FilmSceneDesigner: Chaining Set Design for Procedural Film Scene Generation

Zhifeng Xie[1,2], Keyi Zhang[1], Yiye Yan[1], Yuling Guo[1], Fan Yang[1], Jiting Zhou[1,2], Mengtian Li [1,2*]

[1]Department of Film and Television Engineering, Shanghai University
[2]Shanghai Engineering Research Center of Motion Picture Special Effects
{zhifeng_xie, ky_zhang, 2543912909, 1304422167, yangphan, zjting, mtli}@shu.edu.cn

Abstract

Film set design plays a pivotal role in cinematic storytelling and shaping the visual atmosphere. However, the traditional process depends on expert-driven manual modeling, which is labor-intensive and time-consuming. To address this issue, we introduce **FilmSceneDesigner**, an automated scene generation system that emulates professional film set design workflow. Given a natural language description, including scene type, historical period, and style, we design an agent-based chaining framework to generate structured parameters aligned with film set design workflow, guided by prompt strategies that ensure parameter accuracy and coherence. On the other hand, we propose a procedural generation pipeline which executes a series of dedicated functions with the structured parameters for floorplan and structure generation, material assignment, door and window placement, and object retrieval and layout, ultimately constructing a complete film scene from scratch. Moreover, to enhance cinematic realism and asset diversity, we construct **SetDepot-Pro**, a curated dataset of 6,862 film-specific 3D assets and 733 materials. Experimental results and human evaluations demonstrate that our system produces structurally sound scenes with strong cinematic fidelity, supporting downstream tasks such as virtual previs, construction drawing and mood board creation.

Introduction

Film scenes play a crucial role in cinematic storytelling by conveying narrative structure and visual atmosphere. As shown in Figure 1, in traditional workflows, the art department analyzes film scripts, extracting key elements such as scene type, color palette, historical period, and regional characteristics to guide scene creation. After gathering this information, set designers manually construct the scene using professional modeling tools. However, current film production workflows still rely heavily on expert-driven manual modeling, which is time-consuming and requires substantial artistic expertise. This creates a compelling need for an efficient and automated solution tailored to film set design.

Existing scene generation methods fall short in addressing cinematic requirements. Image-based approaches (Chung et al. 2023; Fang et al. 2023; Fridman et al. 2023; Höllein et al. 2023) rely on depth inference and often produce distorted meshes with artifacts such as holes and stretched surfaces, failing to meet professional standards. Procedural methods (Raistrick et al. 2024) typically require manual parameter configuration and are limited by predefined room templates, lacking structural diversity and flexibility. LLM-based approaches (Yang et al. 2024; Çelen et al. 2024) depend heavily on general-purpose datasets and focus on the functional roles of rooms rather than their structural types (e.g., wall-structure vs. column-structure), resulting in scenes with low cinematic fidelity. In summary, existing approaches suffer from two major challenges when applied to film set design: (1) none of them align with the established workflow of film scene design, making it difficult to integrate into production pipelines; and (2) they primarily generate general-purpose scenes, lacking the cinematic fidelity required for professional use.

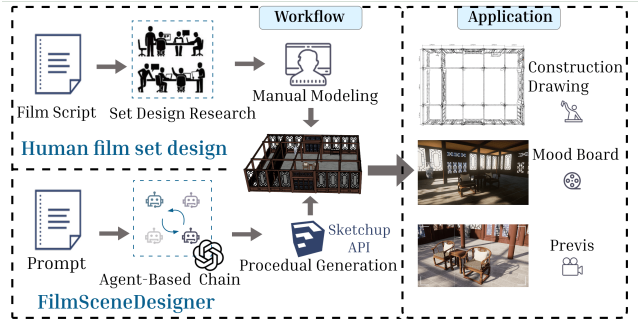

Figure 1: Human Film Set Design: Traditional set design involves labor-intensive script analysis, object research, and manual modeling. FilmSceneDesigner: Our system leverages an agent chain of set design to automate procedural generation, enabling efficient generation of film sets for construction drawing, mood board, and previs.

According to our observations, when professional set designers construct a film scene, the workflow typically involves four sequential stages: (1) constructing the set structure, (2) dressing the surfaces with materials, (3) installing doors and windows, and (4) arranging props in the scene. In addition, we observe that the quality and appropriateness of material and props play a decisive role in determining

*Corresponding author.

whether a scene conveys authentic cinematic realism.

Inspired by these observations and to address the first challenge, we propose **FilmSceneDesigner**, a novel system that integrates procedural generation with an agent-based chaining framework to emulate the traditional film set design workflow. As shown in Figure 2, given a natural language description including scene type, historical period, and style, the framework first determines the appropriate structural category (wall or column) and generates coherent parameters through prompt-guided reasoning. These parameters are then executed in a procedural pipeline that follows the four key stages of set design: floorplan and structure, material assignment, door and window placement, and object retrieval and layout. Each stage is supported by dedicated generation functions in SketchUp, a widely adopted tool in the film set design industry. Furthermore, our system supports downstream tasks such as virtual previs, construction drawing and mood board creation. To address the second challenge, we construct **SetDepot-Pro**, a film-specific dataset spanning diverse historical periods and regions to enhance cinematic fidelity.

Our main contributions are as follows:

- We propose **FilmSceneDesigner**, an automated system for film scene generation that emulates professional set design and seamlessly fits production workflows.
- We combine procedural generation and an agent-based chaining framework to generate structurally diverse, semantically coherent scenes from natural language.
- We construct **SetDepot-Pro**, a film-specific dataset of 6,862 labeled assets and 733 materials supporting the creation of high-fidelity, stylistically rich film scenes.

Related Work

3D Indoor Scene Generation

Research on 3D scene generation spans multiple methodologies. **Image-based** models (Chung et al. 2023; Fang et al. 2023; Fridman et al. 2023; Höllein et al. 2023) leverage pre-trained text-to-image models and depth estimation to lift 2D images to 3D using NeRF (Mildenhall et al. 2021) or Gaussian Splatting (Kerbl et al. 2023) representations. Another major direction is **layout-driven** scene generation, which defines spatial arrangements before populating a scene. Rule-based methods (Yeh et al. 2012; Weiss et al. 2018; Merrell et al. 2011) manually encode spatial constraints, which ensures structural consistency but limits its diversity. On the other hand, some methods (Tang et al. 2024; Zhai et al. 2023; Wu et al. 2024c; Zhai et al. 2024) learn scene priors from datasets and some of them adapt scene graphs (Gao et al. 2024; Lin and Mu 2024) to represent object relationships and enforce spatial constraints, while they are inherently limited by dataset biases. More recently, **LLM-driven** approaches have been introduced for scene generation. Layoutgpt (Feng et al. 2023) prompts LLMs to directly generate absolute object coordinates and some methods (Yang et al. 2024; Fu et al. 2024; Çelen et al. 2024; Littlefair, Dutt, and Mitra 2025) infer relative spatial relationships and incorporate constraint-based object placement to ensure a plausible layout with LLMs. **Procedural modeling** is also a powerful tool for scalable 3D scene generation (Raistrick et al. 2023), Infinigen Indoors (Raistrick et al. 2024) employs constraint-based object arrangement to achieve realistic spatial composition. Nevertheless, existing approaches either generate indoor scenes with fixed room types or focus on general-purpose scenes, neither of which meets the requirements of cinematic set design; moreover, they cannot be seamlessly integrated into real film set design workflows.

Dataset

Datasets for 3D scene generation can be broadly categorized into two types: structured **indoor scene datasets** and open-vocabulary **3D asset datasets**. For **indoor scene**, several datasets provide structured 3D assets specifically for indoor environments. Scan2CAD (Avetisyan et al. 2019) contains CAD-based synthetic models alongside scanned indoor scenes. 3D-FUTURE (Fu et al. 2021b) includes synthetic indoor assets with high-quality textures, enhancing realism in virtual environments. 3D-FRONT (Fu et al. 2021a) offers synthetic indoor scenes along with their associated digital assets, while ScanNet (Dai et al. 2017) consists of scanned indoor scene models. However, these datasets are inherently constrained by predefined room categories, primarily focusing on common household spaces such as bedrooms, kitchens, and living rooms, thereby limiting the diversity of available assets to furniture like tables and chairs. Beyond structured indoor scene datasets, large-scale open-vocabulary **3D asset datasets** provide a broader range of digital assets. Objaverse (Deitke et al. 2023) aggregates 3D models from diverse sources, offering a vast and heterogeneous collection of objects beyond the constraints of specific room types. Additionally, commercial software platforms such as Unreal Engine Marketplace (Inc. 2024b), UE4Arch (UE4Arch 2024), and Adobe Stock (Inc. 2024a) maintain their own curated digital asset libraries, supporting various creative and industrial applications. While these datasets provide valuable digital assets, they mainly focus on general-purpose objects. Asset selection plays a crucial role in achieving cinematic realism. To fill this gap, we build a specialized database of 3D assets and materials tailored for film production. Compared to these datasets above, our dataset is specifically designed for film scenes.

Method

Procedural Scene Generation

Floorplan and Structure. This stage constructs the foundational floorplan and structural framework of the scene. We formalize two architectural structure types to accommodate diverse structural requirements. We define a wall-structure scene as a collection of rooms S:

$$S = [R_1, R_2, \ldots, R_n], \tag{1}$$

where $R_1, R_2, \ldots, R_n$ represent individual rooms within the scene. Each room is composed of a set of boundary edges, which can be either straight lines or arcs, defined as:

$$E_{ij} = \begin{cases} [x_{\text{start}}, y_{\text{start}}, x_{\text{end}}, y_{\text{end}}], & \text{line}, \\ [x_{\text{start}}, y_{\text{start}}, x_{\text{end}}, y_{\text{end}}, h_{\text{chord}}], & \text{arc}, \end{cases} \tag{2}$$

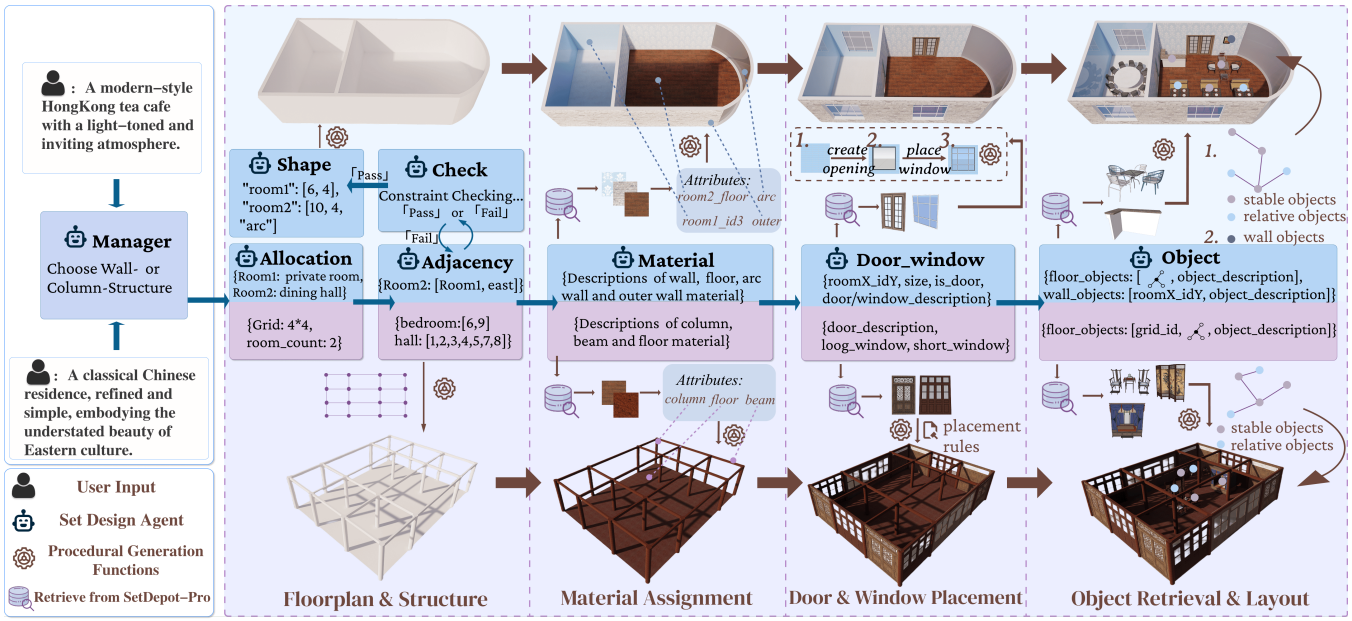

Figure 2: The framework of FilmSceneDesigner. Given a scene description, FilmSceneDesigner constructs an agent-based chaining framework to generate the structured parameters, and then executes the procedural functions with these parameters for floorplan and structure, material assignment, door and window placement, and object retrieval and layout. All required assets, including materials, doors and windows, and objects, are retrieved from SetDepot-Pro to ensure high cinematic fidelity.

where E_{ij} denotes the j-th edge of the i-th room. A line is defined by its start and end coordinates, while an arc additionally requires a chord height h_{chord}. Here, x and y are the 2D coordinates. To represent spatial adjacency relationships between rooms, we use a directed graph representation:

$$A = \{(r_i, r_j, \text{relation}) \mid r_i, r_j \in S, r_i \neq r_j\}, \quad (3)$$

where r_i and r_j denote two rooms, and *relation* specifies their relative positioning, such as east or north. In our modeling system, room layout is based on the 2D coordinate system of SketchUp: the first room (typically `room1`) is placed at the origin $(0, 0)$, and the positions of subsequent rooms are inferred sequentially based on their spatial relationships with previously placed rooms.

For column-structure scenes, the floorplan is represented as a grid G of column points:

$$G = [C_{1,1}, C_{1,2}, \ldots, C_{m,n}], \quad (4)$$

where $C_{i,j}$ denotes the column located at row i and column j. Each column is defined by its center coordinates $(x_{i,j}, y_{i,j})$ and radius r_{column}:

$$C_{i,j} = [x_{i,j}, y_{i,j}, r_{\text{column}}]. \quad (5)$$

According to the above definitions, we design a series of functions to accomplish the generation process. For wall-structure scenes, we first use the `parse_edge` function to infer room boundary edges from room sizes and adjacency relationships. These edges are categorized into two types: external walls and internal walls that partition rooms. We then employ several designed functions to draw straight lines and arcs, followed by the `add_face` function to create enclosed floor surfaces from these boundaries. Wall geometry is generated by the `offset` function to establish inner and outer wall thickness, and finally apply the `pushpull` function to extrude the walls to the specified height. For column-structure scenes, we compute the center coordinates of all columns based on the grid's row-column count and spacing to form a structured grid. The generation then sequentially applies functions to add the ground plane, instantiate columns at computed positions, and connect adjacent columns with beams to complete the column-beam framework, using the aforementioned functions.

Material Assignment. This stage assigns appropriate materials to structural components through a systematic attribute-based approach. Our material assignment stage operates in two phases: attribute definition and material application.

In the attribute definition phase, we employ the `set_attribute` function to assign unique identifiers to each structural element. For wall-structure scenes represented as $S = \{R_1, R_2, \ldots, R_n\}$, we systematically assign attributes to room containers (`roomX`), floor surface of each room (`roomX_floor`), directional inner walls (`roomX_idY` where $Y \in \{1, 2, 3, 4\}$ denotes west, south, north, east walls respectively), curved wall segments (`arc`), and exterior boundaries (`outer`). For column-structure scenes, attributes are assigned to the ground plane (`floor`), columns (`column`), and beams (`beam`).

Following attribute assignment, the `apply_material` function performs batch material application by mapping these structured identifiers to corresponding material properties retrieved from our SetDepot-Pro dataset via Sentence-BERT similarity matching. Specifically, as illustrated in Fig-

ure 3, textual descriptions are encoded into vector representations using a Sentence-BERT (Reimers and Gurevych 2019) encoder. Likewise, assets are pre-encoded and stored in an embedding database. Retrieval is performed by computing similarity scores between query and database embeddings, from which the most relevant candidates are selected.

Door and Window Placement. This stage installs appropriate doors and windows for the current scene. For wall-structure scenes, the process operates in two distinct phases: opening creation and asset placement. In the opening creation phase, we employ the `open_wall` function to create openings on target wall surfaces indexed by `roomX_idY` attributes. The opening position depends on the asset type: doors are opened from the ground level, while windows are positioned at the wall center.

In the asset placement phase, door and window assets are retrieved from our SetDepot-Pro dataset via Sentence-BERT similarity matching, then positioned through a sequence of dedicated functions: adaptive `rotate` based on the `roomX_idY` wall orientation, `scale` to match the opening dimensions, and `translate` to precisely fit the asset into the opening. This two-stage approach ensures accurate geometric alignment between openings and their corresponding assets.

For column-structure scenes, doors and windows are directly inserted into the natural gaps between adjacent columns without requiring opening creation. We categorize placements into three types: doors, long windows, and short windows. Based on extensive analysis of real-world column-structure scenes, we establish a set of placement rules to guide door and window placement. (1) doors are positioned between the middle column pairs of the first and last rows; (2) long windows are placed between the middle column pairs of the first and last columns; (3) remaining perimeter gaps are filled with short windows; (4) short windows are used to partition the entire scene into different rooms. All assets are retrieved from our dataset via similarity matching and positioned with appropriate functions, including rotation, scaling, and translation consistent with the wall-structure scene.

Object Retrieval and Layout. This stage is responsible for selecting semantically appropriate assets and arranging them into spatially coherent layouts. Objects are first retrieved from our SetDepot-Pro dataset using semantic similarity matching. Then the retrieved objects are subsequently arranged according to the underlying structural type. In wall-structure scenes, each room (`roomX`) serves as the placement unit for both floor and wall objects, while in column-structure scenes, units U are defined as rectangular regions enclosed by four columns:

$$U = \{C_{i_1,j_1},\ C_{i_1,j_2},\ C_{i_2,j_1},\ C_{i_2,j_2}\}. \tag{6}$$

Only floor objects are placed within each column-structure unit. Floor objects are further categorized into *stable objects* and *relative objects*. Stable objects include corner, edge, and center placements, with edge and center serving as *anchor objects*. Each anchor is assigned a unique `anchor_ID`, which can be referenced by relative objects. The placement of a relative object is determined by its anchor reference,

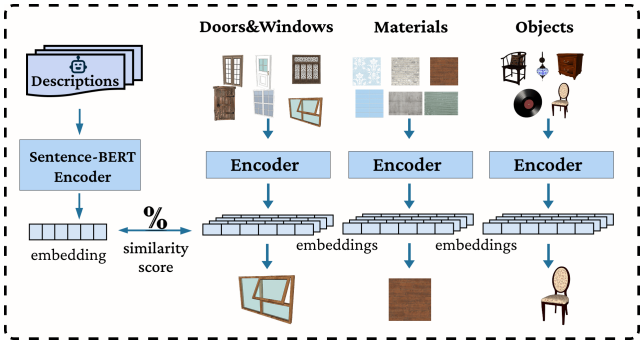

Figure 3: Retrieval Process: Textual descriptions from agent responses are encoded into vector representations using a Sentence-BERT encoder. Similarly, doors, windows, materials, and objects are encoded and stored in an embedding database. By computing similarity scores between the two embeddings, the highest-scoring assets are retrieved.

combined with a spatial relation (e.g., *left*, *right*, *in front of*, *behind*, *above*) and a distance level (e.g., *near*, *far*). Formally, layout relations are represented as a triplet $\mathcal{L}$:

$$\mathcal{L} = (o_a, r, o_r), \tag{7}$$

where $o_a \in \{edge, center\}$ is the anchor object, o_r the relative object, and $r = (s, d)$ the composite relation consisting of spatial relation s and distance level d. The relative position p is inferred as:

$$p(o_r) = p(o_a) + \lambda(d) \cdot \vec{v}(s), \tag{8}$$

where $\vec{v}(s)$ denotes the direction vector of s, and $\lambda(d)$ is the scaling factor associated with d. Based on this inference, the system applies the previously introduced placement functions to perform model importation, rotation, scaling, and translation. After these operations, two refinement functions, `avoid_collision` and `refine_orientation`, are invoked to further adjust object placement, ensuring collision-free layouts.

For wall-structure scenes, wall objects follow a dedicated strategy: each object is assigned to a wall surface identified by `roomX_idY`, adaptively rotated to face the room interior, then scaled and translated to the center of target wall, resulting in spatial plausibility and visual uniformity.

Agent-Based Chaining Framework

Set Design Agents. To support procedural generation at the conceptual design stage, we construct an agent-based chaining framework that progressively transforms natural language descriptions into structured parameters required by procedural generation functions. Within this framework, multiple set design agents are defined, each responsible for a specific sub-task. Following the workflow decomposition in professional film set design, the complex generation task is divided into sub-tasks with clearly defined responsibilities, each handled by one or more dedicated agents. The overall process is initiated and coordinated by the *Manager*, which sequentially invokes the agents to complete the four

Agent	Duties
Manager	Choose wall-structure or column-structure according to user input.
Allocation	Wall: assign number and functions of rooms; Column: define grid and room count.
Adjacency	Wall: Define spatial adjacency between rooms; Column: Assign rooms to occupied grid cells.
Check	Validate adjacency logic and return to Adjacency if constraints are violated(Wall only).
Shape	Generate room sizes and determine whether to add arc walls (Wall only).
Material	Wall: Select materials for floors and walls; Column: Select materials for floors, columns, and beams.
Door_Window	Wall: Select walls, plan opening sizes, and describe door/window styles; Column: Describe door/window styles.
Object	Wall: Select floor (stable and relative) and wall objects; Column: Select floor (stable and relative) objects.

Table 1: Agent roles and their duties (Wall for wall-structure scene and Column for column-structure scene).

stages of procedural generation. The duties of all set design agents are summarized in Table 1. Beyond the general role assignment, we further incorporate specialized mechanisms to enhance the accuracy of agent responses during generation. During adjacency planning, we design a looped verification mechanism between *Adjacency* and *Check*, forming a generate–verify–revise cycle that enforces spatial alignment under constraints. To ensure agents consistently fulfill their designated roles, we employ prompt engineering strategies: **Role-Play** and **Few-Shot Prompting** reinforce behavioral boundaries and contextual grounding, while the *door_window* agent uniquely adopts a **Chain-of-Thought (CoT)** strategy to reason from shared wall analysis to opening placement and style description. These strategies collectively enhance the controllability and semantic coherence of the agent-based chaining framework.

Agent Turn Control. Our agent-based chaining framework follows the film set design pipeline in a strictly sequential manner. To regulate the speaking order of agents, we implement a Finite State Machine (FSM) (Wu et al. 2023) with a state transition dictionary (`speaker_transitions_dict`) specifying the valid next speaker(s) for each agent. Moreover, each agent declares its context and speaking conditions through a natural language `description` field (e.g., "I can only speak after *Adjacency*, and the next speaker is *Shape*"). These semantic constraints, together with FSM, provide a robust control mechanism for stable and predictable agent responses.

Hooks for Data Bridging. To seamlessly connect agent responses with the procedural generation pipeline, we adopt the hook mechanism from AutoGen (Wu et al. 2024a), which extracts agent response and converts it into standardized structured parameters in real time. For key agents, custom hooks continuously monitor responses and extract valid parameters that are directly forwarded to downstream generation functions. Beyond internal parameter passing, hooks also trigger external retrieval routines (including material selection, door/window asset matching, and object retrieval from SetDepot-Pro), thereby providing a unified interface that grounds high-level language descriptions into concrete generation operations for automated scene construction.

SetDepot-Pro

Collection and Pre-processing. The material library covers 733 materials across 12 categories (e.g., aged concrete, brick wall, flooring, marble, wallpaper, glass), sourced from professional repositories. The object library contains 6,862 assets obtained through three primary channels: (1) acquisitions from architectural studios, (2) purchases from model platforms, and (3) contributions from film art departments, all secured with appropriate usage rights. To enhance diversity, we collected not only general-purpose models (e.g., furniture for kitchens, offices, or restaurants) but also culturally distinctive items (e.g., Persian carpets, farming tools, ancient statues) and period-specific props from film sets and traditional Chinese palaces. For composite items such as table–chair ensembles, we retained them as unified assets to preserve contextual integrity. All assets underwent manual pre-processing, including resetting orientation, smoothing redundant geometry and scaling to real-world dimensions to ensure accurate spatial integration.

Image Capture and Annotation. Standardized images were captured for each asset using automated camera controls in SketchUp, ensuring complete framing and consistent viewpoints. Object images were taken after orientation normalization, using the `active_view` and `zoom_extents` functions for consistency. Annotations were generated with GPT-4V, covering attributes such as category, style, cultural origin, and era. To ensure reliability, domain experts from film art departments reviewed critical labels, particularly those involving historical and cultural attributes. Additionally, materials with patterned textures were manually annotated with scale factors to preserve realistic proportions.

Feature Encoding. After annotation, all labels were concatenated into unified textual descriptions for each asset. To enable semantic retrieval, we employed Sentence-BERT (all-mpnet-base-v2) to encode these descriptions into dense vector representations. The resulting features were normalized and stored in a precomputed embedding database for retrieval tasks. For the object library, features of door and window models were extracted and maintained as an independent category for door and window placement.

Experiment

Implementation Details. For the LLM, we employ OpenAI's GPT-4o (Hurst et al. 2024) with a temperature setting of 0.7. For semantic retrieval, the Sentence-BERT model employed is all-mpnet-base-v2 (Reimers and Gurevych 2019). All reported results were obtained on a MacBook equipped with an Apple M2 chip and 16GB of memory. Pro-

Figure 4: Qualitative comparison: A visual comparison of film scenes generated by our method, HOLODECK, and DreamScene. The results show that our method achieves stronger expressiveness in film scene generation, enabling scenes with distinct era, style, and regional characteristics, thereby ensuring greater historical and cinematic authenticity.

cedural generation was carried out within SketchUp 2024 (Trimble Inc. 2024), and the rendered visualizations were produced using Enscape (Enscape GmbH 2024).

Metric. Following prior work (Çelen et al. 2024; Wu et al. 2024b) demonstrating GPT-4V's alignment with human judgments in 3D evaluation, we adopt a GPT-4V-based criteria to evaluate the quality of scenes generated by our method and the baseline works (HOLODECK (Yang et al. 2024) and DreamScene (Li et al. 2024)), assessing their suitability for cinematic use. The evaluation covers five core aspects: *Object Layout*, *Material Selection*, *Style Consistency*, *Object Selection*, and *Overall Aesthetic Atmosphere*. Specifically, Object Layout examines spatial balance and composition; Material Selection evaluates whether the chosen materials match the style described in the natural language description; Style Consistency checks alignment with historical or thematic intent; Object Selection evaluates whether the chosen objects align with the natural language description and maintain cinematic realism; Overall Aesthetic Atmosphere evaluates the overall harmony among material, door and window, and object selection within the scene. Each aspect is rated on a scale of 0-10, with higher scores indicating stronger cinematic alignment. To ensure robustness, each scene is scored 10 times independently (by setting $n = 10$ in the GPT-4V API), and we report the average score and standard deviation for each metric.

Quantitative Analysis. As shown in Table 2, our method consistently achieves higher scores in layout design, material realism, style consistency, object selection, and overall aesthetic atmosphere. The *chinese residence* setting demonstrates the most notable improvements. Our method achieves scores of 8.4 in layout, 8.6 in material, and 9.0 in style, substantially outperforming HOLODECK (5.4, 4.4, 3.4) and DreamScene (7.0, 6.4, 8.2) on the same metrics. These gains are attributed to our adoption of a culturally accurate column-structure scene, which better reflects the historical and architectural semantics of traditional Chinese design. In other scenes, our method continues to outperform. For the *vintage room*, it achieves 7.4 in layout and 8.6 in atmosphere, exceeding DreamScene (6.4, 6.2) and HOLODECK (6.0, 5.6). In the *asian-style bedroom*, it leads with 7.6 and 8.6. These consistent advantages underscore our superior spatial planning and aesthetic control. Overall, the results demonstrate our framework's capability to generate semantically grounded and stylistically faithful film sets across diverse cinematic contexts. We also evaluate HOLODECK on SetDepot-Pro (see supplementary material).

Qualitative Analysis. Figure 4 presents a visual comparison among our method, HOLODECK, and DreamScene across 4 scene types under identical prompts. In the European-style room, our design captures warm tones and temporal depth, while HOLODECK loses structural integrity and DreamScene over-saturates colors. The 1980s livingroom generated by our method retains authentic wear and dim tones, whereas HOLODECK fails to reflect aging, and DreamScene appears visually inconsistent. For the Southeast Asian bedroom, our method incorporates regionally appropriate furniture and decorative patterns. DreamScene, while visually rich, suffers from cluttered layout, and HOLODECK lacks tropical characteristics. The classical Chinese residence best highlights our structural advantage: we adopt column-structure and traditional furnishings aligned with historical aesthetics, in contrast to the wall-structure and stylistically mismatched outputs of

Method	Room Type	Layout↑	Material↑	Style↑	Object↑	Atmosphere↑
HOLODECK	western guestroom	7.2 ± 0.4	6.0 ± 0.0	8.0 ± 0.0	7.0 ± 0.0	7.4 ± 0.49
	vintage room	6.0 ± 0.0	5.4 ± 0.8	6.8 ± 0.75	5.8 ± 0.75	5.6 ± 0.49
	asian-style bedroom	7.0 ± 0.0	6.0 ± 0.0	8.0 ± 0.0	7.2 ± 0.4	6.8 ± 0.4
	chinese residence	5.4 ± 0.49	4.4 ± 0.49	3.4 ± 0.49	3.4 ± 0.49	4.4 ± 0.49
DreamScene	western guestroom	8.2 ± 0.4	8.2 ± 0.4	9.0 ± 0.0	8.0 ± 0.0	9.0 ± 0.0
	vintage room	6.4 ± 0.49	5.4 ± 0.49	7.8 ± 0.4	6.8 ± 0.4	6.2 ± 0.75
	asian-style bedroom	7.0 ± 0.0	6.8 ± 0.98	8.6 ± 0.49	7.8 ± 0.4	7.8 ± 0.98
	chinese residence	7.0 ± 0.0	6.4 ± 0.8	8.2 ± 0.4	7.6 ± 0.49	7.6 ± 0.49
Ours	western guestroom	8.2 ± 0.4	8.4 ± 0.49	9.0 ± 0.0	8.2 ± 0.4	9.0 ± 0.4
	vintage room	7.4 ± 0.9	8.2 ± 0.4	9.0 ± 0.0	8.2 ± 0.4	8.6 ± 0.49
	asian-style bedroom	7.6 ± 0.49	7.8 ± 0.4	9.0 ± 0.0	7.8 ± 0.4	8.6 ± 0.49
	chinese residence	8.4 ± 0.49	8.6 ± 0.49	9.0 ± 0.0	8.2 ± 0.4	9.0 ± 0.0

Table 2: Quantitative results: A comparison among our method, HOLODECK, and DreamScene across five key evaluation metrics: Object Layout, Material Selection, Style Consistency, Object Selection, and Overall Aesthetic Atmosphere.

Strategy	3 Rooms	4 Rooms	5 Rooms
Adjacency	90%	50%	30%
Adjacency + Check	100%	60%	50%

Table 3: Ablation study on *Adjacency* and *Check*, conducted to evaluate the impact of *Check* on improving the accuracy of parameters generated by *Adjacency*.

Agent	Prompt	+Role-Play	+Few-Shot	+CoT
Material	0.77	0.80	0.89	/
Door_Window	0.73	0.82	0.87	0.95
Object	0.75	0.80	0.93	/

Table 4: Normalized scores for each agent under different prompting strategies. "/" indicates that the strategy was not applied, as these tasks did not require multi-step reasoning.

HOLODECK and DreamScene. These results highlight our strength in cultural specificity and cinematic coherence.

Ablation Study. In our framework, the *Adjacency* agent proposes room-to-room relationships, while the *Check* agent validates and refines them. We compare *Adjacency* alone with *Adjacency* + *Check* across layouts of 3–5 rooms. As room count grows, *Adjacency* alone frequently produces invalid connections, whereas incorporating *Check* systematically corrects errors. Over 10 samples per setting, the combined setup achieves higher correctness rates (Table 3), particularly in more complex room adjacency.

To evaluate the impact of different prompting strategies, we conducted an ablation study on three key agents as shown in Table 4. For each agent, we defined three evaluation criteria and scored three samples per setting on a scale of 1-5. The scores were summed across the three criteria (maximum 15) and then normalized to the range $[0, 1]$. Final results were averaged across all samples. Detailed scoring criteria are provided in the supplementary material. All prompting strategies improved performance, with Few-Shot prompting showing the most consistent gains, and Chain of Thought proving particularly effective for *Door_Window*.

User Study. To evaluate the effectiveness of our method in realistic film set design scenario, we conducted a user study involving 32 film industry professionals. The study covered 4 representative scene types, each generated by our method, HOLODECK, and DreamScene. Participants rated the results across five key criteria: text-structure consistency, material selection, object selection, aesthetic atmosphere, and spatial composition. As summarized in Table 5, our method achieved the highest average scores across all evaluation dimensions, indicating superior structural coherence, visual realism, and alignment with cinematic design goals.

Method	Struc.	Mat.	Obj.	Atmos.	Spa.
HOLODECK	2.59	2.67	2.74	2.66	2.85
DreamScene	2.71	2.34	2.41	2.21	2.41
Ours	4.88	4.62	4.66	4.55	4.53

Table 5: User study results across five evaluation criteria: structural coherence, material selection, object selection, aesthetic atmosphere, and spatial composition. Scores are on a scale of 1-5, with the higher the better.

Conclusion

We presented FilmSceneDesigner, a novel scene generation system that emulates professional film set design workflow. By combining an agent-based chaining framework with procedural generation, our method enables dynamic parameter selection, ensuring greater flexibility in scene composition. To further enhance cinematic authenticity, we constructed SetDepot-Pro, a film-specific dataset spanning different historical periods and architectural styles, allowing more contextually accurate scene generation. Experimental results demonstrate that our approach effectively produces cinematically authentic scenes that meet industry standards of film set design. User studies further validate that our system generates visually coherent and structurally realistic environments, making it a valuable tool for film set design.

Acknowledgments

This work is supported by the Natural Science Foundation of Shanghai (Grant No. 25ZR1401130 and No. 24ZR1422400), the National Natural Science Foundation of China (Grant No. 62402306), and the Open Research Project of the State Key Laboratory of Industrial Control Technology, China (Grant No. ICT2024B72).

References

Avetisyan, A.; Dahnert, M.; Dai, A.; Savva, M.; Chang, A. X.; and Nießner, M. 2019. Scan2cad: Learning cad model alignment in rgb-d scans. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, 2614–2623.

Çelen, A.; Han, G.; Schindler, K.; Van Gool, L.; Armeni, I.; Obukhov, A.; and Wang, X. 2024. I-design: Personalized llm interior designer. In *European Conference on Computer Vision*, 217–234. Springer.

Chung, J.; Lee, S.; Nam, H.; Lee, J.; and Lee, K. M. 2023. Luciddreamer: Domain-free generation of 3d gaussian splatting scenes. *arXiv preprint arXiv:2311.13384*.

Dai, A.; Chang, A. X.; Savva, M.; Halber, M.; Funkhouser, T.; and Nießner, M. 2017. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5828–5839.

Deitke, M.; Schwenk, D.; Salvador, J.; Weihs, L.; Michel, O.; VanderBilt, E.; Schmidt, L.; Ehsani, K.; Kembhavi, A.; and Farhadi, A. 2023. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 13142–13153.

Enscape GmbH. 2024. Enscape. Available at: https://enscape3d.com/.

Fang, C.; Dong, Y.; Luo, K.; Hu, X.; Shrestha, R.; and Tan, P. 2023. Ctrl-room: Controllable text-to-3d room meshes generation with layout constraints. *arXiv preprint arXiv:2310.03602*.

Feng, W.; Zhu, W.; Fu, T.-j.; Jampani, V.; Akula, A.; He, X.; Basu, S.; Wang, X. E.; and Wang, W. Y. 2023. Layoutgpt: Compositional visual planning and generation with large language models. *Advances in Neural Information Processing Systems*, 36: 18225–18250.

Fridman, R.; Abecasis, A.; Kasten, Y.; and Dekel, T. 2023. Scenescape: Text-driven consistent scene generation. *Advances in Neural Information Processing Systems*, 36: 39897–39914.

Fu, H.; Cai, B.; Gao, L.; Zhang, L.-X.; Wang, J.; Li, C.; Zeng, Q.; Sun, C.; Jia, R.; Zhao, B.; et al. 2021a. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10933–10942.

Fu, H.; Jia, R.; Gao, L.; Gong, M.; Zhao, B.; Maybank, S.; and Tao, D. 2021b. 3d-future: 3d furniture shape with texture. *International Journal of Computer Vision*, 129(12): 3313–3337.

Fu, R.; Wen, Z.; Liu, Z.; and Sridhar, S. 2024. Anyhome: Open-vocabulary generation of structured and textured 3d homes. In *European Conference on Computer Vision*, 52–70. Springer.

Gao, G.; Liu, W.; Chen, A.; Geiger, A.; and Schölkopf, B. 2024. Graphdreamer: Compositional 3d scene synthesis from scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 21295–21304.

Höllein, L.; Cao, A.; Owens, A.; Johnson, J.; and Nießner, M. 2023. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7909–7920.

Hurst, A.; Lerer, A.; Goucher, A. P.; Perelman, A.; Ramesh, A.; Clark, A.; Ostrow, A.; Welihinda, A.; Hayes, A.; Radford, A.; et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Inc., A. 2024a. Adobe Stock. https://stock.adobe.com/3d-assets. Accessed: 2025-07-24.

Inc., E. G. 2024b. Unreal Engine Marketplace. https://www.unrealengine.com/marketplace/en-US/store. Accessed: 2025-07-24.

Kerbl, B.; Kopanas, G.; Leimkühler, T.; and Drettakis, G. 2023. 3D Gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4): 139–1.

Li, H.; Shi, H.; Zhang, W.; Wu, W.; Liao, Y.; Wang, L.; Lee, L.-h.; and Zhou, P. Y. 2024. Dreamscene: 3d gaussian-based text-to-3d scene generation via formation pattern sampling. In *European Conference on Computer Vision*, 214–230. Springer.

Lin, C.; and Mu, Y. 2024. Instructscene: Instruction-driven 3d indoor scene synthesis with semantic graph prior. *arXiv preprint arXiv:2402.04717*.

Littlefair, G.; Dutt, N. S.; and Mitra, N. J. 2025. FlairGPT: Repurposing LLMs for interior designs. In *Computer Graphics Forum*, e70036. Wiley Online Library.

Merrell, P.; Schkufza, E.; Li, Z.; Agrawala, M.; and Koltun, V. 2011. Interactive furniture layout using interior design guidelines. *ACM transactions on graphics (TOG)*, 30(4): 1–10.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1): 99–106.

Raistrick, A.; Lipson, L.; Ma, Z.; Mei, L.; Wang, M.; Zuo, Y.; Kayan, K.; Wen, H.; Han, B.; Wang, Y.; et al. 2023. Infinite photorealistic worlds using procedural generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12630–12641.

Raistrick, A.; Mei, L.; Kayan, K.; Yan, D.; Zuo, Y.; Han, B.; Wen, H.; Parakh, M.; Alexandropoulos, S.; Lipson, L.; et al. 2024. Infinigen indoors: Photorealistic indoor scenes using procedural generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 21783–21794.

Reimers, N.; and Gurevych, I. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Tang, J.; Nie, Y.; Markhasin, L.; Dai, A.; Thies, J.; and Nießner, M. 2024. Diffuscene: Denoising diffusion models for generative indoor scene synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 20507–20518.

Trimble Inc. 2024. SketchUp. Available at: https://www.sketchup.com/en.

UE4Arch. 2024. UE4Arch. https://ue4arch.com/. Accessed: 2025-07-24.

Weiss, T.; Litteneker, A.; Duncan, N.; Nakada, M.; Jiang, C.; Yu, L.-F.; and Terzopoulos, D. 2018. Fast and scalable position-based layout synthesis. *IEEE Transactions on Visualization and Computer Graphics*, 25(12): 3231–3243.

Wu, Q.; Bansal, G.; Zhang, J.; Wu, Y.; Li, B.; Zhu, E.; Jiang, L.; Zhang, X.; Zhang, S.; Liu, J.; et al. 2024a. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*.

Wu, Q.; Bansal, G.; Zhang, J.; Wu, Y.; Zhang, S.; Zhu, E.; Li, B.; Jiang, L.; Zhang, X.; and Wang, C. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 3(4).

Wu, T.; Yang, G.; Li, Z.; Zhang, K.; Liu, Z.; Guibas, L.; Lin, D.; and Wetzstein, G. 2024b. Gpt-4v (ision) is a human-aligned evaluator for text-to-3d generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 22227–22238.

Wu, Z.; Li, Y.; Yan, H.; Shang, T.; Sun, W.; Wang, S.; Cui, R.; Liu, W.; Sato, H.; Li, H.; et al. 2024c. Blockfusion: Expandable 3d scene generation using latent tri-plane extrapolation. *ACM Transactions on Graphics (ToG)*, 43(4): 1–17.

Yang, Y.; Sun, F.-Y.; Weihs, L.; VanderBilt, E.; Herrasti, A.; Han, W.; Wu, J.; Haber, N.; Krishna, R.; Liu, L.; et al. 2024. Holodeck: Language guided generation of 3D embodied AI environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16227–16237.

Yeh, Y.-T.; Yang, L.; Watson, M.; Goodman, N. D.; and Hanrahan, P. 2012. Synthesizing open worlds with constraints using locally annealed reversible jump mcmc. *ACM Transactions on Graphics (TOG)*, 31(4): 1–11.

Zhai, G.; Örnek, E. P.; Chen, D. Z.; Liao, R.; Di, Y.; Navab, N.; Tombari, F.; and Busam, B. 2024. Echoscene: Indoor scene generation via information echo over scene graph diffusion. In *European Conference on Computer Vision*, 167–184. Springer.

Zhai, G.; Örnek, E. P.; Wu, S.-C.; Di, Y.; Tombari, F.; Navab, N.; and Busam, B. 2023. Commonscenes: Generating commonsense 3d indoor scenes with scene graph diffusion. *Advances in Neural Information Processing Systems*, 36: 30026–30038.