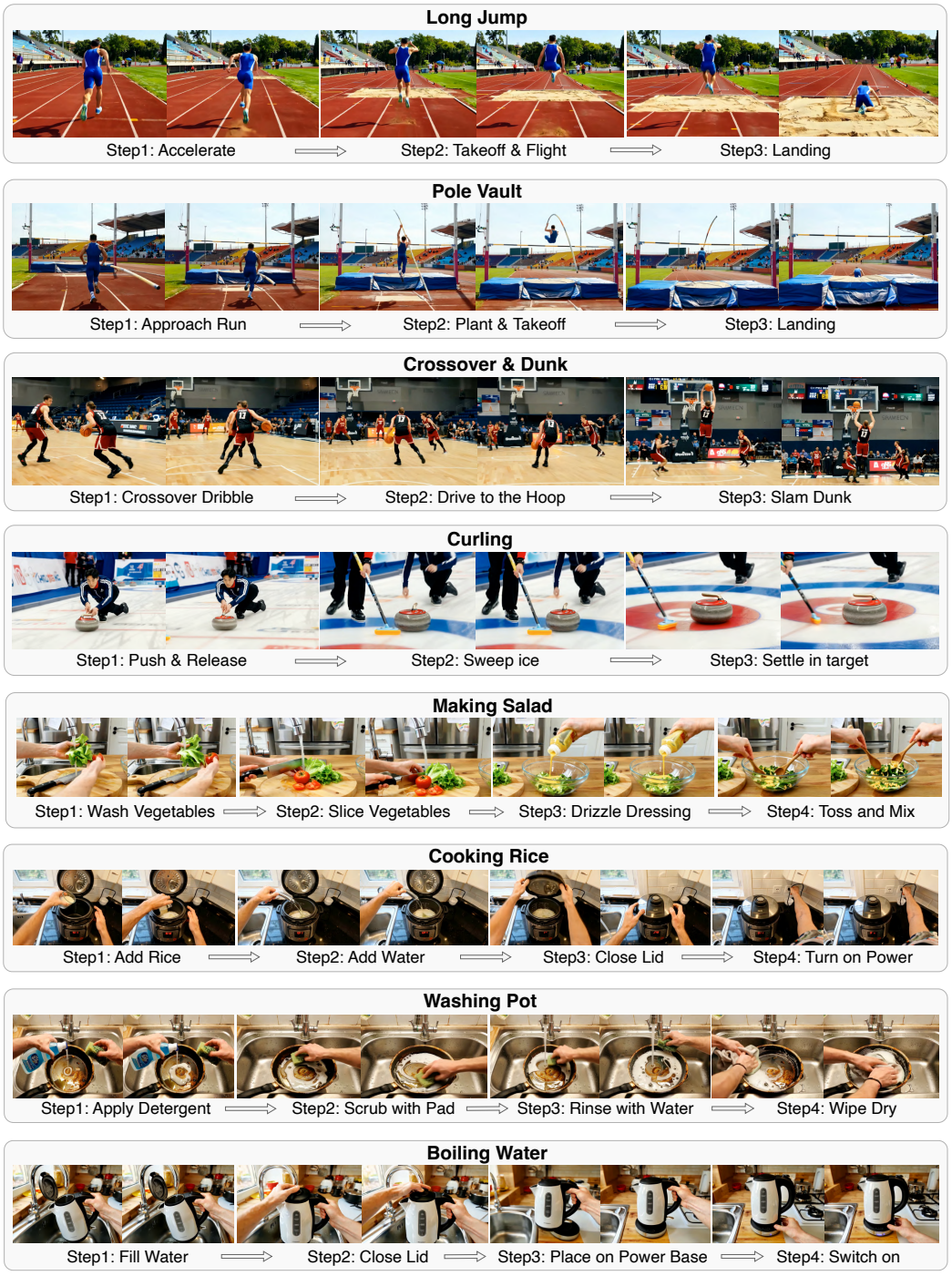


Figure 18: **Procedural action video generation results.** We visualize two procedural generation modes: *third-person human kinematics*, where actions control full-body motion (e.g., sports behaviors); and *egocentric behavior generation*, where actions are executed from a first-person perspective.



Figure 19: **Ultra-long procedural generation results.** We visualize two eight-step kitchen procedures: tomato preparation and storage, and making tomato and cucumber salad. SPIRAL maintains coherent scene context, object states, and action ordering across extended multi-step manipulation.

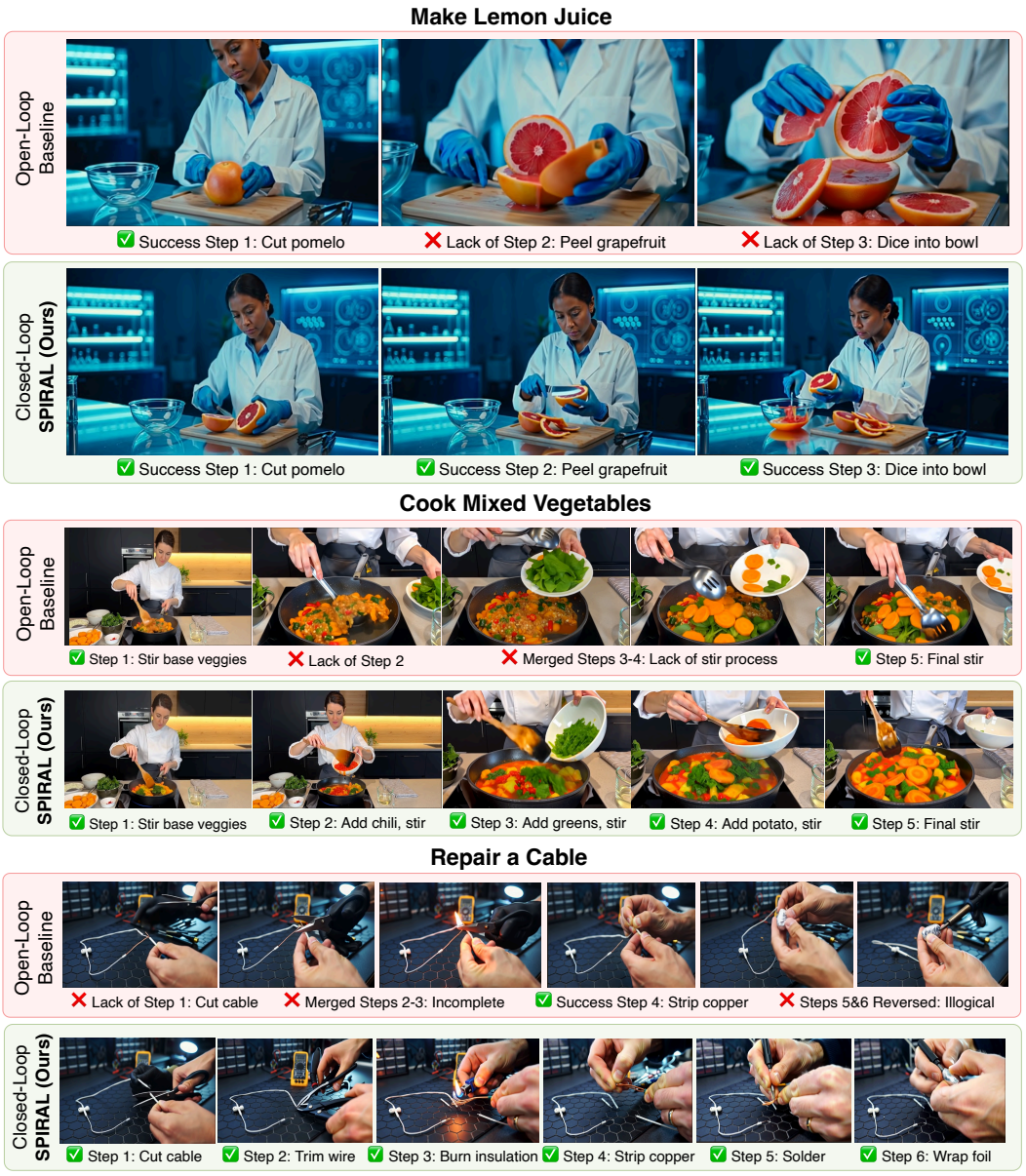


Figure 20: **Single-shot open-loop vs. closed-loop generation on long-horizon procedural tasks.** The open-loop baseline often fails to complete the full procedure, exhibiting missing actions, incorrect ordering, and implausible transitions. In contrast, SPIRAL leverages explicit planning and verification, generating complete and long-chain action sequences with coherent ordering and procedural fidelity.

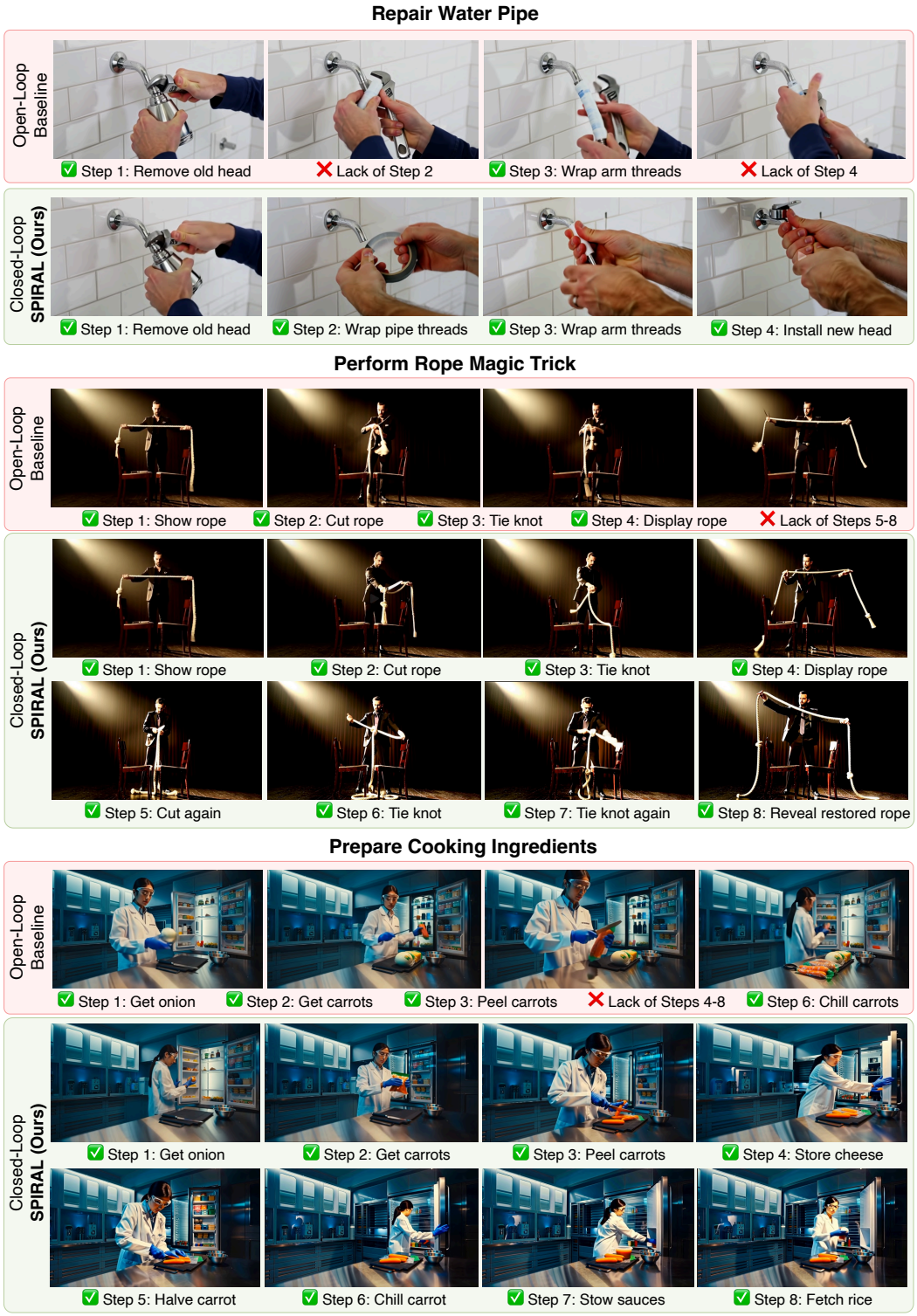


Figure 21: **Single-shot open-loop vs. closed-loop generation on long-horizon procedural tasks.** The open-loop baseline often fails to complete the full procedure, exhibiting missing actions, incorrect ordering, and implausible transitions. In contrast, SPIRAL leverages explicit planning and verification, generating complete and long-chain action sequences with coherent ordering and procedural fidelity.

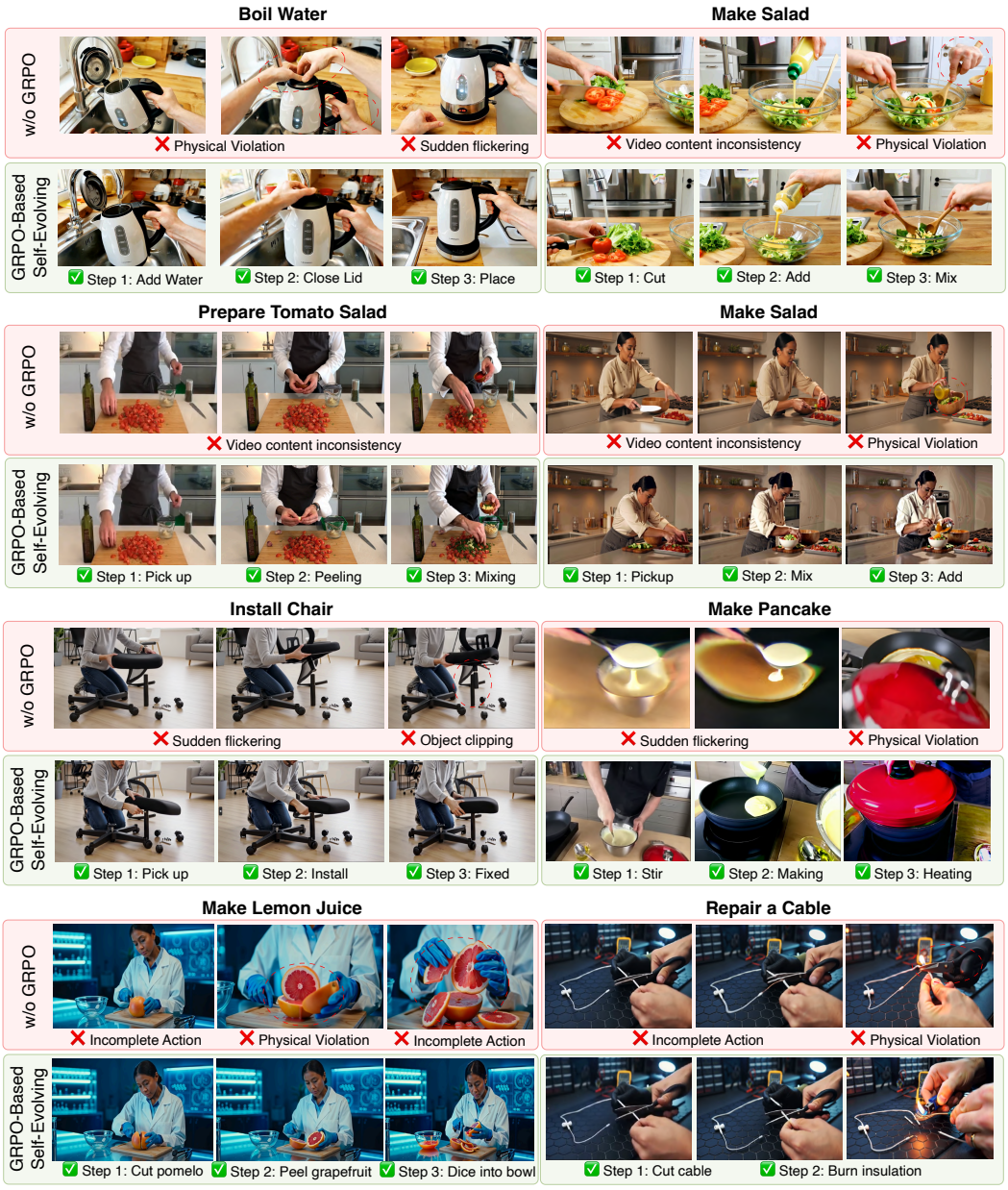


Figure 22: **Effect of GRPO-based self-evolving.** Compared with the model without GRPO, GRPO-based self-evolving improves action completion, temporal coherence, and physical plausibility across long-horizon procedural generation by leveraging critic-derived reward signals during post-training.

G Potential Societal Impact & Limitations

G.1 Societal Impact

SPIRAL aims to improve controllable long-horizon video generation by explicit action planning, intermediate verification, and corrective feedback. Its societal impacts include both potential benefits and risks.

- **Positive impact.** SPIRAL can support content creation, education, and procedural simulation by synthesizing step-wise videos from high-level goals. Its explicit plan-generate-critic loop also provides interpretable traces, helping users diagnose action failures. It also encourages evaluation beyond generic visual fidelity toward action execution, object interaction, and physical consistency, bridging the gap toward more advanced action-conditioned video world models.
- **Negative impact.** More controllable and temporally coherent video generation may increase risks of misleading synthetic media, impersonation, or unsafe instructional content if deployed without safeguards. Models may also inherit social biases, cultural stereotypes, or privacy-sensitive patterns from large-scale video data. Practical deployment should therefore include prompt filtering, provenance or watermarking mechanisms, human review for sensitive domains, and restrictions on harmful or deceptive use cases.

G.2 Known Limitations

While SPIRAL demonstrates strong performance in long-horizon action-conditioned video generation, it still has several limitations.

- **Inference cost.** The closed-loop think–act–reflect process introduces additional system complexity and inference latency compared with single-shot generation. A promising direction is to amortize planning and verification into the video generator itself.
- **Action representation.** The current formulation focuses on sequential, high-level semantic actions and does not explicitly model fine-grained control signals or tightly coupled physical interactions. Richer action representations and tighter low-level motion modeling remain open directions.
- **Automatic evaluation.** Our evaluation protocol relies on VLM-based agents. Although Appendix E.2 shows reasonable consistency, automatic evaluation can still be affected by model bias, imperfect visual grounding, and ambiguity in open-ended procedural tasks.
- **Generalization scope.** Our experiments mainly focus on the distribution covered by ActVideoGen-Dataset and ActVideoGen-Bench. Generalization to specialized domains, safety-critical procedures, rare object interactions, or culturally diverse activities remains to be systematically studied.

Addressing these limitations is essential for building more robust, efficient, and responsibly deployable long-horizon video generation models.

H Public Resources Used

H.1 Public Datasets Used

- Ego4D[1] ..... Ego4D Asset License Agreement
- Ego-Exo4D[2] ..... Ego-Exo4D Asset License Agreement
- COIN[3] ..... COIN Asset License Agreement
- EPIC-KITCHENS[4] ..... CC BY-NC 4.0
- VideoVerse[5] ..... CC BY-NC 4.0
- GAIA[6] ..... Apache License 2.0

H.2 Public Implementations Used

- Qwen3[7] ..... Apache License 2.0
- Qwen3-VL[8] ..... Apache License 2.0
- GLM-4.5V[9] ..... Apache License 2.0
- EgoPlan-Bench[10] ..... BSD 3-Clause License
- VideoGen-RewardBench[11] ..... MIT License
- VBench[12] ..... Apache License 2.0
- LLaMA-Factory[13] ..... Apache License 2.0
- LongLive[14] ..... Apache License 2.0
- SVI[15] ..... MIT License
- SANA-Video[16] ..... Apache License 2.0
- Wan2.1[17] ..... Apache License 2.0
- Wan2.2[18] ..... Apache License 2.0
- DanceGRPO[19] ..... Apache License 2.0

[1] https://ego4d-data.org/
[2] https://ego-exo4d-data.org/
[3] https://coin-dataset.github.io/
[4] https://epic-kitchens.github.io/
[5] https://github.com/Zeqing-Wang/VideoVerse
[6] https://github.com/zijianchen98/GAIA
[7] https://github.com/QwenLM/qwen3
[8] https://github.com/QwenLM/Qwen3-VL
[9] https://github.com/zai-org/GLM-V/
[10] https://github.com/ChenYi99/EgoPlan
[11] https://github.com/KlingAIResearch/VideoAlign
[12] https://github.com/Vchitect/VBench
[13] https://github.com/hiyouga/LlamaFactory
[14] https://github.com/NVlabs/LongLive
[15] https://github.com/vita-epfl/Stable-Video-Infinity
[16] https://github.com/NVlabs/Sana
[17] https://github.com/Wan-Video/Wan2.1
[18] https://github.com/Wan-Video/Wan2.2
[19] https://github.com/XueZeyue/DanceGRPO