INTERACTIVE VIDEO STORYTELLING WITH CONTEXTUAL GROUNDING

Anonymous authors
Paper under double-blind review

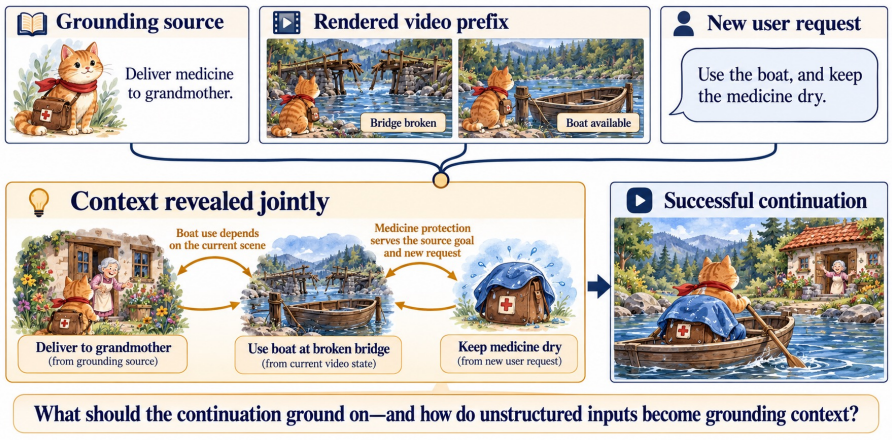

Figure 1: In interactive video storytelling, a viewer's request has complex latent dependency with the grounding source and the states established in a previously generated video prefix. These dependencies become constraints when they are projected onto a candidate future video continuation. In this work, we formulate the task as **Interactive Video Storytelling with Contextual Grounding**.

ABSTRACT

Interactive video storytelling enables viewers to actively steer how a video unfolds. However, once we allow viewers to intervene during generation, a new challenge arises: The viewer's request can have latent dependencies on both the grounding source and the current rendered video state. These dependencies may not be explicitly stated in any individual input, but emerge only when the source, rendered history, and new viewer intent are considered jointly. Existing interactive video generation systems primarily emphasize following viewer instructions, while source-grounded video generation methods focus on aligning generated content with an external narrative or knowledge source. This leaves a fundamental question underexplored: What context should a generation model ground on during interactive continuation, and how can heterogeneous, unstructured inputs be transformed into such grounding context? In this work, we formulate contextual grounding as the process of transforming heterogeneous input context into an executable constraint model for video generation. To address this challenge, we introduce ***VideoTaleInteract***, which performs Structured Context Abstraction to recover grounded states and cross-context dependencies, Generation Constraints Induction to project relevant dependencies into candidate-specific constraints, and Constrained Video Generation to enforce these constraints through planning, verification, revision, and rendering. Across three video generation backbones, ***VideoTaleInteract*** consistently achieves the highest overall composite score, reaching an average absolute improvement of 10.3 points over the strongest per-backbone baselines. Detailed analysis further shows gains across both narrative and knowledge grounding, and reveals remaining challenges in dependency extraction, and faithful realization during video rendering.

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

1 INTRODUCTION

Video storytelling communicates narratives and explains complex ideas through language, imagery, motion, and sound. Yet a viewer's interests are rarely fixed before watching begins: A narrative choice may invite questions about consequences, or an unfamiliar concept may prompt a need for concrete examples. Such requests arise from the content just observed, motivating storytelling systems that allow viewers to shape how a video continues. While human–AI writing tools like Wordcraft demonstrate the value of interaction during story creation (Yuan et al., 2022), this remains underexplored in video storytelling. In our formative study of knowledge-grounded video storytelling, the reported mean pre-to-post gains were 47.9 percentage points with interaction and 32.1 without interaction. These results motivate further study of interaction as a way to support viewers' understanding.

Supporting such interaction, however, is not simply a matter of appending a new prompt to a video generator. Once part of a video has been rendered, it establishes concrete visual states, events, and information that subsequent generations must respect. At the same time, an underlying narrative or factual source may impose additional dependencies that are not directly visible in the video. A viewer request therefore cannot be interpreted in isolation: some aspects of the continuation may adapt freely, while others are constrained by what has already happened or by what the grounding material supports. Direct generation from the source, video prefix, and latest request leaves these relationships implicit, making it easy to introduce incompatible state changes, contradict previously communicated information, or satisfy the request in a way unsupported by the source. The central challenge is therefore not only *what to generate next*, but *how to determine which contextual information matters and how it should constrain the continuation*.

We formulate this problem as **Interactive Video Storytelling with Contextual Grounding**. Given a grounding source (e.g., a narrative, paper, wiki page, or reference document), a previously rendered video prefix, and a new viewer request, the system generates a continuation that responds to the viewer while remaining compatible with the established context. To address the viewer's new request, it requires jointly reasoning over three heterogeneous forms of information: the grounding source specifies what is supported, the video prefix records what the viewer has actually observed, and the evolving story captures how events and information relate over time.

Existing work provides important pieces of this capability, but does not directly address their combination. Recent video storytelling systems develop increasingly capable pipelines for story planning, multimodal production, and coherent long-form generation (Xu et al., 2025; Wu et al., 2025b; Zheng et al., 2024; Song et al., 2026; Huang et al., 2026). Their primary formulations, however, generate videos from an initially specified story, script, or creative intent. Complementary work on interactive video generation enables generation to evolve with changing text prompts (Yin et al., 2025; Kodaira et al., 2026; Yang et al., 2025; Feng et al., 2026), while interactive world models support action-conditioned exploration and controllable world events (Bruce et al., 2024; Decart & Etched, 2024; He et al., 2025; Huang et al., 2025; Google DeepMind, 2025a). These advances make video generation increasingly responsive, but responsiveness alone does not determine whether a newly requested continuation is compatible with the source and with what has already been shown. What is missing is an mechanism for transforming these heterogeneous inputs into a grounded representation of the current context and reasoning about the consequences they impose on future generation.

Motivated by this intuition, we introduce ***VideoTaleInteract***, a framework that separates *what the available context establishes* from *what its relationships imply for generation*. ***VideoTaleInteract*** first performs **Structured Context Abstraction**, transforming them into aligned video, story, and source states that make relevant entities, events, claims, state transitions, and cross-input relationships explicit. It then performs **Generation Constraints Induction**, reasoning over this structure to derive instance-specific conditions that a valid continuation should satisfy. These grounded constraints provide an explicit interface between contextual understanding and generation: they guide script planning, identify unsupported or conflicting proposals during verification, and provide targeted feedback for revision. Finally, because a valid script does not guarantee that a video generator faithfully realizes it, we apply contextual verification both before and after rendering through an iterative plan–verify–revise–render process.

To systematically study this problem and address the absence of dedicated benchmarks for the task, we further introduce *IVS-Bench*, a benchmark pairing grounding sources, observed video prefixes,

and viewer interventions across narrative- and knowledge-grounded settings. Our evaluation measures source faithfulness, story continuity, and interaction fulfillment individually and jointly, together with visual consistency and video quality, allowing us to distinguish contextual grounding from perceptual generation quality. Across three different video generation backbones (Omni-1.1-Flash (Google, 2026c), Veo 3.1 (Google DeepMind, 2025b), and MiniMax-H3 (MiniMax, 2026)) ***VideoTaleInteract*** achieves the highest overall composite score with each backbone. The gains over the strongest corresponding baselines are 12.3 points for Omni, 15.3 for Veo, and 3.4 for MiniMax.

2 Interactive Video Storytelling with Contextual Grounding

2.1 Formative Study

Prior work on human–AI co-creation suggests that viewer intent often evolves after generated content becomes visible. Analyses of large-scale human–AI interaction logs find that viewers revise, extend, or reformulate their intentions after observing generated outputs in 18–30% of sessions (Mysore et al., 2025).

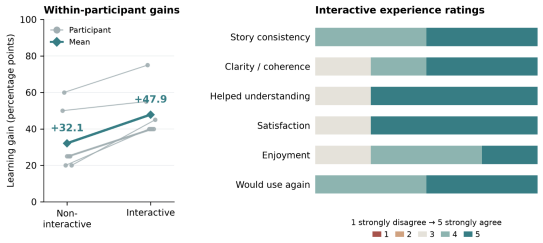

Figure 2: **Formative Study Results.** The study enrolled nine volunteers. *Left:* Learning gains for the seven participants who passed the attention test. *Right:* Experience ratings from four respondents.

We conducted a formative study with nine volunteers to examine interaction in knowledge-based video storytelling. Figure 2 reports learning gains for the seven participants who passed the attention test. Among these seven participants, mean gains were higher with interaction; the experience ratings were concentrated toward the positive end of the five-point scale. These exploratory observations motivate studying how viewers can ask questions and redirect a continuation while preserving its grounding source and evolving video context. They do not establish a learning-gain estimate for all nine participants. See Appendix F.1 for details.

2.2 Task Formulation

Given a grounding source $\mathcal{G}$, a previously generated video prefix v_{prefix}, and a current viewer request u, the model generates a video continuation v_{cont}: $v_{\text{cont}} \sim p_\theta(v_{\text{cont}} \mid \mathcal{G}, v_{\text{prefix}}, u)$ Here, $\mathcal{G}$ provides persistent context the generated story should respect, and v_{prefix} provides the visual and narrative context for the continuation. Rather than explicitly conditioning on all the previous viewer instructions, each continuation relies on v_{prefix} as a sufficient record of earlier instructions.

3 Method

We introduce ***VideoTaleInteract***, a framework for interactive video storytelling that treats contextual grounding as a *context compilation* problem.

Our central idea is to compile heterogeneous context into a *grounded dependency model* and use this model to control subsequent generation. As illustrated in Figure 3, ***VideoTaleInteract*** first converts the raw inputs into grounded facts and cross-context dependencies (Sec. 3.1). These dependencies are then projected onto a candidate continuation, where only the relations made relevant by the proposed future are instantiated as generation constraints (Sec. 3.2). Finally, the resulting constraints guide planning, script repair, rendering, and output-level correction (Sec. 3.3). Appendix C expands the representation and repair procedure, with an illustrative dependency trace in Appendix C.3.

3.1 Compiling Context into Grounded Dependencies

The grounding source, video prefix, and viewer request encode different views of an evolving situation. The source may establish factual or narrative relations, the rendered prefix determines the current visible state, and the viewer request specifies how the story should continue. The challenge

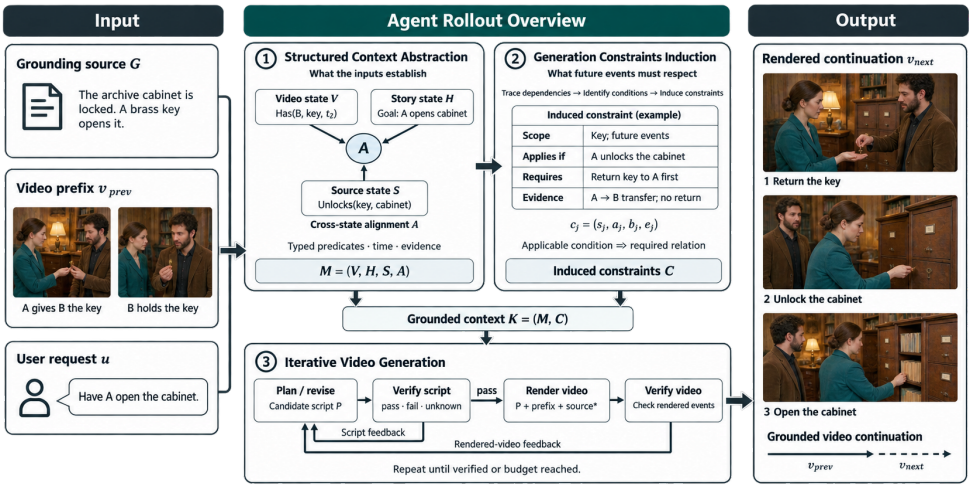

Figure 3: **Overview of *VideoTaleInteract*.**

is not merely to summarize each input, but to recover the dependencies that emerge from their interaction.

Grounded State Extraction. We first transform each input into a structured set of grounded facts. We extract entities, attributes, events, relations, and state transitions using typed predicates with arguments, temporal scopes, and evidence links with VLMs. We then organize these predicates into three complementary states:

- **Video state** $\mathcal{V}$ represents what has been visually established in the rendered prefix, including entities, attributes, locations, possession, actions, and state transitions. Temporal annotations distinguish prior observations from the latest established state.
- **Story state** $\mathcal{H}$ represents the evolving narrative state, including completed events, active goals, unresolved relations, causal dependencies, and information already conveyed.
- **Source state** $\mathcal{S}$ represents atomic claims, events, definitions, and relations supported by the grounding source, while preserving qualifications, conditions, and supporting evidence.

Cross-Context Dependency Construction. The aligned states are compiled into a grounded dependency model $\mathcal{M} = (\mathcal{V}, \mathcal{H}, \mathcal{S}, \mathcal{A})$, where $\mathcal{A}$ contains cross-context alignments and $\mathcal{R}$ contains dependencies inferred from the aligned predicates. The dependency expresses how one established fact constrains another event or state. These relations include temporal precedence, causal prerequisites, state-transition requirements, persistence relations, compatibility conditions, and source-supported dependencies. Importantly, many such dependencies are not explicitly stated in any individual input. They arise only after the inputs are jointly grounded.

3.2 Projecting Dependencies into Generation Constraints

We induce constraints from dependencies $\mathcal{R}$ in the grounded model $\mathcal{M}$, activating those relevant to each candidate continuation. These dependencies link the source, rendered prefix, and viewer request. Our method does not directly optimize final-video evaluation metrics: their criteria are neither optimization objectives nor predefined categories for constraint induction.

Candidate-Conditioned Activation. Let $\mathcal{P}^{(k)}$ denote a candidate continuation script. For each dependency, we determine whether the candidate introduces an event, relation, or state transition for which that dependency becomes relevant. An activated dependency is represented as a generation constraint $c_j = (s_j, a_j, b_j, e_j)$, where s_j specifies its entity and temporal scope, a_j is an applicability condition, b_j is the relation that must hold, and e_j contains the grounded evidence supporting the dependency. The constraint is activated when $a_j(\mathcal{P}^{(k)}, \mathcal{M}) = 1$, and once activated, requires $b_j(\mathcal{P}^{(k)}, \mathcal{M})$ to hold within scope s_j.

This formulation makes the constraints *candidate-conditioned*. A possession dependency, for example, need not constrain a continuation unless the proposed script introduces an action that requires a

particular character to possess the corresponding object. Similarly, a source-defined prerequisite is activated only when the candidate attempts an event whose validity depends on that prerequisite.

Constraint Evaluation. For an activated constraint c_j, we compare the relations implied by the candidate script with the grounded dependency model. Each constraint evaluation returns$\{\texttt{pass}, \texttt{fail}, \texttt{unknown}\}$. A `pass` indicates that the proposed continuation is compatible with the dependency; a `fail` identifies a conflicting event, state transition, or missing prerequisite; and an `unknown` indicates that the available evidence is insufficient to determine compatibility. Because every constraint retains links to its supporting predicates and evidence, failures can be traced back to the contextual dependency that caused them and used to produce targeted revision feedback.

3.3 Dependency-Guided Video Generation

The grounded dependency model is not used merely as additional conditioning for the video generator. Instead, it remains active throughout generation and provides a common basis for proposing, checking, and repairing the continuation.

Planning and Script Repair. The planner first proposes a candidate script conditioned on the grounded context, $\mathcal{P}^{(0)} \sim p_\theta(\mathcal{P} \mid \mathcal{M}, u)$. The dependencies activated by $\mathcal{P}^{(0)}$ are then instantiated as constraints and evaluated. When a constraint fails, the framework identifies both the conflicting portion of the script and the grounded dependency that it violates. This produces targeted feedback for revision rather than regenerating the continuation from scratch.

The revised script is therefore generated as $\mathcal{P}^{(k+1)} = \mathrm{Revise}_\theta\big(\mathcal{P}^{(k)}, \mathcal{M}, F^{(k)}\big)$, where $F^{(k)}$ contains the failed or unresolved dependencies and their supporting evidence. Because revision may introduce new events or state transitions, constraint activation is recomputed after each update. Planning and verification therefore form an iterative dependency-guided repair process in which the relevant obligations evolve together with the candidate continuation.

Rendering and Output Repair. Once a candidate script $\mathcal{P}^*$ satisfies the activated script-level constraints, it is passed to the video generator: $v_{\text{next}} \sim p_\psi(v \mid \mathcal{P}^*, v_{\text{prev}}, \mathcal{G})$. Script-level validity, however, does not guarantee that the rendered video realizes the intended events or state transitions. We therefore extract the realized relations from v_{next} and project the same grounded dependencies onto the generated output. If the video fails to realize an activated dependency—for example, an object transfer is omitted, a persistent attribute changes, or a prerequisite event is visually absent—the corresponding evidence is converted into output-level feedback. The planner then repairs the script or makes the missing dependency more explicit before rendering again. In this way, ***VideoTaleInteract*** uses a single grounded dependency model across both reasoning and generation. Context is first compiled into dependencies, dependencies are instantiated as constraints only when a proposed future makes them relevant, and violations are used to repair either the planned continuation or its rendered realization. The process terminates when the generated continuation satisfies the applicable grounded dependencies or the computation budget is exhausted.

4 Experiments

4.1 *IVS-Bench*: A New Benchmark for Interactive Video Storytelling

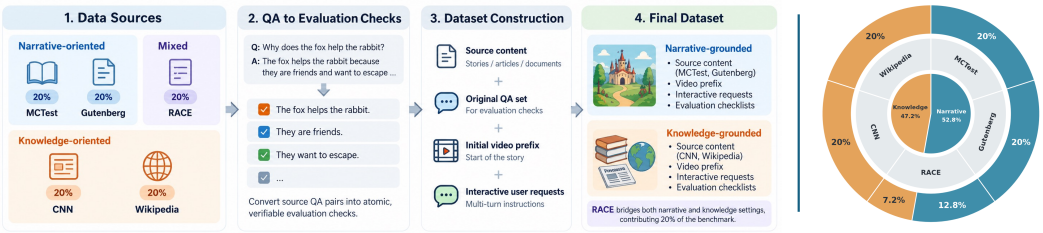

Figure 4: Dataset Construction and Dataset Statistics

To systematically study this problem and address the absence of dedicated benchmarks for the task, we introduce *IVS-Bench*. It considers two grounding settings:

- **Narrative grounding** uses a core narrative, such as a story outline, script, or plot description, that specifies important characters, events, relationships, and intended story progression.
- **Knowledge grounding** uses an external information source, such as a scientific paper, Wikipedia article, or document, whose factual content should be faithfully communicated through the story.

We construct the dataset from CoQA (Reddy et al., 2019). We sample 250 source passages from MCTest, Gutenberg, RACE, CNN, and Wikipedia, covering 132 narrative- grounded and 118 knowledge-grounded sources. Each source passage is converted into a scene-based story with aligned source facts and rendered in two visual styles: cinematic realism and cutout animation. We generate 1,004 interaction instructions at designated interruption points, yielding 2,008 planned instruction–style instances. All instances are assigned to evaluation; no training split is used. Each sample contains the source passage, video prefix, viewer instruction, and relevant reference assets, with target facts and continuation requirements reserved for evaluation. Generated videos undergo quality checks and repair, instruction alignment and information leakage are audited before packaging. See Appendix B for more details.

4.2 Evaluation Metrics

We evaluate each generated video along six dimensions. All metrics are normalized to a 0–100 scale, with higher values indicating better performance. Appendix D.2 gives the rating anchors, Appendix D.3 specifies normalization and judge aggregation

Semantic Metrics. Semantic evaluation is based on metric-specific checklists. Each applicable check is rated on a five-level scale corresponding to *absent or contradictory*, *mostly unsuccessful*, *partially fulfilled*, *mostly fulfilled*, and *fully fulfilled*.

- **Source Faithfulness (SF)** measures whether the generated video remains faithful to the grounding source $\mathcal{G}$. Its checklist is primarily derived from the original source QA annotations and captures source-specific facts, events, relationships, causal dependencies, and other persistent constraints.
- **Story Continuity (SC)** measures whether the rendered continuation is consistent with the video prefix. The evaluation considers established events, object states, character identities and relationships, spatial configurations, and causal history.
- **Interaction Fulfillment (IF)** measures how completely the continuation satisfies the latest viewer instruction. Each instruction is decomposed into atomic requirements, including, where applicable, requested subjects, actions, concepts, explanations, and presentation requirements.
- **Joint Constraint Satisfaction (JCS)** captures the model's overall ability to satisfy the three semantic requirements jointly. It is computed as the equal-weight average of SF, SC, and IF.

Perceptual Metrics. In addition to semantic correctness, we evaluate whether the generated video remains visually coherent and maintains high perceptual quality.

- **Visual Consistency (VC)** measures whether recurring characters, objects, environments, and visual styles remain coherent across the video. The evaluator considers both consistency between the existing video prefix and the generated video and consistency within the generated video itself.
- **Video Quality (VQ)** measures the perceptual quality of the generated video, using sampled-frame image integrity and composition; these scores do not assess native motion or audio quality. VQ evaluates generation quality independently of source or instruction compliance.

4.2.1 Evaluation Protocol

Auto-Evaluator. We use `Gemini 3.8 Flash` (Google, 2026b) and `Claude Opus 5` (Anthropic, 2026) as two independent auto-evaluators. We conduct blinded human evaluation to validate the evaluation rubrics and measure agreement between human judgments and the automatic evaluators. For each output, each judge independently produces per-check ratings together with supporting evidence, we then compute each metric independently for the two judges and average their scores. For the overall benchmark composite, we give the two grounding settings equal weight. Appendix D expands the evaluation rubric and score definitions; Appendix F.3 describes the human-validation protocol and agreement results.

Baselines. Interactive streaming models such as StreamDiT and LongLive (Kodaira et al., 2026; Yang et al., 2025) emphasize low-latency, prompt-driven generation. Our task requires source-

	Narrative Grounding						Knowledge Grounding						Overall
	SF	SC	IF	JCS	VC	VQ	SF	SC	IF	JCS	VC	VQ	Comp. [CI]
Backbone: **Omni-1.1-Flash**													
Direct Generation	83.9	66.1	80.4	76.8	45.1	56.2	59.4	68.8	62.5	63.5	50.2	52.6	70.2 [62.50, 77.83]
MM-StoryAgent	90.6	59.4	81.2	77.1	44.0	59.4	37.5	34.4	40.6	37.5	53.4	64.9	57.3 [41.15, 73.44]
MovieAgent	85.4	67.7	85.4	79.5	46.6	60.9	65.6	46.9	65.6	59.4	47.1	60.4	69.4 [57.29, 81.60]
VideoGen-of-Thought	89.6	56.3	86.5	77.4	42.7	58.1	69.5	54.2	**75.3**	66.3	52.4	63.4	71.9 [64.50, 78.70]
VideoTaleInteract	**92.7**	**81.3**	**90.6**	**88.2**	**53.0**	**62.4**	**87.5**	**78.1**	75.0	**80.2**	**65.4**	**68.2**	**84.2 [77.78, 90.45]**
Backbone: **Veo 3.1**													
Direct Generation	76.8	62.5	80.4	73.2	39.2	49.7	62.5	70.8	62.5	65.3	49.7	58.5	69.3 [57.34, 79.37]
MM-StoryAgent	70.5	41.1	70.5	60.7	32.2	56.1	50.0	66.7	60.4	59.0	**57.4**	62.4	59.9 [45.73, 72.07]
MovieAgent	89.3	57.2	**86.6**	77.7	38.5	59.7	64.6	56.3	58.3	59.7	47.5	**65.7**	68.7 [56.85, 76.39]
VideoGen-of-Thought	78.6	58.0	81.3	72.6	43.7	60.1	84.4	25.0	72.9	60.8	44.3	62.5	66.7 [58.55, 74.08]
VideoTaleInteract	**92.9**	**83.0**	84.8	**86.9**	**55.9**	**62.4**	**93.8**	**75.0**	**78.1**	**82.3**	54.8	64.2	**84.6 [80.95, 88.39]**
Backbone: **MiniMax-H3**													
Direct Generation	67.0	42.0	57.1	55.4	34.3	45.2	25.0	6.3	31.3	20.8	20.5	45.8	38.1 [27.23, 49.11]
MM-StoryAgent	78.6	57.1	**80.4**	72.0	39.5	44.0	81.3	31.3	65.6	59.4	43.8	45.0	65.7 [55.06, 76.34]
MovieAgent	78.6	65.2	75.0	72.9	39.0	46.4	**93.8**	58.3	**81.3**	77.8	**54.2**	57.3	75.4 [68.21, 81.70]
VideoGen-of-Thought	67.7	56.3	74.0	66.0	37.9	42.6	79.2	39.6	72.9	63.9	45.5	50.2	64.9 [57.64, 71.35]
VideoTaleInteract	**84.4**	**70.8**	77.1	**77.4**	**46.1**	**47.8**	87.5	**71.9**	**81.3**	**80.2**	46.7	**58.7**	**78.8 [75.52, 81.42]**

Table 1: **Main results.** We compare each method under narrative and knowledge grounding across three video-generation backbones. Metrics include Source Faithfulness (SF), Story Continuity (SC), Interaction Fulfillment (IF), Joint Constraint Satisfaction (JCS), Visual Consistency (VC), and Video Quality (VQ). **Comp.** denotes the overall composite score, followed by its confidence interval in brackets. Darker cells indicate stronger performance within each backbone and metric. **Bold** denotes the best result, while underlining denotes the second-best result. Higher is better for all metrics.

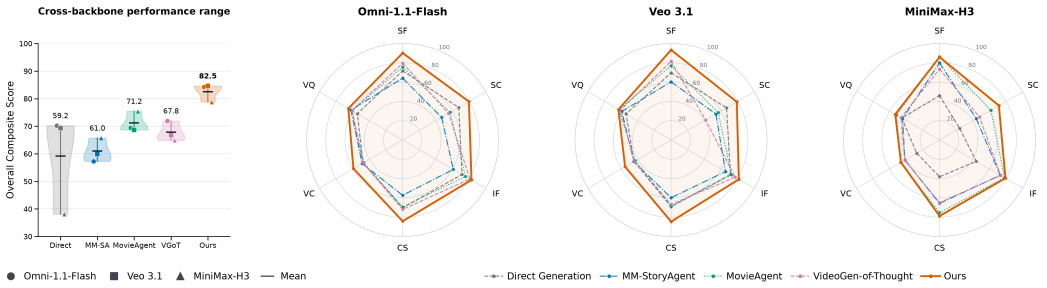

Figure 5: **Cross-backbone comparison. Left:** overall composite-score variation. **Right:** performance across the six evaluation dimensions, averaged over settings for each backbone.

grounded, 48-second multi-shot continuation from a supplied prefix. We select four complementary baselines: *Direct Generation* tests the backbone's native capabilities; *MM-StoryAgent* (Xu et al., 2025) uses multi-agent storytelling; *MovieAgent* (Wu et al., 2025b) uses hierarchical long-form planning; and *VideoGen-of-Thought* (Zheng et al., 2024) uses explicit intermediate reasoning. Although not originally designed for interactive continuation, these baselines test whether direct generation, agent coordination, and structured planning support our task under shared inputs and backbones, isolating contextual reasoning from synthesis capability. Appendix E details the adaptations.

Video Backbones. To separate the effectiveness of the interaction strategy from the capability of a particular video generator, we evaluate each method with three video-generation backbones: `Omni-1.1-Flash` (Google, 2026c), `Veo 3.1` (Google DeepMind, 2025b), and a locally hosted `MiniMax-H3` (MiniMax, 2026).

4.3 Evaluation Results

Table 1 summarizes the main evaluation results across three video-generation backbones. Overall, ***VideoTaleInteract*** improves contextual grounding while maintaining strong visual quality, achieving the highest composite score with every backbone.

Contextual grounding improves interactive continuation quality. The clearest gains appear on metrics that require the model to reconcile the grounding source, previously rendered video, and current viewer request. Across the grounding-related dimensions (SF, SC, IF, and CS), ***VideoTaleInteract*** improves over the strongest competing result by roughly 10% on average across backbones and settings.

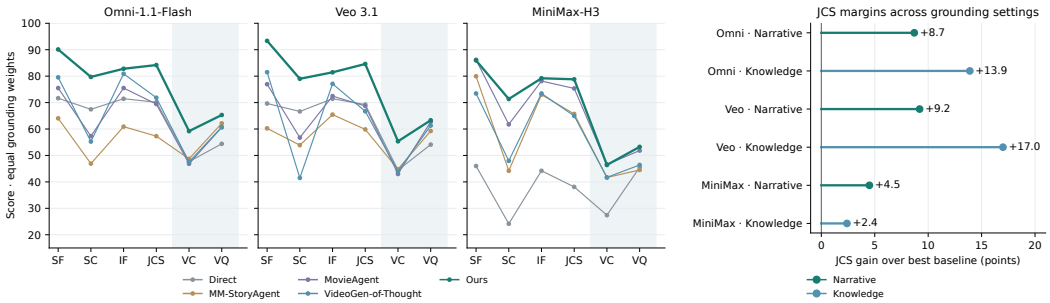

Figure 6: ***VideoTaleInteract* Performance gain over the baselines.** Left: performance across six metrics for three video generators under equal grounding weights. Right: our method's JCS gain over the strongest baseline in narrative and knowledge settings (percentage points).

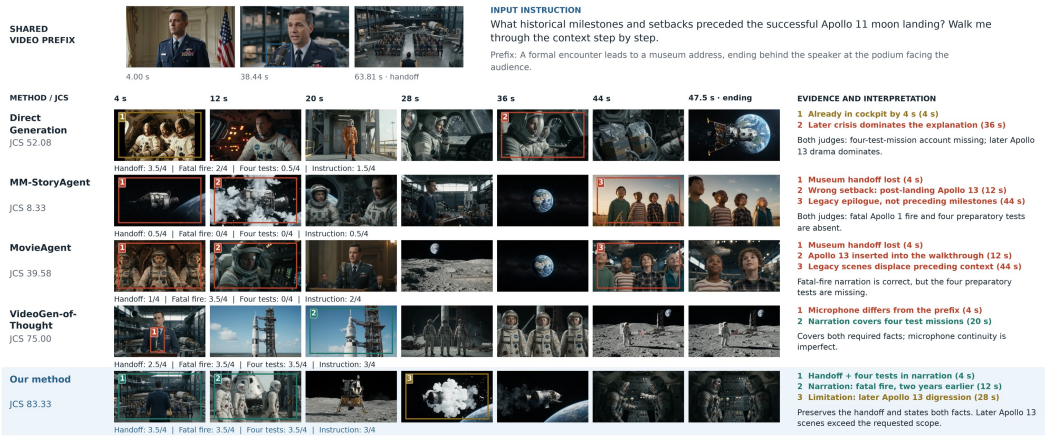

Figure 7: **Qualitative Example.** Colored boxes highlight successful evidence preservation and representative failures.

The mechanism generalizes across grounding sources. We observe consistent improvements in both narrative- and knowledge-grounded settings, even though the two settings emphasize different forms of information. Narrative grounding requires maintaining characters, events, and story progression, while knowledge grounding places greater emphasis on accurately incorporating external information. In both cases, our method retains a similar advantage over competing approaches.

The improvements are robust across video backbones. Figure 5 further shows that baseline performance can change substantially with the underlying video generator. The composite score of Direct Generation spans more than 30 points across the three backbones, whereas the range of ***VideoTaleInteract*** is below 6 points. This corresponds to an approximately 82% reduction in cross-backbone performance variation relative to Direct Generation.

5 Discussion

5.1 Case Study

Figure 7 illustrates how different methods balance the three sources of information in interactive storytelling. The example requires a continuation of an Apollo-related sequence while explaining the historical milestones and setbacks preceding the Apollo 11 landing. Several baselines capture individual facts but lose information already established by the prefix or introduce events in an inappropriate order. For example, MM-StoryAgent (Xu et al., 2025) quickly loses the museum handoff and omits much of the requested historical context, while MovieAgent (Wu et al., 2025b) introduces Apollo 13 before completing the requested preceding milestones. VideoGen-of-Thought provides

substantially better factual coverage, but still exhibits visual discontinuity from the prefix. In contrast, our method preserves the handoff, introduces the Apollo 1 fire and preceding test missions in the requested narrative, and maintains a more coherent progression across the continuation.

5.2 Ablation Study

We examine two design choices in ***VideoTaleInteract***. *Unstructured State* replaces the structured representation $\mathcal{M} = (\mathcal{V}, \mathcal{H}, \mathcal{S}, \mathcal{A})$ with a free-form summary while retaining the remaining pipeline. *One-Pass Planning* plans directly from the structured representation without instance-specific constraint induction. In the narrative-grounded setting, One-Pass Planning and Unstructured State reduce the mean of SF, SC, and IF by 4.36 and 7.60 percentage points, respectively, relative to the full method. The observed reductions in narrative visual consistency for Unstructured State and knowledge-grounded interaction fulfillment for One-Pass Planning are consistent with complementary roles for context representation and dependency-guided planning. These comparisons assess the implemented configurations; they do not independently isolate representation, repair, and reasoning-budget effects. Appendix E.1 explains their interpretation.

5.3 Limitation and Failure Analysis

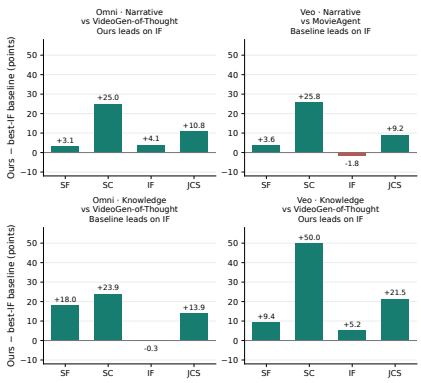

Figure 8: The comparator is the IF-leading baseline, held fixed across all four metrics in each panel.

VideoTaleInteract is not best on every individual metric, including some IF, VC, and VQ comparisons. For IF, our method jointly considers the viewer request with the grounding source and previously rendered video. When a request is ambiguous, aggressive, or partially conflicts with established context, the framework may favor a continuation that remains compatible with prior evidence instead of following the request as literally as possible. In contrast, methods that condition more directly on the latest instruction may obtain higher IF in such cases, but can sacrifice source faithfulness or story continuity. Similarly, visual quality remains largely bounded by the underlying video generator because our method does not modify the synthesis model itself. These metric-level gaps therefore reflect the multi-objective nature of interactive video storytelling: maximizing a single criterion independently can be easier than satisfying the source, prior video, and viewer request jointly. Despite these trade-offs, ***VideoTaleInteract*** achieves the strongest and most balanced overall composite performance across backbones.

Figure 9 analyzes a sample of 72 completed outputs selected for failure analysis; this is an analysis subset, not the total benchmark evaluation set. Within this sample, 67 outputs score at least 75 on Source Faithfulness, while 64 reach that threshold on each of Story Continuity and Interaction Fulfillment. Perfect scores are less common, however, especially for Story Continuity (8 outputs) and Interaction Fulfillment (11 outputs). In this analysis sample, continuations often preserve the main context while missing details of the established sequence or request.

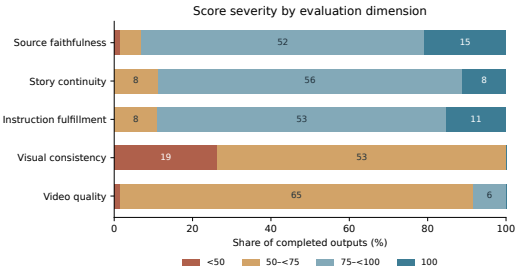

Figure 9: Score distribution in the 72-output analysis sample.

6 Conclusion

We introduced interactive video storytelling with contextual grounding, along with ***VideoTaleInteract*** and *IVS-Bench*. By abstracting the source, video prefix, and viewer request into aligned states and inducing constraints for generation, ***VideoTaleInteract*** improves performance across three video backbones.

AI USE STATEMENT

We used generative AI tools to assist with drafting and polishing the manuscript, literature retrieval and discovery, and the generation of synthetic benchmark materials. AI models also support the generation and automatic evaluation pipelines described in the paper and appendix. The authors take responsibility for the final manuscript, experimental claims, and released materials.

REFERENCES

Anthropic. Introducing Claude Opus 5, 2026. URL https://www.anthropic.com/news/claude-opus-5. Official model documentation. Accessed September 26, 2026.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. *arXiv preprint arXiv:2402.15391*, 2024. URL https://arxiv.org/abs/2402.15391.

Emanuele Bugliarello, Hernan Moraldo, Ruben Villegas, Mohammad Babaeizadeh, Mohammad Taghi Saffar, Han Zhang, Dumitru Erhan, Vittorio Ferrari, Pieter-Jan Kindermans, and Paul Voigtlaender. StoryBench: A multifaceted benchmark for continuous story visualization. *arXiv preprint arXiv:2308.11606*, 2023. URL https://arxiv.org/abs/2308.11606.

Decart and Etched. Oasis: A universe in a transformer. Project website, October 2024. URL https://oasis-model.github.io/.

Tianrui Feng, Zhi Li, Shuo Yang, Haocheng Xi, Muyang Li, Xiuyu Li, Lvmin Zhang, Keting Yang, Kelly Peng, Song Han, Maneesh Agrawala, Kurt Keutzer, Akio Kodaira, and Chenfeng Xu. StreamDiffusionV2: A streaming system for dynamic and interactive video generation. In *Proceedings of Machine Learning and Systems*, 2026. URL https://arxiv.org/abs/2511.07399.

Google. Gemini 2.5 Flash Image (Nano Banana), 2025. URL https://ai.google.dev/gemini-api/docs/models/gemini-2.5-flash-image. Official model documentation. Accessed September 26, 2026.

Google. Gemini 3.1 Pro Preview, 2026a. URL https://ai.google.dev/gemini-api/docs/models/gemini-3.1-pro-preview. Official model documentation. Accessed September 26, 2026.

Google. Gemini 3.8 Flash, 2026b. URL https://ai.google.dev/gemini-api/docs/models/gemini-3.8-flash. Official model documentation. Accessed September 26, 2026.

Google. Build with Gemini Omni 1.1 Flash, 2026c. URL https://blog.google/innovation-and-ai/technology/developers-tools/build-with-gemini-omni-1-1-flash/. Official model documentation. Accessed September 26, 2026.

Google DeepMind. Genie 3: A new frontier for world models. Google DeepMind blog, 2025a. URL https://deepmind.google/discover/blog/genie-3-a-new-frontier-for-world-models/.

Google DeepMind. Veo 3.1, 2025b. URL https://deepmind.google/models/veo/. Official model documentation. Accessed September 26, 2026.

Xianglong He, Chunli Peng, Zexiang Liu, Boyang Wang, Yifan Zhang, Qi Cui, Fei Kang, Biao Jiang, Mengyin An, Yangyang Ren, Baixin Xu, Hao-Xiang Guo, Kaixiong Gong, Size Wu, Wei Li, Xuchen Song, Yang Liu, Yangguang Li, and Yahui Zhou. Matrix-Game 2.0: An open-source real-time and streaming interactive world model. *arXiv preprint arXiv:2508.13009*, 2025. URL https://arxiv.org/abs/2508.13009.

Lingxuan Huang, Sizhe He, Hengji Zhou, Liqiang Nie, Lianghao Xia, and Chao Huang. ViMax: Agentic video generation. *arXiv preprint arXiv:2606.07649*, 2026. doi: 10.48550/arXiv.2606.07649. URL https://arxiv.org/abs/2606.07649.

Siqiao Huang, Jialong Wu, Qixing Zhou, Shangchen Miao, and Mingsheng Long. Vid2World: Crafting video diffusion models to interactive world models. *arXiv preprint arXiv:2505.14357*, 2025. URL https://arxiv.org/abs/2505.14357.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Huang_VBench_Comprehensive_Benchmark_Suite_for_Video_Generative_Models_CVPR_2024_paper.html.

Akio Kodaira, Tingbo Hou, Ji Hou, Markos Georgopoulos, Felix Juefei-Xu, Masayoshi Tomizuka, and Yue Zhao. StreamDiT: Real-time streaming text-to-video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://arxiv.org/abs/2507.03745.

MiniMax. MiniMax-H3: Model Card, 2026. URL https://huggingface.co/MiniMaxAI/MiniMax-H3. Official model documentation. Accessed September 26, 2026.

Sheshera Mysore, Debarati Das, Hancheng Cao, and Bahareh Sarrafzadeh. Prototypical human-AI collaboration behaviors from LLM-assisted writing in the wild. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 16819–16846, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.852.

Siva Reddy, Danqi Chen, and Christopher D. Manning. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266, 2019. doi: 10.1162/tacl_a_00266. URL https://aclanthology.org/Q19-1016/.

Yale Song, Yiwen Song, Nick Losier, Nathan Hodson, Ye Jin, Rhyard Zhu, Yan Xu, Daniel Vlasic, Carina Claassen, Jasmine Leon, Khanh G. LeViet, Zack Chomyn, Joe Timmons, Brett Slatkin, Scott Penberthy, and Tomas Pfister. Co-Director: Agentic generative video storytelling. *arXiv preprint arXiv:2604.24842*, 2026. doi: 10.48550/arXiv.2604.24842. URL https://arxiv.org/abs/2604.24842.

Weijia Wu, Mingyu Liu, Zeyu Zhu, Xi Xia, Haoen Feng, Wen Wang, Kevin Qinghong Lin, Chunhua Shen, and Mike Zheng Shou. MovieBench: A hierarchical movie level dataset for long video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28984–28994, 2025a. URL https://openaccess.thecvf.com/content/CVPR2025/html/Wu_MovieBench_A_Hierarchical_Movie_Level_Dataset_for_Long_Video_Generation_CVPR_2025_paper.html.

Weijia Wu, Zeyu Zhu, and Mike Zheng Shou. Automated movie generation via multi-agent CoT planning. *arXiv preprint arXiv:2503.07314*, 2025b. doi: 10.48550/arXiv.2503.07314. URL https://arxiv.org/abs/2503.07314.

Xuenan Xu, Jiahao Mei, Chenliang Li, Yuning Wu, Ming Yan, Shaopeng Lai, Ji Zhang, and Mengyue Wu. MM-StoryAgent: Immersive narrated storybook video generation with a multi-agent paradigm across text, image and audio. *arXiv preprint arXiv:2503.05242*, 2025. doi: 10.48550/arXiv.2503.05242. URL https://arxiv.org/abs/2503.05242.

Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, Song Han, and Yukang Chen. LongLive: Real-time interactive long video generation. *arXiv preprint arXiv:2509.22622*, 2025. URL https://arxiv.org/abs/2509.22622.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T. Freeman, Frédo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2412.07772.

Kaining Ying, Hengrui Hu, Siyu Ren, Jiamu Li, Fengjiao Chen, Ziwen Wang, Xuezhi Cao, Xunliang Cai, and Henghui Ding. WBench: A comprehensive multi-turn benchmark for interactive video world model evaluation. *arXiv preprint arXiv:2605.25874*, 2026. URL https://arxiv.org/abs/2605.25874.

Ann Yuan, Andy Coenen, Emily Reif, and Daphne Ippolito. Wordcraft: Story writing with large language models. In *Proceedings of the 27th International Conference on Intelligent User Interfaces*, IUI '22, pp. 841–852, Helsinki, Finland, 2022. Association for Computing Machinery. doi: 10.1145/3490099.3511105.

Mingzhe Zheng, Yongqi Xu, Haojian Huang, Xuran Ma, Yexin Liu, Wenjie Shu, Yatian Pang, Feilong Tang, Qifeng Chen, Harry Yang, and Ser-Nam Lim. VideoGen-of-Thought: Step-by-step generating multi-shot video with minimal manual intervention. *arXiv preprint arXiv:2412.02259*, 2024. doi: 10.48550/arXiv.2412.02259. URL https://arxiv.org/abs/2412.02259.

Cailin Zhuang, Ailin Huang, Wei Cheng, Jingwei Wu, Yaoqi Hu, Jiaqi Liao, Zhewei Huang, Hongyuan Wang, Xinyao Liao, Weiwei Cai, Hengyuan Xu, Xuanyang Zhang, Xianfang Zeng, Gang Yu, and Chi Zhang. ViStoryBench: Comprehensive benchmark suite for story visualization. *arXiv preprint arXiv:2505.24862*, 2025. URL https://arxiv.org/abs/2505.24862v1.

A Related Work

Video Storytelling. Video storytelling systems coordinate narrative planning and multimodal generation to produce coherent visual stories. MM-StoryAgent (Xu et al., 2025) orchestrates text, images, and audio for narrated storybook videos, while MovieAgent (Wu et al., 2025b) generates multi-scene movies from scripts and character references. VideoGen-of-Thought (Zheng et al., 2024), Co-Director (Song et al., 2026), and ViMax (Huang et al., 2026) further explore structured planning, agent coordination, and refinement for coherent video production. These approaches primarily organize generation around an initial creative brief, script, or narrative source. Our setting instead considers requests made after part of the video has been generated: each continuation must accommodate the new request while remaining faithful to the grounding source and consistent with the rendered video prefix.

Interactive Video Generation and World Models. Streaming video models enable viewers to influence generation as it unfolds. CausVid (Yin et al., 2025), StreamDiT (Kodaira et al., 2026), and LongLive (Yang et al., 2025) support evolving text prompts, while StreamDiffusionV2 (Feng et al., 2026) supports dynamic generation with streaming inputs. Interactive world models extend this responsiveness to action-conditioned environments: Genie (Bruce et al., 2024), Oasis (Decart & Etched, 2024), Matrix-Game 2.0 (He et al., 2025), and Vid2World (Huang et al., 2025) generate worlds that evolve with viewer actions, and Genie 3 (Google DeepMind, 2025a) also supports promptable world events. Together, these systems advance responsive generation and continuity across interactions. Interactive video storytelling additionally requires interpreting each new request in light of both an external grounding source and the story already rendered, then using their joint context to plan the continuation.

Video Generation and Storytelling Benchmarks. Existing benchmarks evaluate complementary aspects of visual generation. VBench (Huang et al., 2024) measures video generation quality; StoryBench (Bugliarello et al., 2023) evaluates action execution, story continuation, and story generation; and MovieBench (Wu et al., 2025a) provides hierarchical movie data for studying long-form generation. ViStoryBench (Zhuang et al., 2025) evaluates narrative alignment and consistency across image sequences, while GenAD-Bench (Song et al., 2026) evaluates generative advertising workflows. More closely related, WBench (Ying et al., 2026) evaluates interactive world models across setting adherence, interaction adherence, consistency, physics compliance, and video quality, covering navigation, subject actions, event editing, and perspective switching. Our benchmark focuses on the joint satisfaction of source faithfulness, story continuity, and interaction fulfillment in video continuations grounded in narrative or knowledge sources.

B Benchmark Construction and Data Interface

This appendix describes benchmark construction, grounded continuation, evaluation, and study materials for interactive video storytelling. The continuation setting uses two 24-second scenes, each comprising three eight-second shots. Illustrative examples are distinguished from empirical results.

B.1 Video examples

An anonymous repository for supplementary video examples is provided at this link.

B.2 Source collection and instance counts

IVS-Bench is constructed from 250 CoQA (Reddy et al., 2019) source passages drawn from MCTest, Gutenberg, RACE, CNN, and Wikipedia. The collection contains 132 narrative-grounded passages and 118 knowledge-grounded passages. Narrative grounding preserves the characters, events, and relations specified by a core story; knowledge grounding preserves factual claims and their qualifications in an external information source. Each passage is converted into a scene-based story with aligned source facts and rendered in cinematic realism and cutout animation.

The construction comprises 1,004 interaction instructions at designated interruption points. Pairing each instruction with both visual styles gives 2,008 *planned* instruction–style instances. These

	Interaction Type	Source Faithfulness	Story Continuity	Interaction Fulfillment
Video storytelling methods				
MM-StoryAgent (Xu et al., 2025)	—	—	✓	—
MovieAgent (Wu et al., 2025b)	—	✓	✓	—
VideoGen-of-Thought (Zheng et al., 2024)	—	—	✓	—
Co-Director (Song et al., 2026)	—	△[a]	✓	—
ViMax (Huang et al., 2026)	—	✓[b]	✓	—
CausVid (Yin et al., 2025)	Free-Form	—	△[d]	✓
StreamDiT (Kodaira et al., 2026)	Free-Form	—	△[d]	✓
LongLive (Yang et al., 2025)	Free-Form	—	△[d]	✓
StreamDiffusionV2 (Feng et al., 2026)	Free-Form	—	△[d]	△[f]
Genie (Bruce et al., 2024)	Limited Action Space	—	—	△[c]
Oasis (Decart & Etched, 2024)	Limited Action Space	—	—	△[c]
Matrix-Game 2.0 (He et al., 2025)	Limited Action Space	—	—	△[c]
Vid2World (Huang et al., 2025)	Limited Action Space	—	—	△[c]
Video generation and visual storytelling benchmarks				
MovieBench (Wu et al., 2025a)	—	△[a]	△[d]	—
ViStoryBench (Zhuang et al., 2025)	—	△[a]	△[d,g]	—
GenAD-Bench (Song et al., 2026)	—	△[a]	—	—
VBench (Huang et al., 2024)	—	—	—	—
WBench (Ying et al., 2026)	Free-Form	△[e]	—	✓
StoryBench (Bugliarello et al., 2023)	Free-Form	—	△[d]	△[c]
Interactive Video Storytelling with Contexutal Grounding	Free-Form	✓	✓	✓

Table 2: **Comparison with related methods and benchmarks.** ✓: explicitly addressed; △: partial coverage; —: not explicitly established. [a]only asset or script alignment; [b]only screenplay adaptation; [c]action-conditioned control; [d]only temporal coherence; [e]only world-setting adherence; [f]live video interaction; [g]image-sequence evaluation.

counts describe distinct units: a passage can support several instructions, and an instruction can appear in two styles.

Because each configuration requires rendering a 48-second multi-shot continuation, we evaluate a fixed subset of *IVS-Bench* rather than the full 2,008 instances. For each of the three backbones, we sample 8 narrative-grounded and 8 knowledge-grounded instances from distinct source passages, and run all five methods on the same 16 instances, yielding 240 evaluated continuations in total (5 methods $\times$ 3 backbones $\times$ 16 instances).

B.3 Evaluation setting and inputs

All benchmark instances belong to the evaluation split, no training split is used.

An interactive continuation instance consists of a grounding source $\mathcal{G}$, an observed video prefix v_{prefix}, and the current viewer instruction u, together with relevant reference assets. The source remains persistent context, while the prefix records the visual and narrative state already shown to the viewer. The continuation must respond to u without contradicting either source-supported content or established history. The task formulation does not separately condition on the complete history of earlier viewer instructions; it relies on the rendered prefix to carry the relevant observable history.

Target facts and continuation requirements are reserved for evaluation. This distinction is essential: the generator may derive its own context-dependent constraints from its permitted inputs, but access to the evaluator's answer annotations or reference checklist would change the task.

C Grounded Dependencies and Continuation Repair

C.1 Representation and evidence provenance

The purpose of context compilation is to expose dependencies that cannot be recovered reliably from the latest instruction alone. We retain the main paper's notation $\mathcal{M} = (\mathcal{V}, \mathcal{H}, \mathcal{S}, \mathcal{A})$ for aligned contextual states: $\mathcal{V}$ describes the observed video, $\mathcal{H}$ the evolving story, $\mathcal{S}$ source-supported content,

and $\mathcal{A}$ cross-context alignments. We denote dependencies derived from these states by $\mathcal{R}$. This separates the evidence representation from relations inferred over it.

A grounded predicate identifies an entity or event, its attributes or relations, its temporal scope, and supporting evidence. Entity alignment associates references to the same character, object, event, or claim across modalities. Temporal scope is necessary because an earlier observation may no longer describe the current state: possession can change after a transfer, and an action can alter an object's location or condition. Source predicates also retain qualifications and conditions so that a restricted claim is not promoted to an unconditional fact.

Evidence provenance provides the connection between a verification decision and the context that justifies it. Source evidence can identify a supporting passage; video evidence can identify the observation supporting a state or transition. An inferred dependency should remain distinguishable from a directly observed fact. The implementation schemas, prompts, and agent/planner code map this interface to source facts, a prefix ledger, source alignment, instruction requirements, a scene script, and an expanded shot plan. Source alignment separates shown, partially shown, unseen, and uncertain facts. The shot plan records prompts, reference identifiers, audio, camera, and transitions; review traces record defects, revised plans, and selected attempts. The conceptual state model is thus represented through several structured intermediate artifacts.

C.2 Candidate-conditioned activation

For a candidate script $\mathcal{P}^{(k)}$, a dependency is projected into a constraint $c_j = (s_j, a_j, b_j, e_j)$ with scope s_j, applicability condition a_j, required relation b_j, and supporting evidence e_j. The active constraint set is

$$\mathcal{C}^{(k)} = \left\{ c_j : \; a_j(\mathcal{P}^{(k)}, \mathcal{M}) = 1 \right\}. \tag{1}$$

This notation does not prescribe a new activation classifier; it makes explicit the candidate-conditioned operation described in the method. A dependency that is irrelevant to the proposed events need not constrain the continuation. When a repair introduces a new event, activation must be recomputed because the new event can introduce additional prerequisites.

Each active constraint receives one of three outcomes: `pass`, `fail`, or `unknown`. A failure indicates an identifiable conflict or missing prerequisite; an unknown outcome indicates insufficient evidence to decide compatibility. These outcomes must remain distinct. In particular, a lack of evidence should not be reported as verified satisfaction. The main method includes failed and unresolved dependencies in the feedback used to revise a script.

C.3 Illustrative dependency trace

Consider a hypothetical story in which a cabinet can only be opened with a key. The rendered prefix establishes that character A has transferred the key to character B, and the viewer asks to see A open the cabinet. This example illustrates the mechanism; it is not presented as a sampled benchmark instance or an empirical result.

The source supplies the opening prerequisite, while the prefix supplies the latest possession state. A candidate in which A immediately uses the key activates a dependency that conflicts with the established transfer. A possible repair is to show B returning the key before A opens the cabinet, provided this event is compatible with the source and story. Verification must then check the revised event sequence, rather than merely check whether the word "key" appears in the script.

Rendering introduces a separate point of failure. A repaired script may include the return of the key even when the synthesized video omits that action. Output verification therefore checks the realized relations against the same grounded dependencies. If the transfer is not visually established, the output does not demonstrate the planned prerequisite, and the feedback can identify the missing transition for a subsequent repair.

C.4 Repair loop and termination

The generation procedure can be summarized as follows:

1. Extract and align video, story, and source predicates; derive contextual dependencies.
2. Propose a candidate continuation script conditioned on the grounded context and current request.
3. Activate candidate-relevant constraints and evaluate their compatibility with the script.
4. Use failed or unresolved dependencies and their evidence to produce targeted feedback; revise the script and recompute activation.
5. Render a script that satisfies the applicable script-level constraints, and inspect the realized video relations.
6. Repair missing or conflicting realizations and render again, stopping when verification succeeds or the computation budget is exhausted.

This procedure is a control flow, not a proof of convergence or correctness. Extraction can miss evidence, inferred dependencies can be wrong, and a renderer can repeatedly fail to realize a valid script. Budget exhaustion must therefore be distinguished from verified success. Similarly, a request incompatible with persistent source constraints need not admit a continuation that fully satisfies all objectives.

C.5 TWO-SCENE IMPLEMENTATION

State extraction and structural repair. The base agent extracts ordered source facts with supporting quotations, inspects the complete prefix into a conservative ledger, partitions source facts into shown/partially shown/unseen/uncertain categories, and decomposes the request into source-linked requirements. Uncertain observations are kept separately, and unsupported requests are explicitly recorded. The script validator is called after the first proposal and each repair, with at most five repairs. The agent returns the validation result alongside its output, so reaching the repair limit does not itself establish validity.

Plan review. The six-shot refinement function performs at most three reviews and two repairs. It supplies the actual prefix video, public source, request, style, assets, reference limit, public states, and candidate plan to the reviewer. Acceptance requires an affirmative acceptance flag and empty source/instruction, continuity, and renderability error lists. Each repair receives those defects, after which asset validation is rerun. The function records the review history and final pass flag; after its last review it returns the plan even if that flag is false. Consequently, the orchestration must distinguish a budget-exhausted plan from an accepted plan.

Boundary and audio contracts. The first shot uses prefix continuation. Later shots use exact continuation or a motivated hard cut; the first shot of scene 2 follows the explicit inter-scene boundary. Shot normalization restricts fact and requirement references to the corresponding scene and restricts asset references to valid scene assets. A speaking scene assigns speech to exactly one of its three shots; a silent scene uses ambience only. The optional speech audit checks six ordered shots, requires verbatim source quotations for supported factual speech, and allows at most three audit calls before raising a failure.

Table 3: Shared model configuration for continuation generation and evaluation.

Role	Model identifier
Planner	`gemini-3.1-pro-preview` (Google, 2026a)
Reference-image generator	`gemini-2.5-flash-image` (Google, 2025)
Gemini evaluator	`gemini-3.8-flash` (Google, 2026b)
Claude evaluator	`claude-opus-5` (Anthropic, 2026)
Omni renderer	`gemini-omni-1.1-flash-preview` (Google, 2026c)
Veo renderer	`veo-3.1-generate-001` (Google DeepMind, 2025b)
MiniMax renderer	`MiniMax-H3-Ref2VA-Turbo-4step-v0.1-bf16` (MiniMax, 2026)

The shared configuration caps agent calls at 48, image candidates at three, input tokens at 600,000, and output/reasoning tokens at 96,000. These are upper limits rather than measured mean usage.

D Evaluation Checklists and Score Interpretation

D.1 Internal quality control and benchmark separation

Generation-time review and attempt selection use internal quality checks, not the benchmark's evaluation criteria or metric scores. Their role is to identify defects in a proposed plan or rendered attempt using the public source, observed prefix, viewer request, assets, and intermediate states. They do not use the benchmark's private answer annotations, reference checklists, or final evaluator ratings to choose or repair an output.

Internal acceptance flags and quality scores therefore belong to the generation procedure. Final benchmark evaluation is a separate assessment of the selected output under the fixed benchmark rubric described below. Both procedures can inspect related properties, such as continuity or whether a requested action is visible, because those properties are part of the task. This shared task relevance does not give the generator access to benchmark labels or make final benchmark scores its optimization objective. In particular, generation-time quality review must not be interpreted as selecting outputs with the benchmark evaluator.

D.2 Metric-specific evidence

The semantic metrics assess different sources of obligation. Source Faithfulness (SF) uses source-specific checks primarily derived from the original QA annotations, including facts, events, relationships, and causal dependencies. Story Continuity (SC) concerns established events, identities, object states, spatial arrangements, and causal history in the prefix. Interaction Fulfillment (IF) decomposes the latest request into atomic requirements such as subjects, actions, concepts, explanations, and presentation requirements. Keeping these checklists separate makes it possible to identify a continuation that satisfies the request while losing source or prefix information.

Ordered level	Interpretation
0	Absent or contradictory
1	Mostly unsuccessful
2	Partially fulfilled
3	Mostly fulfilled
4	Fully fulfilled

Table 4: Semantic rating anchors. Unknown evidence is recorded separately.

The same rubric is used for 24-second and 48-second continuations, with evidence timestamps interpreted over the corresponding full continuation. A common transcript supplies speech evidence; sampled frames supply visual evidence. Each rating should be accompanied by evidence for the corresponding check. Evidence of an intended action in a script and evidence of a realized action in a video answer different questions. The documented evaluation protocol requires evidence from the rendered output: a required visible action cannot be satisfied solely by narration or by inclusion in a plan. Fixed check identifiers, applicability, metric membership, and evidence modality are determined independently of method outputs.

Visual Consistency (VC) assesses recurring characters, objects, environments, and style, both across the prefix–continuation boundary and within the continuation. Video Quality (VQ) assesses visible image integrity and composition in the sampled frames. These frame-based scores do not establish fine-motion or audio quality. These perceptual criteria should not substitute for semantic checks: a visually polished continuation can still contradict its source or omit the requested explanation.

D.3 Normalization and judge aggregation

The documented scoring protocol codes each applicable semantic check as $r \in \{0, 1, 2, 3, 4\}$ and normalizes complete ratings as

$$M_i^{(q)} = \frac{25}{K_{i,M}} \sum_{k=1}^{K_{i,M}} r_{i,k}^{(q)}, \qquad M_i = \frac{M_i^{(1)} + M_i^{(2)}}{2}. \tag{2}$$

Here $K_{i,M} > 0$ is the number of fixed applicable checks for metric M on instance i. Applicability is not changed after observing a method output. The two judges independently supply ratings and evidence, and their metric scores are averaged.

Unknown evidence is represented as null rather than a zero or midpoint. Incomplete applicable ratings produce a missing complete score; an observed-only score and coverage can be reported separately. If the known-rating sum is R and u of the K applicable ratings remain unresolved, the documented lower and upper bounds are $25R/K$ and $25(R + 4u)/K$. These bounds describe the contribution of unresolved checks.

D.4 Joint score and overall composite

Joint Constraint Satisfaction is defined in the manuscript as

$$\mathrm{JCS} = \frac{\mathrm{SF} + \mathrm{SC} + \mathrm{IF}}{3}. \tag{3}$$

Despite its name, this arithmetic mean is not the fraction of outputs that satisfy every constraint. A high score in one semantic dimension can compensate numerically for a low score in another. The individual semantic metrics must therefore remain visible alongside JCS.

The documented overall composite is the equal-weight mean of setting-level mean JCS:

$$C_{\mathrm{overall}} = \frac{1}{2}\left(\frac{1}{n_N}\sum_{i\in N}\mathrm{JCS}_i + \frac{1}{n_K}\sum_{i\in K}\mathrm{JCS}_i\right). \tag{4}$$

VC and VQ are reported separately and do not contribute to this composite. Repeated styles or interventions derived from a single source are not independent source observations.

Although the planner shares a model family with one evaluator, the ranking of methods is unchanged when scored by Claude Opus 5 alone, the internal reviewer never sees the evaluation rubric or benchmark labels, and JCS is reported alongside a conjunctive variant (the fraction of instances with SF, SC, and IF all ≥ 75) that yields the same ordering.

E Baseline Adaptations and Implementation Reporting

The four baselines cover direct generation, coordinated story production, hierarchical planning, and intermediate reasoning. Their original formulations do not by themselves specify how to resume a partially rendered story after a viewer intervention. An interactive adaptation must therefore state where the source, observed prefix, and new request enter each pipeline and how the continuation is attached to the prefix.

Direct Generation. Mechanical shot prompts insert the public context, requested action, and temporal interval without an intermediate LLM story-planning call. The method still uses rendering and generation-time QA.

MM-StoryAgent. (Xu et al., 2025) Up to three writer–expert exchanges precede narrative outlining and chapter expansion. Image, sound, speech, and music roles feed a shared shot plan and keyframe design. The expert observes the original prefix video. Audio-role text is realized through the common renderer's native audio, rather than separate upstream synthesis pipelines.

MovieAgent. (Wu et al., 2025b) Source-to-script preparation observes the original prefix and is followed by screenwriting, scene planning, shot planning, and keyframe design. Public references replace a separate character bank; upstream lip synchronization is not reproduced. No additional storyline-repair pass follows these stages.

VideoGen-of-Thought. (Zheng et al., 2024) An initial outline observes the prefix, followed by character specifications and elaboration into Character, Background, Relation, Camera Pose, and HDR Description. Up to two critique/refinement rounds precede keyframe design. Native latent propagation, IP-adapter behavior, and smoothing are not reproduced through the common endpoint.

E.1 Comparison design and interpretation

Why these baseline families. Direct generation, multi-agent story production, hierarchical planning, and intermediate reasoning provide complementary ways to turn the same source, prefix, and request into a continuation. Adapting them tests whether these established planning strategies support contextual grounding under a shared video-generation interface. Interactive streaming methods such as StreamDiT and LongLive (Kodaira et al., 2026; Yang et al., 2025) instead emphasize low-latency generation with evolving prompts. Their ability to generate long streams is not disputed; the comparison here concerns source-grounded, multi-shot continuation from a supplied prefix, rather than streaming throughput or maximum video duration.

What the adaptations control. The common rendering interface separates the planning strategy from the chosen synthesis backbone. The adaptations preserve the corresponding story-planning or reasoning organization while translating its output into the shared shot interface. Renderer-specific features that cannot be expressed through that interface, including latent propagation, separate lip synchronization, and smoothing, are explicitly identified above. The results consequently compare these adapted pipelines on the present task; they are not evaluations of the original systems with every native component intact.

What the ablations establish. Unstructured State probes the role of explicit contextual structure by replacing it with a free-form summary. One-Pass Planning probes the combined role of candidate-specific constraints and the associated planning procedure. These comparisons help interpret the implemented pipeline, but they do not separately estimate the causal effect of every representation field, repair stage, or reasoning call. In particular, the one-pass comparison should not be read as a compute-matched test of constraint induction alone.

Computation and claim scope. Shared inputs and video backbones control access to task information and the synthesis endpoint. They do not imply identical realized token usage, agent calls, retries, or latency across planning strategies. The runtime limits reported in Appendix C.5 specify configuration caps rather than measured equal expenditure. The reported comparisons therefore support conclusions about the evaluated end-to-end configurations, without establishing an efficiency advantage or attributing the entire gain to contextual representation independently of computation.

F Formative Study and Human Evaluation

F.1 Formative study

The formative study enrolled nine volunteers. Two participants did not pass the attention test and were excluded from the learning-gain analysis, leaving seven participants in the analyzed sample. Among these seven participants, four were master's students in computer science, biology, and physics, and three were AI scientists working in industry. Learning gains and experience ratings measure distinct outcomes: the former assess changes in content understanding, while the latter describe the reported interaction experience.

We define learning gain as post-test percentage accuracy minus pre-test percentage accuracy, averaging sessions within each condition before calculating a participant's contrast:

$$L_{pc} = \overline{\mathrm{Post}}_{pc} - \overline{\mathrm{Pre}}_{pc}, \qquad D_p = L_{p,\mathrm{interactive}} - L_{p,\mathrm{noninteractive}}. \tag{5}$$

Under this definition, gains are measured in percentage points rather than relative percent improvement. Within-participant contrasts compare the gain in the interactive condition with the gain in the noninteractive condition. The reported mean learning gains of 47.9 percentage points with interaction and 32.1 without interaction, a difference of 15.8 points, summarize the seven participants who passed the attention test. Figure 2 in the main text reports this seven-participant learning-gain sample; its experience-rating panel contains four respondents. These summaries should not be interpreted as all-nine estimates.

F.2 Formative-study interface

Figures 10–12 document the interface used in the formative learning study: the pre-test, noninteractive viewing, and interactive question-entry screens. These screenshots illustrate the participant-facing workflow; the attention-test exclusion criterion is described above.

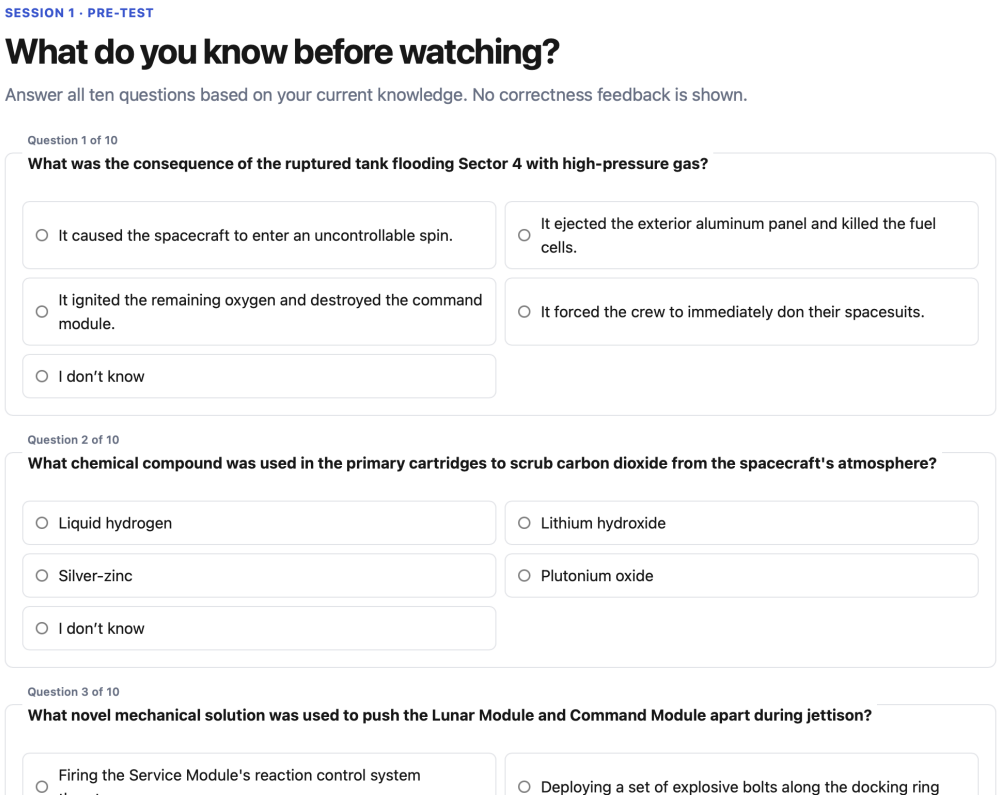

Figure 10: **Pre-test interface.** The pre-test screen asks participants to answer ten questions using their current knowledge, without correctness feedback. Each visible question includes an "I don't know" option.



Figure 11: **Noninteractive viewing interface.** The noninteractive screen presents the video, subtitles, scene title, and viewing progress. The screenshot shows an Apollo 13 example.

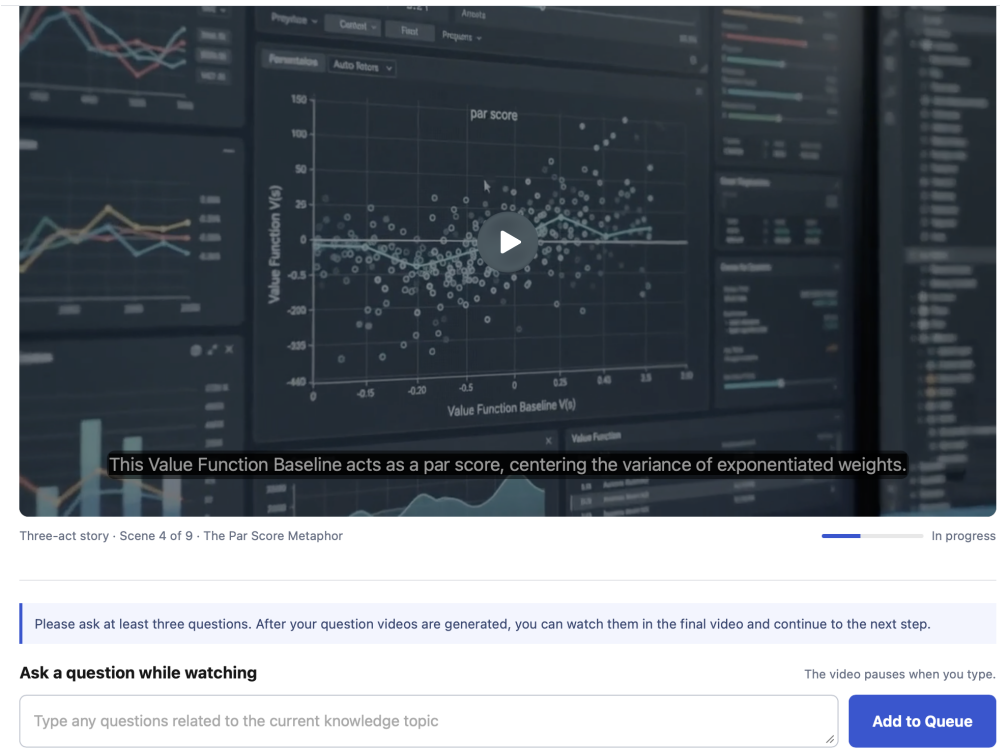

Figure 12: **Interactive viewing interface.** The interactive screen provides a question field and an "Add to Queue" control below the video. The displayed instructions ask participants to submit at least three questions and state that playback pauses while they type.

F.3 HUMAN VALIDATION OF AUTOMATIC EVALUATION

We additionally conduct a human validation study to examine whether the automatic evaluators produce judgments consistent with human assessment. This analysis is separate from the formative learning study and focuses only on agreement in output-level quality judgments. We sample 50 generated outputs spanning both grounding settings, multiple generation methods, and a range of automatic evaluation scores. Each output is independently rated by three human annotators using the same metric definitions and ordinal rating scale as the automatic evaluation. Agreement is measured using Krippendorff's α, treating the human-majority rating as the human reference when comparing against automatic evaluators.

After the evaluation prompt calibration, human annotators exhibit strong inter-rater agreement across all five metrics ($\alpha = 0.78$–0.86). Both automatic evaluators also align closely with human judgments, with Gemini reaching $\alpha = 0.75$–0.83 and Claude reaching $\alpha = 0.64$–0.76. The results suggest that the automatic evaluation provides a reliable approximation of human judgment while enabling evaluation at substantially larger scale.

Field	**Study details**
Agreement statistic and scale	Krippendorff's α computed over the same ordinal rating scale used by the automatic evaluators.
Output sampling and coverage	50 generated outputs sampled across narrative and knowledge grounding, multiple generation methods and backbones, and different automatic-score ranges to obtain broad coverage of output quality.
Rater recruitment and training	Three human annotators familiar with multimodal content evaluation. Annotators first reviewed the metric definitions, rubric, and several worked examples before independently completing the validation set.
Ratings per output	Each output received three independent human ratings for SF, SC, IF, VC, and VQ.
Blinding and presentation order	Annotators were not shown the generation method, model identity, or automatic evaluator scores. Outputs were presented in independently randomized order.
Disagreement handling	Ratings were collected independently without discussion. For human–automatic agreement, the median human rating across the three annotators was used as the reference judgment.
Rubric calibration / validation separation	A small calibration set was used only to familiarize annotators with the rubric and was excluded from the reported validation results. The final validation set was rated independently after calibration.

Table 5: Protocol used for human validation of the automatic evaluation.

F.4 PROMPT AND IMPLEMENTATION AVAILABILITY

The prompt manifest distinguishes shared extraction templates, two-scene planning and refinement, the separate one-scene implementation, optional audio grounding, and final evaluation. Appendix G reproduces eight extraction and two-scene planning/review templates.

G PROMPT TEMPLATES

The following templates specify state extraction and two-scene planning and repair. Braced fields denote runtime substitutions, and output-schema names refer to the structured implementation interfaces. The first four templates extract and align the grounding context; the final four specify the 48-second narrative, shot expansion, review, and repair. Prompt instructions express intended behavior rather than guarantees of generated-output correctness.

G.1 ATOMIC SOURCE FACTS

```
Extract atomic ordered facts from the original source for source-grounded video continuation.
↪  Preserve exact supporting quotes. For narrative text, include events, actions, speech
↪  acts, states, relationships, locations, objects, and causal dependencies. For knowledge
↪  text, include claims, mechanisms, causes, effects, examples, comparisons, and explanatory
↪  dependencies. Assign F001, F002, ... in source order. Do not add external knowledge.
↪  Return only SourceFacts JSON.

GROUNDING: {grounding_type}
SOURCE:
{source_text}
```

G.2 OBSERVED PREFIX LEDGER

```
Inspect the attached complete video prefix, especially its final seconds. Build a conservative
↪  state ledger for planning the immediate continuation. Record visible identities,
↪  appearance, location, object and character states, spatial relationships, recent and
↪  unresolved events, style, composition, and screen direction. Distinguish persistent from
↪  momentary state. Do not infer unseen outcomes. Put uncertain claims only in
↪  excluded_uncertain_claims. Return only PrefixStateLedger JSON.

STORY SUMMARY: {story_summary}
AVAILABLE ASSET IDENTITIES: {assets}
```

G.3 SOURCE–PREFIX ALIGNMENT

```
Align atomic source facts to the prefix ledger. Partition every fact ID exactly once into
↪  shown, partially shown, unseen, or uncertain. "Shown" means semantically realized in the
↪  prefix, not merely mentioned in metadata. "Partially shown" is reserved for an event
↪  actively underway at the cut. Explain the continuation boundary conservatively. Return
↪  only SourceAlignment JSON.

INTERACTION SCENE LABEL: {interaction_after_scene}
FACTS: {source_facts}
PREFIX LEDGER: {prefix_ledger}
```

G.4 INSTRUCTION REQUIREMENTS

```
Decompose the viewer instruction into atomic assessable requirements and map each semantic
↪  requirement only to relevant unseen or partially shown source facts. Presentation
↪  preferences may have no fact ID. Report requests contradicted or unsupported by the source
↪  in unsupported_requests. Do not map to already shown facts. Return only
↪  InstructionRequirements JSON.

GROUNDING: {grounding_type}
INSTRUCTION: {instruction}
FACTS: {source_facts}
ALIGNMENT: {source_alignment}
```

G.5 TWO-SCENE PLANNING

```
You are a source-grounded narrative director. Expand the frozen interactive-video continuation
↪  into exactly two coherent scenes, each 24 seconds long. Each scene will later be divided
↪  into three 8-second shots.

GROUNDING: {grounding_type}
PREFIX SUMMARY: {story_summary}
ORIGINAL SOURCE:
{source_text}

VIEWER INSTRUCTION: {instruction}

FROZEN AGENT SCENE:
{scene_script}

AVAILABLE ASSETS:
{assets}

STRICT RULES
1. Return exactly two ordered scenes with indexes 1 and 2 and total_duration_seconds=48.
2. Both scenes together must fulfill the viewer instruction. Scene 1 begins fulfillment
↪  immediately; scene 2 advances or resolves it without replay.
3. Use only facts supported by the original source and only source_fact_ids, requirement_ids,
↪  and selected_asset_ids available in the frozen agent scene and asset list.
```

```
4. Scene 1 starts exactly from the attached prefix terminal state. Scene 2 must follow
↪  causally from scene 1, with an explicit boundary classified as exact_continuation,
↪  identity_composition_anchor, or intentional_jump.
5. Preserve identities, clothing, object state, relationships, environment logic, screen
↪  direction, and all persistent state unless the visible action explicitly changes them.
6. Each scene has a clear narrative purpose, a compact set of visible actions, and a precise
↪  initial_state/final_state contract. Scene 2 final_state must equal the frozen agent scene
↪  final_state.
7. Speech must be source-supported, concise enough for a single 8-second shot, and must not
↪  invent quotations or claims. Use narrator when sourced exposition is needed. A silent
↪  scene has no narration or dialogue.
8. Use exactly one exclusive audio mode per scene. Character dialogue contains exactly one
↪  line and must use a selected character asset as speaker.
9. Do not add unsupported characters, props, locations, facts, dialogue, outcomes, captions,
↪  or readable text.
10. Return only structured JSON conforming exactly to TwoSceneNarrativePlan.
```

G.6 SIX-SHOT EXPANSION

```
You are a master cinematographer and continuity director. Turn the frozen two-scene narrative
↪  below into exactly six generation-ready shots: three ordered 8-second shots per scene.

PREFIX SUMMARY: {story_summary}
VIEWER INSTRUCTION: {instruction}
TWO-SCENE NARRATIVE:
{two_scene_plan}

AVAILABLE ASSETS:
{assets}

STRICT RULES
1. Return two SceneShotPlans. Scene 1 uses 0-8, 8-16, 16-24 seconds. Scene 2 uses 24-32,
↪  32-40, 40-48 seconds. Every shot duration is exactly 8 seconds.
2. Each scene has exactly three shots indexed 1, 2, 3. Each shot has one dominant narrative
↪  beat, one focal action, readable composition, precise camera movement, and a rich
↪  standalone video_prompt.
3. Scene 1 shot 1 uses prefix_continuation. Within a scene, use exact_continuation for
↪  continuous motion/composition or hard_cut for a motivated camera-angle change.
4. Scene 2 shot 1 follows the supplied SceneBoundary. Use exact_continuation for an exact
↪  boundary; otherwise use hard_cut while preserving the boundary's required persistent
↪  state.
5. Preserve identities, clothing, props, environment, lighting logic, screen direction,
↪  spatial relationships, scale, and causal state across shots and across the scene boundary.
6. Use only fact IDs, requirement IDs, and asset IDs assigned to the corresponding narrative
↪  scene. Do not add facts, dialogue, characters, props, locations, or outcomes.
7. Distribute each scene's visible actions across its three shots without replay. The last
↪  shot of each scene must reach that scene's frozen final_state.
8. For a narrator or dialogue scene, exactly one of its three shots uses scene_audio; the
↪  other two use ambient_only. For a silent scene, all three use ambient_only.
9. In ambient_only shots, nobody speaks, talks, addresses an audience, narrates, or lip-syncs;
↪  prompts specify closed, non-speaking mouths.
10. Each video_prompt describes only its own 8-second shot and prohibits subtitles, captions,
↪  title cards, logos, watermarks, and unsupported readable text.
11. Keep human faces natural and anatomically normal with restrained expressions. Avoid
↪  identity, costume, object, and scale drift, extra limbs, and implausible motion.
12. Return only structured JSON conforming exactly to TwoSceneShotPlan.
```

G.7 PLAN REVIEW

```
Audit the complete six-shot continuation using only the public source, instruction,
actual prefix and extracted public states. Never use external biographical or historical
↪  knowledge.
Check that required visual actions actually happen instead of being narrated or replaced by
↪  decorative motion.
Every shot must advance a distinct part of the requested continuation. Avoid spending several
↪  shots on a static presentation.
Check initial-to-final action states, motivated transitions, persistent
↪  identity/wardrobe/object geometry,
screen direction and literal material style. A paper-cutout scene must not become glossy
↪  realistic 3D.
Check exact speech fits its eight-second shot at a natural pace (usually at most 18{20 words),
and that any audio-only shot is intentional. Do not demand on-screen text unless the
↪  instruction requires it.
Distinguish scene changes from identity drift. Report concrete defects with affected shot
↪  numbers.
Accept only if no material defect is found. These are generation checks, not private
↪  evaluation labels.
```

G.8 PLAN REPAIR

```
Repair the six-shot Plan using the public evidence and listed defects. Preserve its 48-second,
two-scene structure and every source-grounded requirement.
Mandatory structural fields: shot_id exactly [1,2,3,4,5,6]; scene_id exactly [1,1,1,2,2,2];
duration exactly 8 for every shot. Shot 1 transition MUST be prefix_continuation and must
↪  visually
start from the actual prefix terminal state. No other shot may use prefix_continuation.
If the story needs a new setting, establish that change with a hard_cut in shot 2 or later;
never label shot 1 hard_cut. These fixed fields override any informal editing suggestion.
Use only listed asset IDs and at most the
supplied reference limit per shot. State literal visual materials and stable identifying
↪  attributes
in each standalone shot prompt. Give each shot one executable focal action with a visible
↪  start and end.
Make later shots continue the achieved state without replay. Use a motivated hard_cut when the
↪  setting
changes; do not ask for an impossible continuous transformation of location or identity. Keep
↪  speech
brief and source-supported; never invent facts to increase detail. Return the complete Plan.
```