

StoryBlender: Inter-Shot Consistent and Editable 3D Storyboard with Spatial-temporal Dynamics

Bingliang Li^{1*}, Zhenhong Sun^{2*}, Jiaming Bian³, Yuehao Wu¹, Yifu Wang⁴, Hongdong Li², Yatao Bian⁵, Huadong Mo^{1**}, and Daoyi Dong⁶

¹ University of New South Wales, Australia. lbingle@outlook.com,
{yuehao.wu,huadong.mo}@unsw.edu.au

² Australian National University, Australia.
{zhenhong.sun,hongdong.li}@anu.edu.au

³ Central South University, China. bianjiaming@csu.edu.cn

⁴ Vertex Lab, China. usasuper@126.com

⁵ National University of Singapore, Singapore. ybian@nus.edu.sg

⁶ University of Technology Sydney, Australia. daoyi.dong@uts.edu.au

Abstract. Storyboarding is a core skill in visual storytelling for film, animation, and games. However, automating this process requires a system to achieve two properties that current approaches rarely satisfy simultaneously: *inter-shot consistency* and *explicit editability*. While 2D diffusion-based generators produce vivid imagery, they often suffer from identity drift along with limited geometric control; conversely, traditional 3D animation workflows are consistent and editable but require expert-heavy, labor-intensive authoring. We present **StoryBlender**, a grounded 3D storyboard generation framework governed by a Story-centric Reflection Scheme. At its core, we propose *StoryBlender* system, which is built on a three-stage pipeline: (1) Semantic-Spatial Grounding, to construct a continuity memory graph to decouple global assets from shot-specific variables for long-horizon consistency; (2) Canonical Asset Materialization, to instantiate entities in a unified coordinate space to maintain visual identity; and (3) Spatial-Temporal Dynamics, to achieve layout design and cinematic evolution through visual metrics. By orchestrating multiple agents in a hierarchical manner within a verification loop, StoryBlender iteratively self-corrects spatial hallucinations via engine-verified feedback. The resulting native 3D scenes support direct, precise editing of cameras and visual assets while preserving unwavering multi-shot continuity. Experiments demonstrate that StoryBlender significantly improves consistency and editability over both diffusion-based and 3D-grounded baselines. Code and dataset are available at <https://engineeringai-lab.github.io/StoryBlender>.

Keywords: 3D Storyboard · Scene Generation · Multi-Agent Planning

* Equal Contribution

** Corresponding author

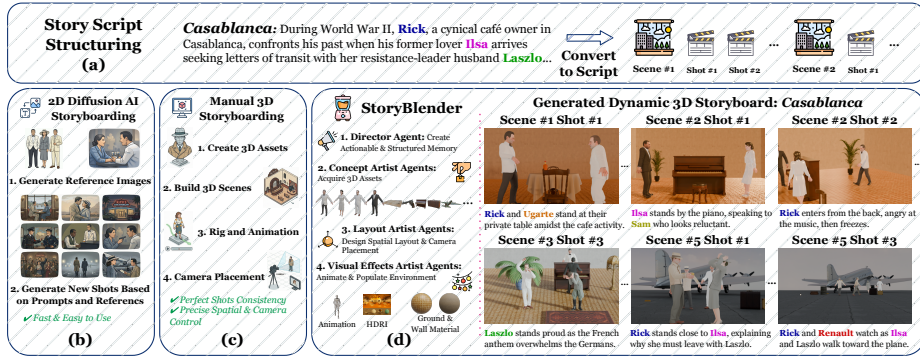


Fig. 1: Overview of StoryBlender compared to existing storyboarding methods. **Left:** diffusion-based generation in pixel space; improving consistency typically requires reference inputs. **Middle:** traditional 3D workflow; strong control but a complex, labor-intensive pipeline. **Right (ours):** StoryBlender uses a hierarchical multi-agent planning framework to create consistent, editable 3D storyboards across shots.

1 Introduction

Visual storytelling aims to translate a narrative (e.g., a script or outline) into a sequence of coherent visual shots. In practice, this is realized through storyboarding, an explicit, shot-by-shot plan that externalizes key creative decisions (characters, actions, layout, and camera intent) to coordinate teams and de-risk production during pre-production and pre-visualization in film, animation, and games. At its essence, storyboarding demands two key capabilities: **consistency** and **editability**. Consistency means maintaining character identity, visual style, and 3D scene state (layout, props, lighting, etc.) across shots and largely determines whether the storyboard is practically usable. Editability means these elements are explicitly controllable and revisable (e.g., adjusting blocking) without redoing the entire sequence, and determines the practical usability of the system.

Existing solutions for automated storyboarding can be broadly grouped into two paradigms that trade off these two requirements: (i) 2D diffusion-based generation can render vivid, high-quality shots [4, 5, 8, 24, 30, 43, 44, 46], but achieving shot-to-shot consistency is difficult and often depends on reference inputs, while the pixel-space representation provides limited editability for geometric/cinematic changes. (ii) Traditional 3D workflows can provide explicit scene representations that naturally support consistency and editability, but they typically require a complex pipeline and substantial manual labor from skilled professionals, making the workload prohibitive at scale. This motivates using Large Language Model (LLM) agents to translate scripts into structured 3D actions (asset selection, placement, camera design) and automate expert-heavy steps while retaining controllability [7, 9, 10, 13, 16, 26]. However, LLMs alone cannot solve the entire pipeline end-to-end. They fundamentally lack a persistent, grounded world model to bridge the gap between semantic narra-

tive planning and precise geometric execution. Specifically, LLMs often exhibit spatial hallucinations, proposing layouts or actions that conflict with physical constraints, and they lack an internal verification loop to correct these geometric errors. Moreover, maintaining narrative coherence requires tracking an evolving world state beyond the limited context window of a standard LLM. This raises a question: *Do general-purpose LLMs truly understand 3D geometry and spatial information to ensure inter-shot consistency and editability, or do they merely hallucinate semantic plausibility without using spatial grounding?*

To answer this, we introduce StoryBlender, a novel framework that reformulates storyboard generation not as stochastic image synthesis but as a Hierarchical Multi-Agent Planning process governed by a Story-centric Reflection Scheme. Unlike prior art, StoryBlender treats the narrative as a persistent, grounded simulation where agents collaborate within a 3D engine to ensure the output is both inter-shot consistent and editable. Our methodology is organized into three progressive stages that systematically operationalize the spatial-temporal dynamics of the story: First, to establish a world model that transcends the limited context windows of LLMs, we propose Semantic-Spatial Grounding. Here, a Director Agent decomposes the narrative into a structured continuity memory graph, serving as a digital script supervisor that ensures precise information flow and maintains consistency across long horizons. Second, to eliminate generative hallucinations, Canonical Asset Materialization instantiates entities from the memory graph into a unified coordinate space, ensuring that character identities and props remain visually identical throughout the sequence. Finally, Spatial-Temporal Dynamics orchestrates the physical arrangement and cinematic evolution of these assets. By employing discrete visual servoing and programmatic engine verification, this stage resolves the complex dynamics of characters, cameras, and visual effects. By integrating this persistent memory with a closed-loop reflection system, StoryBlender replaces mere semantic plausibility with rigorous physical grounding. This architecture also ensures that every creative decision remains explicitly editable within the 3D environment, providing a robust solution for consistent 3D storyboarding.

The main contributions are summarized as follows:

- We reformulate 3D storyboarding as a closed-loop optimization process by pairing LLM reasoning with in-engine physical verification to iteratively self-correct spatial hallucinations.
- We introduce a structured hierarchical memory to decouple global assets from shot-specific variables, enabling seamless asset materialization and ensuring long-horizon identity consistency across the entire narrative.
- We propose a modular pipeline that resolves complex layout and cinematic evolution through visual metrics, achieving a storyboard representation that is both physically grounded and explicitly editable.

2 Related Work

Generative Visual Storytelling. Recent progress has pushed visual storytelling from single images to coherent narrative sequences, driven by the emergence of powerful video foundation models such as Stable Video Diffusion [1] and Sora [17]. To adapt these models for narrative control, prior work improves character consistency with visual adapters (e.g., IP-Adapter [41], T2I-Adapter [19], StoryAdapter [18]) to strengthen cross-frame alignment through mechanisms such as consistent attention and latent anchoring (e.g., StoryDiffusion [46], StoryGen [15], Story2Board [4]). For multi-shot narratives, agent-style pipelines increasingly use LLMs to plan semantic layouts and orchestrate the generation process step-by-step (e.g., VideoDirectorGPT [14], MovieAgent [34], DreamFactory [35], AniMaker [24], CineVision [32]). Recent diffusion-control studies further indicate that reliable complex visual generation often requires explicit instance-, sketch-, and motion-level constraints [27, 28, 40, 42]. However, because these 2D-centric methods operate entirely in pixel or latent space, they fundamentally lack an underlying physical model. This inherently leaves them prone to geometric hallucinations, viewpoint inconsistencies, and weak environmental persistence during editing, limiting their reliability for rigorous pre-visualization workflows.

LLM-Driven 3D Scene Synthesis. To gain spatial consistency, recent systems couple LLMs with 3D representations to generate scenes through scripting or layout planning (e.g., SceneCraft [7], I-Design [47], SceneWeaver [37], WorldCraft [16], PAT3D [13], Physcene [38]). In these pipelines, the LLM typically acts as a high-level planner, decomposing text into object inventories and spatial relations. Recent advancements have further integrated LLMs directly into the generation pipeline using diverse strategies. For instance, SceneTeller [21] predicts 3D layouts via in-context learning. Other approaches leverage Chain-of-Thought for direct numerical spatial reasoning [22], or connect multi-agent frameworks to professional game engines for procedural generation (UnrealLLM [25]). Yet most treat generation as a one-off, *static* arrangement problem, optimizing a single snapshot without mechanisms to maintain state across shots or to reuse scene constraints (e.g., persistent props, character interactions, and camera settings). Reliable character interaction and motion grounding also motivate open-world HOI detection and real-world 3D human pose benchmarking [31, 36].

3D Cinematography and Pre-visualization. Engine-based pre-visualization methods, such as Virtual Dynamic Storyboard (VDS) [23], often rely on pre-selected assets and manual setup. Recently, AI-driven approaches have sought to automate and improve this pipeline. Generative methods are increasingly integrating with 3D blocking to provide robust cinematic control, such as CinePreGen [2], which utilizes engine-powered diffusion for camera-controllable pre-visualization, and PrevizWhiz [6], which combines rough 3D scenes with 2D references. Other works explore the transformation of 2D sketch storyboards into 3D animations (Sketch2Anim [45]). Story3D-Agent [10] scripts 3D actions with LLMs as linear sequences in a pre-configured environment. Despite these

advances, existing systems lack a global context manager for multi-scene transitions. StoryBlender addresses these gaps via a continuity memory graph that maintains persistent states and enforces inter-shot consistency.

3 Methodology

3.1 Overview

We formulate the dynamic 3D storyboard generation task as a mapping process $F : \mathcal{T}_{story} \rightarrow \mathcal{V}_{3D}$, where a raw textual narrative script $\mathcal{T}_{story}$ is synthesized into a spatiotemporally coherent and editable 3D dynamic video sequence $\mathcal{V}_{3D}$. To bridge the gap between abstract semantics and precise spatial execution, we propose the Hierarchical Multi-Agent Planning Framework, illustrated in Fig. 2. This framework mimics professional film production workflows by decoupling the production pipeline into specialized roles, ensuring that each stage, from initial planning to final rendering, is handled with domain-specific expertise. Central to this framework is the Story-centric Reflection Scheme, which shifts the generative process from a conventional feed-forward pass to an iterative optimization problem. Rather than producing a single-shot output, our agents operate within a closed-loop verification system. By executing actions within a 3D engine and verifying the results via multi-modal feedback, the system systematically mitigates spatial randomness and ensures that the final output $\mathcal{V}_{3D}$ remains faithful to the original narrative intent $\mathcal{T}_{story}$.

For any subagent $a \in \mathcal{A}_{agents}$ operating on a specific task context $\mathcal{C} \subset \mathcal{G}_{cm}$, the initial state is generated as $o_0 = \Phi_a(\mathcal{C})$, where Φ_a denotes the generation policy of agent a . To prevent spatial hallucinations and ensure physical plausibility, o_t is evaluated iteratively at each step t by a reflection scoring function $R(o_t, \mathcal{C}) \in \mathbb{R}$, which yields diagnostic feedback $\mathcal{F}_t$:

$$o_{t+1} = \begin{cases} o_t, & \text{if } R(o_t, \mathcal{C}) \geq \tau_a \\ \Psi_a(o_t, \mathcal{F}_t, \mathcal{C}), & \text{if } R(o_t, \mathcal{C}) < \tau_a \text{ and } t < T_{\max} \\ \Omega(a_{fb}, \mathcal{C}), & \text{otherwise.} \end{cases} \quad (1)$$

Here, τ_a represents the task-specific acceptance threshold, and $T_{\max}$ is the maximum allowable reflection horizon to prevent non-convergent loops. If the criteria are not met within $T_{\max}$, the system triggers Ω , a handover operator that delegates the context to a designated fallback agent a_{fb} (e.g., reverting from retrieval to generative synthesis). Depending on the subagent’s domain, the feedback $\mathcal{F}_t$ and the scoring function R are derived from either engine constraints (e.g., deterministic 3D engine detailing bounding box collisions) or aesthetic critiques (e.g., VLM-driven semantic alignment scores scaled to inform τ_a). Driven by $\mathcal{F}_t$, the refinement operator Ψ_a iteratively corrects the spatial or visual effects until the geometric and narrative requirements are satisfied.

Guided by this unified scheme, our framework is organized into three progressive modules, which are detailed in the subsequent subsections: 1) Semantic-Spatial Grounding decomposes the story into a structured continuity memory

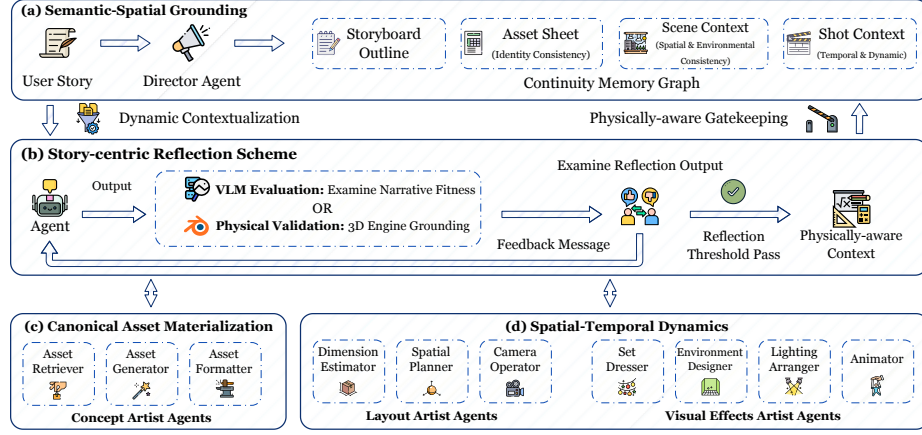


Fig. 2: Hierarchical Multi-Agent Planning Framework. Governed by a **Story-centric Reflection Scheme** (b), our system utilizes iterative feedback from 3D engines, such as Blender, and Vision-Language Models to ensure geometric and narrative consistency. We translate narrative $\mathcal{T}_{story}$ into 3D storyboards $\mathcal{V}_{3D}$ via a three-stage pipeline: (a) **Semantic-Spatial Grounding**, where the Director Agent decomposes the story into a structured continuity memory graph ($\mathcal{G}_{cm}$) to ensure precise information flow to downstream agents; (c) **Canonical Asset Materialization**, which instantiates entities from $\mathcal{G}_{cm}$ to maintain global asset consistency; and (d) **Spatial-Temporal Dynamics**, which performs spatial layout of assets from memory and enhances cinematic visual effects. (Details of all agents are provided in **Appendix**.)

graph, enabling precise and reliable information flow to downstream agents; 2) Canonical Asset Materialization instantiates entities from the memory graph, ensuring global asset consistency; 3) Spatial-Temporal Dynamics arranges assets in space and time based on memory and enhances cinematic visual effects.

3.2 Semantic-Spatial Grounding

The Director Agent (a_{dir}) serves as the high-level orchestrator, responsible for mapping unstructured text into a structured, persistent representation: $a_{dir} : \mathcal{T}_{story} \rightarrow \mathcal{G}_{cm}$. To systematically prevent semantic degradation over long-horizon generation, a_{dir} conducts this mapping through three core mechanisms.

Continuity Memory Graph. Instead of relying on the transient context windows of LLMs, the director agent organizes the narrative into the continuity memory graph ($\mathcal{G}_{cm}$). This graph acts as a hierarchical state comprising four distinct layers: (1) **Storyboard Outline** ($\mathcal{M}_{outline}$) for the temporal sequencing of scenes and shots; (2) **Asset Sheet** ($\mathcal{M}_{asset}$) for registering globally unique entity identifiers; (3) **Scene Context** ($\mathcal{M}_{scene}$) for maintaining static environmental states (e.g., layouts $\mathbf{L}_{layout}$ and sets $\mathbf{E}_{env}$); and (4) **Shot Context** ($\mathcal{M}_{shot}$) for localizing dynamic camera parameters and actions. This structure establishes a definitive source of truth by decoupling global assets from shot-specific variables.

Dynamic Contextualization. To actively manage information flow and prevent cognitive overload for downstream subagents, the Director employs a dynamic contextualization mechanism. We define a projection operator $\Pi_{task}(\mathcal{G}_{cm})$ that extracts only the task-relevant subgraphs for a specific agent. Rather than exposing the entire global state, Π_{task} supplies precise, limited constraints (e.g., providing only the active bounding boxes from $\mathcal{M}_{scene}$ to the Layout Artist Agents) while masking unrelated narrative data. By grounding subagents within a minimally sufficient working memory, this targeted routing drastically curtails semantic noise and mitigates spatial hallucinations.

Physically-Aware Gatekeeping. $\mathcal{G}_{cm}$ is not a static repository but a self-evolving world representation governed by constraint-based gatekeeping. To safely integrate new agent actions, the Director utilizes a strict validation function $V(o_{new}, \mathcal{S}_{exist}) \rightarrow \{0, 1\}$. Any proposed 3D output o_{new} must pass programmatic, engine-verified evaluations against the existing global state $\mathcal{S}_{exist}$ before it is committed to memory. For instance, a newly generated layout must be rigorously verified to avoid bounding box collisions. Invalid updates ($V = 0$) are rejected and routed back to the offending agent’s reflection loop (Eq. 1), ensuring long-horizon geometric consistency across the storyboard.

3.3 Canonical Asset Materialization

The goal of asset materialization is to instantiate abstract semantic entities from the memory graph into standardized, physical 3D meshes. Serving as the semantic-to-geometric bridge, the Concept Artist Agents ($a_{concept}$) populate the Asset Sheet ($\mathcal{M}_{asset}$) with canonical 3D geometries (Ω_{3D}):

$$a_{concept} : (\mathcal{M}_{outline}, \mathcal{M}_{asset}) \rightarrow \Omega_{3D}. \quad (2)$$

To avoid unpredictable artifacts from direct 3D generation and to guarantee unified spatial alignment, $a_{concept}$ executes this process via a cascaded workflow as follows.

Asset Retriever. To mitigate topological errors, the Retriever first queries high-quality existing assets using attributes from $\mathcal{M}_{asset}$. Following Eq. 1, retrieved candidates o_t are evaluated by a VLM reflection function $R(o_t, \mathcal{M}_{asset})$ for spatial quality and stylistic alignment with $\mathcal{M}_{outline}$. Candidates exceeding the threshold τ_{ret} are selected, ensuring optimal fidelity for standard objects without generative hallucinations.

Asset Generator. If retrieval fails to meet τ_{ret} within T_{max} attempts, the system triggers a *Handover* (Ω) to the Asset Generator. This fallback employs a two-stage synthesis: it first generates a VLM-refined 2D reference image conditioned on $\mathcal{M}_{asset}$, which is subsequently lifted into a 3D mesh. Anchoring 3D generation on a strictly verified 2D intermediate localizes geometric errors and enforces aesthetic constraints for novel assets.

Asset Formatter. Raw assets often reside in arbitrary local coordinate systems with unaligned forward vectors, which can cause downstream layout fail-

ures. To standardize them, the Formatter queries a VLM to identify the semantic canonical front and physical extents. It computes a transformation tuple $(\mathbf{R}_{align}, s_{norm})$, applying rotation $\mathbf{R}_{align} \in SO(3)$ for global axis alignment and uniform scaling s_{norm} for bounding box normalization. This ensures all entities in Ω_{3D} share a unified mathematical space, enabling precise layout operations without orientation ambiguity.

3.4 Spatial-Temporal Dynamics

The final stage of our framework transforms the canonical 3D assets into a cohesive cinematic scene. It resolves both the physical arrangement of objects (Spatial) and their narrative evolution through camera motion and character actions (Temporal), yielding the final 3D storyboard $\mathcal{V}_{3D}$.

Layout Artist Agents. Acting as the spatial architect, the Layout Artist Agents (a_{layout}) resolve scene geometry and cinematic configurations:

$$a_{layout} : (\Omega_{3D}, \mathcal{M}_{scene}, \mathcal{M}_{shot}) \rightarrow (\mathbf{L}_{layout}, \mathbf{C}_{cam}). \quad (3)$$

To achieve this, the agent orchestrates a three-step pipeline. First, to address the lack of scene-specific proportions in isolated assets, the **Dimension Estimator** predicts realistic scaling factors $\mathbf{s} \in \mathbb{R}^3$ based on semantic roles. This ensures physically plausible relative sizes across the environment. Second, to bridge abstract text and precise 3D coordinates, the **Spatial Planner** maps qualitative prepositions into quantitative transformations (translation $\mathbf{t} \in \mathbb{R}^3$, rotation $\mathbf{r} \in SO(3)$) to form the layout $\mathbf{L}_{layout}$. The 3D engine programmatically verifies this proposed layout, detecting physical violations such as collisions. By returning exact error logs to the reflection loop, the planner iteratively corrects the arrangement, ensuring geometric validity.

Finally, to translate the static layout into a cinematic experience, the **Camera Operator** determines the camera state $\mathbf{C}_{cam}$ via VLM-guided Discrete Semantic Visual Servoing. To ensure initial visibility, the camera pivot initializes at the target’s bounding box centroid with an orbital distance geometrically derived to guarantee a specific field of view (FOV). To dynamically refine this framing, the VLM evaluates rendered frames and issues discrete camera-control commands (e.g., *Orbit*, *Pan*) mapped to differential spatial updates. This closed-loop servoing optimizes the semantic alignment between the rendered view and the directorial intent. (Details are provided in **Appendix**.)

Visual Effects Artist Agents. To elevate the geometrically valid layout into a fully realized cinematic sequence, the Visual Effects (VFX) Artist Agents (a_{vfx}) finalize the atmospheric and temporal dimensions:

$$a_{vfx} : (\mathcal{M}_{scene}, \mathcal{M}_{shot}) \rightarrow (\mathbf{I}_{light}, \mathbf{E}_{env}, \mathbf{P}_{action}). \quad (4)$$

To achieve this, the agent executes a four-step enhancement pipeline. First, to spatially ground the isolated primary assets, the **Environment Designer** constructs structural backdrops ($\mathbf{E}_{env}$). This establishes clear spatial boundaries

and contextual realism for the scene. Second, to overcome the visual sparsity of the initial layout, the **Set Dresser** dynamically places supplementary props without violating existing bounding box constraints. This targeted dressing enriches fine-grained scene detail and enhances overall visual plausibility.

Third, to drive the cinematic storytelling, the **Lighting Arranger** configures environmental lighting (I_{light}) using suitable high dynamic range image (HDRI) maps and discrete light sources. By iteratively refining light intensity and color via VLM-guided aesthetic reflection, it ensures precise emotional alignment with the script’s intended mood. Finally, to transition the scene from static frames to a dynamic narrative, the **Animator** retrieves appropriate motion sequences from an animation database and retargets them to the characters (P_{action}). This comprehensive pipeline successfully drives the temporal evolution of the storyboard, ultimately yielding a rich, physically grounded, and narratively expressive 3D video sequence. (Details are provided in **Appendix**.)

4 Experiments

4.1 Implementation Details

Baselines. We evaluate StoryBlender against two groups of baselines representing distinct generation schemes. The first group consists of leading 2D generative models and AI storyboarding tools (AniMaker [24], Qwen-Edit [33], Story2Board [4], StoryDiffusion [46]), which serve as representatives of state-of-the-art pixel-space visual storytelling. The second group comprises contemporary 3D scene generation and reconstruction frameworks (SceneWeaver [37] and a cascaded pipeline consisting of Qwen-Edit [33] for initial image generation followed by CAST [39] for 3D reconstruction), which serve as representative methods for isolated 3D scene generation and post-hoc 2D-to-3D reconstruction.

Dataset. To evaluate our framework in a high-fidelity filmmaking context, we introduce **CineBoard3D**, a novel dataset curated from eight iconic feature films, such as *The Godfather* and *Pulp Fiction*. CineBoard3D comprises 51 scenes and 178 shots, featuring 95 characters and 81 props. With an average of 22.25 shots per story and rigorous continuity requirements across diverse environments, it provides a significantly more complex and robust testbed for multi-shot spatial consistency than existing text-to-visual benchmarks. Fig. 3 visualizes the word clouds for the asset, layout, environment, and action descriptions found within the dataset.

Evaluation Metrics. We evaluate our framework using three key metric categories. **Consistency Metrics** measure sequence stability across two dimensions: **Self** evaluates intra-sequence coherence (consistency among the generated shots), while **Cross** measures reference-anchored fidelity (adherence to the initial global asset). We apply these dimensions to both **Character Identification Similarity (CIDS)**, which computes the cosine similarity of character features, and **CLIP Style Disentanglement (CSD)**, which uses a style-trained

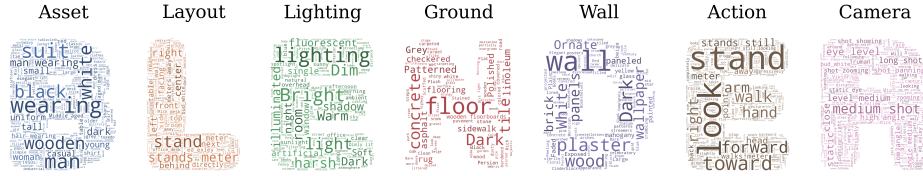


Fig. 3: Narrative Concept Visualization of CineBoard3D.

Table 1: Quantitative comparison on the CineBoard3D dataset. We evaluate performance across multiple dimensions: Character Identification Similarity (CIDS), Onstage Character Count Matching (OCCM), Style Similarity (CSD), Prompt Alignment (PA), and User Study (rank 1–7, Details in the **Appendix**). **Bold** and underlined indicate the best and second-best results.

Method	CIDS $\uparrow$		OCCM $\uparrow$	CSD $\uparrow$		PA $\uparrow$		User $\downarrow$
	Self	Cross	Shots	Self	Cross	Scene	Action	Rank
AniMaker	0.801	0.613	<u>73.91</u>	0.561	0.309	3.689	3.369	3.46
Qwen-Edit	0.849	0.719	67.94	0.503	0.296	3.736	3.613	<u>2.23</u>
Story2Board	0.842	0.772	49.30	0.513	0.264	3.189	2.281	5.25
StoryDiffusion	0.816	0.758	62.89	0.612	<u>0.343</u>	2.547	1.815	5.80
Qwen-Edit+CAST	0.726	0.631	56.39	0.727	0.262	0.800	1.866	6.58
SceneWeaver	<u>0.872</u>	<u>0.773</u>	55.66	<u>0.758</u>	0.216	2.824	1.933	4.13
StoryBlender	0.917	0.803	78.16	0.772	0.352	3.176	2.629	1.84

CLIP encoder to assess global aesthetic coherence. **Onstage Character Count Matching (OCCM)** calculates the numerical accuracy of generated characters against script requirements to penalize spatial hallucinations or omissions. Finally, **Prompt Alignment (PA)** employs a large multimodal model (GPT-4.1) as a judge to rate generated images on a 0–4 scale, assessing both scene alignment (environment/layout) and action alignment (character interactions).

Setup. We conduct our experiments on the proposed CineBoard3D dataset. Our multi-agent framework employs Gemini 3 Pro [29] for high-level reasoning and Gemini 3 Flash to drive the remaining subagents, and the iterative Story-centric Reflection loop, which is capped at a maximum of 5 iterations, with the reflection threshold τ_a set to 8. Scene population utilizes assets retrieved from Poly Haven and Sketchfab. For assets, we use Nano Banana [29] for text-to-image synthesis, followed by Hunyuan 3D [11] for image-to-3D generation. The end-to-end pipeline is integrated within Blender via the Model Context Protocol.

4.2 Main Results

Quantitative Comparisons. StoryBlender’s reliance on a deterministic 3D world model resolves the stability and consistency issues in pixel-based gener-

ative baselines, yielding stronger quantitative performance across key metrics. Specifically, the framework achieves **better identity preservation**, scoring 0.917 in CIDS (Self) and 0.803 in CIDS (Cross), demonstrating that the continuity memory graph anchors identities to persistent 3D meshes rather than relying on pixel-space mechanisms for multi-view consistency, as in StoryDiffusion or Story2Board. Compared with single-shot 3D generation methods like SceneWeaver, StoryBlender shows greater consistency across these metrics, validating the utility of the memory graph in maintaining coherence across multiple cuts. This advantage also extends to global lighting and atmospheric stability, reflected in a CSD score of 0.772. Furthermore, StoryBlender mitigates the common failure mode of hallucinating or omitting characters, achieving the highest **Onstage Character Count Matching** with a score of 78.16 by instantiating discrete models from the scene registry rather than relying on pixel probabilities. While pixel-centric methods like Qwen-Edit score higher in **Prompt Alignment** by bending physical constraints to satisfy prompts, StoryBlender trades this freedom for physical plausibility, prioritizing spatial validity over semantic compliance. These metrics and the user study on asset consistency, environmental consistency, and prompt adherence show that StoryBlender balances narrative requirements with geometric constraints for consistent, spatially coherent storyboards.

Qualitative Analysis. As illustrated by the multi-shot case study of *Casablanca* (see Fig. 4), StoryBlender outperforms 2D visual narrative baselines by prioritizing physical logic over semantic approximation. Unlike pixel-based models that suffer from environmental drift and hallucinate background changes between camera cuts, StoryBlender achieves **geometric stability** by anchoring architectural features and spatial constraints within the $\mathcal{M}_{scene}$ layer, keeping them consistent regardless of camera trajectory. Additionally, the framework executes **precise spatial blocking** by translating semantic prepositions into Euclidean transformations, allowing it to resolve constraints like depth ordering and occlusion that 2D methods flatten or distort. This approach also ensures **reliable numerical and entity management**; by instantiating the required number of 3D meshes cataloged in the $\mathcal{M}_{asset}$ registry, the system addresses the “vanishing” or “multiplying” entity problems common in probabilistic generation, reducing unprompted additions or omissions. Consequently, by maintaining entity fidelity, spatial execution, and environmental stability across dynamic shots, StoryBlender provides a dependable and geometrically sound foundation that surpasses traditional generative baselines.

4.3 Ablation Study

Semantic-Spatial Grounding. We evaluate the role of the hierarchical state machine in maintaining multi-shot continuity by ablating the continuity memory graph. We compare four configurations: (1) **w/o $\mathcal{G}_{cm}$** : a pure LLM baseline processing shot narratives independently; (2) **w/o $\mathcal{M}_{asset}$** : shares the scene layout but lacks a persistent asset registry; (3) **w/o L_{layout} in $\mathcal{M}_{scene}$** : uses the global



Fig. 4: Comparison of baselines on a complex multi-shot sequence from the film *Casablanca*. StoryBlender demonstrates stronger geometric consistency and entity management across shots, maintaining the architectural layout and correct character count in each frame. In contrast, StoryDiffusion and Story2Board capture the general semantic atmosphere but exhibit spatial inconsistencies, hallucinating background changes between camera cuts, and failing to preserve the correct number of characters. (More stories and shots are presented in the **Appendix**.)

asset registry for mesh consistency but computes spatial layout L_{layout} per shot; and (4) **Full $\mathcal{G}_{cm}$** . To quantify geometric instability, we introduce **Spatial Drift Error (SDE)**, measuring the average Euclidean distance of the 8 world-space bounding box corners $v_{j,i}$ for static assets $j \in \mathcal{S}_{static}$ across K consecutive shots:

$$\text{SDE} = \frac{1}{|\mathcal{S}_{static}|} \sum_{j \in \mathcal{S}_{static}} \frac{1}{K-1} \sum_{k=1}^{K-1} \left(\frac{1}{8} \sum_{i=1}^8 \|v_{j,i}^{(k+1)} - v_{j,i}^{(k)}\|_2 \right). \quad (5)$$

Table 2: Ablation of the continuity memory graph on the CineBoard3D.

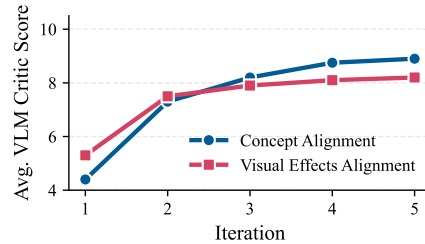
Configuration	CIDS (Self) $\uparrow$	CIDS (Cross) $\uparrow$	SDE $\downarrow$
w/o $\mathcal{G}_{cm}$	0.547	0.382	4.87
w/o $\mathcal{M}_{asset}$	0.532	0.414	0.95
w/o $\mathcal{L}_{layout}$	0.908	0.788	4.12
Full $\mathcal{G}_{cm}$ (Ours)	0.917	0.803	0.28

Table 3: Ablation of Reflection and Visual Effects Agents.

Configuration	PA (Scene) $\uparrow$	PA (Action) $\uparrow$
w/o Visual Effects Agents	0.561	1.374
w/o Reflection ($t = 0$)	2.867	2.540
w/ Reflection ($t = 3$)	3.114	2.621
Full Pipeline (Ours)	3.176	2.629

As shown in Table 2, removing $\mathcal{G}_{cm}$ causes severe identity and geometric failures, indicating that standard LLMs operating over natural-language context windows struggle to maintain multi-shot continuity. Without the asset registry (w/o $\mathcal{M}_{asset}$), CIDS drops significantly, showing that text prompts alone cannot prevent 3D mesh identity drift. Omitting the shared layout (w/o $\mathcal{L}_{layout}$ in $\mathcal{M}_{scene}$) preserves identity but increases SDE to 4.12, revealing spatial inconsistency where background objects shift between shots without persistent spatial context. The full $\mathcal{G}_{cm}$ resolves both issues, achieving consistent identity and stable geometry.

Visual Effects Artist Agents. To assess the impact of specialized roles, we remove the Visual Effects Artist Agents from the pipeline. We replace the atmospheric lighting and character animations with a default Nishita sky texture [20] and static rest poses, causing Prompt Alignment scores to drop (see Table 3). This highlights that geometric layout alone is insufficient. Specialized subagents are required to translate semantic mood and motion into perceptible visual signals. (Visualization cases in the Appendix.)

**Fig. 5: Impact of Story-centric Reflection for the Concept Artist and Visual Effects Artist Agents.**

Story-centric Reflection. We analyze the efficacy of the iterative feedback loop by halting generation at $t = 0$ and $t = 3$. Removing this refinement ($t = 0$) causes Prompt Alignment scores to regress (see Table 3), indicating that zero-shot generation often misses fine-grained constraints. Permitting a partial reflection horizon ($t = 3$) notably improves the scores to 3.114 (Scene) and 2.621 (Action), demonstrating that intermediate feedback effectively corrects errors, though the full five-turn pipeline yields the best overall alignment. Fig. 5 visualizes the system’s recovery, plotting the consistent upward trajectory of VLM critic scores for Concept and Visual Effects Alignment across five iterations to demonstrate semantic convergence. To evaluate the Layout Artist’s spatial reasoning, we compare our engine-driven physical reflection against a naive reflection baseline, which tasks the LLM with fixing previous errors without any

Table 4: Spatial error over five reflection turns for naive and physical methods. D: Direction, R: Relationship, O: Occlusion, C: Contact.

Method	Error	Turn				
		1	2	3	4	5
Naive	D	0.341	0.212	0.188	0.129	0.129
	R	0.224	0.129	0.141	0.094	0.094
	O	0.012	0.000	0.012	0.000	0.012
	C	0.000	0.012	0.000	0.000	0.000
Physical	D	0.377	0.049	0.012	0.008	0.008
	R	0.224	0.110	0.000	0.000	0.000
	O	0.041	0.012	0.000	0.004	0.000
	C	0.008	0.008	0.000	0.000	0.000

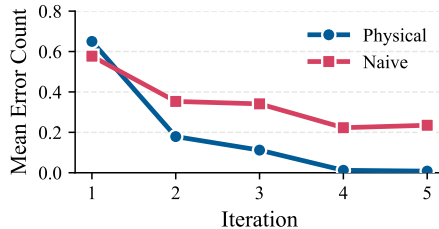


Fig. 6: Impact of Story-centric Reflection for the Layout Artist Agents. Compared to a naive reflection approach, 3D physical reflection rapidly minimizes spatial errors across reflection turns.

spatial context from the 3D engine. As illustrated in Fig. 6, the physical reflection achieves a significantly faster error reduction rate. Table 4 quantifies this geometric self-correction: relying on engine feedback, spatial errors (direction, relationship, contact, and occlusion) drop substantially after the first iteration and reach near-zero by the third verification turn. In contrast, the naive approach plateaus, confirming that LLMs struggle to resolve complex layout constraints without grounded 3D state feedback. (Visualization cases in the **Appendix**.)

4.4 In-Place Scene Editing

A limitation of 2D generative pipelines is their destructive editing process: modifying an element (e.g., camera angle) typically requires re-sampling the entire image, often removing existing details. In contrast, StoryBlender produces a parametric 3D scene graph that supports non-destructive edits through a dual-mode workflow, combining agent-assisted updates and manual editing (Fig. 7). Scene attributes can be modified through natural language instructions. For example, “switch to a close-up shot” updates camera extrinsics ($\mathbf{C}_{cam}$) while preserving geometry. Similarly, instructions such as “change the lighting to mid-day” or “replace the concrete wall with wood” update lighting HDRI ($\mathbf{I}_{light}$) and backdrop textures ($\mathbf{E}_{env}$). These edits remain localized and do not affect other elements. For finer control, StoryBlender exposes the native 3D project file for direct scene graph editing. Users can add or remove props and adjust camera parameters directly in the engine, preserving geometric consistency and enabling the storyboard to function as an editable production asset. This capability allows iterative adjustments without regenerating the scene, improving practical usability for storyboard design.

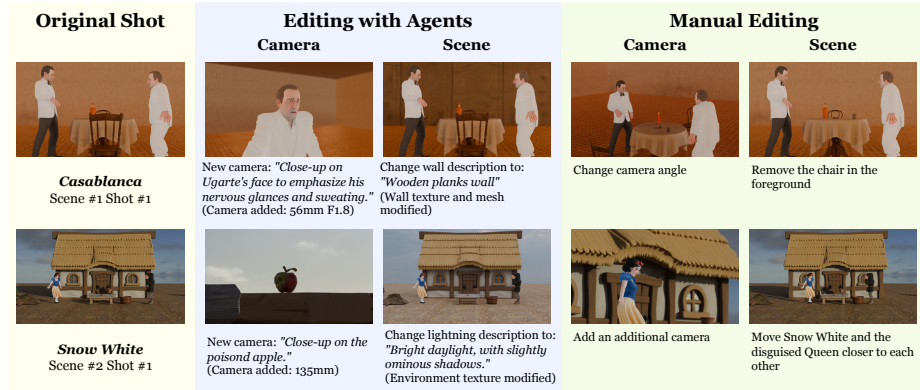


Fig. 7: Non-destructive editing workflow. Starting from an initial generation (left), the figure shows two modes of modification. **Middle:** agent-assisted edits, where natural language updates camera placement, lighting, and textures without changing unrelated geometry. **Right:** manual edits in the 3D engine for asset addition/removal, asset transformation, and camera adjustment. This dual-mode design demonstrates precise editability.



Fig. 8: Videos stylized with Wan 2.7.

5 Conclusion

We introduced StoryBlender, a hierarchical multi-agent framework for generating consistent and editable 3D storyboards from narrative scripts. By grounding generation in a persistent continuity memory graph and a reflection-based planning loop, the system bridges semantic story understanding with physically valid spatial execution, enabling reliable coordination among narrative planning, asset reasoning, and geometric validation. Our pipeline decomposes story generation into semantic grounding, asset materialization, and spatial-temporal dynamics, enabling multi-shot consistency and explicit control within a 3D engine while providing a structured and interpretable workflow for cinematic scene construction. Future work will extend the continuity memory graph with richer semantic and physical attributes, enabling more complex scene reasoning, such as fine-grained object interactions and long-horizon narrative dependencies. We will explore combine generated 3D storyboards with video generation models (see Fig. 8), add audio to rendered clips through video-to-audio generation [3, 12]. We also plan to expand the agent pipeline with stronger motion modeling and interactive editability to support dynamic character behaviors, more expressive cinematography, and tighter human-AI collaboration in production workflows.

References

1. Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., et al.: Stable video diffusion: Scaling latent video diffusion models to large datasets. arXiv preprint arXiv:2311.15127 (2023)
2. Chen, Y., Rao, A., Jiang, X., Xiao, S., Ma, R., Wang, Z., Xiong, H., Dai, B.: Cinepregen: Camera controllable video previsualization via engine-powered diffusion. arXiv preprint arXiv:2408.17424 (2024)
3. Cheng, H.K., Ishii, M., Hayakawa, A., Shibuya, T., Schwing, A., Mitsufuji, Y.: Taming multimodal joint training for high-quality video-to-audio synthesis. arXiv e-prints pp. arXiv–2412 (2024)
4. Dinkevich, D., Levy, M., Avrahami, O., Samuel, D., Lischinski, D.: Story2board: A training-free approach for expressive storyboard generation (2025), <https://arxiv.org/abs/2508.09983>
5. He, H., Yang, H., Tuo, Z., Zhou, Y., Wang, Q., Zhang, Y., Liu, Z., Huang, W., Chao, H., Yin, J.: Dreamstory: Open-domain story visualization by llm-guided multi-subject consistent diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence **47**(12), 11874–11891 (2025). <https://doi.org/10.1109/TPAMI.2025.3600149>
6. Hu, E., Brudy, F., Ledo, D., Fitzmaurice, G., Anderson, F.: Previzwhiz: Combining rough 3d scenes and 2d video to guide generative video previsualization. arXiv preprint arXiv:2602.03838 (2026)
7. Hu, Z., Iscen, A., Jain, A., Kipf, T., Yue, Y., Ross, D.A., Schmid, C., Fathi, A.: Scenecraft: An llm agent for synthesizing 3d scenes as blender code. In: Forty-first International Conference on Machine Learning (2024)
8. Huang, L., Wang, W., Wu, Z.F., Shi, Y., Dou, H., Liang, C., Feng, Y., Liu, Y., Zhou, J.: In-context lora for diffusion transformers (2024), <https://arxiv.org/abs/2410.23775>
9. Huang, Y.H., Wang, W., Huang, S.Y., Wang, Y.C.F.: Toward scene graph and layout guided complex 3d scene generation (2024), <https://arxiv.org/abs/2412.20473>
10. Huang, Y., Qin, Y., Lu, S., Wang, X., Huang, R., Shan, Y., Zhang, R.: Story3d-agent: Exploring 3d storytelling visualization with large language models. arXiv preprint arXiv:2408.11801 (2024)
11. Lai, Z., Zhao, Y., Liu, H., Zhao, Z., Lin, Q., Shi, H., Yang, X., Yang, M., Yang, S., Feng, Y., et al.: Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details. arXiv preprint arXiv:2506.16504 (2025)
12. Li, B., Yang, F., Mao, Y., Ye, Q., Chen, H., Zhong, Y.: Tri-ergon: Fine-grained video-to-audio generation with multi-modal conditions and lufs control. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 4616–4624 (2025)
13. Lin, G., Huang, K., Liu, M., Gao, R., Chen, H., Chen, L., Lu, B., Komura, T., Liu, Y., Zhu, J.Y., Li, M.: Pat3d: Physics-augmented text-to-3d scene generation (2025), <https://arxiv.org/abs/2511.21978>
14. Lin, H., Zala, A., Cho, J., Bansal, M.: Videodirectorgpt: Consistent multi-scene video generation via llm-guided planning. arXiv preprint arXiv:2309.15091 (2023)
15. Liu, C., Wu, H., Zhong, Y., Zhang, X., Wang, Y., Xie, W.: Intelligent grimm-open-ended visual storytelling via latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6190–6200 (2024)

16. Liu, X., Tang, C.K., Tai, Y.W.: Worldcraft: Photo-realistic 3d world creation and customization via llm agents (2025), <https://arxiv.org/abs/2502.15601>
17. Liu, Y., Zhang, K., Li, Y., Yan, Z., Gao, C., Chen, R., Yuan, Z., Huang, Y., Sun, H., Gao, J., et al.: Sora: A review on background, technology, limitations, and opportunities of large vision models. arXiv preprint arXiv:2402.17177 (2024)
18. Mao, J., Huang, X., Xie, Y., Chang, Y., Hui, M., Xu, B., Zhou, Y.: Story-adapter: A training-free iterative framework for long story visualization. arXiv preprint arXiv:2410.06244 (2024)
19. Mou, C., Wang, X., Xie, L., Wu, Y., Zhang, J., Qi, Z., Shan, Y.: T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In: Proceedings of the AAAI conference on artificial intelligence. vol. 38, pp. 4296–4304 (2024)
20. Nishita, T., Sirai, T., Tadamura, K., Nakamae, E.: Display of the earth taking into account atmospheric scattering. In: Proceedings of the 20th annual conference on Computer graphics and interactive techniques. pp. 175–182 (1993)
21. Öcal, B.M., Tatarchenko, M., Karaoglu, S., Gevers, T.: Sceneteller: Language-to-3d scene generation. arXiv preprint arXiv:2407.20727 (2024)
22. Ran, X., Li, Y., Xu, L., Yu, M., Dai, B.: Direct numerical layout generation for 3d indoor scene synthesis via spatial reasoning (2025), <https://arxiv.org/abs/2506.05341>
23. Rao, A., Jiang, X., Guo, Y., Xu, L., Yang, L., Jin, L., Lin, D., Dai, B.: Dynamic storyboard generation in an engine-based virtual environment for video production. In: ACM SIGGRAPH 2023 Posters, pp. 1–2 (2023)
24. Shi, H., Li, Y., Chen, X., Wang, L., Hu, B., Zhang, M.: Animaker: Multi-agent animated storytelling with mcts-driven clip generation. In: Proceedings of the SIGGRAPH Asia 2025 Conference Papers. pp. 1–11 (2025)
25. SongTang, S., Zhao, K., Wang, L., Li, Y., Liu, X., Zou, J., Wang, Q., Chu, X.: Unrealllm: Towards highly controllable and interactable 3d scene generation by llm-powered procedural content generation. In: Findings of the Association for Computational Linguistics: ACL 2025. pp. 19417–19435 (2025)
26. Sun, F.Y., Wu, S., Jacobsen, C., Yim, T., Zou, H., Zook, A., Li, S., Chou, Y.H., Can, E., Wu, X., Eppner, C., Blukis, V., Tremblay, J., Wu, J., Birchfield, S., Haber, N.: 3d-generalist: Self-improving vision-language-action models for crafting 3d worlds (2025), <https://arxiv.org/abs/2507.06484>
27. Sun, Z., Wang, J., Tan, Z., Dong, D., Ma, H., Li, H., Gong, D.: Eggen: Image generation with multi-entity prior learning through entity guidance. In: Proceedings of the 32nd ACM International Conference on Multimedia. pp. 6637–6645 (2024)
28. Sun, Z., Wang, Y., Ng, Y., Xu, Y., Dong, D., Li, H., Ji, P.: T3-s2s: Training-free triplet tuning for sketch to scene synthesis in controllable concept art generation. arXiv preprint arXiv:2412.13486 (2024)
29. Team, G., Anil, R., Borgeaud, S., Alayrac, J.B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A.M., Hauth, A., Millican, K., et al.: Gemini: a family of highly capable multimodal models. arXiv preprint arXiv:2312.11805 (2023)
30. Tewel, Y., Kaduri, O., Gal, R., Kasten, Y., Wolf, L., Chechik, G., Atzmon, Y.: Training-free consistent text-to-image generation (2024), <https://arxiv.org/abs/2402.03286>
31. Wang, J., Yang, F., Li, B., Gou, W., Yan, D., Zeng, A., Gao, Y., Wang, J., Jing, Y., Zhang, R.: Freeman: Towards benchmarking 3d human pose estimation under real-world conditions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21978–21988 (2024)

32. Wei, Z., Wu, H., Zhang, L., Xu, X., Zheng, Y., Hui, P., Agrawala, M., Qu, H., Rao, A.: Cinevision: An interactive pre-visualization storyboard system for director–cinematographer collaboration. In: *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*. pp. 1–18 (2025)
33. Wu, C., Li, J., Zhou, J., Lin, J., Gao, K., Yan, K., Yin, S.m., Bai, S., Xu, X., Chen, Y., et al.: Qwen-image technical report. *arXiv preprint arXiv:2508.02324* (2025)
34. Wu, W., Zhu, Z., Shou, M.Z.: Automated movie generation via multi-agent cot planning. *arXiv preprint arXiv:2503.07314* (2025)
35. Xie, Z., Tang, D., Tan, D., Klein, J., Bisschand, T.F., Ezzini, S.: Dreamfactory: Pioneering multi-scene long video generation with a multi-agent framework. *arXiv preprint arXiv:2408.11788* (2024)
36. Yang, J., Li, B., Zeng, A., Zhang, L., Zhang, R.: Open-world human-object interaction detection via multi-modal prompts. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 16954–16964 (2024)
37. Yang, Y., Jia, B., Zhang, S., Huang, S.: Sceneweaver: All-in-one 3d scene synthesis with an extensible and self-reflective agent (2025), <https://arxiv.org/abs/2509.20414>
38. Yang, Y., Jia, B., Zhi, P., Huang, S.: Physcene: Physically interactable 3d scene synthesis for embodied ai. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 16262–16272 (2024)
39. Yao, K., Zhang, L., Yan, X., Zeng, Y., Zhang, Q., Xu, L., Yang, W., Gu, J., Yu, J.: Cast: Component-aligned 3d scene reconstruction from an rgb image. *ACM Transactions on Graphics (TOG)* **44**(4), 1–19 (2025)
40. Yao, S., Sun, M., Li, B., Yang, F., Wang, J., Zhang, R.: Dance with you: The diversity controllable dancer generation via diffusion models. In: *Proceedings of the 31st ACM International Conference on Multimedia*. pp. 8504–8514 (2023)
41. Ye, H., Zhang, J., Liu, S., Han, X., Yang, W.: Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721* (2023)
42. Zhang, C., Sun, Z., Zhang, Z., Wang, J., Zhang, Y., Gong, D., Mo, H., Dong, D.: Hierarchical and step-layer-wise tuning of attention specialty for multi-instance synthesis in diffusion transformers. *arXiv preprint arXiv:2504.10148* (2025)
43. Zhang, K., Jiang, L., Wang, A., Fang, J.Z., Zhi, T., Yan, Q., Kang, H., Lu, X., Pan, X.: Storymem: Multi-shot long video storytelling with memory (2025), <https://arxiv.org/abs/2512.19539>
44. Zheng, J., Cun, X.: Fairygen: Storied cartoon video from a single child-drawn character (2025), <https://arxiv.org/abs/2506.21272>
45. Zhong, L., Guo, C., Xie, Y., Wang, J., Li, C.: Sketch2anim: Towards transferring sketch storyboards into 3d animation. *ACM Transactions on Graphics (TOG)* **44**(4), 1–15 (2025)
46. Zhou, Y., Zhou, D., Cheng, M.M., Feng, J., Hou, Q.: Storydiffusion: Consistent self-attention for long-range image and video generation. *Advances in Neural Information Processing Systems* **37**, 110315–110340 (2024)
47. Çelen, A., Han, G., Schindler, K., Gool, L.V., Armeni, I., Obukhov, A., Wang, X.: I-design: Personalized llm interior designer (2024)