ACT: AGENTIC TEXT–SCENE CO-DESIGN AND DUAL EVOLUTION FOR INFORMATIVE VIDEO GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Informative videos, such as tutorials, advertisements, and public-service videos, combine visual content and on-screen text to deliver specific messages. Generating such videos requires exact text to appear at the requested times and locations without obscuring the main subject. However, current methods often produce incorrect or poorly placed text and fail to coordinate it with the generated scene. To systematically study this problem, we introduce **IVBench**, a benchmark across five domains that jointly evaluates on-screen text, scene content and layout, and text–scene coordination. To improve text–scene coordination, we propose **ACT**, an agentic framework for *text–scene co-design* that plans text and scene together but renders them separately. ACT uses a shared layout contract to reserve space for text and guide subject placement before scene generation. It then adapts text placement to the generated scene, renders the required strings deterministically, and uses verification feedback for targeted repair. To reduce recurring errors and enable persistent system improvement, we further introduce *verifier-guided dual evolution*, which uses shared execution feedback to update both the harness and the video generator. Experiments on IVBench show that ACT consistently improves text control and text–scene coordination over strong commercial and open-weight baselines, with further gains from dual evolution.

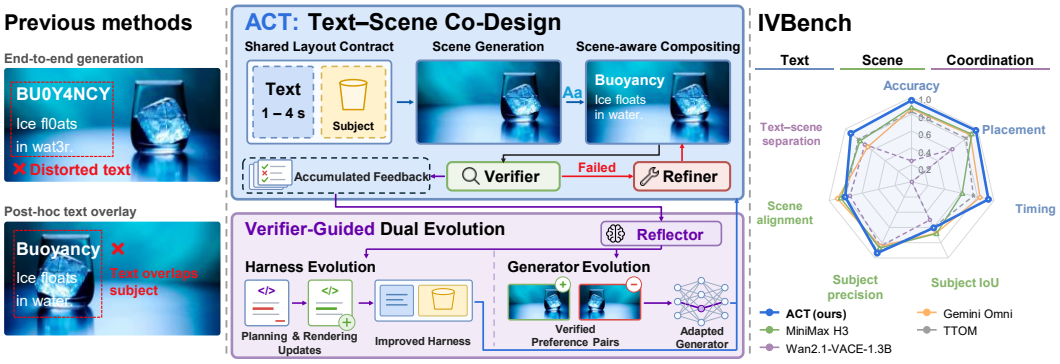


Figure 1: **ACT: text–scene co-design with dual evolution.** ACT plans text and scene together, renders them separately, and uses verifier feedback for both video repair and persistent system evolution. On IVBench, ACT achieves strong text control and text–scene coordination.

1 INTRODUCTION

Recent video generation models can produce visually compelling videos with complex motion and rich control (Wan et al., 2025; Seedance et al., 2026; HaCohen et al., 2026). However, visual quality alone is not enough for videos that need to deliver a specific message. Tutorials, advertisements, and public-service videos rely on readable text that appears at the right time and place. The text must convey the intended message while keeping the subject and relevant actions visible.

Meeting these requirements remains challenging. Native video generation synthesizes text and scene together, but may distort characters, change words across frames, or place text over the main sub-

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

ject (Guo et al., 2025). Post-hoc compositing allows precise control over text content and timing. However, the generated scene may not leave enough room for text in the requested region, so the added text may cover the main subject. The challenge is therefore not text rendering alone, but *text–scene coordination*: the message must remain readable without obscuring the scene (Fig. 1).

Existing benchmarks do not fully capture this need for text–scene coordination. They focus on visual text or video composition separately (Fallah et al., 2025; Guo et al., 2025; Sun et al., 2025), without jointly assessing text readability, timing, placement, and separation from the subject. To fill this gap, we introduce **IVBench**, a benchmark spanning five informative-video domains. Its seven metrics evaluate on-screen text, scene content and layout, and text–scene coordination. Our evaluation shows that even SOTA video generation models still struggle to fully meet these requirements.

To address these limitations, our key insight is to *plan text and scene together, but render them separately*. Joint planning allows text requirements to guide scene composition, while separate rendering provides precise control over text content and timing. Building on this insight, we propose **ACT** (**A**gentic **C**o-design of **T**ext and Scene), a multi-agent framework for text–scene co-design. Specifically, ACT uses a shared layout contract to jointly plan subject placement and reserve space for text before scene generation, where the contract guides scene generation. ACT then uses scene-aware compositing to adjust the text layout based on the subject's position and the available space in the generated video. A deterministic renderer adds the required strings at their requested times. Since the generated scene may not fully follow the contract, ACT verifies the text, scene, and their interaction, then uses detected mismatches to guide targeted refinement. The shared contract establishes how text and scene should fit together, while verification and refinement address the gap between the plan and the generated result.

However, repairing individual videos leaves the system itself unchanged, so similar mistakes may recur across requests. To learn from these recurring failures, we introduce **verifier-guided dual evolution**, where a unified **Reflector** drives the evolution of both the harness and the generator. Given verifier feedback and failure evidence, the Reflector plays two complementary roles. First, it proposes improvements to the planning and rendering strategies, allowing the harness to better coordinate text and scene in future requests. Second, it turns diagnosed spatial failures into *direction-verified preference pairs* for generator adaptation, so the generator learns to better follow the shared layout contract. The updated harness and generator are then applied to subsequent requests, whose new feedback supports further evolution. In this way, verification not only repairs individual outputs, but also provides the feedback signal that continuously improves the overall system. Experiments show that ACT consistently improves text control and text–scene coordination over strong commercial and open-weight baselines, with further gains from verifier-guided dual evolution.

In summary, our contributions are:

- We construct **IVBench**, a benchmark across five informative-video domains that jointly evaluates on-screen text, scene content and layout, and text–scene coordination.
- We propose **ACT**, a multi-agent framework that combines joint text–scene planning through a shared layout contract with separate rendering and verifier-guided refinement.
- We introduce **verifier-guided dual evolution**, which couples harness and generator updates through shared execution feedback, extending verification from individual video repair to persistent system improvement.

2 Related Work

Visual text generation and evaluation. Visual text generation has been studied mainly in images. TextDiffuser uses character-level masks, GlyphControl conditions on rendered glyph images, AnyText supports multilingual generation and editing, and TextDiffuser-2 uses language-model reasoning to plan layouts and improve rendering (Chen et al., 2023; Yang et al., 2023; Tuo et al., 2024; Chen et al., 2024). For videos, Text-Animator controls text appearance and motion across frames, while KineTy focuses on animated letterforms (Liu et al., 2024a; Park et al., 2024). Existing benchmarks cover different parts of this problem: TextInVision evaluates generated image text, T2VTextBench measures video text fidelity and temporal consistency, and ViTeX-Bench evaluates text replacement in existing videos (Fallah et al., 2025; Guo et al., 2025; Chen et al., 2026c). T2V-CompBench evaluates broader video composition, while MME-VideoOCR studies text reading and

reasoning in existing videos (Sun et al., 2025; Shi et al., 2026). However, they do not jointly test whether text is readable, appears at the requested time and location, and remains separate from the main subject. IVBench evaluates these requirements together.

Structured and agentic video generation. Some related works improve video controllability through planning, structured controls, and test-time feedback. LLM-grounded Video Diffusion and VideoDirectorGPT derive layouts or scene plans, while VideoTetris and Boximator introduce spatiotemporal controls for objects and motion (Lian et al., 2024; Lin et al., 2023; Tian et al., 2024; Wang et al., 2024). Frame Guidance and TTOM emphasize test-time control; GENMAC, VideoRepair, and VISTA use planning, verification, repair, or feedback; Code2Video takes a programmatic route to educational video generation (Jang et al., 2026; Qu et al., 2026; Huang et al., 2026; Lee et al., 2026; Long et al., 2026; Chen et al., 2025b). These methods make scene generation more controllable and allow outputs to be refined after generation. ACT instead focuses on coordinating required text with the generated scene through a shared text–scene contract, scene-aware compositing, and deterministic rendering.

Harness evolution and preference learning. Recent work has explored how agent systems can improve their prompts, components, and workflows from execution feedback. Promptbreeder, Automated Design of Agentic Systems, AFlow, and PACE optimize prompts, reusable agent components, or persistent workflows through iterative evaluation and refinement (Fernando et al., 2023; Hu et al., 2025; Zhang et al., 2025; Ling et al., 2026). In parallel, feedback-based learning has been used to adapt generative models to task-specific objectives (Wallace et al., 2024; Du et al., 2024; Chen et al., 2025a), while VideoDPO extends preference learning to video generation (Liu et al., 2025). Another line of work jointly optimizes the external harness and the underlying model or policy. Co-Harness (Chen et al., 2026d) and HarnessForge (Chen et al., 2026b) explore this direction, showing the benefit of adapting both parts of an agent system together. ACT focuses on persistent improvement for informative video generation, where recurring verifier feedback is used to refine the planning and rendering harness and to adapt the video generator with direction-verified preference pairs for better text–scene coordination.

3 IVBENCH

We introduce **IVBench**, a benchmark for informative video generation. The task requires a system to generate the requested scene and present readable text at specified times and locations, without covering the main subject. Unlike existing benchmarks that assess visual text or video composition separately, IVBench jointly evaluates *on-screen text*, *scene content and layout*, and *text–scene coordination* (Fig. 2).

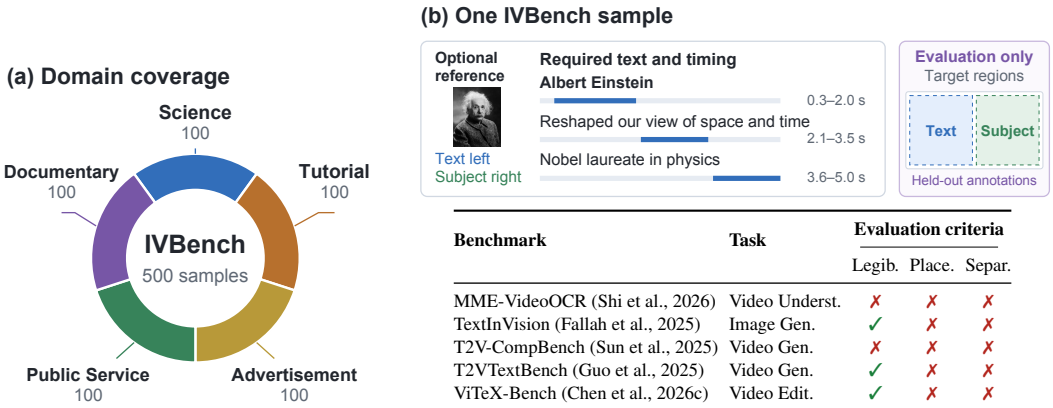
3.1 COMPOSITION AND CONSTRUCTION

IVBench contains 500 human-written samples, with 100 each in **Science**, **Public Service**, **Tutorial**, **Advertisement**, and **Documentary** (Fig. 2(a)). Each sample provides a scene description and two to four *text events*, each specifying an exact string and a display interval on a five-second timeline. Display intervals may overlap, requiring multiple strings to appear together. Optional references include a subject image, a first frame, or a skeleton video. Spatial annotations specify target regions for text and the subject. Generation models receive qualitative spatial instructions such as "upper half" or "left third". The corresponding numerical target regions are used only for offline evaluation (Fig. 2(b)). All samples undergo human review and LLM-assisted checks for content compliance and task usability. More construction details are provided in Appendix A.

3.2 METRICS

IVBench reports seven metrics in three groups, covering text requirements, scene requirements, and their spatial interaction.

On-screen text. *Text accuracy* measures character-level agreement between OCR results and required strings during their requested display intervals. *Placement* measures how accurately the rendered text follows the requested spatial region. *Timing* measures temporal intersection-over-union



Benchmark	Task	Evaluation criteria		
		Legib.	Place.	Separ.
MME-VideoOCR (Shi et al., 2026)	Video Underst.	✗	✗	✗
TextInVision (Fallah et al., 2025)	Image Gen.	✓	✗	✗
T2V-CompBench (Sun et al., 2025)	Video Gen.	✗	✗	✗
T2VTextBench (Guo et al., 2025)	Video Gen.	✓	✗	✗
ViTeX-Bench (Chen et al., 2026c)	Video Edit.	✓	✗	✗
IVBench (ours)	Video Gen.	✓	✓	✓

(c)

Figure 2: **Overview of IVBench.** **(a)** Five domains with 100 samples each. **(b)** A sample with generation inputs and evaluation-only target regions. **(c)** Comparison with related benchmarks on text legibility, placement, and text–subject separation. Underst. = understanding; Gen. = generation; Edit. = editing.

(IoU) between detected and requested display intervals, penalizing both missing text and text shown outside its requested interval.

Scene. *Subject layout* measures how well the detected subject matches its target region using IoU. *Subject precision* measures the fraction of the detected subject that lies within the target region. Together, these metrics capture both subject placement and how well the subject stays within its target region. *Scene alignment* measures how well the generated scene matches the scene description. Specifically, a VLM evaluates a set of visual claims extracted from the description over sampled frames.

Text–scene coordination. *Separation* measures how well the detected text and subject bounding boxes remain spatially separate during the requested display intervals. Their intersection is normalized by both the text and subject areas, and the 95th percentile emphasizes severe overlap while reducing sensitivity to isolated outlier frames.

All scores lie in $[0, 1]$, with higher values indicating better performance. Measurements are first aggregated within each sample across key frames, then averaged over samples. Detailed equations and evaluation settings are provided in Appendix A.

4 Method

ACT combines *text–scene co-design* with *verifier-guided dual evolution* (Fig. 3).

4.1 Text–Scene Co-design with a Layout Contract

An informative-video request is $x = (d, \mathcal{T}, \tau, \ell, r)$, consisting of a scene description d, required strings $\mathcal{T}$, display intervals τ, qualitative spatial instructions ℓ, and an optional reference r. Given x, the **Planner** jointly plans the scene and text, including their visual style and spatial arrangement. It reserves space for $\mathcal{T}$ and plans the subject's position accordingly, producing a shared layout contract:

$$c = (\mathcal{T}, \tau, R_{\text{text}}, R_{\text{subj}}, \sigma, q)\,, \tag{1}$$

where R_{text} and R_{subj} are the planned text and subject regions, σ describes the scene, and q identifies the subject. The Planner then converts c into a scene-generation prompt $\pi(c)$. This prompt describes the scene σ and subject q, and expresses the spatial arrangement defined by R_{text} and R_{subj} in natural language. Conditioned on $\pi(c)$ and the optional reference r, the generator produces a raw scene $v \sim p_\theta(\cdot \mid \pi(c), r)$.

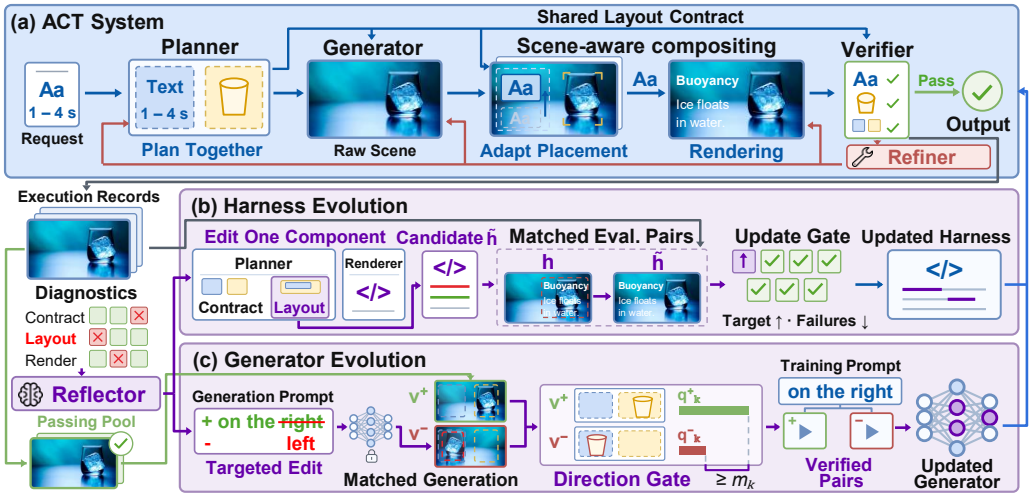


Figure 3: **Overview of ACT. (a)** Text–scene planning establishes a shared contract that guides scene generation and scene-aware compositing, followed by verification and targeted repair. **(b)** Harness evolution uses verifier feedback to propose component-wise updates, which are evaluated through matched comparisons and retained only when targeted diagnostics improve and overall failures decrease. **(c)** Generator evolution uses targeted prompt edits and matched generation to construct direction-verified preference pairs, which are used for preference learning under the original prompt.

4.2 Scene-Aware Compositing and Deterministic Rendering

The generated scene v may not faithfully follow the contract c. We therefore introduce **scene-aware compositing (SA)** to adapt text placement to the actual video. The Planner inspects sampled frames to locate the subject q and identify clear, stable regions suitable for text. Then it creates an HTML overlay for $\mathcal{T}$ within R_{text}. A deterministic HTML-based renderer (Liu et al., 2026b) composites the text overlay onto v according to τ, producing the video y. During this process, the renderer fits text within R_{text} and improves legibility. Implementation details are provided in Appendix B.1.

4.3 Verification and Instance-Level Repair

The **Verifier** checks y against the request x and contract c, combining rule-based text and geometry checks with VLM-based visual inspection. Text checks assess correctness against $\mathcal{T}$, timing against τ, placement within R_{text}, and legibility. Scene checks assess the position of subject q relative to R_{subj} and whether the scene matches σ. The Verifier also checks text–subject overlap. These checks identify failed requirements and the affected regions or intervals. Based on this feedback and preset routing rules, the **Refiner** then adjusts the overlay on v, revises the scene prompt and regenerates under the same c, or revises an infeasible c before regeneration. Each result is verified again. Algorithm 1 gives the repair routes, budgets, and stopping rules.

4.4 Verifier-Guided Dual Evolution

Instance-level repair changes the current output but leaves the harness and generator unchanged. To learn from recurring failures, we introduce **verifier-guided dual evolution** (Fig. 3). As requests are processed, ACT accumulates execution feedback. Once sufficient records are collected, the **Reflector** uses this feedback to guide updates to the harness h and generator p_θ. The retained updates (h', θ') are then applied to subsequent requests, whose executions provide new feedback for further evolution. Algorithm 2 gives the full procedure.

4.4.1 Harness Evolution

The harness $h = (P, R)$ consists of the Planner's contract-planning and text-layout strategies P and the Renderer's text presentation rules R. Given verifier feedback and failure evidence, the **Reflector**

selects a planning or rendering component to evolve, producing a candidate harness $\tilde{h}$. The candidate $\tilde{h}$ is compared with the current harness h on the same development subset by rerunning the affected planning or rendering step on stored videos. For generation-planner updates, candidates that pass this initial evaluation are further validated on a small held-out subset by regenerating videos under matched generation settings.

Let $j \in \mathcal{J}$ index the verifier diagnostics and $j^\star$ the targeted one. We require the target to improve and the total number of failures to decrease:

$$e_{j^\star}(\tilde{h}) < e_{j^\star}(h) \quad \text{and} \quad \sum_{j \in \mathcal{J}} N_j \left[e_j(\tilde{h}) - e_j(h) \right] < 0. \tag{2}$$

where $e_j(h)$ is the failure rate and N_j the number of evaluated samples for diagnostic j. Accepted changes are retained in h'.

4.4.2 Generator Evolution

Harness evolution improves the reusable planning and rendering strategies but leaves the generator unchanged. To improve scene generation itself, the **Reflector** further converts the same verifier feedback into *direction-verified preference pairs* for adapting p_θ. For an execution that passes verification, its raw scene is used as the positive sample v^+. The Reflector then selects a recurring spatial failure identified from the accumulated execution feedback and applies a targeted edit to the corresponding generation prompt, producing a candidate negative v^- under matched generation conditions.

To ensure preference pairs provide a reliable training signal for the diagnosed spatial failure, we introduce a *direction gate* under the original contract c. Let $k \in \mathcal{K}$ index the spatial diagnostics and $q_k(v, c)$ denote the corresponding score, where higher values indicate better performance. A pair is retained only if

$$q_k(v^+, c) - q_k(v^-, c) \geq m_k, \tag{3}$$

where m_k is the margin required for diagnostic k. This retains pairs with the intended spatial ordering, forming direction-verified preferences. To keep the preference signal aligned with the original request, the edited prompt is used only to generate v^-, while both scenes are conditioned on the original prompt $\pi(c)$ during training (Liu et al., 2025).

The generator is then adapted on $\mathcal{D}_{\text{pref}}$ using DPOP (Pal et al., 2024). Using a frozen copy of the current generator as reference p_{ref}, let $s_\theta^\pm = \log \frac{p_\theta(v^\pm|a)}{p_{\text{ref}}(v^\pm|a)}$ for the original conditioning $a = (\pi(c), r)$. The training objective is

$$\mathcal{L}_{\text{DPOP}}(\theta) = -\mathbb{E}_{\mathcal{D}_{\text{pref}}} \left[\log \text{sigmoid} \left(\beta \left(s_\theta^+ - s_\theta^- - \lambda [-s_\theta^+]_+ \right) \right) \right], \tag{4}$$

where $[z]_+ = \max(0, z)$. The objective favors v^+ over v^- while alleviating degradation of the preferred sample relative to the reference. Additional training details are provided in Appendix C.3.

5 Experiments

5.1 Experimental Settings

Benchmark and baselines. We evaluate ACT on IVBench against end-to-end video generators, structured/test-time generation methods, and video-text editing baselines (Team et al., 2025a; Chen et al., 2026a; Seedance et al., 2026; Jiang et al., 2025; Team et al., 2025b; Yang et al., 2024; Chen et al., 2026c; Lian et al., 2024; Huang et al., 2026; Qu et al., 2026; He et al., 2025). LVD, GenMAC, TTOM, and EvoSearch use the same request-aware overlay for text rendering. Full baseline settings are provided in Appendix D.3.

Implementation details. ACT uses Wan2.1-VACE-1.3B (Jiang et al., 2025) to generate 832×480 videos with 81 frames at 16 fps. We use 50 sampling steps, guidance and shift of 5.0, and context scale 1.0, with Claude Opus 4.8 for the agents. For each request, ACT generates one initial scene, allows up to two layout repairs, and at most one scene regeneration. DPOP uses $\beta = 600$ and $\lambda = 0.1$. We perform evolution on a separate training dataset disjoint from IVBench.

Table 1: **IVBench results.** Best per column is **bold** and second is underlined.

Method	Text			Scene			Coordination
	Accuracy	Placement	Timing	IoU	Precision	Alignment	Separation
Proprietary generators							
Gemini Omni	0.863	0.889	0.831	0.672	0.847	**0.898**	0.663
Kling 3.0	0.541	0.892	0.177	0.665	0.885	0.873	0.757
Seedance 2.0	0.855	0.897	0.676	0.633	0.860	0.739	0.706
Wan-3.0	0.851	0.903	0.737	0.651	0.818	0.867	0.639
Open-weight generators							
SANA-Video	0.176	0.438	0.002	0.547	0.657	0.765	0.454
Wan2.1-VACE-1.3B	0.247	0.618	0.004	0.501	0.846	0.747	0.706
LongCat-Video	0.276	0.831	0.002	0.594	0.827	0.793	0.726
CogVideoX1.5-5B	0.113	0.336	0.000	0.476	0.658	0.675	0.289
LTX-Video-2.3	0.617	0.904	0.163	0.636	0.816	0.811	0.655
HunyuanVideo-1.5	0.613	0.558	0.318	0.532	0.677	0.791	0.455
Wan2.2-14B	0.638	0.915	0.175	0.624	0.864	0.801	0.805
MiniMax H3	0.881	0.911	0.616	**0.680**	0.870	0.860	0.779
Structured / test-time generation + request-aware overlay							
LVD	0.866	0.873	0.742	0.470	0.715	0.384	0.541
GenMAC	0.872	0.878	0.745	0.590	0.764	0.648	0.507
EvoSearch	0.814	0.842	0.783	0.538	0.721	0.771	0.452
TTOM	0.830	0.826	0.751	0.600	0.921	0.807	0.804
Video-text editing							
ViTeX-Edit-14B	0.253	0.515	0.087	0.508	0.745	0.694	0.522
ACT (Ours)	**0.963**	**0.974**	**0.933**	0.605	**0.936**	0.805	**0.918**

5.2 BENCHMARKING ON IVBENCH

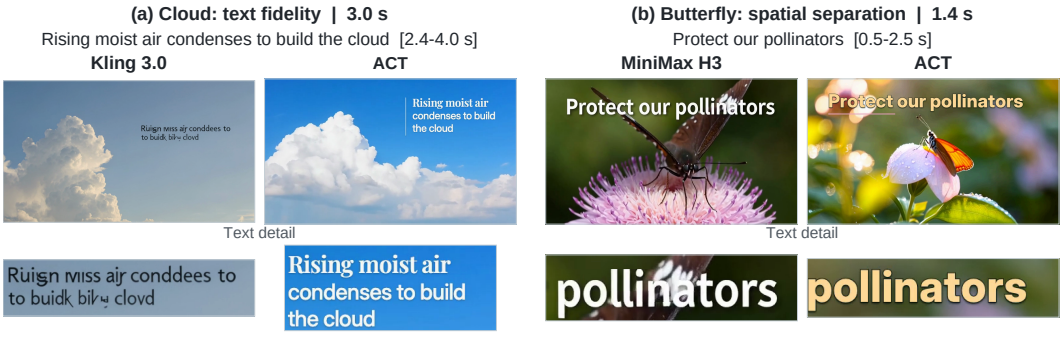


Figure 4: **Qualitative comparisons with representative baselines.** More examples are in Appendix H.

Table 1 shows that ACT delivers strong performance across text fidelity, scene control, and text–scene coordination. It achieves the best results in text accuracy, placement, timing, subject precision, and separation, with particularly clear gains in timing (0.933 vs. 0.831) and separation (0.918 vs. 0.805) over the best competing methods. Compared with its vanilla Wan2.1-VACE-1.3B backbone, ACT improves all seven metrics, demonstrating the effectiveness of the proposed framework beyond the end-to-end generator.

Among proprietary models, Gemini Omni achieves strong scene alignment, while Kling 3.0 shows competitive placement but limited timing control. MiniMax H3 performs strongly across several metrics, reaching 0.881 text accuracy, 0.911 placement, and 0.680 subject IoU. ACT further improves text accuracy, placement, timing, subject precision, and separation, providing a more balanced result across the joint requirements.

Table 2: **Ablation study on the fixed 200-item IVBench subset.** Base ACT uses the original harness and vanilla VACE-1.3B.

Variant	Text			Scene			Coordination
	Accuracy	Placement	Timing	IoU	Precision	Alignment	Separation
(a) Pipeline baselines							
End-to-end	0.233	0.594	0.003	0.481	0.831	0.745	0.693
Request-aware overlay	0.790	0.801	0.776	0.425	0.673	0.732	0.650
Best-of-N	0.814	0.858	0.778	0.501	0.835	0.759	0.695
(b) Component ablations							
Base ACT	0.948	0.912	0.923	0.542	0.875	0.742	0.813
w/o Repair	0.934	0.899	0.905	0.529	0.873	0.741	0.766
w/o Repair & Planner	0.919	0.897	0.897	0.431	0.682	0.725	0.588
w/o Repair & SA	0.876	0.881	0.839	0.524	0.872	0.739	0.732
(c) Dual evolution							
w/ Evolved harness	0.964	0.962	0.919	0.543	0.879	0.755	0.875
w/ Adapted generator	0.951	0.924	0.909	0.579	0.913	0.762	0.856
w/ Both (final ACT)	0.961	0.981	0.931	0.614	0.955	0.781	0.916

Figure 4 compares ACT with representative baselines on two cases. In the cloud example, Kling produces distorted text, while ACT delivers exact information. In the butterfly example, H3 overlaps the text with the butterfly, whereas ACT maintains clearer text–subject separation.

5.3 Ablation Studies

Pipeline baselines. Table 2(a) compares three pipeline baselines built on vanilla VACE-1.3B. End-to-end directly generates the scene and text. Request-aware overlay adds the required text after generation without text-space reservation or scene-aware placement. Best-of-N retains contract-based planning, generates $N = 4$ candidates, and selects the best video. Request-aware overlay substantially improves text accuracy and timing but remains weak in placement (0.801), while Best-of-4 reaches only 0.858 placement and 0.695 separation. Base ACT improves these scores to 0.912 and 0.813, showing the importance of joint planning and scene-aware compositing beyond post-hoc overlay and best-of-N selection. Additional Kling studies are provided in Appendix E.4.

Component ablations. Table 2(b) evaluates the main components before evolution. Instance-level repair improves all three text metrics and raises separation from 0.766 to 0.813. Relative to *w/o Repair*, removing the Planner reduces subject IoU, precision, and separation by 0.098, 0.191, and 0.178, respectively. Replacing SA with request-aware overlay reduces placement from 0.899 to 0.881. These results show that the Planner mainly improves subject layout and text–scene coordination, while SA provides precise text placement on the generated scene.

Dual evolution. Table 2(c) evaluates harness and generator evolution separately and jointly. Harness evolution mainly improves placement and separation, while generator evolution improves subject IoU, precision, and scene alignment. Combining both yields gains of 0.069 in placement, 0.072 in subject IoU, 0.080 in precision, and 0.103 in separation over Base ACT. These results demonstrate the complementary roles of harness and generator evolution in improving text–scene coordination. Detailed settings and the independent-evolution variant are provided in Appendix E.1.

5.4 Human Study

We conduct a blinded ACT–MiniMax H3 comparison with 10 volunteers on 30 randomly selected IVBench samples. The two videos are shown synchronously with an animated reference describing the requested text, timing, and layout. Sample order is randomized and display sides are balanced. Participants choose ACT, H3, or "About the same" for information clarity, text placement, and overall visual quality. Appendix G provides the full setting.

Figure 5(a) shows that ACT receives more preferences on all three criteria. The largest gaps appear in overall quality (50.3% vs. 30.0%) and text placement (48.7% vs. 25.0%). Information clarity is closer (39.7% vs. 31.3%), with 29.0% ties. These results suggest that ACT improves perceived text placement and overall visual quality while maintaining comparable information clarity. Appendix G further reports human evaluations of evolution.

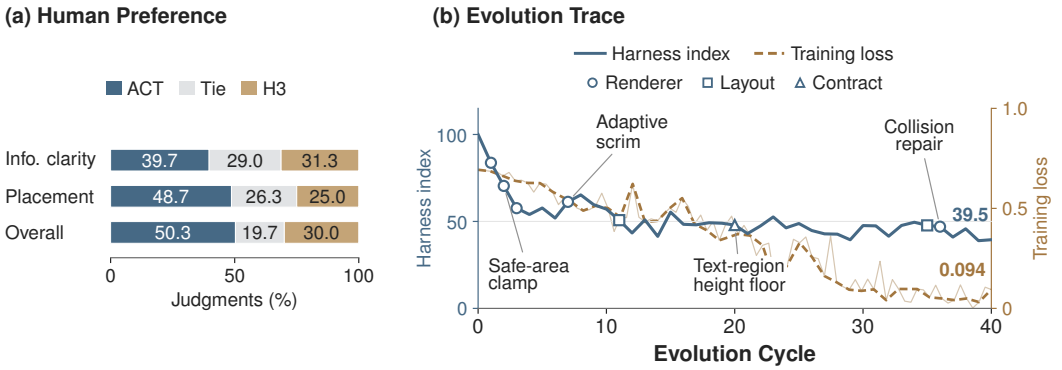


Figure 5: **Human preference and evolution trace.** (a) Human preference between ACT and H3. (b) Harness and generator evolution traces. Markers indicate accepted harness updates (Appendix E.3).

5.5 Evolution Diagnostics

Figure 5(b) visualizes the dual evolution processes. Over 40 harness-search rounds, eight accepted updates reduce the harness diagnostic index (a normalized summary of the errors $e_j(h)$; lower is better) from 100 to 39.5. Generator training loss decreases from 0.693 to 0.094 over the evolution process. Figure 6 compares Base ACT with evolved outputs at matched frames. Both examples show that evolution improves text completeness and rendering consistency. In the sunflower example, the truncated "Sunflowe" is rendered correctly as "Sunflower," while the carrot example shows more complete and consistently placed captions across frames.



Figure 6: **Examples before and after evolution** at matched frames. Additional examples are provided in Appendix H.

6 Conclusion

In this work, we introduced IVBench and ACT for informative video generation with coordinated text and visual content. ACT follows a simple principle: plan text and scene together, render them separately, and use verification to both repair individual outputs and improve the system over time through dual evolution. Our results show that effective text–scene coordination requires text requirements to influence scene composition, rather than treating text as a separate rendering step. They also show that recurring execution feedback can improve both the harness and generator over time. Together, IVBench and ACT provide a unified framework for evaluating and improving informative video generation under explicit text, timing, layout, and scene requirements.

AI USE STATEMENT

We used generative AI assistants to correct the grammar and presentation of the paper. Generative models are also part of the method: ACT uses LLM agents and reflectors. LLMs also assisted the content-compliance and task-usability checks for IVBench samples.

ETHICS STATEMENT

Participation in the human study was voluntary. The interface explained the task before participation, required no account or personal information, and concealed method identities. IVBench samples were manually authored and reviewed for content compliance and task usability, with additional LLM-assisted checks.

REPRODUCIBILITY STATEMENT

We provide the ACT code, IVBench, and evaluation scripts in the supplementary material. Algorithms 1 and 2 describe the online generation and dual-evolution procedures. Detailed generation, evolution, and baseline settings are reported in Appendix D. Appendix A describes benchmark construction and metric definitions, and Appendix G provides the human-study setting.

REFERENCES

Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. Textdiffuser: Diffusion models as text painters. *Advances in Neural Information Processing Systems*, 36:9353–9387, 2023.

Jingye Chen, Yupan Huang, Tengchao Lv, Lei Cui, Qifeng Chen, and Furu Wei. Textdiffuser-2: Unleashing the power of language models for text rendering. In *European Conference on Computer Vision*, pp. 386–402. Springer, 2024.

Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Zhou Daquan, Huan Ling, et al. Sana-video: Efficient video generation with block linear diffusion transformer. In *International Conference on Learning Representations*, volume 2026, pp. 38973–38997, 2026a.

Mingju Chen, Can Lv, Guibin Zhang, Heng Chang, and Shiji Zhou. Harnessforge: Joint harness and policy evolution for adaptive agent systems. *arXiv preprint arXiv:2606.01779*, 2026b.

Xinghao Chen, Xiangbo Gao, Jiongze Yu, Yuheng Wu, and Zhengzhong Tu. Vitex-bench: Benchmarking high fidelity video scene text editing. In *Advances in Neural Information Processing Systems (NeurIPS), Track on Evaluations and Datasets*, 2026c. URL `https://vitex-bench.github.io/`.

Xingye Chen, Wei Feng, Zhenbang Du, Weizhen Wang, Yanyin Chen, Haohan Wang, Linkai Liu, Yaoyu Li, Jinyuan Zhao, Yu Li, et al. Ctr-driven advertising image generation with multimodal large language models. In *Proceedings of the ACM on Web Conference 2025*, pp. 2262–2275, 2025a.

Yanzhe Chen, Kevin Qinghong Lin, and Mike Zheng Shou. Code2video: A code-centric paradigm for educational video generation. *arXiv preprint arXiv:2510.01174*, 2025b.

Zhengyu Chen, Teng Xiao, Huaisheng Zhu, Yige Yuan, Luan Zhang, and Jingang Wang. Co-harness: Co-evolving harnesses and model weights for llm agents. *arXiv preprint arXiv:2607.22688*, 2026d.

Zhenbang Du, Wei Feng, Haohan Wang, Yaoyu Li, Jingsen Wang, Jian Li, Zheng Zhang, Jingjing Lv, Xin Zhu, Junsheng Jin, et al. Towards reliable advertising image generation using human feedback. In *European Conference on Computer Vision*, pp. 399–415. Springer, 2024.

Forouzan Fallah, Maitreya Patel, Agneet Chatterjee, Vlad Morariu, Chitta Baral, and Yezhou Yang. Textinvision: Text and prompt complexity driven visual text generation benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 525–534, 2025.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*, 2023.

Xuyang Guo, Jiayan Huo, Zhenmei Shi, Zhao Song, Jiahao Zhang, and Jiale Zhao. T2vtextbench: A human evaluation benchmark for textual control in video generation models. *arXiv preprint arXiv:2505.04946*, 2025.

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

Haoran He, Jiajun Liang, Xintao Wang, Pengfei Wan, Di Zhang, Kun Gai, and Ling Pan. Scaling image and video generation via test-time evolutionary search. *arXiv preprint arXiv:2505.17618*, 2025.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

Kaiyi Huang, Yukun Huang, Xuefei Ning, Zinan Lin, Yu Wang, and Xihui Liu. Genmac: compositional text-to-video generation with multi-agent collaboration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 5049–5057, 2026.

Sangwon Jang, Taekyung Ki, Jaehyeong Jo, Jaehong Yoon, Soo Ye Kim, Zhe Lin, and Sung Ju Hwang. Frame guidance: Training-free guidance for frame-level control in video diffusion models. In *International Conference on Learning Representations*, volume 2026, pp. 60977–61003, 2026.

Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. Vace: All-in-one video creation and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17191–17202, 2025.

Daeun Lee, Jaehong Yoon, Jaemin Cho, and Mohit Bansal. Self-correcting text-to-video generation with misalignment detection and localized refinement. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 36464–36489, 2026.

Long Lian, Baifeng Shi, Adam Yala, Boyi Li, et al. Llm-grounded video diffusion models. In *International Conference on Learning Representations*, volume 2024, pp. 50207–50227, 2024.

Han Lin, Abhay Zala, Jaemin Cho, and Mohit Bansal. Videodirectorgpt: Consistent multi-scene video generation via llm-guided planning. *arXiv preprint arXiv:2309.15091*, 2023.

Chen Ling, Pei Chen, Albert Guan, Jiaming Qu, Shayan Ali Akbar, Madhu Gopinathan, and Erwin Cornejo. Pace: Two-timescale self-evolution for small language model agents. *arXiv preprint arXiv:2605.23019*, 2026.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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. *Advances in Neural Information Processing Systems*, 38:82155–82192, 2026a.

Junle Liu, Peirong Zhang, Yuyi Zhang, Pengyu Yan, Hui Zhou, Xinyue Zhou, Fengjun Guo, and Lianwen Jin. Posterverse: A full-workflow framework for commercial-grade poster generation with html-based scalable typography. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 7197–7205, 2026b.

Lin Liu, Quande Liu, Shengju Qian, Yuan Zhou, Wengang Zhou, Houqiang Li, Lingxi Xie, and Qi Tian. Text-animator: Controllable visual text video generation. *arXiv preprint arXiv:2406.17777*, 2024a.

Runtao Liu, Haoyu Wu, Ziqiang Zheng, Chen Wei, Yingqing He, Renjie Pi, and Qifeng Chen. Videodpo: Omni-preference alignment for video diffusion generation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8009–8019. IEEE, 2025.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pp. 38–55. Springer, 2024b.

Do Xuan Long, Xingchen Wan, Hootan Nakhost, Chen-Yu Lee, Tomas Pfister, and Sercan Ö Arik. Vista: A test-time self-improving video generation agent. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6021–6032, 2026.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*, 2024.

Seonmi Park, Inhwan Bae, Seunghyun Shin, and Hae-Gon Jeon. Kinetic typography diffusion model. In *European Conference on Computer Vision*, pp. 166–185. Springer, 2024.

Leigang Qu, Ziyang Wang, Na Zheng, Wenjie Wang, Liqiang Nie, and Tat-Seng Chua. Ttom: Test-time optimization and memorization for compositional video generation. In *International Conference on Learning Representations*, volume 2026, pp. 70225–70247, 2026.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. In *International Conference on Learning Representations*, volume 2025, pp. 28085–28128, 2025.

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.

Yang Shi, Huanqian Wang, Xie Xie, Huanyao Zhang, Lijie Zhao, Xinfeng Li, Chaoyou Fu, Zhuoer Wen, Wenting Liu, Zhuoran Zhang, et al. Mme-videoocr: Evaluating ocr-based capabilities of multimodal llms in video scenarios. *Advances in Neural Information Processing Systems*, 38, 2026.

Kaiyue Sun, Kaiyi Huang, Xian Liu, Yue Wu, Zihan Xu, Zhenguo Li, and Xihui Liu. T2v-compbench: A comprehensive benchmark for compositional text-to-video generation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8406–8416. IEEE, 2025.

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*, 2025a.

Meituan LongCat Team, Xunliang Cai, Qilong Huang, Zhuoliang Kang, Hongyu Li, Shijun Liang, Liya Ma, Siyu Ren, Xiaoming Wei, Rixu Xie, and Tong Zhang. Longcat-video technical report, 2025b. URL https://arxiv.org/abs/2510.22200.

Ye Tian, Ling Yang, Haotian Yang, Yuan Gao, Yufan Deng, Jingmin Chen, Xintao Wang, Zhaochen Yu, Xin Tao, Pengfei Wan, et al. Videotetris: Towards compositional text-to-video generation. *Advances in Neural Information Processing Systems*, 37:29489–29513, 2024.

Yuxiang Tuo, Wangmeng Xiang, Jun-Yan He, Yifeng Geng, and Xuansong Xie. Anytext: Multilingual visual text generation and editing. In *International Conference on Learning Representations*, volume 2024, pp. 56783–56799, 2024.

Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8228–8238. IEEE, 2024.

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.

Jiawei Wang, Yuchen Zhang, Jiaxin Zou, Yan Zeng, Guoqiang Wei, Liping Yuan, and Hang Li. Boximator: Generating rich and controllable motions for video synthesis. *arXiv preprint arXiv:2402.01566*, 2024.

Yukang Yang, Dongnan Gui, Yuhui Yuan, Weicong Liang, Haisong Ding, Han Hu, and Kai Chen. Glyphcontrol: Glyph conditional control for visual text generation. *Advances in Neural Information Processing Systems*, 36:44050–44066, 2023.

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.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. Aflow: Automating agentic workflow generation. In *International Conference on Learning Representations*, volume 2025, pp. 34040–34077, 2025.

Yubo Zhang, Xueqing Wang, Manhui Lin, Yue Zhang, Penglongyi Deng, Ting Sun, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Changda Zhou, et al. Pp-ocrv6: From 1.5 m to 34.5 m parameters, surpassing billion-scale vlms on ocr tasks. *arXiv preprint arXiv:2606.13108*, 2026.

APPENDIX CONTENTS

A More Details on IVBench **15**

A.1 Construction and Annotation 15

A.2 Metrics 15

B ACT Online Execution Details **17**

B.1 Scene-Aware Compositing and Rendering 17

B.2 Generation, Verification, and Repair Settings 17

B.3 Online Execution Algorithm 18

C Verifier-Guided Dual Evolution Details **18**

C.1 Evolution Loop 18

C.2 Harness Evolution Details 18

C.3 Generator Evolution Details 20

D Experimental Settings **23**

D.1 ACT Configuration 23

D.2 Evaluation Protocol 23

D.3 Baseline Implementations 24

D.4 Ablation Study Settings 24

E Additional Experimental Results **24**

E.1 Independent versus Coupled Evolution 24

E.2 Additional Mechanism Comparisons 25

E.3 Evolution Diagnostics 26

E.4 Commercial-Backbone Study 27

E.5 Temporal Sampling Sensitivity 28

F Cost Analysis **29**

G Human Study Details **29**

H Qualitative Comparisons **30**

I Limitations **37**

J Reflector Prompts **37**

A MORE DETAILS ON IVBENCH

A.1 CONSTRUCTION AND ANNOTATION

Construction and quality control. All 500 IVBench items were manually authored and reviewed by humans. Each item was also checked using both GPT-5.6 Sol and Claude Opus 4.8 for content compliance and task usability.

Domain composition. IVBench contains 500 samples across five application domains, with 100 samples in each domain (Fig. 2(a)). **Science** contains explanations (60) and facts (40); **Public Service** contains safety warnings (55) and environmental messages (45); **Tutorial** contains cooking (45), language (45), and fitness (10); **Advertisement** contains service (50) and subject advertisements (50); and **Documentary** contains nature (40), travel (20), history (20), and biography (20).

Sample format. Each sample is stored as a JSON file containing a scene description, required strings, display intervals, qualitative layout instructions, and an optional reference. The dataset contains 1,403 text events, with two to four events per sample, defined over a five-second timeline.

Layouts and references. Regions are represented by normalized (x, y, w, h) coordinates. Generation receives only the corresponding qualitative spatial instruction. Numerical target regions are used only for evaluation. The dataset includes 340 text-to-video, 110 subject-reference-to-video, 40 image-to-video, and 10 skeleton-to-video samples.

A.2 METRICS

Sampling and notation. We uniformly sample $F = 16$ frames from each video and examine sensitivity to the temporal sampling density in Appendix E.5. Each sampled frame f at timestamp t_f represents a temporal cell C_f, whose boundaries are defined by the midpoints between neighboring sampled frames. For each text event e, let s_e, I_e, and R_e denote the required string, display interval, and target region, respectively. Frames whose timestamps fall within I_e are treated as active:

$$A_e = \{f : t_f \in I_e\}. \tag{5}$$

Let B_{ef} denote the matched OCR box for event e in frame f, and let S_f denote the detected subject box. All spatial coordinates are normalized.

Text accuracy. OCR is applied to the full frame, and detections with confidence below 0.30 are discarded. Text matching applies Unicode NFC normalization, whitespace collapsing, and case folding while preserving punctuation. For each event, the target string is compared against individual OCR spans and their reading-order concatenations. Let $\mathcal{W}_f(s_e)$ denote the candidate strings considered by the matcher. The frame-level accuracy is

$$a_{ef} = \max_{u \in \mathcal{W}_f(s_e)} \left[1 - \frac{d_{\mathrm{Lev}}(s_e, u)}{|s_e|}\right]_+, \qquad Q_{\mathrm{text}} = \frac{1}{E}\sum_{e=1}^{E} \frac{1}{|A_e|} \sum_{f \in A_e} a_{ef}, \tag{6}$$

where d_{Lev} denotes the character-level edit distance and E is the number of text events. Missing detections receive a score of zero. Whole-word matching allows extra surrounding words, but requires the target itself to appear as a complete word.

Placement. Placement measures the fraction of the matched text box that lies within the requested region:

$$Q_{\mathrm{place}} = \frac{1}{E}\sum_{e=1}^{E} \frac{1}{|A_e|} \sum_{f \in A_e} \frac{|B_{ef} \cap R_e|}{|B_{ef}|}. \tag{7}$$

Missing or zero-area boxes receive a score of zero. When the best text match consists of multiple OCR spans, we use their enclosing box if the spans form a valid text block; otherwise, we use the largest localized matched span. Matched spans require text accuracy of at least 0.34, and adjacent rows must be within 0.12 of the frame height vertically.

Timing. A target string is considered present in frame f when $a_{ef} \geq 0.75$. Its observed temporal support is $O_e = \bigcup_{f:a_{ef}\geq 0.75} C_f$, which includes detections outside the requested interval. The requested temporal support on the same sampling grid is $G_e = \bigcup_{f\in A_e} C_f$. Timing is measured using temporal IoU:

$$Q_{\text{time}} = \frac{1}{E}\sum_{e=1}^{E} \frac{|O_e \cap G_e|}{|O_e \cup G_e|}. \tag{8}$$

If the target string is never detected, its timing score is zero. Text appearing too early or too late enlarges the union and therefore lowers the score.

Subject layout and precision. Let R_{subj} denote the requested subject region, and let Π_a denote projection onto the constrained axis $a \in \{x, y\}$. For each frame, we compute

$$J_f = \frac{|\Pi_a(S_f) \cap \Pi_a(R_{\text{subj}})|}{|\Pi_a(S_f) \cup \Pi_a(R_{\text{subj}})|}, \qquad P_f = \frac{|\Pi_a(S_f) \cap \Pi_a(R_{\text{subj}})|}{|\Pi_a(S_f)|}. \tag{9}$$

For targets constrained along both axes, we instead use two-dimensional intersection and union areas. The sample-level scores are $Q_{\text{IoU}} = \text{median}_f\, J_f$ and $Q_{\text{prec}} = \text{median}_f\, P_f$. IoU measures agreement in subject position and spatial extent, while precision measures the fraction of the detected subject that lies within the requested region.

Scene alignment. Each scene description is first decomposed by Gemini 2.5 Flash into three to eight visual claims covering entities, attributes, actions, and scene settings. Claims related to text, timing, layout, or text–subject overlap are excluded.

Let $c_1, \ldots, c_M$ denote the claims for a sample and $b_m \in \{0, 1\}$ the judge's support decision for claim m. Scene alignment is computed as

$$Q_{\text{scene}} = \frac{1}{M}\sum_{m=1}^{M} b_m. \tag{10}$$

Gemini 2.5 Flash evaluates these claims at temperature zero using six uniformly sampled frames. Before evaluation, detected text regions are pixelated so that the judge focuses on scene content rather than the rendered text.

Separation. Separation measures the spatial overlap between text and the main subject. For each active event–frame pair with valid detections, let B_{ef} denote the matched text box and S_f the tight bounding box of the detected subject mask. We compute the overlap relative to both regions:

$$r^T_{ef} = \frac{|B_{ef} \cap S_f|}{|B_{ef}|}, \qquad r^S_{ef} = \frac{|B_{ef} \cap S_f|}{|S_f|}. \tag{11}$$

The separation score is

$$Q_{\text{sep}} = 1 - \max\left\{\text{p}_{95}(r^T), \text{p}_{95}(r^S)\right\}. \tag{12}$$

The 95th percentile is computed over all valid event–frame pairs using linear interpolation. Normalizing by both regions captures cases where the overlap is large relative to either the text or the subject. If no valid text or subject box is available throughout the video, we set $Q_{\text{sep}} = 0$.

Evaluation implementation. We use PP-OCRv6 for text recognition (Zhang et al., 2026). For subject detection, we use Grounding-DINO-base (Liu et al., 2024b) followed by SAM 2.1 Hiera-large (Ravi et al., 2025), with both box and text thresholds set to 0.25. The resulting subject mask is converted to a tight axis-aligned bounding box for computing subject layout, precision, and separation. For frames with missed detections, we use the detected subject box with median area from the same clip. If no subject is detected in any frame, subject IoU, precision, and separation are set to zero. Scores are first aggregated within each sample and then averaged across samples.

B ACT ONLINE EXECUTION DETAILS

B.1 SCENE-AWARE COMPOSITING AND RENDERING

Scene-aware compositing. The Planner analyzes sampled frames to estimate the subject location and identify clear, stable regions for text. These observations, together with the planned text region and visual design guidance, guide the HTML overlay layout. This adapts text placement when the generated scene deviates from the layout contract.

Rendering safeguards. The deterministic renderer fits text using measured element geometry and applies font-size, clipping, safe-area, and contrast checks. When needed, it adjusts typography or adds a suitable background to preserve legibility. All safeguards operate on the text overlay layer and leave the generated scene unchanged.

B.2 GENERATION, VERIFICATION, AND REPAIR SETTINGS

Inputs and budgets. Online execution takes a request x, a fixed harness h, and generator p_θ. For each request, ACT generates one initial scene, allows up to two layout repairs, and permits at most one scene regeneration. We use a single scene candidate ($N = 1$) for the evaluation.

Verification and acceptance. After text compositing, the Verifier evaluates the resulting video using programmatic text and geometry checks together with VLM-based visual inspection. Let r denote the resulting verification report. The report contains a Boolean pass decision and an online score in $[0, 1]$ used only for execution-time acceptance and repair.

Before visual-audit constraints are applied, the online score is computed as

$$\begin{aligned} s_{\text{online}} = 0.30\, q_{\text{read}} + 0.18\, q_{\text{contrast}} + 0.22\, q_{\text{nonocc}} + 0.10\, q_{\text{stable}} \\ + 0.10\, q_{\text{layout}} + 0.10\, q_{\text{semantic}}, \end{aligned} \tag{13}$$

where the terms measure text readability, visual contrast, subject non-occlusion, temporal stability, layout intrusiveness, and semantic support, respectively. In the evaluated execution path, the semantic-support term takes its default value when it is not supplied to the programmatic scorer; semantic failures identified by the VLM audit are instead enforced through the rejection rules below.

The final report score is set to zero when the required visual audit is incomplete and is capped at 0.40 when a hard failure is detected. An output is accepted only if a verification report is produced, the audit is complete, no hard failure is present, and

$$r.\text{total_score} \geq \gamma, \qquad \gamma = 0.75. \tag{14}$$

Otherwise, the output is routed to repair. Table 3 summarizes the main rejection conditions. These online verification signals are used only to control generation and repair and are separate from the IVBench evaluation metrics.

Table 3: **Online verification settings.** Main rejection conditions used during online execution and repair.

Check	Condition for rejection
Text and geometry	High-priority text OCR < 0.75; text–subject overlap normalized by text area > 0.15; co-visible text-box IoU > 0.15; panel area > 0.30 of the frame; or clipped/out-of-frame text.
Scene and semantics	Unwanted in-video text, no usable safe region, black or empty footage, missing required scene content, or required VLM judgment < 0.50.
Audit validity	Missing required checks, inconsistent judgments, or an incomplete visual audit.
Optional region check	Text containment in the planned region < 0.80 when this check is enabled.

Repair routing. Failed outputs are routed according to the diagnosed failure. Layout failures trigger overlay refinement while keeping the generated scene fixed. Scene-content failures trigger

prompt revision and scene regeneration when the regeneration budget permits. Contract-level failures trigger contract revision followed by regeneration. Each repaired output is verified again until it passes or the corresponding repair budget is exhausted.

Layout repair allows up to $L = 2$ edits. Scene-prompt and contract repair share a regeneration budget of $G = 1$, allowing at most one additional scene after the initial generation.

B.3 Online Execution Algorithm

Algorithm 1 summarizes the complete online workflow.

Stopping and retention. Layout repair keeps the generated scene fixed and updates only the text overlay. Prompt and contract repairs consume the shared regeneration budget. After each repair, the output is rendered and verified again. Execution stops when the output passes verification or the corresponding repair budget is exhausted. Generation, rendering, and verification errors are recorded as explicit failures.

Here, L, G, and C denote the budgets for layout repair, scene regeneration, and contract revision, respectively. We use $L = 2$, $G = 1$, and $C = 1$ in evaluation. Scene-prompt and contract repairs share the regeneration budget G, so at most one additional scene is generated after the initial sample.

C Verifier-Guided Dual Evolution Details

C.1 Evolution Loop

ACT evolves the harness and generator from the same stream of execution feedback. For each production cycle, the current harness h_t and generator p_{θ_t} process a batch of requests and produce execution records. These records are used both to evolve the harness and to construct direction-verified preference pairs for generator adaptation. The retained updates (h_{t+1}, θ_{t+1}) are then used for the next production cycle. Either component may remain unchanged when no update satisfies its acceptance conditions. Algorithm 2 summarizes this process. Here, $\mathcal{R}_t^{\mathrm{dev}}$ denotes the development subset of $\mathcal{R}_t$.

Execution records are accumulated as requests are processed. Passing raw scenes are retained in the pool $\mathcal{W}$ until enough eligible samples are available for preference-pair construction and generator adaptation.

C.2 Harness Evolution Details

The editable harness is $h = (P, R)$, where P contains the Planner's contract-planning and text-layout strategies and R contains the renderer's text presentation rules. Harness evolution jointly searches these components while changing one component in each proposal.

Search data. Harness evolution uses execution records from the same batches used for generator evolution. Within each batch, requests are split approximately $60/20/20$ into development, held-out, and test subsets. The development subset is used for failure diagnosis, proposal generation, evaluation on stored videos, and preference-pair collection. For generation-planner updates that pass this initial evaluation, a small portion of the held-out subset is used for regeneration-based validation, while the test subset is reserved for post-search evaluation. During regeneration-based validation, the generator state, reference input, random seed, and sampling settings are kept identical between the current and candidate planners.

Diagnostics and failure evidence. For development subset $\mathcal{D}$, the lower-is-better diagnostic indicator is

$$\mathbf{e}(h; \mathcal{D}) = \left(\mathbf{e}_R, \mathbf{e}_{P_{\mathrm{layout}}}, \mathbf{e}_{P_{\mathrm{contract}}}\right), \tag{15}$$

Algorithm 1 ACT online execution

Require: Request x, fixed harness h, generator p_θ; repair budgets L, G, C
Ensure: Output video y

$n_{\text{lay}}, n_{\text{gen}}, n_{\text{con}} \leftarrow 0, 0, 0$
$c \leftarrow \textsc{Planner}(x)$
$v \sim p_\theta(\cdot \mid \pi(c), r)$
$o \leftarrow \textsc{SceneAwareLayout}(v, c)$
$y \leftarrow \textsc{Render}(v, o, \tau)$
$e \leftarrow \textsc{Verifier}(y, x, c)$
while e indicates a repairable failure **do**
  $d \leftarrow \textsc{Route}(e)$
  if $d = \textsc{Layout}$ **then**
    if $n_{\text{lay}} \geq L$ **then**
      break
    end if
    $o \leftarrow \textsc{Refiner}(o, v, c, e)$
    $y \leftarrow \textsc{Render}(v, o, \tau)$
    $n_{\text{lay}} \leftarrow n_{\text{lay}} + 1$
  else if $d = \textsc{Scene}$ **then**
    if $n_{\text{gen}} \geq G$ **then**
      break
    end if
    $\pi' \leftarrow \textsc{Refiner}(\pi(c), c, e)$
    $v \sim p_\theta(\cdot \mid \pi', r)$
    $o \leftarrow \textsc{SceneAwareLayout}(v, c)$
    $y \leftarrow \textsc{Render}(v, o, \tau)$
    $n_{\text{gen}} \leftarrow n_{\text{gen}} + 1$
  else if $d = \textsc{Contract}$ **then**
    if $n_{\text{con}} \geq C$ **or** $n_{\text{gen}} \geq G$ **then**
      break
    end if
    $c' \leftarrow \textsc{Refiner}(c, x, e)$
    if $c' = c$ **then**
      break
    end if
    $c \leftarrow c'$
    $v \sim p_\theta(\cdot \mid \pi(c), r)$
    $o \leftarrow \textsc{SceneAwareLayout}(v, c)$
    $y \leftarrow \textsc{Render}(v, o, \tau)$
    $n_{\text{con}} \leftarrow n_{\text{con}} + 1$
    $n_{\text{gen}} \leftarrow n_{\text{gen}} + 1$
  else
    break
  end if
  $e \leftarrow \textsc{Verifier}(y, x, c)$
end while
return y

where

$$\mathbf{e}_R = (e_{\text{leg}}, e_{\text{frame}}, e_{\text{safe}}, e_{\text{coll}}, e_{\text{font}}), \tag{16}$$

$$\mathbf{e}_{P_{\text{layout}}} = (e_{\text{occ}}, e_{\text{region}}), \tag{17}$$

$$\mathbf{e}_{P_{\text{contract}}} = (e_{\text{clash}}, e_{\text{cap}}, e_{\text{subj}}, e_{\text{busy}}). \tag{18}$$

The three groups measure renderer legibility and geometry, layout-level subject occlusion and region compliance, and contract-level subject–text conflicts, text-region capacity, subject placement, and region busyness, respectively.

Algorithm 2 Verifier-guided dual evolution of ACT

Require: Request stream; initial harness h_0 and generator p_{θ_0}; update budget T; collection trigger K; minimum preference pairs M
Ensure: Evolved harness h_T and generator p_{θ_T}

$\mathcal{W} \leftarrow \emptyset$ ▷ Persistent pool of eligible passing scenes
for $t = 0, \ldots, T-1$ **do**
  $\mathcal{R}_t \leftarrow \textsc{ExecuteBatch}(h_t, \theta_t)$ ▷ Accumulate execution feedback
  $(h_{t+1}, \theta_{t+1}) \leftarrow (h_t, \theta_t)$
  $\tilde{h} \leftarrow \textsc{EvolveHarness}(h_t, \mathcal{R}_t)$
  if $\tilde{h}$ satisfies the admission criteria **then**
    $h_{t+1} \leftarrow \tilde{h}$
  end if
  $\mathcal{W} \leftarrow \mathcal{W} \cup \textsc{EligiblePassingScenes}(\mathcal{R}_t^{\text{dev}})$
  if $|\mathcal{W}| \geq K$ **then**
    $\mathcal{D}_{\text{pref}} \leftarrow \textsc{BuildPreferencePairs}(\mathcal{W}, \theta_t)$
    if $|\mathcal{D}_{\text{pref}}| \geq M$ **then**
      $\theta_{t+1} \leftarrow \text{DPOP}(\mathcal{D}_{\text{pref}}, \theta_t)$
      $\mathcal{W} \leftarrow \emptyset$
    end if
  end if
end for
return (h_T, θ_T)

The Reflector also receives concrete failure examples associated with these diagnostics. These examples localize the observed failure and provide context for proposing a targeted planning or rendering update.

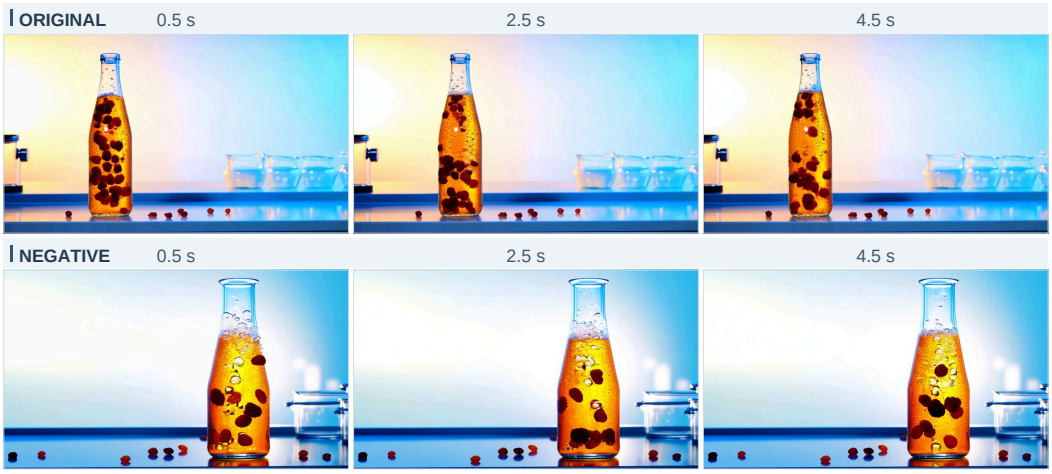
Admission details. Candidate updates are admitted according to Eq. 2. In addition to improving the targeted diagnostic and reducing the overall diagnostic error, an update must keep regressions on non-target diagnostics within predefined bounds and pass validity checks for execution, text preservation, and layout regions.

C.3 Generator Evolution Details

C.3.1 Direction-Verified Preference-Pair Settings

Preference-pair construction follows Sec. 4.4.2. The preferred sample v^+ is the raw scene from a passing execution before text compositing. The negative sample v^- is generated using a targeted prompt edit while keeping the seed, generator, reference input, and sampling settings fixed. Both v^+ and v^- are conditioned on the original prompt $\pi(c)$ during preference training. Figures 7 and 8 show two example pairs, with negative prompts proposed by the Reflector. All pairs are generated with the same Wan2.1-VACE-1.3B model at 81 frames, 16 fps, and 832×480 resolution.

For direction verification, q_k measures the contract-related spatial property targeted by the prompt edit, such as subject placement, subject visibility, or clearance of the reserved text region, with higher values indicating better performance. Pairs are admitted according to Eq. 3 with $m_k = 0.02$.



Original generation prompt	Edited negative generation prompt
Locked camera. A crisp modern science-lab bench in warm afternoon sun: on the left stands a tall clear glass bottle of amber fizzy soda, backlit so streams of tiny gas bubbles glow silver-gold and climb continuously through the liquid, while dark wrinkled raisins bob and sink along a living rising-sinking column, some cloaked in clinging bubbles that lift them until the bubbles pop and drop them again. A faint condensation sheen slides slowly down the glass, warm #F4A020 highlights rim the bottle, and a cool #7FD8E8 backlight glows behind it on the polished bench surface with scattered glass beakers softly out of focus. All the bubble sparkle, raisin motion and bench detail stay confined to the left half. The right half of the frame is a calm, evenly-lit pale blue-to-white gradient of softly out-of-focus lab wall, smooth and low-detail, with nothing crossing into it. Shallow depth of field, crisp clean cool palette of amber, cyan and pale white, bright and modern.	Locked camera. A crisp modern science-lab bench in warm afternoon sun: on the right stands a tall clear glass bottle of amber fizzy soda, backlit so streams of tiny gas bubbles glow silver-gold and climb continuously through the liquid, while dark wrinkled raisins bob and sink along a living rising-sinking column, some cloaked in clinging bubbles that lift them until the bubbles pop and drop them again. A faint condensation sheen slides slowly down the glass, warm #F4A020 highlights rim the bottle, and a cool #7FD8E8 backlight glows behind it on the polished bench surface with scattered glass beakers softly out of focus. All the bubble sparkle, raisin motion and bench detail stay confined to the right half. The left half of the frame is a calm, evenly-lit pale blue-to-white gradient of softly out-of-focus lab wall, smooth and low-detail, with nothing crossing into it. Shallow depth of field, crisp clean cool palette of amber, cyan and pale white, bright and modern.

Figure 7: **Subject-position example (bottle).**



Original generation prompt	Edited negative generation prompt
Locked camera. A cozy rustic kitchen counter bathed in warm golden morning light: a polished marble countertop where a copper saucepan of bubbling amber sugar syrup simmers in the lower half of the frame, glossy translucent strips of orange and lemon peel rolling and swirling through the amber liquid as tiny bubbles rise and burst. Halved lemons and oranges rest beside the pan, loose sugar crystals scatter across the marble catching the light, and blurred wooden shelving and warm cabinetry sit softly out of focus behind. The saucepan drifts gently from the left of the lower region to the right at constant size across the clip. Soft window light spills across the surface. The upper third of the frame is a calm, out-of-focus warm-cream haze from the sunlit kitchen air, softly and evenly lit, with no props or hard edges. Shallow depth of field, warm rustic appetizing food cinematography, glossy amber and jewel-bright citrus tones.	Locked camera. A cozy rustic kitchen counter bathed in warm golden morning light: a polished marble countertop where a copper saucepan of bubbling amber sugar syrup simmers in the lower half of the frame, glossy translucent strips of orange and lemon peel rolling and swirling through the amber liquid as tiny bubbles rise and burst. Halved lemons and oranges rest in front of the pan, partially overlapping and blocking it from view, loose sugar crystals scatter across the marble catching the light, and blurred wooden shelving and warm cabinetry sit softly out of focus behind. The saucepan drifts gently from the left of the lower region to the right at constant size across the clip. Soft window light spills across the surface. The upper third of the frame is a calm, out-of-focus warm-cream haze from the sunlit kitchen air, softly and evenly lit, with no props or hard edges. Shallow depth of field, warm rustic appetizing food cinematography, glossy amber and jewel-bright citrus tones.

Figure 8: **Occlusion example (citrus and pan).**

C.3.2 DIFFUSION DPOP

Flow-matching score. Wan2.1-VACE (Jiang et al., 2025) predicts velocity (Lipman et al., 2022). For a clean video latent $z_0^{\pm}$, we sample Gaussian noise ϵ and a timestep with noise level u_t:

$$z_t^{\pm} = (1 - u_t) z_0^{\pm} + u_t \epsilon, \qquad v_*^{\pm} = \epsilon - z_0^{\pm}. \tag{19}$$

The preferred and rejected videos share the same timestep and noise realization.

For conditioning $a = (\pi(c), r)$, let

$$e_\theta^{\pm} = w_t \frac{1}{N} \left\| v_\theta(z_t^{\pm}, t, a) - v_*^{\pm} \right\|_2^2, \qquad \Delta_{\pm} = e_\theta^{\pm} - e_{\mathrm{ref}}^{\pm}, \tag{20}$$

where N is the number of scored latent elements and w_t is the scheduler-dependent training weight, and p_{ref} is a frozen copy of the generator at the start of training.

Objective. We approximate the log-ratio score in Eq. 4 using $\hat{s}_\theta^{\pm} = -\Delta_{\pm}/2$. The resulting objective is

$$\mathcal{L}_{\mathrm{DPOP}} = \mathbb{E}\left[\mathrm{softplus}\left(\frac{\beta}{2} \left\{ \Delta_+ - \Delta_- + \lambda [\Delta_+]_+ \right\} \right) \right]. \tag{21}$$

The first two terms favor the preferred video over the negative, while the final term penalizes deterioration of the preferred sample relative to the frozen reference. Setting $\lambda = 0$ recovers the corresponding flow-DPO objective (Liu et al., 2026a).

The reference remains frozen throughout each training run, while LoRA updates are applied only to the policy generator. Training hyperparameters are provided in Appendix D.1.

D EXPERIMENTAL SETTINGS

D.1 ACT CONFIGURATION

All experiments run on a server with eight NVIDIA H200 GPUs. Tables 4 and 5 summarize the generation, evolution, and generator-adaptation settings.

Table 4: **Generation and evolution settings.**

Parameter	Setting
Generation and online execution	
Generator / agent model	Wan2.1-VACE-1.3B / Claude Opus 4.8
Online text verification	EasyOCR (online verification and repair)
Benchmark OCR	PP-OCRv6 (IVBench evaluation)
Video format	832×480; 81 frames; 16 fps
Sampling	50 steps; guidance 5.0; shift 5.0; context scale 1.0
Candidate and repair budget	One initial scene; up to two layout repairs; $\leq$ one scene regeneration
Evolution and preference collection	
Production batch	64 samples per cycle
Harness search	40 rounds; up to two proposals per round
Search split	Development/held-out/test: 60/20/20
Negative sampling	Same as positive: 50 steps; guidance 5.0; shift 5.0; context scale 1.0
Negative prompt edits	Up to three attempts
Training triggers	64 admitted pairs

D.2 EVALUATION PROTOCOL

All methods in the main comparison are evaluated on the full 500-item IVBench, while ablations and model-sensitivity studies use a fixed 200-item subset selected by stratified sampling across subtasks. Methods receive the same requested scene content, required text and timing, qualitative layout

Table 5: **Generator-adaptation settings.**

Parameter	Setting
Training hardware	Four NVIDIA H200 GPUs
Training video format	832×480; 81 frames
Optimizer	AdamW; constant learning rate 10^{-5}; weight decay 0.01
LoRA	Rank 64; scaling parameter 128
Adapted modules	Attention query, key, value, and output projections in the video transformer and VACE branch
DPOP coefficients	$\beta = 600$; $\lambda = 0.1$
Batch size	One pair per GPU; 16 accumulation steps; global batch 64
Gradient clipping	Maximum norm 1.0
Gradient checkpointing	Enabled

instructions, and available reference inputs, while numerical target regions are used only for evaluation. Different generators support different conditioning interfaces and native formats, and their specific settings and inference budgets are summarized below. Harness and generator evolution use a separate training dataset disjoint from IVBench. Each metric is first computed within a sample and then averaged across samples; detailed metric definitions are provided in Appendix A.

D.3 Baseline Implementations

Tables 6 and 7 summarize the conditioning interfaces, generation settings, and inference budgets of all baselines. T2V, I2V, and R2V denote text-to-video, first-frame image-to-video, and subject-reference-to-video generation, respectively, while Skeleton2V uses a reference skeleton video for motion conditioning. When a method does not support the required reference input, we use its T2V interface with the same textual request. Numerical evaluation regions are never provided to the generators. For methods with shorter native durations, outputs are temporally resampled to the five-second timeline before text compositing and evaluation.

Shared request-aware overlay. LVD, GenMAC, TTOM, EvoSearch, Request-aware overlay, Best-of-N, and *w/o Repair & SA* use the same frozen text overlay. It renders the requested strings at their specified intervals and qualitative locations without inspecting the generated scene or accessing evaluation-only regions.

D.4 Ablation Study Settings

Pipeline baselines. All pipeline baselines use vanilla VACE-1.3B. *End-to-end* directly generates the scene and text. *Request-aware overlay* adds the requested text after generation without text-space reservation, scene-aware compositing, or repair. *Best-of-N* retains contract planning, generates $N = 4$ candidates, applies the same text overlay, and selects the highest-scoring composite.

Component and evolution variants. Base ACT uses the original harness and vanilla generator with up to two layout repairs and one scene regeneration. All removal variants disable repair, and *w/o Repair & SA* replaces SA with the request-aware overlay. Evolution variants update only the selected component while keeping the others fixed.

E Additional Experimental Results

E.1 Independent versus Coupled Evolution

To examine the effect of coupling, we compare ACT with an *independent-then-combine* variant. Starting from the same base system, this variant evolves the harness with the generator fixed and separately adapts the generator using preference pairs collected with the original harness. The two independently updated components are combined only at evaluation. In contrast, coupled evolution jointly updates the harness and generator within the same loop.

Table 6: **Native-generator configurations.**

Method	Reference / skeleton input	Output setting	Sampler / inference setting
Commercial generators			
Gemini Omni	I2V/R2V images; Skeleton2V via video reference	720p, 16:9; 5 s	gemini-omni-flash-preview
Kling 3.0	I2V first frame / R2V subject image; Skeleton2V via Omni video reference	720p, 16:9, 5 s	3.0 for T2V/I2V; Omni for image/video reference; one shot; audio off
Seedance 2.0	I2V/R2V images; Skeleton2V via video reference	480p, 16:9, 5 s	seedance-2.0; audio off
Wan-3.0	I2V/R2V images; Skeleton2V via video reference	480p, 16:9, 5 s	wan3.0-video; prompt extension off
Open-weight generators			
SANA-Video 2B	R2V as first frame; Skeleton2V: T2V adaptation	832×480; 81 frames at 16 fps	50 steps; CFG 6
Wan2.1-VACE-1.3B	I2V/R2V images; Skeleton2V via pose-video control	832×480; 81 frames at 16 fps	50 steps; CFG 5; shift 5; context scale 1
LongCat-Video	I2V / R2V as first frame; Skeleton2V: T2V adaptation	832×480; 81 frames at 16 fps	25 steps; CFG 4
CogVideoX1.5-5B	I2V / R2V as first frame; Skeleton2V: T2V adaptation	832×480; 81 frames at 16 fps	50 steps; CFG 6
LTX-Video-2.3	I2V/R2V at frame 0; Skeleton2V via pose IC-LoRA	832×480; 121 frames at 24 fps	50 steps; CFG 3; STG 1 (block 28); modality guidance 3; rescale 0.7
HunyuanVideo-1.5	I2V / R2V as first frame; Skeleton2V: T2V adaptation	832×480 T2V; 816×480 conditioned; 121 frames at 24 fps	50 steps; checkpoint-default guider; no super-resolution stage
Wan2.2-14B	VACE-Fun-A14B; I2V/R2V; Skeleton2V via pose control	832×480; 81 frames at 16 fps	50 steps; CFG 5; shift 12; context scale 1
MiniMax H3	I2V/R2V images; Skeleton2V via Ref2VA video reference	832×480; 124 frames at 24 fps	50 sigma points (49 denoiser evaluations); guidance-distilled

Both settings use the same initialization, request batches, and per-component search and training budgets. Compared with independent-then-combine, coupled evolution improves six metrics and achieves comparable text accuracy (0.961 vs. 0.959). The gains are most evident in placement, subject IoU, and separation, showing that jointly evolving the harness and generator is more effective than optimizing them separately and combining them only at evaluation.

E.2 Additional Mechanism Comparisons

Table 9 further evaluates three mechanisms used in dual evolution. *w/o Failure exhibits* removes concrete failure evidence from harness reflection. *Target-only admission* accepts harness edits based only on improvement of the targeted diagnostic, without non-target guards. *w/o Direction gate* removes direction-based selection and filtering during preference-pair construction.

Removing failure exhibits weakens the Reflector's ability to diagnose actionable failures, with the largest drop in subject precision. More importantly, target-only admission sharply reduces placement and separation, showing that optimizing a single failure mode without guarding other dimensions can damage text–scene coordination. Removing the direction gate also consistently degrades spatial metrics, indicating that unverified preference pairs provide a weaker training signal for generator adaptation. Together, these results show that effective dual evolution depends not only on

Table 7: **Structured, editing, and pipeline baselines.**

Method	Backbone and conditioning	Generation / optimization / output selection
LVD	ZeroScope v2 576w; scene text; Skeleton2V: T2V adaptation	One sample; 40 steps, CFG 9; up to 5 latent-guidance updates per step for the first 10 steps. Native 576×320, 24 frames (about 8.33 fps). No candidate reranking.
GenMAC	VideoCrafter2; scene text; Skeleton2V: T2V adaptation	Up to 5 design–generate–verify/redesign rounds, one sample per round; 50 DDIM steps, CFG 12, $\eta = 1$. 1024×576, 16 frames at 10 fps; retain the terminal round.
TTOM	Wan2.1-T2V-1.3B / VACE-1.3B; I2V/R2V and Skeleton2V pose control through VACE	One 50-step video, CFG 5, shift 5. Rank-32 LoRA with AdamW at 10^{-3}; up to 8 updates at each of the first 5 denoising steps with early stopping. 832×480, 81 frames at 16 fps.
EvoSearch	Wan2.1-T2V-1.3B / VACE-1.3B; I2V/R2V and Skeleton2V pose control through VACE	50-step DPM++ sampler, CFG 5; populations $(4, 2, 2)$ with evolution at steps $(15, 30)$, one elite, and mutation 0.2. Eight decoded candidates are scored by VideoReward, and the highest-scoring one is retained. 832×480, 81 frames at 16 fps.
ViTeX-Edit-14B	VACE-1.3B source generator with Skeleton2V pose control; VACE-14B glyph editor	One 50-step source generation followed by one 50-step edit per caption track, CFG 5. Editing uses 1280×720, 121 frames at 24 fps; the result is composited and returned at 832×480, 81 frames at 16 fps. No candidate reranking.
End-to-end	Vanilla VACE-1.3B; Skeleton2V via pose control	One native generation; no ACT text compositor or repair.
Request-aware overlay	Vanilla VACE-1.3B; Skeleton2V via pose control	One generation without text-space reservation; request-aware overlay; no scene-aware placement or repair.
Best-of-N	Vanilla VACE-1.3B; contract-planned candidates	$N = 4$ sampled candidates; request-aware overlay on each; select the highest online-verifier score. No layout repair or regeneration.
ACT	Wan2.1-VACE-1.3B; I2V/R2V images; Skeleton2V via pose control	One initial 50-step scene, CFG 5, shift 5; up to two layout repairs; at most one additional scene regeneration.

Table 8: **Independent vs. coupled evolution.**

Variant	Text			Scene			Coordination
	Accuracy	Placement	Timing	IoU	Precision	Alignment	Separation
Base ACT	0.948	0.912	0.923	0.542	0.875	0.742	0.813
w/ Evolved harness	0.964	0.962	0.919	0.543	0.879	0.755	0.875
w/ Adapted generator	0.951	0.924	0.909	0.579	0.913	0.762	0.856
Independent-then-combine	0.959	0.965	0.927	0.588	0.938	0.770	0.891
Coupled evolution (final ACT)	0.961	0.981	0.931	0.614	0.955	0.781	0.916

updating the harness and generator, but also on reliable failure diagnosis, guarded update admission, and direction-verified supervision.

E.3 Evolution Diagnostics

Figure 9 tracks the two evolution paths. For harness evolution, let $d_{t,k}$ denote the normalized error of diagnostic k after round t. We report the display-only index

$$I_t = 100 \frac{\bar{d}_t}{\bar{d}_0}, \qquad \bar{d}_t = \frac{1}{12} \sum_{k=1}^{12} d_{t,k}, \tag{22}$$

Table 9: **Evolution mechanism comparisons**.

Variant	Text			Scene			Coordination
	Accuracy	Placement	Timing	IoU	Precision	Alignment	Separation
Full	0.961	0.981	0.931	0.614	0.955	0.781	0.916
w/o Failure exhibits	0.952	0.967	0.908	0.601	0.915	0.776	0.891
w/ Target-only admission	0.950	0.929	0.912	0.608	0.934	0.770	0.860
w/o Direction gate	0.958	0.968	0.912	0.601	0.928	0.772	0.901

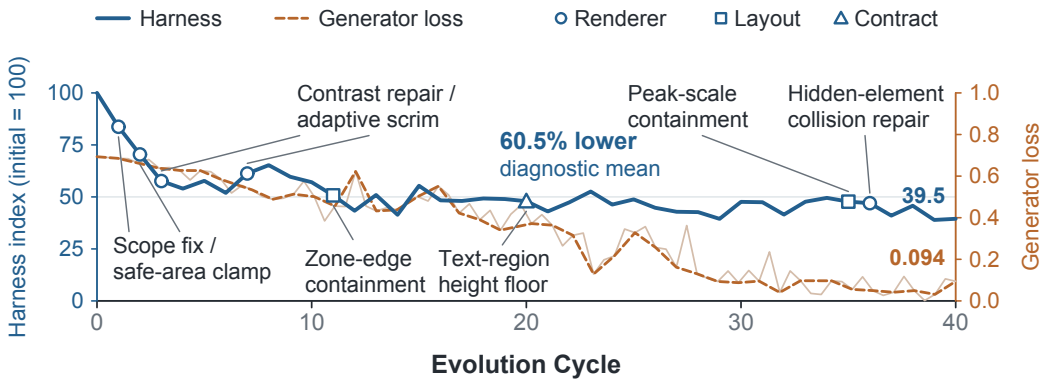


Figure 9: **Evolution diagnostics.** Harness diagnostic index and generator training loss over 40 evolution rounds. Markers denote accepted harness updates.

which averages the 11 diagnostics above together with the renderer execution-error rate. The index is computed on rotating development subsets.

Over 40 search rounds, eight harness updates are accepted and the index decreases from 100 to 39.5, corresponding to a 60.5% reduction in the diagnostic mean. In parallel, generator training loss decreases from 0.693 to 0.094. These traces show the optimization progress of harness and generator evolution. Representative harness updates include safe-area and contrast improvements in rendering, zone-edge containment in text layout, and a height floor for reserved text space in planning. Later updates further address peak-scale containment and hidden-element collisions.

Agent LLM sensitivity. Table 10 evaluates ACT with different LLMs across all agent calls on the same fixed 200-item IVBench subset. Opus 4.8, Opus 4.6, and GPT-5.6-Sol achieve similar performance across most metrics, showing that ACT is not tied to a specific proprietary LLM. Their differences are mainly reflected in spatial metrics, while text accuracy and timing remain stable. Qwen3-VL-235B shows larger drops in placement and separation, indicating that weaker agent primarily affects spatial planning and text–scene coordination. Nevertheless, it remains competitive on text accuracy, timing, subject precision, and scene alignment. These results suggest that ACT can also operate with an open-weight agent model, providing a practical option for reducing reliance on proprietary APIs and controlling inference cost.

E.4 Commercial-Backbone Study

Settings. We evaluate four variants on the same 100-sample IVBench subset, all using Kling as the frozen video generator with one scene candidate and no repair.

1. *Native Kling* receives the complete request and directly generates both the scene and on-screen text.
2. *Request-aware overlay* removes text from the generation prompt, generates a text-free scene, and adds the requested strings afterward at the specified times and qualitative locations.

Table 10: **Agent LLM sensitivity study.**

Model	Text			Scene			Coordination
	Accuracy	Placement	Timing	IoU	Precision	Alignment	Separation
Opus 4.8	0.961	0.981	0.931	0.614	0.955	0.781	0.916
Opus 4.6	0.964	0.968	0.923	0.575	0.940	0.789	0.920
GPT-5.6-Sol	0.953	0.983	0.926	0.565	0.933	0.777	0.922
Qwen3-VL-235B	0.941	0.935	0.920	0.593	0.931	0.768	0.864

3. *Contract guidance* further incorporates ACT's layout contract into scene generation, encouraging the subject to follow its planned region and leave the requested text area clear, while using the same overlay as the *Request-aware overlay*.
4. *ACT* uses the same raw scenes as *Contract guidance*, but replaces the request-aware overlay with scene-aware layout and deterministic rendering from the final harness.

Table 11: **Kling commercial-backbone controls.** Higher is better and best scores are bold. B/C share the request-aware overlay; C/D share the generated scene.

Variant	Text			Scene			Coordination
	Accuracy	Placement	Timing	IoU	Precision	Alignment	Separation
A: Native Kling	0.544	0.888	0.184	0.621	0.835	0.874	0.699
B: w/ Request-aware overlay	0.905	0.910	0.893	0.602	0.794	0.878	0.585
C: w/ Contract guidance	0.916	0.902	0.916	0.628	0.839	0.899	0.715
D: w/ ACT	**0.962**	**0.938**	**0.942**	**0.630**	**0.839**	**0.902**	**0.745**

Results. Request-aware overlay greatly improves text accuracy and timing over native Kling, but weakens text–scene separation, showing that accurate post-hoc rendering alone does not solve coordination. Adding contract guidance improves subject layout and separation, demonstrating the benefit of shaping scene composition before text rendering. Using the same raw scenes, ACT further improves text placement and separation through scene-aware compositing and deterministic rendering. Together, these results further show that ACT's planning and rendering harness could be transferred to a frozen commercial generator.

E.5 Temporal Sampling Sensitivity

Our default timing evaluation uniformly samples $F = 16$ frames from each five-second video. To test sensitivity to the temporal sampling density, for each saved video, we re-extract $F \in \{16, 32, 64\}$ uniformly spaced frames, run the same PP-OCRv6 recognizer and text-match threshold, and recompute timing IoU for Kling 3.0, MiniMax H3, and ACT. Table 12 reports the mean per-video timing score.

Table 12: **Temporal sampling sensitivity.**

Method	$F = 16$	$F = 32$	$F = 64$
Kling 3.0	0.177	0.170	0.170
MiniMax H3	0.616	0.615	0.618
ACT (Ours)	0.933	0.958	0.951

The method ordering remains ACT (Ours) $>$ MiniMax H3 $>$ Kling 3.0 at all three sampling densities. ACT varies by at most 0.025 across the three settings, while MiniMax H3 and Kling 3.0 vary by 0.003 and 0.007, respectively. These results show that denser temporal sampling can slightly change the absolute timing scores, but does not affect the comparative conclusions, supporting $F = 16$ as a reasonable default for evaluation.

F Cost Analysis

Online execution cost. We report the per-request online execution cost. The evaluated configuration generates one 832×480 scene with 81 frames using 50 diffusion steps, followed by at most two layout repairs and one video regeneration. Table 13 summarizes the averaged statistical values. The mean end-to-end latency is 246.2 s, of which scene generation accounts for 102.5 s on average.

Table 13: **Online execution cost.** Statistics are computed on 500 samples.

Statistic	Mean $\pm$ SD	Median
End-to-end request time (s)	246.18 ± 87.23	227.19
Scene-generation time (s)	102.47 ± 59.25	81.83

Repair workload. Our evolution also reduces the amount of online correction required. Under the same online budget, the mean number of repairs decreases from 1.48 to 0.15, while first-pass acceptance increases from 14.8% to 74.2% and final-pass acceptance from 37.2% to 86.4%.

Table 14: **Repair workload before and after evolution.**

System	Mean repairs	First-pass (%)	Final-pass (%)
Initial ACT	1.48	14.8	37.2
Final ACT	0.15	74.2	86.4

API cost. Using Claude Opus 4.8 for agent inference, ACT consumes an average of 40,211 input tokens and 8,078 output tokens per request, corresponding to an API cost of $0.403.

G Human Study Details

Participants and protocol. We randomly select 30 items from IVBench for matched ACT–MiniMax H3 comparisons. Ten volunteers from different industries participate in the study. Participation is voluntary, and no email or personal information is collected. Each trial compares the two methods on the same request and five-second timeline. Method identities are hidden using Left-/Right labels (Top/Bottom in portrait mode), with ACT shown on each side in 15 of the 30 trials.

Viewing interface. Participants view both muted videos alongside an animated reference showing the requested text, timing, and layout (Fig. 10). The reference serves as a guide to the request. Both videos share synchronized playback, pause, and replay controls.

Questions. Each trial asks participants to compare the two videos on three criteria:

1. **Information clarity:** "Which video makes the information easier to read and understand?" Hint: "Is the text correct, clear, and shown at the right time?"
2. **Text placement:** "Which video puts the text in a better place?" Hint: "Is the text where it should be, without covering the main subject?"
3. **Overall visual quality:** "Which video looks better overall?" Hint: "Think about both the scene and the text."

Each question allows either video or "About the same" as the response. All three responses are required, and participants may replay or revisit trials before submission.

Evolution comparison. Another ten participants completed 30 random before/after video pairs with the same three questions. Figure 11 shows a consistent preference for the after-evolution videos

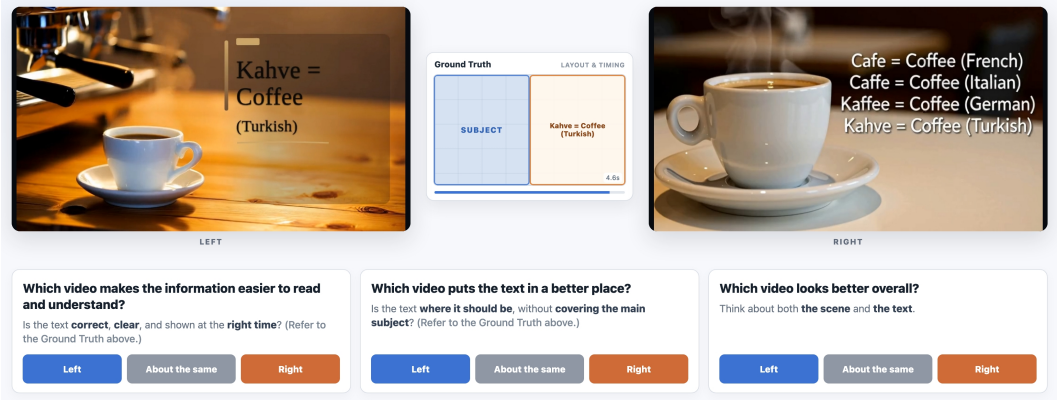


Figure 10: **Human-study interface.** Anonymous videos are shown with the text, timing, and layout reference and three preference questions.

across all three criteria. For information clarity, 58.0% of judgments prefer the after-evolution outputs, compared with 18.7% for the before-evolution outputs. The gap is larger for text placement (59.7% vs. 15.0%) and overall quality (68.7% vs. 17.7%). Overall quality also has the lowest tie rate (13.7%), compared with 23.3% for information clarity and 25.3% for text placement, indicating a more visible difference in the final presentation. These human results are also consistent with the automatic results in Table 2, where the evolved ACT achieves higher average IVBench scores than the pre-evolution system. Together, the automatic and human results show that evolution improves ACT at the system level, with the gains also being perceptible to human viewers.

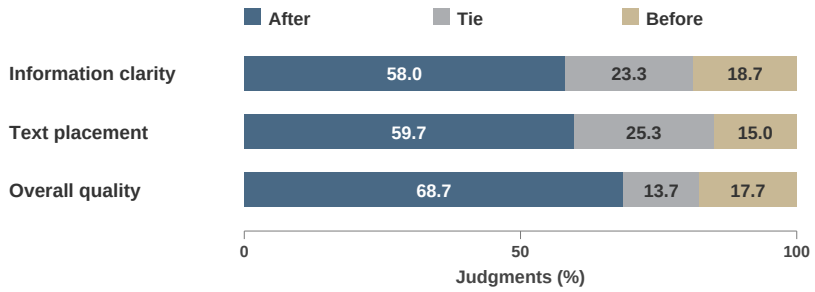


Figure 11: **Human preference before and after evolution.** Response distributions for information clarity, text placement, and overall quality.

H Qualitative Comparisons

Baseline comparisons. Figs. 12–15 show baseline comparisons on four IVBench requests. The grid displays three frames per method, with the requested text, display intervals, and spatial regions shown above it. Additional qualitative comparisons across baselines are provided as videos in the supplementary material.

Evolution examples. We show four Base ACT and evolved ACT comparisons—lightning, Portuguese vocabulary for the sun, testing a smoke alarm, and a toaster in Figs. 16–19. Each example includes the target strings, display intervals, and spatial annotations.

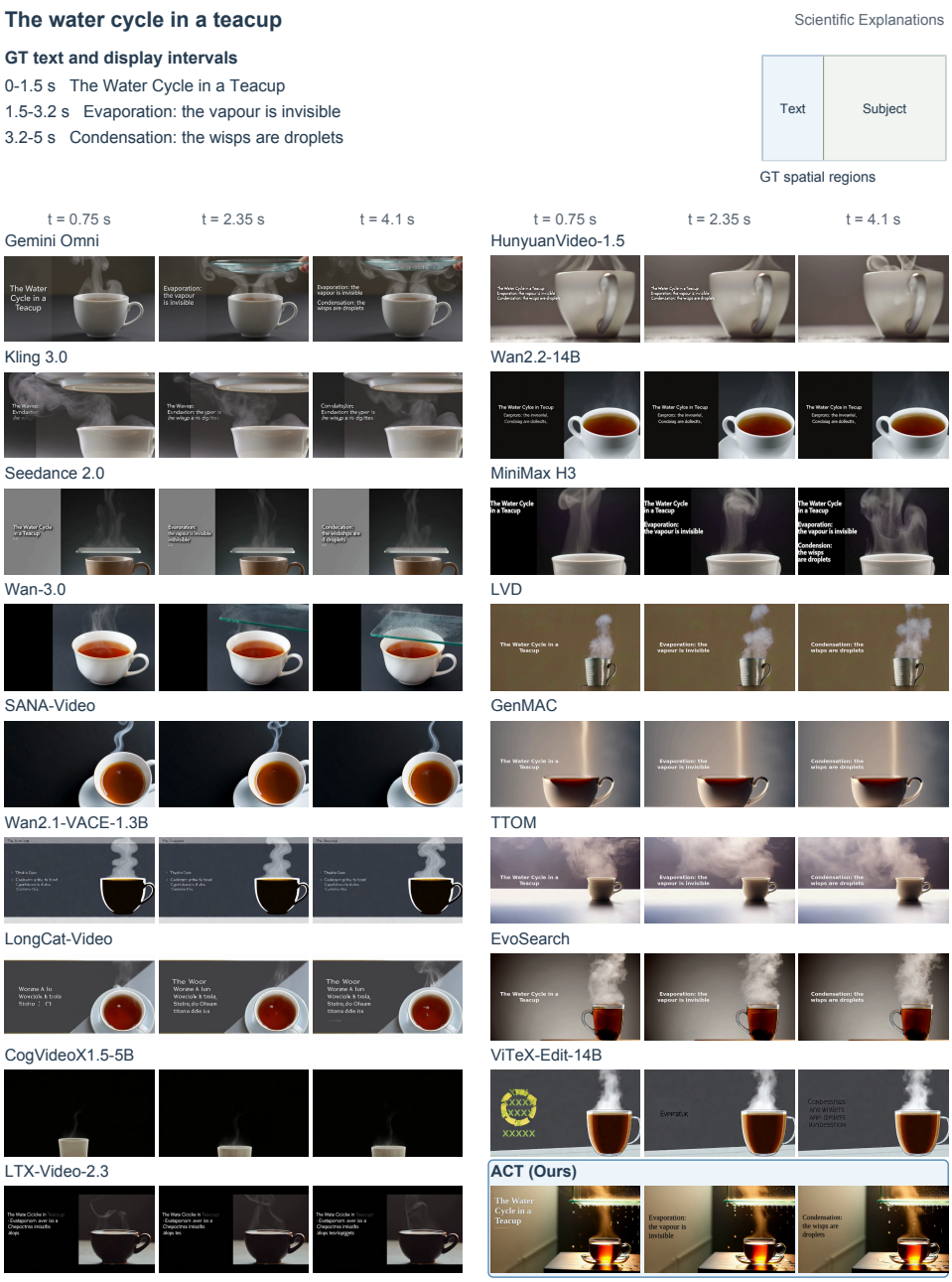


Figure 12: **Temporal baseline comparison: The water cycle in a teacup.** Frames are sampled at the indicated elapsed times; GT specifies the requested text, display intervals, and spatial regions. *Please zoom in for a better view.*

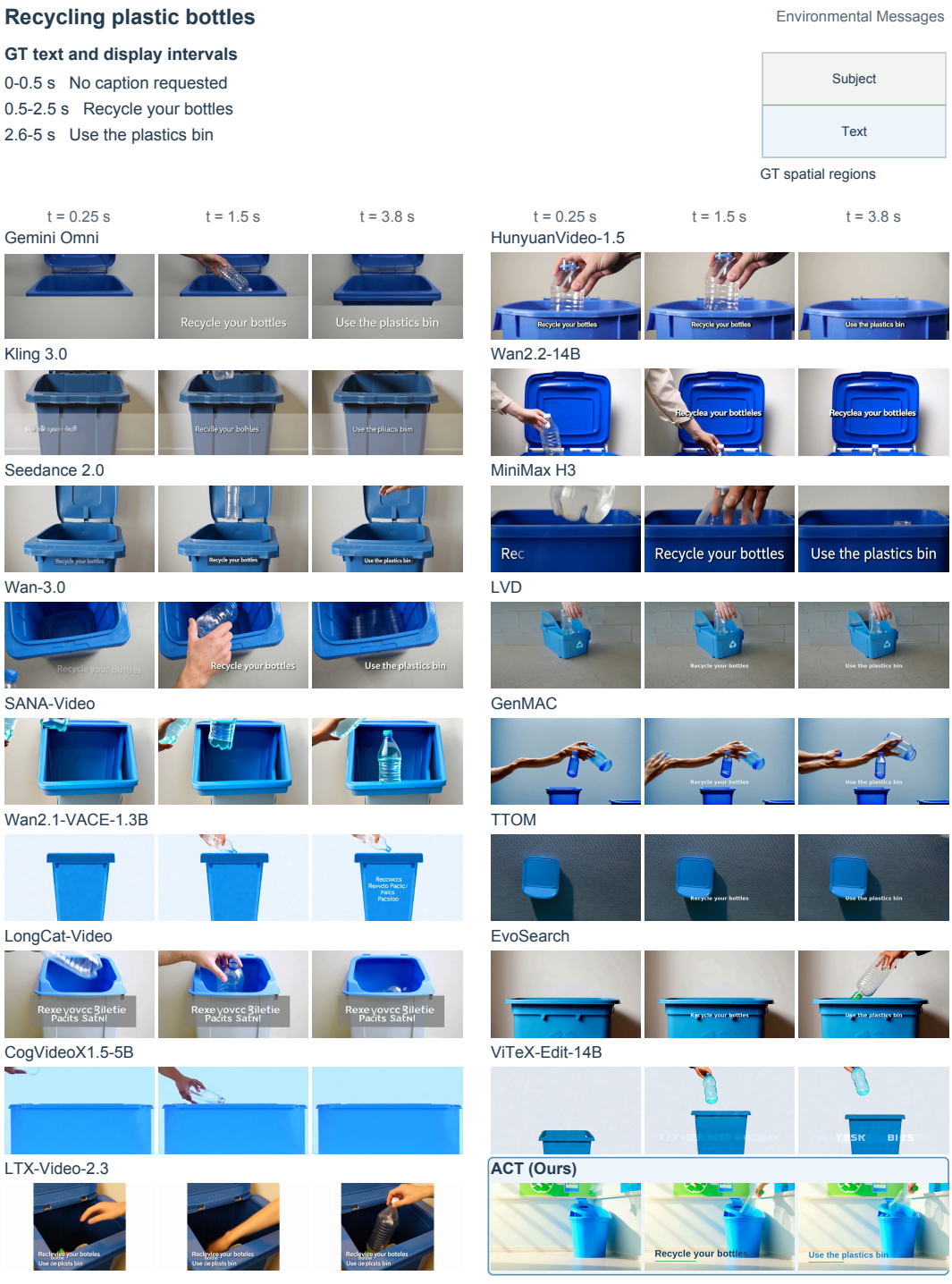


Figure 13: **Temporal baseline comparison: Recycling plastic bottles.** Frames are sampled at the indicated elapsed times; GT specifies the requested text, display intervals, and spatial regions. *Please zoom in for a better view.*



Figure 14: **Temporal baseline comparison: A group of crows.** Frames are sampled at the indicated elapsed times; GT specifies the requested text, display intervals, and spatial regions. *Please zoom in for a better view.*

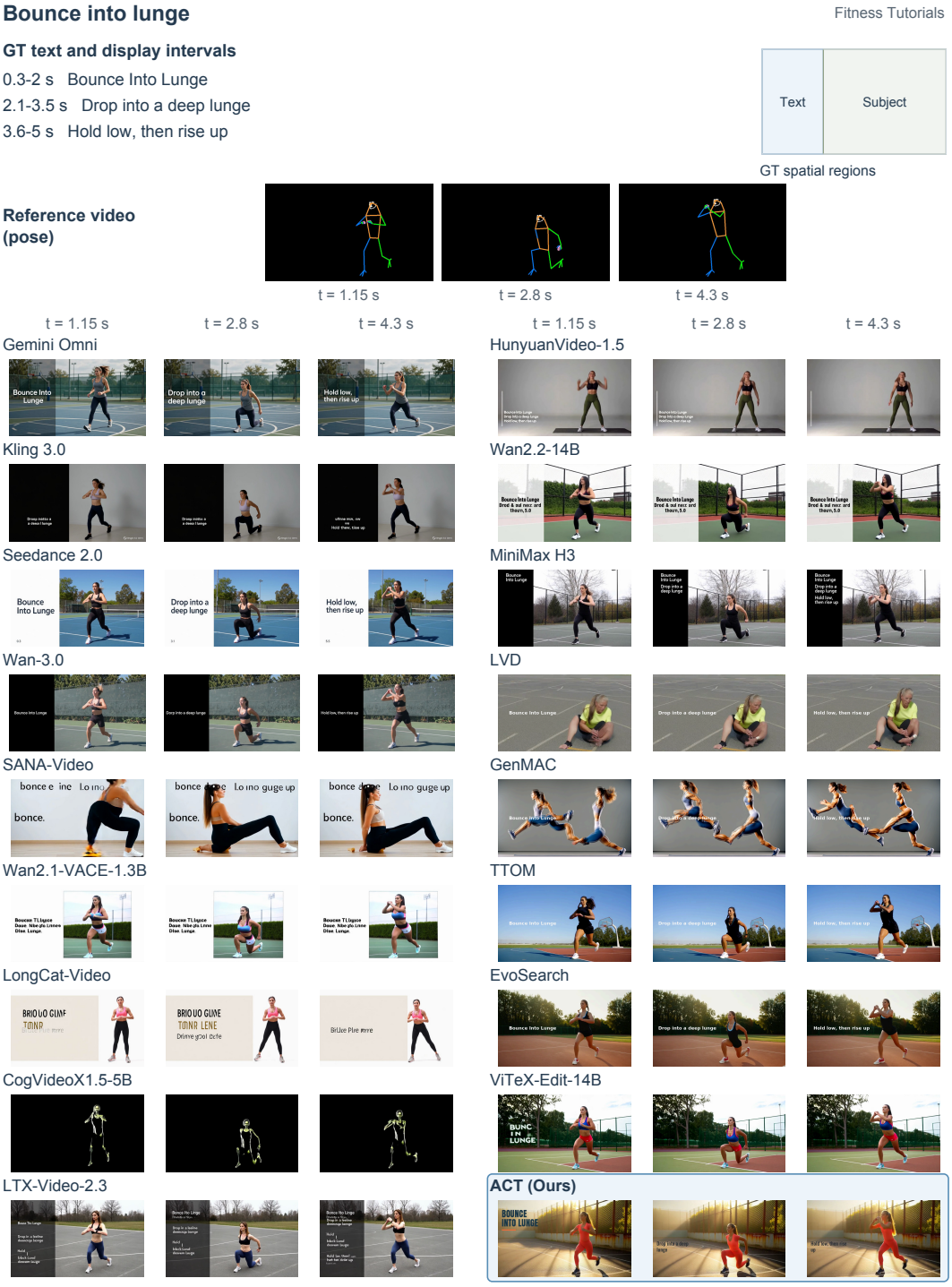


Figure 15: **Temporal baseline comparison: Bounce into lunge.** Frames are sampled at the indicated elapsed times, including the reference pose video; GT specifies the requested text, display intervals, and spatial regions. *Please zoom in for a better view.*

Lightning

Science | T2V

Introduce lightning as an atmospheric electric discharge by showing a bright bolt flashing through dark storm clouds at night, naming the phenomenon and the clouds it illuminates.

Requested text and timing (GT)
0.6-2.2 s | Lightning
2.3-3.7 s | Electric discharge
3.8-5 s | Storm cloud

GT layout



T: text S: subject

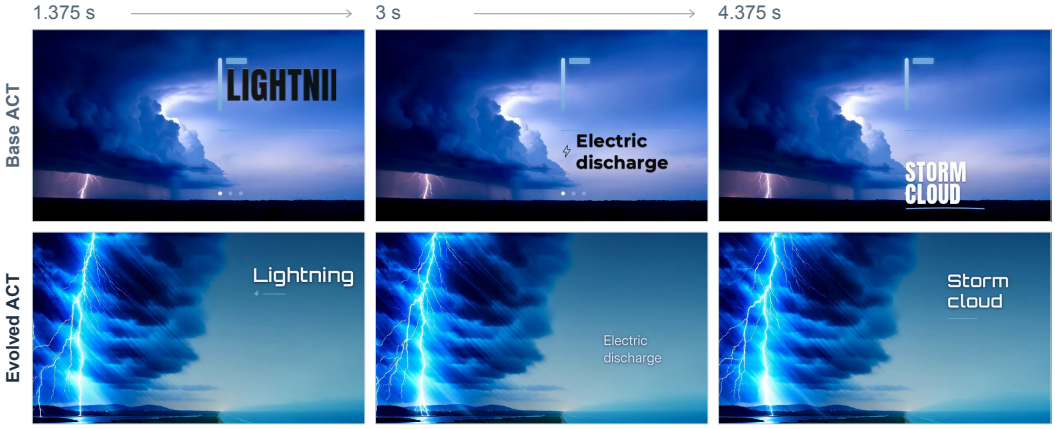


Figure 16: **Lightning.**

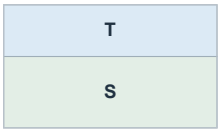
Portuguese: Sol

Language | T2V

Introduce this language sample clearly, with the overlay text legible in its region.

Requested text and timing (GT)
0-1.67 s | Learn Portuguese: Sol
1.67-3.33 s | Sol = Sun
3.33-5 s | Say it: Sol

GT layout



T: text S: subject

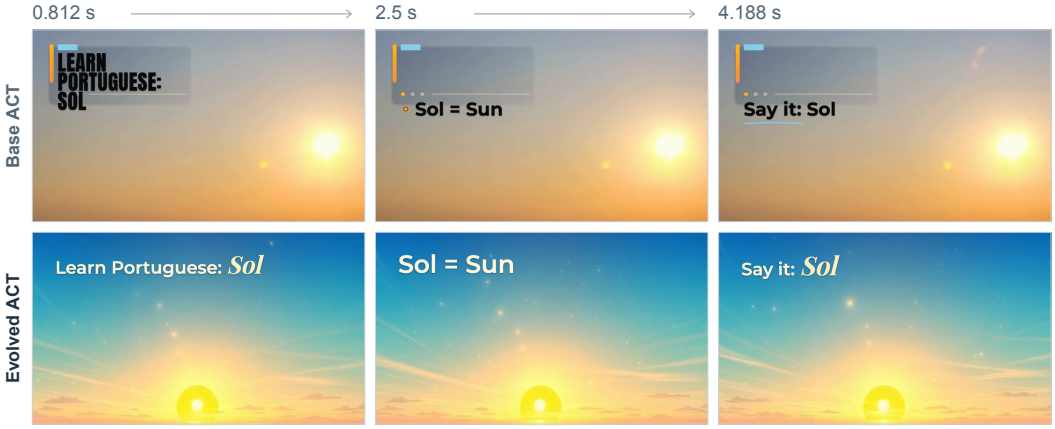


Figure 17: **Portuguese: Sol (Sun).**

Testing a smoke alarm

Safety | T2V

Demonstrate in a neutral, instructional manner how to test a ceiling-mounted smoke alarm by reaching up and pressing its central test button, while reminding the viewer to test alarms on a regular monthly cadence.

Requested text and timing (GT)
0-2.4 s | Press to test the smoke alarm
2.4-5 s | Test alarms monthly

GT layout



T: text S: subject

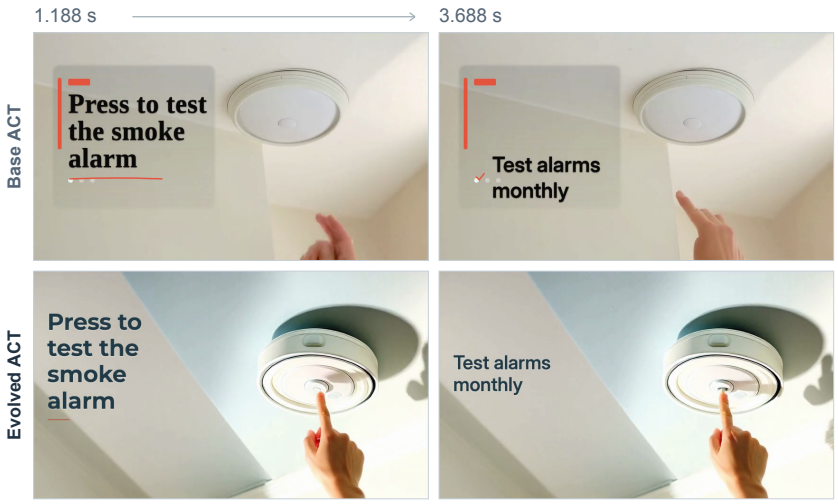


Figure 18: **Testing a smoke alarm.**

Toaster

Advertisement | R2V

Showcase the sleek matte white design with soft lighting, leaving a narrow strip along the right edge clear for overlay text.

Requested text and timing (GT)
0.3-3.6 s | Simple. Sleek. Daily.
3.2-5 s | Elevate your morning

Reference image



GT layout



T: text
S: subject

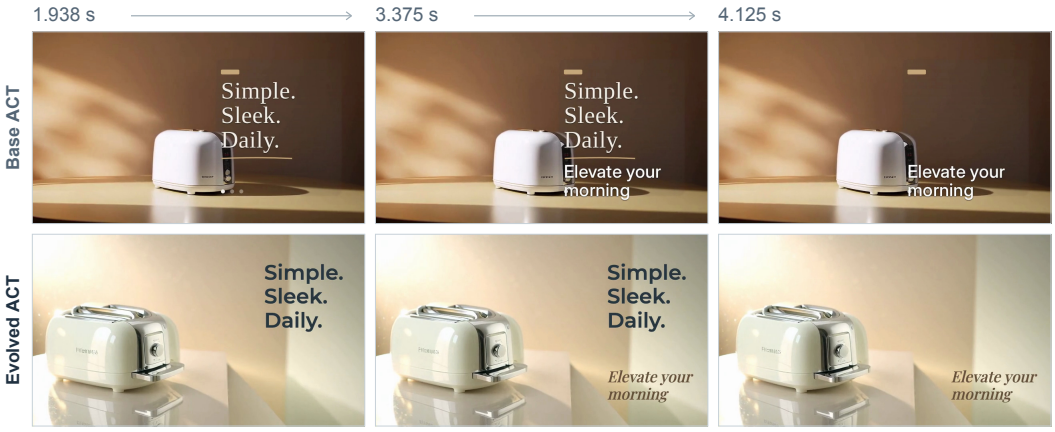


Figure 19: **Toaster.**

I LIMITATIONS

ACT uses Wan2.1-VACE-1.3B to enable efficient generator adaptation, but the limited capacity of this backbone can lead to imperfect scene content and motion that cannot be corrected by text rendering or layout coordination. ACT also introduces additional inference cost from agent calls, verification, and repair, increasing token usage and latency. Reducing context length and adjusting verification budgets based on request difficulty may improve efficiency. ACT relies on OCR, subject detection, and vision-language models for verification and evolution. Errors in these components can propagate to repair decisions and training feedback, limiting overall system performance. Finally, ACT currently focuses on short, single-shot videos with screen-aligned graphic overlays; extending it to physical scene text, multi-shot narratives, and audio-synchronized captions remains future work.

J REFLECTOR PROMPTS

We provide the two task-specific prompt templates used by the unified Reflector for dual evolution. Boxes reproduce prompt excerpts; [. . .] marks omitted text, and whitespace is reflowed for readability. Braced names denote runtime placeholders, and doubled braces represent literal JSON objects.

For harness evolution, the Reflector proposes reusable edits to the planning or rendering harness based on diagnostic profiles, failure exhibits, allowed surfaces, source text, and rejection feedback. For generator evolution, the same Reflector proposes a targeted prompt edit for constructing a preference pair, expressed as a small program executed under a fixed primitive budget.

Reflector Prompt for Harness Evolution

Instruction template

Runtime input: Profile, integrity errors, failure exhibits, target menu, surface restrictions, current source, and prior feedback.

Response: JSON with `target`, `surface`, exact `find/replace` edits, and rationale.

You are the Harness Reflector. Propose one reusable update to an allowed harness surface: renderer JavaScript, layout-agent instructions, or generation-planner instructions. Choose the surface whose responsibility matches the diagnosed failure, using the surface note below. Candidate harness edits are evaluated on stored raw footage; this search does not regenerate that footage or update generator weights. Planner edits change the contract or layout inferred during this evaluation. Accepted edits persist in the harness used for subsequent requests.

[. . .]

{profile}

{integrity}

{exhibits}

{surf_note}

YOUR TASK THIS ROUND. Decide for yourself which single dimension to attack, and say why. Choose the one you believe you can actually move with one patch, which is not always the largest one: a dimension is a good choice when the exhibits point at a specific mechanism you can name, and a poor choice when you would be guessing. State the dimension in the "target" field using exactly one of these names: {menu}. If the only name offered is "integrity", the choice has already been made for you by the exception above.

Your patch is accepted only if all three hold:

- the target dimension improves;

- the TOTAL number of failing items summed over every dimension listed above goes down, so a gain of six items in one dimension may cost at most five items spread over the others. Each dimension is counted in ITEMS on its own denominator, which is printed beside it: the two planner dimensions are

Reflector Prompt for Harness Evolution (continued)

measured on a smaller cohort than the renderer ones, so one item there is worth more rate than one item here;

- no single dimension rises by more than {collapse:.0%} of items, and no dimension that is already worse than it was at the start of this run gets worse still. Trading a little of one dimension for a lot of another is allowed and sometimes necessary. Wrecking one to win another is not, and a dimension you damage will have to be repaid before you are allowed to look elsewhere. A single patch may contain several edits, so if a fix needs a precondition changed as well as a mechanism – for instance a guarantee that is applied to the wrong set of elements AND cannot measure them – put both in the one patch, because a patch carrying only half of it will measure zero and be rejected.

[. . .]

{rules}

CURRENT SOURCE: {bodies}

Return ONLY a JSON object, no prose and no markdown fence:

{{"target": "<one of {menu}>", "why_this_dimension": "<one sentence on why this one is the most movable right now>", "surface": "<one allowed surface from the surface note>", "edits": [{{"find": "<exact existing text>", "replace": "<new text>"}}], "rationale": "<one sentence on the mechanism>"}} Set "surface" to exactly one allowed name: renderer, layout_agent, or generation_planner. Every "find" must occur EXACTLY ONCE in the selected surface's source. Include enough surrounding text to be unique and copy it character for character. Listing headers are not source text and must not appear in a "find". At most 6 edits; all edits must target the same surface.

{prior}

[. . .]

Audited design constraints. A patch that violates one is discarded regardless of its measurements:

- no glyph outline: do not set -webkit-text-stroke to a non-zero width, and do not build a halo out of several stacked text-shadow layers

- no panel over the subject: a backing panel may not be introduced across the middle of the frame

- do not remove or rewrite text: a candidate showing fewer visible strings than the start is rejected

- do not special-case a particular item id, brief, or string

Reflector Prompt for Generator Evolution

Instruction template

Runtime input: The selected failure diagnosis and the preferred raw scene's video prompt.

Response: A restricted program that edits the prompt, reuses the seed, generates once, and assigns `LOSER`.

You construct HARD NEGATIVE examples for preference training of a text-to-video generator, using ONLY natural edits to the prompt (then re-generating). This generalizes a known-good recipe: for a "subject on the left vs right" failure, one flips the direction word ("left"->"right") and re-generates with the same seed, checkpoint, and decoding settings. This controls the generation inputs but does not guarantee that non-target visual attributes remain unchanged; the resulting pair must pass the downstream verification filters before admission. Your job is to find the analogous MINIMAL, NATURAL edit for THIS failure – it need NOT be a direction word; invent whatever single wording change makes the failure appear (a spatial-relation phrase, a position/placement clause, a count, an ordering, an orientation, a "in the foreground/background" clause, etc.).

FAILURE DIAGNOSIS (all you get about what went wrong):

{diagnosis}

THE WINNER PROMPT (the video WITHOUT the failure was generated from this – edit a span of it):

{winner_prompt}

Reflector Prompt for Generator Evolution (continued)

REQUIREMENTS

* The edit must be a SINGLE, natural, minimal change to the wording – the kind a person would make by rephrasing one clause – that causes the diagnosed failure to appear.

* The generation inputs must differ only in the intervened prompt wording; reuse the seed and fixed generation settings.

* Do NOT write a generic "make it worse" prompt. Touch exactly the one variable the diagnosis names and leave the rest of the prompt verbatim.

* Prefer a replace on the exact offending span over inserting new sentences.

{contract}

[. . .]

You may call ONLY these three primitives. Nothing else exists.

edit_prompt(base=None, op="insert"|"delete"|"replace", span="<exact substring>", value="<text>") -> the edited prompt string. base=None edits the winner's prompt. 'span' MUST be an exact substring of the text you edit. Make the SMALLEST edit that flips the named variable.

seed(reuse=True) -> the winner's seed (matched-seed: keep this so only the prompt differs).

generate(prompt=None, seed_value=None) -> path to a generated .mp4 (fixed checkpoint/decoding).

BUDGET: at most 4 primitive calls, at most 1 generate call.

RULES

* Emit a FLAT program: no imports/def/loops/comprehensions/attributes/subscripts.

* You MUST call edit_prompt at least once, then generate once, then assign the result to LOSER.

* seed(reuse=True) – matched seed, so the loser differs ONLY in the intervened wording.

* Change the ONE causal variable the diagnosis names; keep every other word intact.

Output ONLY the program text. No markdown fence, no commentary.