

D2GS: A Data-Centric Framework for LLM-based Creative Game Script Generation

Jiali Wang
Tencent
ShenZhen, China
friggawang@tencent.com

Junbao Huang
Tencent
ShenZhen, China
gumbohuang@tencent.com

Minghao Chen*
Tencent
ShenZhen, China
monychen@tencent.com

Abstract

Although Large Language Models (LLMs) offer significant advantages for creative tasks like game narrative design, their application remains largely confined to academic exploration. Existing methods often fail to integrate with specific game assets, production constraints, and iterative workflows essential for industrial development, limiting real deployment in professional studios. To bridge this gap, we present D2GS, a data-centric framework for LLM-based game script generation designed for industry. D2GS supports narrative designers end-to-end, from ideation to engine-ready scripts. It grounds creation in multi-source, game-specific data and mimics human iterative refinement. The framework includes four modules: (1) knowledge augmentation for multi-source data preparation; (2) logline and plot outline generation with a critic-feedback revision loop; (3) character and world enrichment using role-assigned LLM agents to simulate interactions; and (4) script rephrasing to ensure game-engine compatibility. Experiments show D2GS outperforms strong baselines in coherence, relevance, creativity, and overall writing quality, while staying faithful to predefined settings and assets. Industrial-scale validation further confirms its practicality for real-world narrative design pipelines.

1 Introduction

Game script generation remains a labor-intensive, expertise-heavy pipeline. Studios still rely on humans to ideate storylines, draft plots, revise dialog, and then align the final script with concrete game assets before it can run in-engine, shuttling between creative documents and production tools to reconcile character arcs with branching logic and wire text to sprites, backgrounds, animations, and audio cues. A game script is more than dialog: it interleaves conversation and narration with

expressions, camera directions, and fine-grained event commands that bind text to assets. Unlike conventional screenplays written for human production, game scripts must be executable by the engine, which makes them a central development artifact. Large language models (LLMs) can accelerate ideation and drafting (Li et al., 2024; Tian et al., 2024), but industrial-grade game script generation remains challenging. First, generation must be grounded in game-specific information scattered across heterogeneous sources (design docs, tables, images, legacy scripts, asset catalogs) whose integration and normalization are nontrivial (Hu et al., 2024; Jeong and Lee, 2025). Second, most prior work covers only fragments of the pipeline, e.g., story generation from simplified premises or script writing without upstream planning or downstream asset configuration (Ran et al., 2025; Yang et al., 2023), and rarely models game-specific characters and worldview explicitly, instead inheriting settings from novels (Lin et al., 2025; Ran et al., 2025; Chen et al., 2024) or inventing them freely (Yang et al., 2023, 2022), which misaligns outputs with real game constraints and forces manual rework. Finally, direct generation or unconstrained role-play tends to produce inconsistency, bland dialog, and violations of character and world rules. To address these gaps, we present D2GS, a data-centric framework that turns creative intent into engine-ready game scripts. Our contributions are:

- We propose the first framework that grounds LLM-based script generation in multi-source, game-specific data, turning professional planning inputs (genre, themes, characters) into loglines, plot outlines, structured simulation records, and finally executable scripts.
- We design a role-driven multi-agent pipeline in which a world agent and multiple character agents draft scenes under explicit narrative constraints; the transcripts are rewritten into

* Corresponding author.

polished scripts with natural dialog and coherent beats that adhere to character motivations and world rules.

- Experiments on industrial data from a commercial game show that D2GS consistently outperforms strong baselines. The verdicts are reproduced by two additional cross-family LLM judges and by a blinded panel of professional narrative designers, and transfer to BookWorld’s public benchmark under its own human-calibrated protocol.
- We report deployment evidence rarely available in academic work: as an internal authoring tool in a commercial game’s production workflow, D2GS attains 99.73% frame-event command legality and 93.5% automatic asset-reference resolution at ~ 0.5 h of post-editing per chapter.

2 Related Work

2.1 Automated Script Generation

LLMs increasingly assist in crafting stories and screenplays (Mirowski et al., 2023; Chakrabarty et al., 2024) via two paradigms. Top-down methods plan-and-write, expanding high-level artifacts (loglines, outlines, scene sketches) into full scripts through hierarchical prompting or multi-agent planner/reviewer/writer setups (Wan et al., 2025; Yang et al., 2023, 2022). Bottom-up methods simulate character interactions into transcripts later rewritten into screenplays, yielding vivid dialog but needing stronger consistency control; hybrids combine both (Chen et al., 2024; Ran et al., 2025). Most efforts, however, operate offline without conditioning on concrete game assets.

2.2 LLM-based Game Narrative

LLM-based game narrative splits by whether players shape generation at runtime. Offline generation authors outlines, scenes, and branching scripts in advance through schema-controlled generation, outline-to-script pipelines, multi-agent collaboration, and post-generation validation (Hu et al., 2024; Huot et al., 2025; Ran et al., 2025), at the cost of weak reactivity and combinatorially many branches to maintain. Player-driven emergent generation instead adapts the plot to player actions in real time (Wu et al., 2024; Han et al., 2024; Phillips et al., 2025; Jeong and Lee, 2025; Peng et al., 2024; Wu et al., 2025), typically via planner-controller

or director architectures with retrieval-augmented memory and state tracking, but faces latency, long-horizon coherence, safety moderation, and stable grounding of live text to the available assets.

2.3 Multi-agent System

Multi-agent systems are widely used to simulate social dynamics and study emergent behavior in controlled environments, from small-scale simulations supporting fine-grained analysis (Chuang et al., 2024) and sandbox generative agents with realistic human-agent interaction (Park et al., 2023) to nearly a thousand personality-distinct agents in a shared virtual world (AL et al., 2024) and large-scale social-network simulations of information diffusion and network formation (Yang et al., 2024).

3 Methodology

In this section, we first elaborate on our proposed data-centric framework for game script generation, namely D2GS. Given a complex script generation task, we generate the output $\mathcal{T}$ by decomposing the game script generation process into multiple subtasks that are tackled by specialized agents. The overall pipeline is illustrated in Fig. 1.

3.1 Multi-source Game Knowledge Integration and Representation

Game projects produce heterogeneous narrative assets—character design documents, worldview documents, and legacy scripts of events already played in the game—that vary widely in format, granularity, and intended use. To make them available for downstream narrative reasoning and retrieval, we unify these inputs into two complementary repositories: a character-centric knowledge base capturing identity, attributes, and interpersonal relations, and a worldview knowledge base encoding locations, domain-specific terminology, and the historical evolution of the game world.

The pipeline begins with ingestion and normalization: all inputs become plain text, PDFs through OCR and layout-aware parsing, tabular data linearized while retaining cell provenance, and every fragment tagged with metadata (document type, version, timestamps) for traceability. Documents are then chunked at semantic boundaries with a sliding overlap window that mitigates pronoun drift and cross-paragraph dependencies, chunk sizes trading context sufficiency against extraction precision. The LLM runs in a constrained extraction

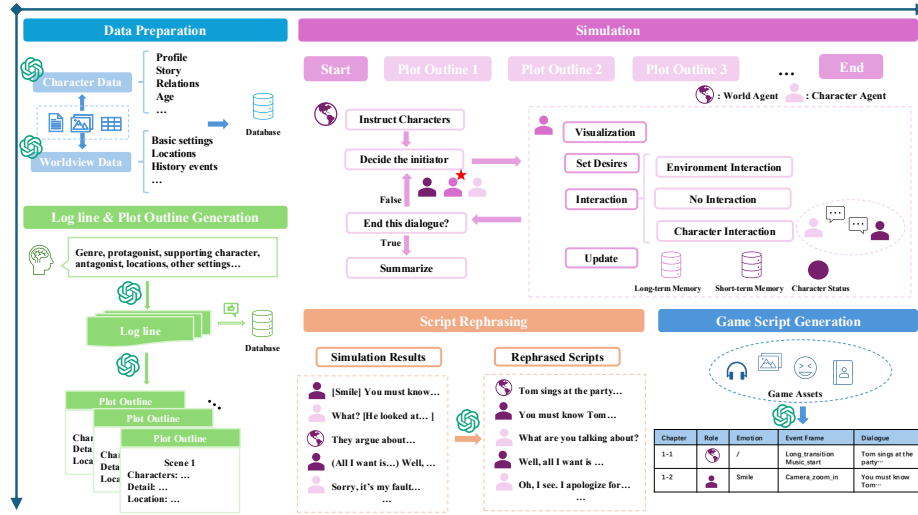


Figure 1: The complete paradigm of the proposed data-centric framework for LLM-based creative game script generation.

mode emitting schema-conforming JSON into the knowledge database; inter-character relations are extracted from the same text. Worldview data are partitioned into a historical database (events, eras, causal and temporal edges) and a terminology database (terms, locations, definitions), with locations in a containment hierarchy enabling spatial reasoning and scoped retrieval. Schema and integrity constraints keep pace with evolving narratives while preserving provenance and reliability for production use.

3.2 Logline and Plot Outline Generation with Critic-Feedback Revision

We regard story ideation as a two-stage workflow. In the first stage the narrative designer sets the genre, the characters (protagonists, supporting characters, antagonists), and the primary location. The system treats these as structured constraints and prompts an LLM to generate M candidate loglines $\mathcal{L} = \{L_1, \dots, L_M\}$, each a one- or two-sentence summary of the main plot and key elements, which the designer may edit. To reduce the writer’s reading burden, the LLM also scores every candidate along four dimensions—coherence, relevance, creativity, and writing quality—and loglines marked good are persisted in a logline database for few-shot retrieval. In the second stage we expand the chosen logline L_g into an N -chapter plot outline $\mathcal{C} = \{C_1, \dots, C_N\}$ structured by a narrative framework (Gustav Freytag’s pyramid), where each chapter instantiates a beat and each entry carries a short synopsis plus the required characters

and locations. Accepted outlines are persisted in a plot-outline database as in-domain exemplars for retrieval-augmented few-shot guidance, which matters for reproducibility, governance, and large-scale deployment.

Critic-feedback revision. Both stages share one *critic-conditioned revision* loop. Prompted as a senior narrative editor, the critic returns ~ 300 words of itemized notes on five aspects: fit of the characterization to the established storyline, plausibility and depth of the character relationships, audience appeal, logical soundness, and any further issue it finds; at the outline level it additionally applies a three-act and hero’s-journey lens. The reviser regenerates the draft from the original plus these notes under the *same* schema, leaving downstream interfaces unchanged. We use a single round ($R_0 \rightarrow R_1$), which already suffices (Sec. 4.2.2); prompts and a before/after case are in Appendix C.

3.3 Multi-agent simulation

In this subsection, we introduce the simulation process based on the generated plot outlines. The simulation process is orchestrated by a world agent and a group of character agents selected in a specific way. The combination of narrative intent and character agents interactions enables the simulation to advance plot goals without degenerating into unconstrained free play. The algorithm is illustrated in Appendix A.

Initialization. Character agents $\mathcal{R}$ and a world agent are initialized with the multi-source knowl-

edge of Sec. 3.1. Each character agent holds a long-term memory (backstories, relationships, goals) for behavioral consistency and a short-term memory (recent interactions, current objectives, environmental changes) for dynamic reactions. The world agent, initialized with the game environment (geography, rules, worldview), keeps the world coherent, mediates character–environment interaction, and propagates events by updating the world state after each action. The simulation text $\mathcal{T}_{sim}$ starts empty.

Simulation Process. Each chapter $C_k \in \mathcal{C}$ is simulated in turn under the guidance of the outline. Its content is loaded first—scene description, the characters present $\mathcal{R}_k$, and related details—after which every agent $r \in \mathcal{R}_k$ visualizes the scene from C_k and the global state $\mathcal{S}$ (what can be seen, sensed, and smelled, plus its current status), yielding $\mathcal{V}_r$, and determines its extrinsic and intrinsic desires $\mathcal{D}_r$ from $\mathcal{V}_r$, the logline, and C_k . Visualization and desire-setting let characters decide from the current situation rather than from generic roleplay. The world agent then compiles for each character an action guide $\mathcal{G}_r = \{g_{r,1}, g_{r,2}, \dots\}$ fulfilling $\mathcal{D}_r$ under the constraints of $\mathcal{S}$, $\mathcal{V}_r$, and C_k , and selects the first character to act. At each turn the acting character executes the next step of its guide and picks dialog targets; its action is appended to $\mathcal{T}_{sim}$, and every target reacts according to its own visualization, desires, and plan, those reactions (dialog or movement) likewise being appended. Once the chapter is complete, a summary is appended and $\mathcal{S}$ is updated with the chapter’s locations and environmental cues.

Cost and scalability. Three mechanisms bound simulation cost: the active cast of a chapter is exactly the outline’s $\mathcal{R}_k$ (typically 2–5), with visualization, desire-setting, and action-guide compilation each run once per chapter rather than per turn; the world agent applies dialog- and chapter-level termination checks; and loglines and chapters are independent, hence scheduled concurrently. Over the 55 GPT-5 production runs a script averages 206 interaction records (min 64, max 453), roughly 320–350 LLM calls for 6–10 chapters—well within tolerance for *offline* authoring on multi-week version-event cycles.

3.4 Script Rephrasing

Script rephrasing transforms structured simulation outputs into coherent, human-readable game scripts. Its input is the simulation records—speaker identifier, semantic content, action, and emotional

context—serialized into a schema-aligned prompt so that the LLM receives both factual content and contextual metadata. The LLM then emits dialog lines in a standardized format, using environmental cues and emotional states to adjust lexical choice and tone while preserving the original narrative intent, so that dialog carries plot while revealing personality and subtext.

3.5 Game Script Generation with frame events

We finally integrate the narrative script with the game’s multimedia assets—character sprites, scene backgrounds, music tracks, transition animations, and facial expressions—to produce engine-ready, frame-event-annotated scripts, associating each dialog line and scene with the corresponding pre-existing assets so that the narrative can be performed in the game’s script editor. Guided by the frame-event annotation rules of the target engine, we prompt LLMs to emit enriched scripts in which every narrative unit is paired with the appropriate visual, auditory, and transitional elements, preserving narrative fidelity while conforming to the technical specification required for direct import.

4 Experiments

4.1 Experiment Setup

4.1.1 Dataset

We build a dataset spanning 6 genres—romance, comedy, horror, science fiction, drama, and crime (Chen et al., 2024). Characters, antagonists, worldviews, and character relationships are extracted from an existing commercial game, whose character and worldview data arrive as documents, images, and tables. For each genre we use LLMs to generate 10 unique loglines, reflecting that genre’s thematic and narrative conventions, as the initial input. All source documents and generated scripts are in Chinese. The distilled knowledge bases hold 23 character profiles, 73 relation pairs, 175 locations, and 404 terminology and history entries; on a hand-verified sample of 150 records, extraction reaches 87.3% strict and 94.6% partial-inclusive accuracy (Appendix G). Across 6 genres, 60 loglines, 5 base-lines, and 4 backbones we analyse ~880 completed full-script pairwise comparisons on real production assets—a scope we accordingly call *validated on industrial data* rather than industrial-scale validation.

Table 1: Win rate (%) of D2GS against each baseline. **Avg.** is the unweighted mean of the five higher-is-better dimensions (Misc. Problem excluded, being lower-is-better); the bottom block macro-averages each baseline over the four backbones, giving an overall ranking.

Model	Method	Coherence $\uparrow$	Relevance $\uparrow$	Interestingness $\uparrow$	Humanlike $\uparrow$	Writing Quality $\uparrow$	Avg. $\uparrow$	Misc. Problem $\downarrow$
GPT-5	Direct prompt	92.22	72.60	89.90	98.18	70.87	84.75	17.47
	Plan-then-write	89.50	79.56	76.43	90.01	80.00	83.10	19.09
	DOC-screen	80.01	70.90	75.32	82.98	76.91	77.22	13.70
	HoLLMWood	62.72	62.09	72.72	74.54	81.81	70.78	31.53
	BookWorld	59.88	60.44	75.00	72.44	61.11	65.77	39.25
GPT-4o	Direct prompt	80.87	74.34	78.43	86.52	73.91	78.81	13.33
	Plan-then-write	73.78	74.78	83.47	83.47	75.65	78.23	15.00
	DOC-screen	80.00	88.88	91.73	90.00	97.77	89.68	28.98
	HoLLMWood	61.27	60.87	76.52	85.21	63.04	69.38	31.66
	BookWorld	54.28	51.00	71.42	74.28	60.71	62.34	40.92
DS-r1	Direct prompt	96.29	85.18	81.85	85.18	80.40	85.78	9.13
	Plan-then-write	85.05	96.29	90.00	85.00	79.00	87.07	14.34
	DOC-screen	96.29	88.78	86.24	92.29	86.29	89.98	21.74
	HoLLMWood	84.61	83.07	65.38	70.76	74.61	75.69	30.43
	BookWorld	68.75	58.00	56.25	62.50	62.50	61.60	45.00
DS-v3	Direct prompt	95.00	81.66	41.66	78.33	82.50	75.83	11.81
	Plan-then-write	79.04	96.65	91.67	96.87	76.40	88.13	16.89
	DOC-screen	98.32	98.53	98.31	80.05	65.90	88.22	18.91
	HoLLMWood	63.93	68.33	51.67	58.03	68.45	62.08	16.34
	BookWorld	62.00	59.00	80.00	86.00	82.00	73.80	57.78
Macro-avg.	Direct prompt	91.09	78.44	72.96	87.05	76.92	81.29	12.94
	Plan-then-write	81.84	86.82	85.39	88.84	77.76	84.13	16.33
	DOC-screen	88.66	86.77	87.90	86.33	81.72	86.28	20.83
	HoLLMWood	68.13	68.59	66.57	72.13	71.98	69.48	27.49
	BookWorld	61.23	57.11	70.67	73.81	66.58	65.88	45.74

4.1.2 Evaluation Metrics

Script quality requires nuanced judgment across several dimensions, so we adopt the pairwise comparison framework of prior work (Chen et al., 2024; Yang et al., 2023), in which an LLM decides which of two scripts is better; the approach has been validated against human evaluation (Ran et al., 2025; Chen et al., 2024; Yang et al., 2023; Huot et al., 2025). Our metrics are as follows. **Coherence**: Coherence of plot structure, character description, scene transitions, and setup consistency. **Relevance**: Faithful to the genre and logline. **Creativity**: The originality and unexpectedness of plots (for logline generation). **Interestingness**: Interesting to the reader (for script generation). **Humanlike**: Judged to be human-written. **Writing Quality**: Assess writing quality. **Misc. Problem** (Yang et al., 2022): Writing issues, including narration, inconsistent, confusing, repetitive, and disfluent. Judges are prompted in Chinese, matching the scripts’ language, with the English metric definitions above embedded verbatim (Appendix D). LLM judging covers narrative quality only; frame-event legality and asset constraints are checked by a rule-based validator (Sec. 4.3), and Sec. 4.2.3 establishes reliability for this setting rather than borrowing figures from other domains.

Table 2: Comparison of loglines and outlines under one round of critic-feedback revision.

Content	Method	Coherence $\uparrow$	Relevance $\uparrow$	Creativity $\uparrow$	Writing Quality $\uparrow$
Logline	R1 Wins	92.13	89.30	92.92	91.10
	R0 Wins	2.34	4.13	3.90	3.28
	Ties	5.53	6.57	3.18	5.62
Outline	R1 Wins	93.10	85.59	96.55	98.28
	R0 Wins	5.17	8.69	1.72	0.00
	Ties	1.73	5.72	1.73	1.72

4.1.3 Baselines and models

In our study, we compare scripts generated using six different methods to assess their effectiveness in producing high-quality screenplays. The methods includes **Direct prompt**, **Plan-then-write**, **DOC-screen** (Yang et al., 2023), **HoLLMWood** (Chen et al., 2024), **BookWorld** with script instruction (Ran et al., 2025), and **D2GS** in Sec. 3. For models, we adopt multiple state-of-the-art LLMs, including GPT-5, GPT-4o, DS (Deepseek)-v3, and DS (Deepseek)-r1. We utilize GPT-5 for the evaluation, and re-run the complete evaluation with two further judges in Sec. 4.2.3.

Input parity. Since D2GS is data-centric, one may ask whether its win rates merely reflect richer inputs. We audited every baseline against the experimental code (Appendix F). All six methods start from the same 60 detailed loglines, produced from the full character, relation, and worldview data, and every baseline whose design accepts explicit character or world inputs receives them through

the same loader and files as D2GS; BookWorld is mounted on the same worldview knowledge base and retrieves from it during generation. One asymmetry remains, which we state rather than leave implicit: structured character relations are consumed only by D2GS — the contribution of Sec. 3.1 rather than an evaluation artifact, whose worth Sec. 4.2.5 measures.

4.2 Evaluation Results

4.2.1 Effect of Script Generation

Table 1 reports the win rates of D2GS against the baselines across six metrics. D2GS consistently outperforms direct prompting on every metric and every model. Against Plan-then-write and DOC-screen it keeps a clear edge in humanlike and interestingness, writing quality being its narrowest margin over both; against HoLLMWood and BookWorld its clearest advantage is again humanlike text. Coherence and relevance are closer and depend on the backbone, since multi-agent simulation demands stronger structured-output and instruction-following ability than direct generation. On Misc. Problem, D2GS makes fewer errors in every pairing but one. Output length, finally, is governed mainly by the backbone rather than the method, and since longer text does not imply higher quality we defer the word-count statistics to Appendix B. The macro-average block of Table 1 gives the overall ranking: averaged over backbones and the five higher-is-better dimensions, D2GS beats every baseline, from 65.88% against the strongest competitor (BookWorld) to 86.28% against DOC-screen, while producing fewer miscellaneous writing problems than all of them. The ordering is stable across backbones, the two simulation-based systems being hardest to beat.

4.2.2 Effect of critic-feedback revision for logline and plot outline generation

We compare loglines and outlines produced at round 1 (R1) against round 0 (R0). As Table 2 shows, revised content overwhelmingly beats unrevised content. Notably the gain is not confined to coherence and relevance: creativity and writing quality improve as well, indicating that critic-feedback revision also facilitates the downstream script development.

4.2.3 Reliability of the Evaluation Protocol

Using GPT-5 both as generator and as judge invites a self-preference concern, and human-LLM

agreement reported elsewhere does not establish reliability for Chinese game scripts written under real character and world constraints.¹

Multiple judges. Re-running all ~ 880 comparisons with two further judges from other model families gives verdict agreement of 90.4% (Cohen’s $\kappa=0.614$) with Claude-Sonnet-4.5, 79.4% ($\kappa=0.471$) with DeepSeek-V4-Flash, and 78.9% ($\kappa=0.440$) between the two new judges; per-dimension agreement is 73.0–88.5% with the former and 67.9–79.7% with the latter. No conclusion in Table 1 is reversed by changing the judge or the judge family.

Human evaluation. Three professional narrative designers judged 30 blinded pairs in randomized order, pitting D2GS against BookWorld and HoLLMWood. Human preference for D2GS is 67.8%, inter-annotator agreement 43.3–70.0%, and human-LLM agreement 69.2%, so the judge agrees with the designers about as often as they agree with each other (Appendix E). Deleting all length-related phrasing from the judge prompt leaves 92% of verdicts unchanged, so the win rates are not an artifact of verbosity.

4.2.4 Cross-Dataset Validation on Public BookWorld Benchmark

Table 3: Pairwise comparison (%) of D2GS against BookWorld on the public *A Song of Ice and Fire* benchmark ($N=6$ scenarios), under BookWorld’s own five-dimension protocol with GPT-5 as judge.

Result	An $\uparrow$	CF $\uparrow$	IS $\uparrow$	WQ $\uparrow$	SQ $\uparrow$	Avg $\uparrow$
D2GS Wins	83.33	50.00	100.00	100.00	50.00	76.67
BookWorld Wins	16.67	50.00	0.00	0.00	50.00	23.33

To validate our evaluation beyond proprietary game data, we compare D2GS against BookWorld on its own public benchmark: both methods take the same loglines from the publicly available *A Song of Ice and Fire* setting and share the same DS-v3 actor, differing only in their generation pipeline, and are judged by GPT-5 under BookWorld’s original five-dimension protocol—Anthropomorphism (An), Character Fidelity (CF), Immersion & Setting (IS), Writing Quality (WQ), and Storyline Quality (SQ). Although limited to $N=6$ scenarios, each script is a full multi-chapter storyline, the two methods together producing 2,945 turns the judge reads

¹We release the evaluation code, the judge and critic prompts, and the validator at <https://anonymous.4open.science/r/d2gs-2027>; the industrial dataset cannot be released, being covered by commercial IP.

in full. As Table 3 shows, D2GS attains a 76.67% average win rate, with the clearest margins on immersion/setting and writing quality and parity on character fidelity and storyline quality. Since this protocol was calibrated against a five-annotator human panel in the original study (Cohen’s $\kappa \geq 0.64$), the result rests on a human-validated metric rather than an ad-hoc LLM judge; together with Table 1, it indicates that the advantage of D2GS generalizes across different protocols and datasets.

4.2.5 Ablating the Knowledge Base and the Simulation

Table 4: Rule-based, LLM-free measurements. **Top:** grounding ablation (GPT-5, 30 loglines), counting distinct canonical archive entities per script. **Bottom:** frame-event validation over 10 deployed authoring sessions.

Grounding ablation	Entities / script [†]
w/o knowledge base, w/o simulation	3.8
w/ knowledge base, w/o simulation	5.6
Full D2GS (knowledge base + simulation)	11.8
Engine-readiness	Result
Frame-event command legality	99.73% (749/751)
Automatic asset-reference resolution	93.5% (545/583)
Expression validity	87.56%

Table 2 already isolates the critic-feedback module. For the other two we measure grounding *objectively*, string-matching generated scripts against the knowledge base, so the ablation inherits none of the judge’s bias. Removing character profiles and relations drops the no-simulation variant from 5.6 to 3.8 distinct canonical setting entities per script (−32%, Table 4), and mentions of profile-exclusive facts — e.g., the protagonist’s mercenary backstory, present only in the character archive — fall from 46 to exactly 0: the knowledge base is the sole channel through which archive-level settings reach the scripts. Adding the simulation, whose agents retrieve during generation, raises the count to 11.8, $2.1\times$ the no-simulation variant. Grounding is not free: direct generation keeps an edge in the free-flowing fluency LLM judges reward, but cannot meet production’s grounding and asset-executability requirements — hence the full framework.

4.3 Deployment and Engine-Readiness

“Engine-ready” deserves evidence rather than assertion. D2GS runs as an internal web tool in a commercial game’s production workflow: designers upload the script breakdown and asset sheets,

and the tool returns frame-event-annotated scripts exported back into the pipeline. It has been in production since August 2025 across roughly six version events, one every two months, and every designer-reviewed output has shipped in the live game.

Frame events come from a closed instruction set and every asset name must occur in the production resource sheets, so correctness is decidable without an LLM. A rule-based validator over 10 authoring sessions reports 99.73% command legality (749/751), 93.5% automatic asset-reference resolution (545/583), and 87.56% expression validity (Table 4); table-format import never fails, by construction. Unresolved references are flagged with their row number and fixed during designer review, costing review time rather than breaking imports. Post-editing averages ~ 0.5 h per chapter, about four hours per version-event script; the studio internally estimates a 25–50% efficiency gain, an internal estimate rather than a controlled measurement (Appendix H).

5 Conclusion

This paper introduced D2GS, a data-centric framework for LLM-based game script generation aimed at real-world narrative design pipelines. Unlike prior text-only approaches, D2GS grounds generation in game-specific assets and production constraints, carrying the workflow from ideation to engine-ready scripts: it generates a logline, constructs a plot outline, and runs role-playing simulations with character agents coordinated by a world agent. Extensive experiments show consistent improvements over existing methods under three independent LLM judges and a blinded panel of professional narrative designers, and a deployed authoring tool turns these into production value: 99.73% frame-event command legality, 93.5% asset-reference resolution, and about half an hour of post-editing per chapter in a commercial game shipping D2GS-assisted scripts since August 2025.

Limitations

Scope of the industrial evidence. All proprietary experiments come from a single commercial game. The evaluation spans 6 genres, 60 loglines, 5 base-lines, and 4 backbones (~ 880 full-script pairwise comparisons) over production assets of 23 character profiles, 73 relation pairs, 175 locations, and

404 terminology and history entries, but one game world cannot establish transfer to other titles, art styles, or engines; this is why we claim validation *on industrial data* rather than industrial-scale validation. The public cross-check against BookWorld covers only $N=6$ scenarios of a single corpus, so broader validation across scenarios, genres, and corpora remains future work.

Scale of the human validation. The blinded study uses 3 professional narrative designers and 30 pairs, with 43.3–70.0% inter-annotator agreement. This suffices to show that the LLM judge tracks expert preference about as well as experts track each other, but not to resolve fine-grained differences among the stronger baselines; moreover, the designers work on the game the data come from, so they are expert but not fully independent raters.

The efficiency figure is an internal estimate. The 25–50% scripting-efficiency gain is the studio’s own assessment, not a controlled A/B study against unassisted authoring, and the ~ 0.5 h per chapter of post-editing is measured on designers already familiar with the tool. The frame-event legality and asset-resolution numbers, by contrast, come from a deterministic validator and are reproducible from the released code.

Reproducibility. We release the evaluation code, the judge and critic prompts, and the validator, but commercial IP and confidentiality prevent releasing the game data, so the proprietary half of the evaluation cannot be re-run externally; the public BookWorld comparison partly compensates.

Dependence on structured sources. Being data-centric, D2GS inherits the quality of its inputs: schema-constrained extraction assumes reasonably well-structured worldbuilding documents and reaches 87.3% strict accuracy on our sample, so noisier or sparser sources will degrade the knowledge base and, through it, the grounding of the scripts.

Inference cost. The simulation issues roughly 320–350 LLM calls per script, far more than single-pass prompting. This is acceptable for offline authoring on multi-week cycles but rules out the framework, as it stands, for latency-sensitive or runtime player-driven narrative.

Language and model coverage. Our experiments use four backbones and Chinese-language scenarios throughout; behavior under other languages, writing traditions, and model families remains to be characterized.

Ethical Considerations

Data. The proprietary material is fictional world-building documentation, character sheets, and asset catalogs owned by the studio, used with permission under confidentiality obligations. It contains no personal data and no user-generated content, which is also why it cannot be released. The public comparison uses the *A Song of Ice and Fire* setting distributed with BookWorld under its original terms.

Human participants. The blinded evaluation was performed by three in-house professional narrative designers as part of their ordinary compensated duties, on fiction they already work with; no personal data was collected and participation was voluntary.

Effect on creative labor. D2GS is an assistive, designer-in-the-loop tool rather than an autonomous writer: designers choose the logline, edit the outline, and review every script before it ships, and the reported efficiency gain comes from removing mechanical asset wiring and first-draft dialog, not authorship. We nonetheless acknowledge that such tools shift creative work from drafting toward editing, a shift that should be negotiated with the writers it affects.

Content risk. Generated scripts inherit the biases and failure modes of their backbone LLMs and may produce stereotyped characterization or content unsuitable for the game’s rating. In deployment every script passes both content-moderation and designer review before reaching players; we regard this human gate as a requirement, not an optional safeguard.

References

- Altera AL, Andrew Ahn, Nic Becker, Stephanie Carroll, Nico Christie, Manuel Cortes, Arda Demirci, Melissa Du, Frankie Li, Shuying Luo, Peter Y. Wang, Mathew Willows, Feitong Yang, and Guangyu Robert Yang. 2024. [Project Sid: Many-agent simulations toward AI civilization](#). *arXiv preprint arXiv:2411.00114*.
- Tuhin Chakrabarty, Vishakh Padmakumar, Faeze Brahman, and Smaranda Muresan. 2024. [Creativity support in the age of large language models: An empirical study involving professional writers](#). In *Proceedings of the 16th Conference on Creativity & Cognition*, pages 132–155. ACM.
- Jing Chen, Xinyu Zhu, Cheng Yang, Chufan Shi, Yadong Xi, Yuxiang Zhang, Junjie Wang, Jiashu Pu, Tian Feng, Yujiu Yang, and Rongsheng Zhang. 2024.

- HoLLMwood: Unleashing the creativity of large language models in screenwriting via role playing. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8075–8121, Miami, Florida, USA. Association for Computational Linguistics.
- Yun-Shiuan Chuang, Nikunj Harlalka, Siddharth Suresh, Agam Goyal, Robert Hawkins, Sijia Yang, Dhavan Shah, Junjie Hu, and Timothy T. Rogers. 2024. [The wisdom of partisan crowds: Comparing collective intelligence in humans and LLM-based agents](#). In *Proceedings of the 46th Annual Meeting of the Cognitive Science Society*, Rotterdam, The Netherlands. Cognitive Science Society.
- Senyu Han, Lu Chen, Li-Min Lin, Zhengshan Xu, and Kai Yu. 2024. [IBSEN: Director-actor agent collaboration for controllable and interactive drama script generation](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1607–1619, Bangkok, Thailand. Association for Computational Linguistics.
- Ziniu Hu, Ahmet Iscen, Aashi Jain, Thomas Kipf, Yisong Yue, David A. Ross, Cordelia Schmid, and Alireza Fathi. 2024. [SceneCraft: An LLM agent for synthesizing 3D scenes as blender code](#). In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 19252–19282. PMLR.
- Fantine Huot, Reinald Kim Amplayo, Jennimaria Palomaki, Alice Shoshana Jakobovits, Elizabeth Clark, and Mirella Lapata. 2025. [Agents’ room: Narrative generation through multi-step collaboration](#). In *The Thirteenth International Conference on Learning Representations*. OpenReview.net.
- Jin Jeong and Tak Yeon Lee. 2025. [LIGS: Developing an LLM-infused game system for emergent narrative](#). In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–12. ACM.
- Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024. [Pre-trained language models for text generation: A survey](#). *ACM Computing Surveys*, 56(9):1–39.
- Zefeng Lin, Yi Xiao, Zhiqiang Mo, Qifan Zhang, Jie Wang, Jiayang Chen, Jiajing Zhang, Hui Zhang, Zhengyi Liu, Xianyong Fang, and Xiaohua Xu. 2025. [R²: A LLM based novel-to-screenplay generation framework with causal plot graphs](#). *arXiv preprint arXiv:2503.15655*.
- Piotr Mirowski, Kory W. Mathewson, Jaylen Pittman, and Richard Evans. 2023. [Co-writing screenplays and theatre scripts with language models: Evaluation by industry professionals](#). In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–34. ACM.
- Joon Sung Park, Joseph O’Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. [Generative agents: Interactive simulacra of human behavior](#). In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22. ACM.
- Xiangyu Peng, Jessica Quaye, Sudha Rao, Weijia Xu, Portia Botchway, Chris Brockett, Nebojsa Jojic, Gabriel DesGarennes, Ken Lobb, Michael Xu, Jorge Leandro, Claire Jin, and Bill Dolan. 2024. [Player-driven emergence in LLM-driven game narrative](#). In *2024 IEEE Conference on Games (CoG)*, pages 1–8. IEEE.
- Adon Phillips, Jochen Lang, and David Mould. 2025. [Goal-oriented interactions in games using LLMs](#). *IEEE Transactions on Games*, 17(2):510–521.
- Yiting Ran, Xintao Wang, Tian Qiu, Jiaqing Liang, Yanghua Xiao, and Deqing Yang. 2025. [BookWorld: From novels to interactive agent societies for story creation](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15898–15912, Vienna, Austria. Association for Computational Linguistics.
- Yufei Tian, Tenghao Huang, Miri Liu, Derek Jiang, Alexander Spangher, Muhao Chen, Jonathan May, and Nanyun Peng. 2024. [Are large language models capable of generating human-level narratives?](#) In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17659–17681, Miami, Florida, USA. Association for Computational Linguistics.
- Kaiyang Wan, Honglin Mu, Rui Hao, Haoran Luo, Tianle Gu, and Xiuying Chen. 2025. [A cognitive writing perspective for constrained long-form text generation](#). In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 9832–9844, Vienna, Austria. Association for Computational Linguistics.
- Hongqiu Wu, Weiqi Wu, Tianyang Xu, Jiameng Zhang, and Hai Zhao. 2025. [Towards enhanced immersion and agency for LLM-based interactive drama](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11166–11182, Vienna, Austria. Association for Computational Linguistics.
- Weiqi Wu, Hongqiu Wu, Lai Jiang, Xingyuan Liu, Hai Zhao, and Min Zhang. 2024. [From role-play to drama-interaction: An LLM solution](#). In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3271–3290, Bangkok, Thailand. Association for Computational Linguistics.
- Kevin Yang, Dan Klein, Nanyun Peng, and Yuandong Tian. 2023. [DOC: Improving long story coherence with detailed outline control](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3378–3465, Toronto, Canada. Association for Computational Linguistics.

Kevin Yang, Yuandong Tian, Nanyun Peng, and Dan Klein. 2022. [Re3: Generating longer stories with recursive reprompting and revision](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4393–4479, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Ziyi Yang, Zaibin Zhang, Zirui Zheng, Yuxian Jiang, Ziyue Gan, Zhiyu Wang, Zijian Ling, Jinsong Chen, Martz Ma, Bowen Dong, Prateek Gupta, Shuyue Hu, Zhenfei Yin, Guohao Li, Xu Jia, Lijun Wang, Bernard Ghanem, Huchuan Lu, Chaochao Lu, and 4 others. 2024. [OASIS: Open agent social interaction simulations with one million agents](#). *arXiv preprint arXiv:2411.11581*.

A Outline-Guided Simulation Algorithm

Algorithm 1 provides the detailed procedure of our outline-guided simulation. Given the generated plot outline and the narrative intent for the current step, a world agent first selects and initializes a set of character agents under asset and role constraints, then orchestrates multi-round interactions among them. The world agent monitors progress and applies control signals to ensure the simulation advances plot goals rather than drifting into unconstrained free play. The algorithm outputs a simulation transcript and updated world/character states, which are used for subsequent rewriting into engine-ready scripts.

Notation. The plot outline is $\mathcal{C} = \{C_1, \dots, C_N\}$, where each chapter C_k packages a scene description, the set of participating characters $\mathcal{R}_k$, a location, and a short synopsis that serves as the local narrative goal. $\mathcal{S}$ is the evolving global state (locations, environmental cues, and what has happened so far); $\mathcal{V}_r$, $\mathcal{D}_r$, and $\mathcal{G}_r$ denote, respectively, the scene character r perceives, its determined desires, and the world-agent-issued action guide; and $\mathcal{T}_{sim}$ is the running simulation transcript.

The chapter as the unit of control. Rather than letting agents free-associate, each chapter C_k scopes *who* is on stage ($\mathcal{R}_k$), *where* the scene takes place, and *what* it must accomplish (its synopsis), so character behavior is constrained to advance that specific beat. Within a chapter, every present character first *visualizes* the scene ($\mathcal{V}_r$) and forms *desires* ($\mathcal{D}_r$) conditioned on the logline and C_k ; this grounds subsequent decisions in the current situation instead of generic role-play. The world agent then compiles per-character action guides $\mathcal{G}_r$ that translate these desires into concrete, goal-directed steps under the constraints of $\mathcal{S}$ and C_k .

Interaction loop and termination. The acting character executes the next step of its guide and selects dialog targets; the targeted characters react, and all utterances are appended to $\mathcal{T}_{sim}$. After each turn the world agent updates $\mathcal{S}$ and decides whether to continue, acting as a director that keeps the exchange on-beat and ends the chapter once its narrative goal is met—preventing the runaway or off-topic dialog typical of unconstrained multi-agent role-play. A chapter summary is then appended and the updated $\mathcal{S}$ is carried into the next chapter, preserving long-horizon consistency. Once all chapters are simulated, the full transcript $\mathcal{T}_{sim}$ is rephrased (Sec. 3) and annotated with frame events to yield the engine-ready script $\mathcal{T}$.

B Length Statistics of Generated Scripts

Table 5 reports the average number of words in scripts generated by different methods and backbone models under the same experimental setting. The word counts are computed by tokenizing the final generated script text and splitting by whitespace; thus, they reflect overall script verbosity rather than semantic quality. Note that different methods may impose different output formats (e.g., explicit scene boundaries, shot descriptions, or structured planning artifacts). Such formatting requirements can systematically increase word counts because they introduce additional descriptive or organizational text. Therefore, length should be interpreted primarily as a proxy for output volume and format overhead, not as a direct indicator of coherence, creativity, or factual correctness.

We also observe that model-inherent verbosity affects the statistics: larger/stronger models tend to generate longer responses given the same stopping criteria, which partially explains why GPT-5 produces substantially more words than the other backbones. Finally, since some pipelines are multi-stage and rely on intermediate planning or rewriting, they may amplify length through repeated elaboration; this can lead to higher variance across models even when the method name is the same.

C Critic-Feedback Prompts and a Before/After Case

The loop of Sec. 3.2 uses four prompts — a logline critic, a logline reviser, an outline critic, and an outline reviser. The critic and the generator share a backbone but are invoked independently, with

Algorithm 1: Framework of game script generation.

Input: Genre, characters, and locations.
Output: Game script $\mathcal{T}$.

Generate multiple loglines $\mathcal{L} = \{L_1, \dots, L_M\}$ based on the given genre, characters and locations and choose the best logline L_g ;
Generate the outline $\mathcal{C} = \{C_1, C_2, \dots, C_N\}$ with critic-feedback revision;
Initialize multiple character agents and a world agent with multi-source game knowledge and $\mathcal{T}_{sim}$ to be empty;

for each chapter $C_k \in \mathcal{C}$ **do**
 Load chapter content C_k (scene description, characters $\mathcal{R}_k$ in chapter C_k , etc.);
 for each character agent $r \in \mathcal{R}_k$ **do**
 $\mathcal{V}_r \leftarrow$ Visualize scene: r imagines based on C_k and $\mathcal{S}$, including what can be seen, sensed, smelled, and the current status;
 $\mathcal{D}_r \leftarrow$ Determine desires: r 's extrinsic and intrinsic desires based on $\mathcal{V}_r$, logline, and C_k ;
 The world agent generates a sequence of action guides $\mathcal{G}_r = \{g_{r,1}, g_{r,2}, \dots, g_{r,t}, \dots\}$ for each character r to achieve $\mathcal{D}_r$;
 The world agent selects initial acting character r ;
 while the world agent decides to continue the performance do
 $a_r, \mathcal{R}' \leftarrow$ Execute action: r performs the next step in $\mathcal{G}_r$ and selects dialog targets;
 Append a_r (description of r 's action a_r) to $\mathcal{T}_{sim}$;
 for each character agent $r' \in \mathcal{R}'$ **do**
 $a_{r'} \leftarrow$ React to a_r ;
 Append r' 's reaction (e.g., dialog, movement) to $\mathcal{T}_{sim}$;
 Update global state $\mathcal{S}$ based on the history text;
 The world agent selects next acting character r .
 Append chapter summary to $\mathcal{T}_{sim}$;
 Update global state $\mathcal{S}$ with the history text;
Rephrase the simulation text to meet the script quality;
Produce frame-event-annotated game scripts within game assets;
return $\mathcal{T}$

Table 5: The average length statistics of scripts generated by various methods.

Model	Method	Words
GPT-5	Direct prompt	2137
	Plan-then-write	11613
	DOC-screen	10669
	HoLLMWood	10496
	BookWorld	13506
	D2GS	14044
GPT-4o	Direct prompt	1251
	Plan-then-write	5736
	DOC-screen	5750
	HoLLMWood	4052
	BookWorld	7486
	D2GS	7512
DS-r1	Direct prompt	1112
	Plan-then-write	5890
	DOC-screen	6133
	HoLLMWood	3817
	BookWorld	9334
	D2GS	9002
DS-v3	Direct prompt	1261
	Plan-then-write	5599
	DOC-screen	5901
	HoLLMWood	2911
	BookWorld	8037
	D2GS	7747

no shared context. The main experiments use a single round ($R_0 \rightarrow R_1$). We reproduce all four prompts below, translated from the Chinese originals that are released verbatim with the evaluation code; $\{\dots\}$ marks a runtime slot.

The **critic** prompts cast the model as a senior narrative editor and require itemized, actionable output of about 300 words organized along five aspects: (a) consistency of the characterization with the established storyline; (b) plausibility, depth, and complexity of the character relationships; (c) audience appeal; (d) logical soundness; and (e) any further issue the critic identifies. The outline critic additionally requires an analysis through a three-act structure and a hero's-journey lens, checking that each character's desires remain consistent across chapters and actually drive the plot rather than being restated. The **reviser** prompts receive the original draft together with the critic's itemized notes and must regenerate it under the same structured schema as the first pass, so that no downstream component needs to know whether a revision took place.

Logline critic.

You are an editor, skilled at guiding screenwriters to create better stories in film storylines. Your responsibility is to provide constructive suggestions for improving the characters the screenwriter has shaped from the logline.

When providing feedback, pay close attention to the following aspects of the character design: (1) assess whether the designed character introductions fit harmoniously with the established storyline; (2) assess the plausibility, depth, interest, and complexity of the character relationships, rather than being overly simplistic — consider whether the story is consistent with the characters’ personas; (3) assess whether the designed characters are compelling and can effectively engage the audience; (4) assess whether the character introductions are logically sound; (5) any other aspect you consider important. If you find problems of these kinds in what the screenwriter provided, give precise and concise revision suggestions targeting those problems.

Genre: {genre}
Logline: {logline}
Detail: {logline_detail}

Output the suggestions directly, itemized, in about 300 words:

Logline reviser.

Revise and improve the original storyline according to the revision suggestions, and output the revised result.

Genre: {genre}
Original logline: {logline}
Original detailed plot: {logline_detail}
Revision suggestions: {advice}

Output a JSON object without markdown formatting, with the fields logline and detail.

Outline critic. Identical in role and output contract to the logline critic, with the narrative-theory aspect replacing aspect (2):

Script content: {script}

(1) Assess whether the characters’ behaviour and performance fit harmoniously with the established storyline. (2) Analyse the plot from the perspective of the three-act structure and the hero’s journey, so that the plot is full of conflict and suspense; is the plot compelling? Are the characters’ desires consistent throughout, and do they effectively drive the plot forward? (3) Assess whether the designed characters are compelling and can effectively engage the audience. (4) Assess whether the plot is logically sound. (5) Any other aspect you consider important.

Output the suggestions directly, itemized, in about 300 words:

Outline reviser.

Revise and improve the original plot according to the revision suggestions, and output the revised result.

Original plot: {script}
Revision suggestions: {advice}

Output JSONL without markdown formatting; the fields of each chapter are plot (string, “Chapter X”), role (list of strings), description (string describing the plot), and location (string, where the plot takes place).

A representative case. On a romance-genre logline the draft R_0 reads (translated; character and organisation names replaced with placeholders):

On the eve of the tournament held by Nova Entertainment, Alex meets again with Morgan, the senior he had once misjudged. To prove himself he trains hard but is hurt in an accident, and finally, encouraged by Iris, he pulls himself together, reconciles with Morgan, and wins the tournament.

After one critic-feedback round, R_1 reads:

On the eve of the tournament held by Nova Entertainment, Alex meets again with Morgan, the senior he had once misjudged. At first he trains hard with the single aim of proving himself, and is injured through a flaw in his own character. Encouraged by Iris, herself a former competitor who speaks his language, he overcomes his inner wound and his weakness and recovers. During the tournament he discovers the reason Morgan has been deliberately avoiding him and clears up the misunderstanding, turning from merely wanting to prove himself to grasping the transmission of the craft itself; in a contest with Morgan that is half rivalry and half friendship he wins the tournament, gaining both love and growth.

Four changes are observable. The injury is converted from an external accident into a consequence of the protagonist’s own flaw, so the character arc acquires a causal driver; the supporting character is given a competitor background that motivates her mentorship instead of leaving it unexplained; a suspense hook (why Morgan avoids him) is added; and the theme is raised from “proving oneself” to “inheriting a craft”. The detail field grows from 298 to 740 characters. These are the kind of structural notes a human editor gives, which is consistent with the large R_1 -over- R_0 win rates of Table 2.

D Judge Prompt and Evaluation Protocol

Scripts are compared pairwise. For each comparison the judge receives the shared logline and the two complete scripts, and returns a single JSON object of 24 fields: an overall verdict (A, B, or TIE), a justification of at most 100 characters, and a 1–5 score for each of 11 dimensions for each script. The per-dimension win/lose/tie used in Table 1 is derived by comparing the two scores. The dimension definitions of Sec. 4.1.2 are embedded verbatim in English inside an otherwise Chinese prompt matching the language of the scripts:

Judge impartially the quality of the scripts generated by two methods. You should choose the better script. Your assessment should focus on the following 11 aspects; the first 6 are positive aspects and the last 5 are negative. The richness

and the length of a script should be taken into account, as should its fit to the logline; a script markedly shorter than the other is not good.

1. **Coherence**: Evaluate the coherence from plot structure, character description, scene transitions and setup consistency. 2. **Relevance**: Evaluate the relevance from the relationship between characters and genre of the story. 3. **Interestingness**: Evaluate the interestingness to readers. 4. **Humanlike**: Judged to be human-written. 5. **Details**: Evaluate whether the script provides sufficient detail to clearly illustrate the progression of the plot. 6. **Overall Quality**: Synthesize the coherence, relevance, and interestingness of a screenplay to assess the overall quality. 7. **Narration**: Jarring changes in narration and/or style. 8. **Inconsistency**: Factually inconsistent or containing very odd details. 9. **Confusing**: Confusing or difficult to follow. 10. **Repetitive**: Highly repetitive. 11. **Disfluent**: Frequent grammatical errors.

logline: {logline}

Script A: {s1}

Script B: {s2}

Score script A and script B separately. Aspects 1–6 are scored positively from 1 to 5, where 5 is best, 4 better than average, 3 average, 2 worse than average, 1 worst. Aspects 7–11 are scored negatively from 1 to 5, where 5 denotes the worst behaviour with severe problems and 1 the best. Your assessment should compare the two scripts and give a brief explanation. Avoid any positional bias and ensure that the order in which the scripts are presented does not influence your decision. After the explanation, output your final conclusion strictly in the given format ... Output a JSON object without markdown formatting.

Dimensions 1–4 are reported directly in Table 1. Dimension 5 is how the prompt operationalises **Writing Quality** as defined in Sec. 4.1.2: rather than asking for a bare quality rating, it makes the judge assess how well the writing carries the progression of the plot, and its score is reported in that column. Dimensions 7–11 are aggregated into Misc. Problem, following Yang et al. (2022). One property of the prompt deserves explicit statement, since it is released as run: the preamble explicitly tells the judge to penalize a markedly shorter script, which could favour verbose systems; this is why we ran the length-neutral re-scoring reported below.

Three judges were run with this identical prompt: GPT-5 (used in Table 1), Claude-Sonnet-4.5, and DeepSeek-V4-Flash. Pairwise verdict agreement is 90.4% ($\kappa=0.614$) between GPT-5 and Claude-Sonnet-4.5, 79.4% ($\kappa=0.471$) between GPT-5 and DeepSeek-V4-Flash, and 78.9% ($\kappa=0.440$) between the two new judges. For the length-robustness check we deleted every clause of the preamble mentioning length, detail, or richness, re-scored a stratified subset covering all genres and

baselines, and found 92% of verdicts unchanged, so the win rates are not an artifact of that clause.

E Human Evaluation Protocol

Three professional narrative designers, all working on the game the data originate from, evaluated 30 blinded pairs. Each pair contrasts a D2GS script with one from the two strongest baselines (BookWorld or HoLLMWood) generated from the same logline. System identity was hidden, presentation order was randomized independently per pair, and annotators judged the same six dimensions reported in Table 1 with a forced choice plus an optional tie.

Aggregated over pairs and dimensions, D2GS is preferred in 67.8% of judgments. Inter-annotator agreement ranges from 43.3% to 70.0% across dimensions, highest on coherence and lowest on the more subjective interestingness, which is the usual pattern in creative-writing evaluation. Agreement between the human majority verdict and the GPT-5 judge is 69.2%, within the range of agreement among the annotators themselves. We therefore treat the LLM judge as a proxy for expert preference of roughly the reliability of one additional expert, and not as a substitute for expert review in production.

F Baseline Input Parity

Table 6: Input configuration of each method, audited against the experimental code. ✓: supplied directly through the same data loader and files; ◦: conveyed indirectly through the shared logline and its detailed description, as the method’s original design requires; –: not used. DP: Direct prompt, PtW: Plan-then-write, DOC: DOC-screen, HW: HoLLMWood, BW: BookWorld.

Input	DP	PtW	DOC	HW	BW	D2GS
Genre	✓	✓	✓	✓	✓	✓
Logline with detailed plot	✓	✓	✓	✓	✓	✓
Locations with descriptions	✓	✓	◦	✓	✓	✓
Character profiles	✓	◦	◦	✓	✓	✓
Worldview knowledge base (retrieval)	–	–	–	–	✓	✓
Character long/short-term memory	–	–	–	–	✓	✓
Structured character relations	–	–	–	–	–	✓

Table 6 records the input configuration of all six methods, audited item by item against the experimental code. Every method starts from the same 60 loglines, themselves generated from the full character, relation, and worldview data, so no method is deprived of the creative premise; all use the same backbone set, the same JSONL output contract, and the same retry-on-parse-failure policy. Where a baseline’s original design accepts explicit character or world inputs, it receives the same files

through the same loader as D2GS: Direct prompt and HoLLMWood read character profiles through the identical helper, and BookWorld’s world agent and role agents are mounted on the same world-view knowledge base and retrieve from it during generation, exactly as D2GS does. Where a baseline’s design does not consume such inputs — Plan-then-write expands from a premise by construction, and DOC-screen generates its own character descriptions at the outline stage, which is its original design — we deliberately did not re-engineer its interface, since doing so would compare our re-implementation rather than the published method; the character and location information those two methods need reaches them through the shared log-line and its detailed plot. The single genuine asymmetry is the last row: structured character relations are consumed only by D2GS. That is the contribution of Sec. 3.1 rather than a handicap imposed on the baselines, and Sec. 4.2.5 measures how much it is worth. All baseline prompts are released with the evaluation code.

G Knowledge Extraction Accuracy

To check the reliability of the schema-constrained extraction of Sec. 3.1, we sampled 150 extracted records — all 23 character profiles, 27 character relations, 50 terminology entries, and 50 locations — and verified each against the source documents by hand. A record is *strictly correct* if every field matches the source, and *partially correct* if it identifies the right entity with correct substantive fields but an incomplete or over-specific secondary field. Strict accuracy is 87.3%, and 94.6% of records are strictly or partially correct. The dominant residual error is over-segmentation of long relationship descriptions, which splits one relation into two records; since retrieval is by entity, such fragments degrade recall gracefully rather than injecting false facts.

H Frame-Event Validator

The target engine consumes scripts as a table whose rows carry a dialog or narration line plus optional frame events drawn from a closed instruction set (sprite and pose changes, background switches, background music and sound effects, transitions, expressions, and timing markers). Every asset name referenced by a frame event must occur in the production resource sheets for that version event. Both properties are decidable, so we check them

with a deterministic validator rather than an LLM.

Command legality is the fraction of emitted frame-event instructions whose command belongs to the closed instruction set and whose arguments are well-formed: $749/751 = 99.73\%$ over the 10 authoring sessions. *Automatic asset-reference resolution* is the fraction of asset references resolving to an entry of the resource sheets without human intervention: $545/583 = 93.5\%$. *Expression validity*, the fraction of facial-expression tags that exist for the particular character sprite addressed, is 87.56%; it is the lowest of the three because expression inventories differ per character and minor characters have sparse inventories. Import of the table format never fails, since the exporter emits the schema by construction. The residual failures fall into three kinds: a command placed in the wrong category, a sprite requested for an unregistered minor character, and an invented asset name. Every unresolved reference is surfaced by the validator together with its row number, so the failure mode is a review item rather than a broken build.