

ELECT: Settling Identity Before Rendering for Scalable Story Visualization

Mehrajul Abadin Miraj*
Aniket Joarder
Md. Ahasanul Alam
Md Mahmudur Rahman

Md. Mahmudul Hasan
Mahmudul Hasan
Redwan Ahmed Rizvee
Md. Tanvir Alam
University of Dhaka

Meherun Farzana
Abir Chakraborty Partha
Md. Fahim Arefin
Md Mosaddek Khan

*Corresponding author: mehrajulabadin-2020115630@cs.du.ac.bd

Abstract

Training-free story visualization without reference images must preserve recurring entities across frames while scaling to long narratives. Existing methods typically maintain consistency by conditioning each frame on other frames of the same story, coupling generation time, VRAM, or both to story length. We introduce ELECT¹, which instead settles cross-frame identity before final rendering through a language-guided scene–entity mapping and stationary anchors elected from the model’s own drafts. This makes the per-frame cost of the final rendering pass independent of story length. Joint attention exceeds 96 GB of VRAM at 32 scenes, whereas ELECT remains at 8.3 s per frame and under 20 GB VRAM from 10 to 1000 scenes. On the Story50 long-story benchmark, ELECT reaches 0.833 average Character Consistency Similarity (aCCS), compared with 0.826 for ten-round refinement, at one seventh of its generation cost, with lower text alignment (0.327 vs. 0.340). At 1000 scenes, to our knowledge, the first evaluation of training-free story visualization at this length, ELECT exceeds ten-round refinement by 0.017 aCCS and the nearest flat-cost baseline by 0.091, while also surpassing refinement in text alignment at one twenty-fourth of its generation time. These results show that long-horizon consistency need not require increasingly expensive inter-frame conditioning as stories grow.

1. Introduction

A coherent visual story requires recurring entities to remain consistent across scenes; otherwise, identity drift breaks narrative continuity even when individual frames are visually plausible. Story visualization addresses this chal-

lenge by rendering narrative text as a sequence of images while preserving entity identity across frames. The task was introduced in a closed-world form by StoryGAN [16], where the cast is fixed and seen during training. We study the open-ended form: the cast is unknown until the text is read, no reference images are provided, and recurring entities may disappear for long stretches before reappearing later in the story. This setting better reflects practical story-generation scenarios, where the cast emerges from the narrative and character-specific visual references are not assumed to be available. Given only raw prose, a system must determine which entities recur, establish their visual identities, and preserve those identities throughout the narrative.

Training-based approaches [13, 27, 28] bind identity into model parameters, but admitting a new character then requires additional optimization, which is poorly suited to open-ended narratives in which new characters may be encountered throughout the story. Moreover, requiring a photograph or appearance sheet for each character as a reference would instead shift identity specification to the user, becoming increasingly impractical as the cast grows. We therefore work in the training-free, reference-free setting, with the prose as the only input.

In this setting, the central technical difficulty is that independently generated diffusion frames have no persistent channel through which entity identity can be carried across scenes. A character described in scene i may be absent for many scenes and return in scene j , with $j \gg i$; each frame follows a separate denoising trajectory from independent noise, and nothing in the generation of scene j knows what was drawn in scene i . Training-free methods therefore differ mainly in how they construct such a channel. Autoregressive conditioning [18, 24] passes each frame to its successor, propagating appearance and errors alike. Attention sharing [29, 38] jointly generates a batch and extends self-attention across frames, reducing compounding drift at

¹Code and the long-story dataset are provided in the supplementary material and will be made publicly available upon acceptance.

the cost of keeping the batch resident in VRAM. Story-Iter [21] instead regenerates the story over multiple rounds, conditioning each frame on all frames from the previous round. The mechanisms differ, but the underlying premise is shared: maintaining consistency requires visual information from other frames of the same story to remain in the conditioning path during generation.

This shared premise makes long-horizon consistency increasingly costly in generation time, VRAM, or both. Under all-pairs conditioning, the N reference image embeddings from previous round form a sequence attended at every denoising step, so one refinement round incurs $O(N^2)$ attention work and a key/value cache that grows linearly with N ; repeated rounds compound these costs. The effect becomes pronounced as stories grow: Story-Iter’s per-frame time increases by a factor of 3.9 between 10 and 1000 scenes, while ConsiStory’s joint attention exceeds 96 GB of VRAM at 32 scenes (Sec. 4.3). These bottlenecks arise from the mechanism used to maintain cross-frame consistency, rather than from any intrinsic requirement of the narrative itself.

To break this coupling, we propose ELECT, a training-free, reference-free approach that settles cross-frame identity before final rendering rather than maintaining it through inter-frame visual conditioning. This allows each frame to be rendered independently, decoupling per-frame generation cost from story length. ELECT remains competitive with iterative refinement at benchmark length, while at 1000 scenes it surpasses ten-round refinement in both identity consistency and text alignment using one twenty-fourth of the generation time and one third of the VRAM. The relative advantage therefore grows substantially as the story horizon increases. Our contributions are as follows:

- ELECT operationalizes pre-render identity settlement through a symbolic scene–entity map and a stationary visual anchor for each recurring entity, elected from the model’s own drafts. This avoids both per-character optimization and user-provided visual references while enabling independent final-frame generation.
- At benchmark length, ELECT reaches 0.833 in average Character Consistency Similarity (aCCS), outperforming ten-round refinement by 0.007 at one seventh of its generation cost, while giving up only 0.013 in text alignment. This demonstrates that stationary identity conditioning can match or exceed iterative refinement without repeated cross-frame generation.
- We introduce a long-horizon evaluation suite of 1000-scene stories and, to our knowledge, provide the first evaluation of training-free story visualization at this scale. ELECT reaches 0.805 aCCS, 0.017 above ten-round refinement and 0.091 above the nearest flat-cost baseline, while maintaining 8.3 s per frame and 19.7

GB VRAM, at \$20 in GPU cost for the suite versus \$487 for ten-round refinement. At this length, refinement takes 55.6 hours, while joint attention exhausts VRAM at 32 scenes.

- Through controlled ablations, we identify where ELECT’s gains arise: anchored conditioning contributes +0.100 aCCS at benchmark length and +0.081 at 1000 scenes; electing the anchor improves aCCS by 0.027 over random selection and by 0.100 over the worst candidate, while reversing the injection schedule costs 0.048 aCCS.

2. Related Work

Prior work relevant to ELECT spans open-ended story visualization, methods for preserving character identity, and training-free mechanisms for maintaining that identity across frames.

Story visualization began as a closed-world problem. StoryGAN [16] fixed the cast in advance and showed it during training, and the work that followed improved the generator or its conditioning without disturbing that assumption [5, 15, 20]. StoryGen [18] named the open-ended form we study here, in which characters exist only as descriptions in the prose, and released a large dataset of illustrated stories spanning hundreds of character categories. Once the cast is unknown until the text is read, the question becomes where an entity’s appearance is stored and how it reaches a frame that needs it.

One answer stores it in weights. Subject-driven personalization [11, 17, 27, 31] learns a token or a low-rank update per character, and story systems built on it compose the learned characters into scenes [13, 28, 32]. Identity fidelity can be strong, but admitting a new character costs an optimization run, and continual customization [9, 36] reports identity degrading as learned characters accumulate, which is the wrong direction for a narrative whose cast keeps growing.

The training-free alternative stores appearance in the frames themselves and manufactures a channel between them. Autoregressive conditioning [18, 24] hands each frame to its successor, keeping neighbours consistent while inheriting every error it is given. Attention sharing [29, 38] generates a batch jointly and keeps it resident in VRAM; StoryDiffusion additionally bootstraps long stories from its first generated frames, tying everything later to those initial outputs. Story-Iter [21] instead regenerates the story over several rounds, conditioning each frame on all frames of the previous round compressed to a fixed number of global tokens, which keeps per-frame conditioning cheap at the price of repeated full passes over the story (Sec. 4.3) which is still quadratic. The injection mechanism is largely shared, IP-Adapter’s decoupled cross-attention [35]; what separates

these methods is which visual references reach each frame, and that choice is what couples their cost to story length.

A second family breaks that coupling by conditioning on the character rather than the frame, fixing the conditioning before rendering: clustering samples into an identity, minting a portrait from an LLM description, sharing one reference panel, or grounding mentions to entity features [1, 8, 10, 14]; concurrent multi-shot video work accumulates per-entity banks across shots. ELECT belongs here, but where these approaches accept the reference obtained at construction time, we elect it from evidence distributed across the complete story. A parallel line delegates narrative structure to a language model [6, 7, 34, 37], and existing benchmarks [4, 10, 39] run to tens of frames, so neither the mechanism nor the evaluation has yet been pushed to the horizon we target.

Taken together, prior work addresses long horizon identity consistency through either per-character optimization or story-length-dependent inter-frame visual conditioning, which becomes impractical in generation time and VRAM as stories grow.

3. The ELECT Framework

We now describe ELECT, which transforms a sequence of story scenes into a visually consistent image sequence through three stages: script construction, drafting and anchor election, and anchored regeneration. Figure 1 provides an overview of the complete pipeline.

3.1. Problem Setting

A story is a sequence of N scene descriptions $p_1, \dots, p_N$, one per frame, with no cast list, reference photographs, or guarantee that an entity’s first mention provides a usable visual description. We produce frames $x_1, \dots, x_N$ in which recurring entities remain visually consistent, each x_i remains faithful to p_i , and the cost of producing x_i depends on neither i nor N . Every component is a frozen pre-trained model [23, 25, 26, 30, 35] composed at inference; no weights are updated at any point.

3.2. Script Construction

ELECT begins by constructing a symbolic representation of the story. A local language model (Qwen3-14B [33]) reads the full prompt sequence in one pass and emits a *Script* $\mathcal{S}$.

$$\mathcal{S} = \text{LLM}(p_{1:N}) = (\mathcal{E}, \Pi) \quad (1)$$

As defined in Eq. (1), $\mathcal{E} = \{e_1, \dots, e_M\}$ is the cast of recurring entities, each a character the story returns to, and $\Pi : \{1, \dots, N\} \rightarrow 2^{\mathcal{E}}$ assigns to each frame the entities it casts. The model is asked for exactly two things: a cast of recurring entities, each carrying a one-line description $d(e)$,

and, for every frame, the subset of that cast supported by the frame’s own text. The prompt is reproduced in full in the supplementary material.

Everything required by the later stages is read from this pair. An entity’s description $d(e)$ is used unchanged as the grounding phrase that queries the detector, while its frame set $F(e) = \{i : e \in \Pi(i)\}$ is obtained by scanning Π . Both come directly from $\mathcal{S}$, with no further judgment asked of the model once it has named the cast and placed its members scene by scene.

The map Π is where the cross-frame channel is established. Other training-free methods construct it in pixel space, implicitly asserting that entity e in frames i and j is the same by making their latents attend to one another; Eq. (1) asserts it symbolically. Deciding that “the knight,” “he”, and “Sir Aldric” denote one entity in scene j as in scene i , with $j \gg i$, is a coreference-resolution problem, handled notably more reliably in text than in latent space.

3.3. Drafting and Election

Each frame is first generated once from its own prompt with a pretrained text-to-image diffusion model [25] under a fixed style template, with no conditioning on any other frame. These drafts are not shown; instead, they provide the candidate pool from which entity anchors are selected. Anchor construction proceeds in two steps.

Harvesting and filtering. An open-vocabulary detector [30] runs on every draft, queried only with that frame’s Script participants, yielding a pool of candidate crops for each entity across the story. Each pool is embedded with DINOv2 [23] and clustered with k -means. Crops outside the dominant cluster are discarded, leaving a candidate pool $\mathcal{C}(e)$ for each entity. This filtering also removes two common errors: detections of the wrong object and lookalike characters that match the same grounding phrase. Such incorrect crops appear as outliers in the embedding space.

Election. Given the filtered candidate pool $\mathcal{C}(e)$, the anchor for entity e is selected as the highest-scoring surviving crop, $a(e) = \arg \max_{c \in \mathcal{C}(e)} q(c)$, where the quality score combines three per-crop signals,

$$q(c) = w_c \text{conf}(c) + w_s \text{sharp}(c) + w_a \text{area}(c), \quad (2)$$

with the balanced default $(w_c, w_s, w_a) = (0.4, 0.3, 0.3)$. The three terms capture complementary aspects of reference quality. $\text{conf}(c)$ is the detector’s confidence, which is low for partial, occluded, and weakly grounded instances; $\text{sharp}(c)$ is a normalised Laplacian response, which is low for blurred and out-of-focus crops; and $\text{area}(c)$ is the crop’s area as a fraction of its frame, which is low for distant instances that contain too few pixels to encode appearance

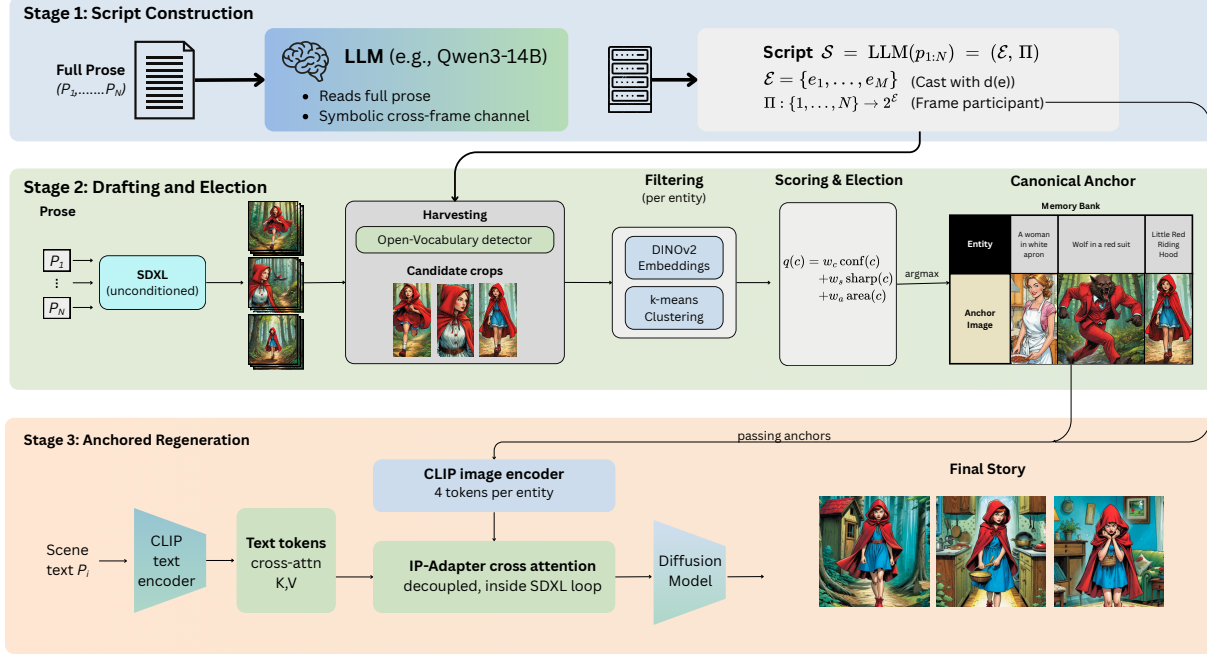


Figure 1. **Overview of ELECT.** *Stage 1, script construction:* an LLM reads the full story and writes a Script, a recurring cast with short descriptions and a per-frame participant map, establishing cross-frame consistency symbolically. *Stage 2, drafting and election:* SDXL drafts each frame from text alone, a detector proposes crops for that frame’s cast, DINOv2 filtering removes misfires, and the best crop per entity is elected as its canonical anchor. *Stage 3, anchored regeneration:* each frame is regenerated from scratch using its own text and its participants’ anchors via IP-Adapter, independently of other frames, so the pass stays parallel and flat in N .

reliably. These criteria matter because the elected anchor is re-encoded and injected into every frame in which its entity appears, so defects in the selected crop can propagate throughout the story. Section 4.4 varies (w_c, w_s, w_a) and shows that no single signal suffices.

By the time Eq. (2) is applied, two earlier decisions have already been made. Harvesting determines whether a crop corresponds to the target entity, since a candidate exists only because the detector matched it to the entity’s grounding phrase. Filtering then determines whether that crop is representative of the entity: restricting $\mathcal{C}(e)$ to the dominant DINOv2 cluster removes unrepresentative crops as a hard constraint rather than treating representativeness as a soft penalty traded against image quality. Election therefore addresses the remaining question: which of the surviving, matched, and representative crops provides the cleanest instance to re-encode.

3.4. Anchored Regeneration

With the anchor bank fixed, every frame is regenerated independently from fresh Gaussian noise and its own scene text. Each frame is conditioned through IP-Adapter’s decoupled cross attention [35] only on the bank entries of the entities participating in that scene, with one token block per active entity built from the CLIP embedding of its an-

chor. Because the bank entries are entity crops rather than whole frames, composition and background are not transferred along with identity. Moreover, because each frame is conditioned only on the entities in its own scene, the conditioning set does not grow with story length. Because a scene casts a bounded number of entities, what a frame is conditioned on does not grow with the number of frames preceding it, and a 50-scene and a 1000-scene story pay the same per-frame cost.

Injection schedule. The remaining design choice is how strongly, and when, the anchor should influence generation. We scale the anchor token block by a factor $\lambda_{\text{ent}}(t)$ inside the decoupled cross-attention and vary it linearly over the sampling trajectory,

$$\lambda_{\text{ent}}(t) = \lambda_{\text{hi}} + (\lambda_{\text{lo}} - \lambda_{\text{hi}}) \frac{T - t}{T}, \quad (3)$$

with $t = T, \dots, 0$ over the T sampling steps, from $\lambda_{\text{hi}} = 0.85$ at the first, highest-noise step to $\lambda_{\text{lo}} = 0.35$ at the last. Early steps determine coarse structure, where silhouette, build, and palette are established and weak anchor guidance can lose identity; later steps determine texture and local detail, where reducing anchor strength leaves more capacity for the scene text. Tab. 4 compares this decay with constant scales and the reverse sweep.

Table 1. Story50 [21] benchmark comparison. Rows below the rule are our own reproductions, run on identical stories, seeds and hardware; rows above are quoted from prior publications.

Method	CLIP-T $\uparrow$	aCCS $\uparrow$
AR-LDM [24]	0.216	0.673
StoryGen [18]	0.223	0.740
IP-Adapter [35]	0.274	0.751
FLUX.1.Kontext [2]	0.290	0.783
Nano Banana	0.310	0.791
IP-AdapterXL [35]	0.297	0.787
DALL-E 3 [3]	0.301	0.674
SEED-LLaMA [12]	0.279	0.688
ConsiStory [29]	0.316	0.761
1Prompt1Story [19]	0.340	0.7287
StoryDiffusion [38]	0.351	0.8103
DreamStory [14]	0.3193	0.7627
Story-Iter [21]	0.340	0.8264
ELECT (ours)	0.3273	0.8334

4. Experiments

We evaluate ELECT along the dimensions that motivate its design: identity consistency and text alignment, scalability in generation time and VRAM, and the contribution of its individual components. We first describe the evaluation setting and implementation.

4.1. Setup

We evaluate on the Story50 long-story benchmark (20 stories of 50 frames each) [21] and on five 1000-scene stories adapted from public-domain books: *The Mysterious Island*, *Taras Bulba*, the *Shahnameh*, a Russian fairy tale, and the *Völsunga* saga. Their construction is described in the supplementary material. We report CLIP-T for text-image alignment and aCCS for cross-frame identity consistency, following [7].

Implementation details. All runs use SDXL [25] at 1024×1024 with 50 sampling steps on a single NVIDIA RTX 6000 PRO Blackwell (96 GB). Anchors are injected through base IP-Adapter with four pooled tokens per entity using the schedule in Eq. (3). Script extraction uses Qwen3-14B. Harvesting uses YOLOE [30], and candidate crops are embedded with DINOv2 [23] for the cluster filter in Sec. 3.3. In preliminary experiments, YOLOE grounded the Script descriptions more accurately than the alternatives tested, while DINOv2 separated visually similar characters more reliably than CLIP

4.2. What does anchoring buy?

Anchoring drives consistency. Anchoring replaces per-frame identity guesswork with a single settled reference: Pass 0 drafts every scene independently from text, one crop per entity is elected as the anchor, and the second pass conditions all remaining scenes on it according to the map-

Table 2. Effect of anchored conditioning. Pass 0 is ELECT’s own draft: identical prompts, backbone and seeds, generated independently from text with no cross-frame conditioning. The 50-scene row is the mean over 20 benchmark stories.

	aCCS $\uparrow$		CLIP-T $\uparrow$	
	Pass 0	ELECT	Pass 0	ELECT
<i>50-scene benchmark</i>				
Mean over 20 stories	0.7334	0.8334	0.352	0.3273
Δ	+0.100		−0.025	
<i>1000 scenes</i>				
Mean	0.7239	0.8050	0.318	0.306
Δ	+0.081		−0.012	

Table 3. Quality and cost at $N=1000$ scenes (1024^2 , RTX 6000 PRO Blackwell, 96 GB). ConsiStory exceeds VRAM at 32 scenes; its peak is reported at $N=16$. **s/frame is end-to-end wall clock divided by N** , covering every stage each method runs: for ELECT, Script extraction, the Pass 0 drafts, detection, embedding and clustering, and the anchored regeneration pass; for Story-Iter, all ten refinement rounds. Total (h) is this average per story generation time CLIP-T and aCCS are mean over 5 stories.

Method	CLIP-T $\uparrow$	aCCS $\uparrow$	s/frame	Total (h)	VRAM (GB)
<i>Cost grows with N</i>					
ConsiStory [29]	OOM	OOM	–	–	92.7 ($N=16$)
Story-Iter [21]	0.3044	0.7885	200.1	55.6	59.7
<i>Cost flat in N</i>					
1Prompt1Story [19]	0.309	0.714	7.37	2.05	16.7
StoryDiffusion [38]	0.315	0.639	4.40	1.22	66.4
DreamStory [14]	0.3089	0.6976	38.42	10.7	32.9
ELECT (ours)	0.306	0.8050	8.30	2.30	19.7

ping. On the standard story50 benchmark (Tab. 1), this puts ELECT at 0.8334 aCCS, the highest of any method we compare against, ahead of Story-Iter (0.8264) and StoryDiffusion (0.8103), while CLIP-T sits at 0.3273, within 0.024 of the best value and above most baselines. The benchmark therefore places us where anchoring should: identity is held tighter than by iterative refinement or reference-injection methods, at a small and expected cost in per-frame prompt adherence, since conditioning on a fixed appearance necessarily pulls each frame slightly away from its own text.

Pass 0 isolates the anchor’s contribution: it shares prompts, backbone, style template and seeds with the final pass and differs only in that no anchor is injected (Tab. 2). Anchoring raises aCCS by 0.100 at 50 scenes and by 0.081 at 1000, while CLIP-T falls by only 0.025 and 0.012. The trade is strongly asymmetric, and it does not degrade with story length.

4.3. What does story length cost?

Existing benchmarks typically cover only tens of frames, leaving the long-story regime largely unexplored. We therefore evaluate stories up to 1000 scenes to examine how qual-

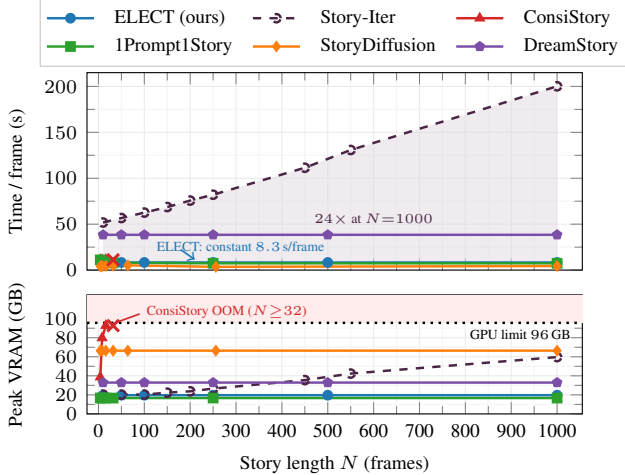


Figure 2. **Scaling from 4 to 1000 scenes.** *Top:* amortized cost per frame. ELECT is constant at 8.3 s/frame while Story-Iter degrades from 51.8 to 200.1 s/frame, a $24\times$ gap at $N=1000$ (8.3 s vs. 200 s end-to-end); the shaded band is the resulting speedup. *Bottom:* peak VRAM. ELECT is constant at 19.7 GB, whereas ConsistencyStory exceeds the 96 GB device budget at $N=32$ (cross), while StoryDiffusion and DreamStory hold fixed budgets of 66.4 and 32.9 GB. Baselines limited to short horizons are plotted only over the range they complete.

ity and computational cost scale with narrative length. Figure 2 reports scaling from 4 to 1000 scenes, while Tab. 3 summarizes quality and cost at 1000 scenes.

ELECT keeps its cost fixed as stories grow. The difference is clearest in per-frame cost. ConsistencyStory cannot reach 32 scenes because its shared attention state exceeds 96 GB of VRAM (Fig. 2). Story-Iter reaches 1000 scenes, but its per-frame cost grows from 51.8 s at 10 scenes to 200.1 s at 1000, requiring 55.6 hours for one story. ELECT instead remains at 8.3 s per frame and 19.7 GB VRAM from 10 to 1000 scenes, completing the same story in 2.30 hours.

This behavior follows from how ELECT carries cross-frame information. The Script and frozen anchor bank fix the relevant entity information before rendering, so each scene is conditioned only on the entities it contains and the conditioning size does not grow with story length. IPrompt1Story and StoryDiffusion also maintain nearly constant per-frame cost and are faster than ELECT at 1000 scenes, at 7.37 s and 4.40 s per frame, respectively. However, this speed advantage comes with substantially lower identity consistency: ELECT reaches 0.8050 aCCS, compared with 0.714 for IPrompt1Story and 0.639 for StoryDiffusion. Thus, ELECT retains markedly stronger identity consistency while keeping its per-frame cost nearly constant as stories grow.

The efficiency does not come at the cost of identity. At 1000 scenes, ELECT reaches 0.8050 aCCS, which is 0.091 higher than IPrompt1Story, 0.166 higher than StoryDiffusion, and 0.017 higher than Story-Iter (Tab. 3). CLIP-T is similar across the three flat-cost methods, with ELECT only 0.009 below StoryDiffusion. Thus, the methods that keep their cost fixed do not all preserve identity equally well. StoryDiffusion, for example, is only 0.023 behind ELECT at the benchmark length but falls 0.166 behind at 1000 scenes. Against Story-Iter, ELECT is better on both quality measures at 1000 scenes while using about $24\times$ less generation time and one-third the peak VRAM.

4.4. How hard should the anchor push, and which crop should it push with?

Results in Tab. 2 already show that anchoring improves identity consistency. We next examine two choices that determine how much of this gain is realised: how the anchor is injected during sampling (Tab. 4) and which crop is elected as the anchor (Tab. 5).

When the anchor pushes matters more than how hard.

At a constant scale, increasing λ_{ent} from 0.35 to 0.85 raises identity monotonically, from 0.787 to 0.829 aCCS, while reducing alignment from 0.343 to 0.327 CLIP-T. A fixed scale therefore acts largely as a dial between the two objectives.

The scheduled variants reveal that timing also matters. Both use the same endpoints and differ only in direction, yet they finish 0.048 aCCS apart: decaying $0.85 \rightarrow 0.35$ is the best arm in the table, while rising $0.35 \rightarrow 0.85$ is the worst. Because the two schedules have the same average anchor strength, the difference comes from when that strength is applied. The rising schedule performs no better on either metric than holding the weakest scale throughout, consistent with identity-relevant structure being established early in denoising. Conversely, decay outperforms a constant 0.85 on both metrics despite using lower strength for most of the trajectory, suggesting that strong early guidance and weaker late guidance provide a better balance. The flat schedules remain useful for other operating points: a constant 0.50 reaches 0.808 aCCS and 0.338 CLIP-T, matching our Story-Iter reproduction’s alignment to within 0.002 at 0.018 lower consistency.

Anchor election is necessary, not merely anchoring. To separate having an anchor from having a good one, we cross the election rule with the pool it draws from: argmax of Eq. (2), a random survivor, or argmin, over either the cluster-filtered pool or the raw one (Tab. 5). aCCS is monotonic in the rule on both pools, 0.8334/0.8065/0.7911 filtered and 0.8244/0.7951/0.7335 raw, and the best cell requires both stages: filtering without a careful election, or

Table 4. Anchor strength during regeneration. λ_{ent} scales the anchor token block inside IP-Adapter’s decoupled cross-attention, either held flat across the 50 sampling steps or swept linearly by Eq. (3). The two scheduled arms share their endpoints and differ only in direction. Mean over the story50 stories; everything else fixed, including Script, drafts, harvest, election and seed. The last row is the configuration reported throughout.

λ_{ent}	aCCS $\uparrow$	CLIP-T $\uparrow$
<i>Flat</i>		
0.35	0.7872	0.3427
0.50	0.8083	0.3382
0.75	0.8107	0.3300
0.85	0.8293	0.3268
<i>Scheduled</i>		
rising 0.35 $\rightarrow$ 0.85	0.7850	0.3426
decaying 0.85 $\rightarrow$ 0.35	0.8334	0.3273

election over an unfiltered pool, each falls short of the two together. A random anchor already recovers much of the benefit of anchoring, but a poor one gives most of it back, especially in the raw pool, where argmin falls 0.091 below argmax.

The two stages guard against different failures. Filtering helps every rule, but by 0.009 aCCS for argmax and 0.011 for random against 0.058 for argmin, so its work is almost entirely the removal of unusable candidates, while the election decides which of the survivors is actually re-encoded. Alignment is essentially unaffected by the choice: argmax and random read 0.3273 and 0.3280 CLIP-T on the filtered pool. The identity gain therefore comes from choosing the anchor well, at no meaningful cost in alignment.

Combining the three signals is better than using any one alone. We ablate the three terms in the election score by comparing the balanced configuration with each single-signal variant, keeping all other settings fixed (Tab. 6). The balanced weights achieve the highest aCCS (0.8334), outperforming area only (0.8291), confidence only (0.8118), and sharpness only (0.7980). This shows that no single signal is sufficient to select the best anchor, while combining confidence, sharpness, and area gives the strongest overall selection. CLIP-T follows the same trend except for area only, which is marginally higher than the balanced setting (0.3279 vs. 0.3273). We therefore use the balanced weights (0.4, 0.3, 0.3) throughout.

4.5. Can Stage 3 reuse the draft’s trajectory?

Pass 0 achieves higher CLIP-T (Tab. 2), suggesting that re-entering its trajectory while supplying identity through the anchor might retain both alignment and consistency. We test this following SDEdit [22]: the draft is forward-noised to a start index determined by strength s , and only the remaining steps are run under injection ($s=0.75$ skips

Table 5. Where the anchor’s quality comes from: the rule that picks one crop per entity, crossed with whether the pool it picks from has been cluster-filtered. All cells share the pipeline, prompts, backbone and seeds, and differ only in these two factors. *Random* is the mean of three seeds, whose spread is at most 0.009 aCCS. The top-left cell is the reported configuration of Tab. 1. Mean over the 20 benchmark stories.

Selection rule	aCCS $\uparrow$		CLIP-T $\uparrow$	
	Filtered	Raw pool	Filtered	Raw pool
argmax of Eq. (2)	0.8334	0.8244	0.3273	0.3260
Random survivor	0.8065	0.7951	0.3280	0.3226
argmin of Eq. (2)	0.7911	0.7335	0.3206	0.3136

Table 6. Weights of the election score on the story50 benchmark, driving (w_c, w_s, w_a) of Eq. (2) to each corner of the simplex with everything else fixed. The first row is the reported configuration.

(w_c, w_s, w_a)	aCCS $\uparrow$	CLIP-T $\uparrow$
balanced (0.4, 0.3, 0.3)	0.8334	0.3273
confidence only (1, 0, 0)	0.8118	0.3262
sharpness only (0, 1, 0)	0.7980	0.3243
area only (0, 0, 1)	0.8291	0.3279

13 of 50 steps; our default is $s=1$). The resulting trade-off is less favorable (Tab. 7). Alignment improves because composition is established by a pass with no competition from anchor tokens, and the variant preserves that decision. Identity degrades for the same reason: diffusion commits low-frequency structure early, precisely where Eq. (3) assigns λ_{ent} its highest values. Entering mid-trajectory leaves the anchor only the low-noise tail, where it can restyle surfaces but not re-shape a participant already established by the draft. The ordering in Tab. 4 remains unchanged, showing that the schedule effect is not an artefact of starting from noise.

Table 7. Draft-trajectory reuse ($s=0.75$) instead of fresh noise; story50 benchmark. Against the fresh-noise default (0.8334 aCCS, 0.3273 CLIP-T), reuse costs 0.039 aCCS for 0.017 CLIP-T.

λ_{ent}	aCCS $\uparrow$	CLIP-T $\uparrow$
flat 0.15	0.7626	0.3512
flat 0.50	0.7807	0.3478
flat 0.70	0.7867	0.3457
decaying 0.85 $\rightarrow$ 0.35	0.7947	0.3442

4.6. Is the planning layer a bottleneck?

If the Script has to come from a strong reader, ELECT inherits the cost and deployment constraints of one. We therefore replace Qwen3-14B with Qwen3-4B, a quarter of the parameters, and re-run the pipeline with everything else fixed. The smaller planner costs 0.0118 aCCS, 0.8216 against 0.8334 (Tab. 8), and moves alignment a negligible 0.0007 the other way. Since nothing downstream of

the Script changes, the whole difference lies in what the two models write: which entities they promote to the cast, which frames they credit them to, and how usable each one-line description is as a grounding phrase for the detector. A small model recovers about 99% of the reported score, so the symbolic layer does not depend on frontier-scale reading. It is not free either: the penalty exceeds the 0.007 by which ELECT leads Story-Iter at this length, so the planner is material to that comparison, though it stays small against the 0.100 anchoring supplies and the 0.048 the schedule’s direction is worth (Tab. 4).

Table 8. Planner capacity. The Script is re-extracted with a smaller language model and the pipeline re-run end to end on the 50-scene benchmark, everything else fixed. The first row is the configuration reported throughout.

Script extractor	aCCS $\uparrow$	CLIP-T $\uparrow$
Qwen3-14B	0.8334	0.3273
Qwen3-4B	0.8216	0.3280

4.7. What does it cost to run?

Time and VRAM capture architectural efficiency, but practical deployment also depends on generation cost. At \$1.74 per GPU-hour (Vast.ai rate for an NVIDIA RTX 6000 PRO Blackwell), the difference is small at benchmark scale but compounds sharply with length: on the long-story suite (5 stories $\times$ 1000 frames), ELECT costs \$20 versus \$487 for ten-round refinement (\$4.01 vs. \$97.44 per story), with the gap widening from $6.7\times$ to $24\times$ because refinement’s per-frame cost grows with N while ELECT’s does not. Scaled to 100 thousand-scene stories, this is \$401 versus \$9,744. ELECT is not the cheapest method (StoryDiffusion: \$10.63; 1Prompt1Story: \$17.82 for the suite), but it is the cheapest that preserves identity at this length, with both cheaper alternatives scoring lower on aCCS (by 0.166 and 0.091, Tab. 9).

4.8. What this looks like at 1000 scenes

Figure 3 tracks a single recurring entity in each long-story suite, at frames drawn from across the whole 1000, rather than from one neighbourhood. Silhouette, palette, and costume survive the span even when hundreds of scenes separate two appearances, which is what the flat per-frame cost protects: the anchor is the same crop at frame j as at frame i , with $j \gg i$, so there is no accumulated drift. It is not exact. Garment detail, facial features, and accessories vary between appearances, and the 0.8050 aCCS of Tab. 3 is the quantitative form of that gap. What the anchor fixes is the entity’s identity at recognition level, not its appearance pixel for pixel.

Table 9. Cost of generating each evaluation set, at \$1.74 per GPU-hour (NVIDIA RTX 6000 PRO, on-demand(vast.ai), 22 August 2026). Story50 is 20 stories of 50 frames; the long-story suite is five stories of 1000 frames. Totals are the generation times of Tab. 3 and Fig. 2; hourly rates and hardware availability vary, so the ratios matter more than the absolute figures.

Method	Story50		Long-story suite	
	Hours	USD	Hours	USD
Story-Iter [21]	15.74	27.39	280.0	487.20
1Prompt1Story [19]	2.16	3.76	10.24	17.82
DreamStory [14]	10.99	19.123	53.5	93.09
StoryDiffusion [38]	1.22	2.12	6.11	10.63
ELECT (ours)	2.35	4.09	11.53	20.06

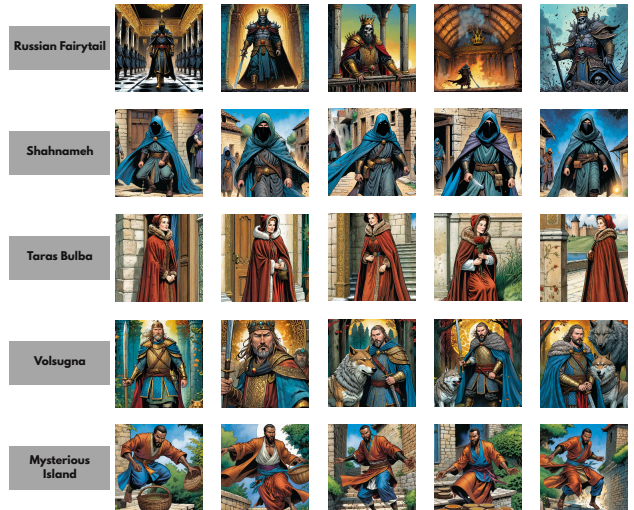


Figure 3. One recurring entity per 1000-scene story, shown at five frames sampled across the full length for the five stories.

5. Conclusions

Training-free story visualization has typically maintained consistency through visual dependencies between frames, making long-horizon generation increasingly expensive. ELECT instead separates entity correspondence from appearance: a language model resolves recurring entities from the prose, while one elected visual anchor per entity fixes appearance before final rendering. The resulting frames can be generated independently, with per-frame cost determined by the cast of a scene rather than the length of the story. At benchmark length, ELECT reaches higher identity consistency than ten-round refinement at one seventh of its generation cost, while trading some text alignment. At 1000 scenes, it exceeds refinement by 0.017 aCCS and 0.002 CLIP-T while using one twenty-fourth of its generation time and one third of its VRAM. Controlled ablations further show that anchored conditioning provides the dominant consistency gain, while anchor selection and injection timing materially affect the final result. These findings suggest that long-horizon consistency need not

rely on increasingly expensive inter-frame conditioning, opening a path toward scalable visual story generation.

References

- [1] Omri Avrahami, Amir Hertz, Yael Vinker, Moab Arar, Shlomi Fruchter, Ohad Fried, Daniel Cohen-Or, and Dani Lischinski. The chosen one: Consistent characters in text-to-image diffusion models. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–12. ACM, 2024. [3](#)
- [2] Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, and Sumith Kulal. Flux.1 kon-text: Flow matching for in-context image generation and editing in latent space. *arXiv e-prints*, 2025. [5](#)
- [3] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science*. <https://cdn.openai.com/papers/dall-e-3.pdf>, 2(3):8, 2023. [5](#)
- [4] Emanuele Bugliarello, H. Hernan Moraldo, Ruben Villegas, Mohammad Babaeizadeh, Mohammad Taghi Saffar, Han Zhang, Dumitru Erhan, Vittorio Ferrari, Pieter-Jan Kin-dermans, and Paul Voigtlaender. Storybench: A multi-faceted benchmark for continuous story visualization. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023. [3](#)
- [5] Hong Chen, Rujun Han, Te-Lin Wu, Hideki Nakayama, and Nanyun Peng. Character-centric story visualization via visual planning and token alignment. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8259–8272, 2022. [2](#)
- [6] Junhao Cheng, Xi Lu, Hanhui Li, Khun Loun Zai, Baiqiao Yin, Yuhao Cheng, Yiqiang Yan, and Xiaodan Liang. Autostudio: Crafting consistent subjects in multi-turn interactive image generation. *arXiv preprint arXiv:2406.01388*, 2024. [3](#)
- [7] Junhao Cheng, Baiqiao Yin, Kaixin Cai, Minbin Huang, Hanhui Li, Yuxin He, Xi Lu, Yue Li, Yifei Li, Yuhao Cheng, Yiqiang Yan, and Xiaodan Liang. Theatergen: Character management with llm for consistent multi-turn image generation. *arXiv preprint arXiv:2404.18919*, 2024. [3](#), [5](#)
- [8] David Dinkevich, Matan Levy, Omri Avrahami, Dvir Samuel, and Dani Lischinski. Story2board: A training-free approach for expressive visual storytelling. *Computer Graphics Forum*, 2026. [3](#)
- [9] Jiahua Dong, Wenqi Liang, Hongliu Li, Duzhen Zhang, Meng Cao, Henghui Ding, Salman Khan, and Fahad Shah-baz Khan. How to continually adapt text-to-image diffusion models for flexible customization? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. [2](#)
- [10] Sibong Dong, Ismail Shaheen, and Sarah Adel Bargal. Freestory: Training-free character consistency for free-form visual storytelling. *arXiv preprint arXiv:2606.25079*, 2026. [3](#)
- [11] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion. In *International Conference on Learning Representations (ICLR)*, 2023. [2](#)
- [12] Yuying Ge, Sijie Zhao, Ziyun Zeng, Yixiao Ge, Chen Li, Xintao Wang, and Ying Shan. Making LLaMA SEE and draw with SEED tokenizer. In *The Twelfth International Conference on Learning Representations*, 2024. [5](#)
- [13] Yuan Gong, Youxin Pang, Xiaodong Cun, Menghan Xia, Yingqing He, Haoxin Chen, Longyue Wang, Yong Zhang, Xintao Wang, Ying Shan, and Yujiu Yang. Talecrafter: Interactive story visualization with multiple characters. In *SIGGRAPH Asia 2023 Conference Papers*. ACM, 2023. [1](#), [2](#)
- [14] Huiguo He, Huan Yang, Zixi Tuo, Yuan Zhou, Qiuyue Wang, Yuhang Zhang, Zeyu Liu, Wenhao Huang, Hongyang Chao, and Jian Yin. Dreamstory: Open-domain story visualization by llm-guided multi-subject consistent diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(12):11874–11891, 2025. [3](#), [5](#), [8](#)
- [15] Bowen Li. Word-level fine-grained story visualization. In *European Conference on Computer Vision (ECCV)*, pages 347–362. Springer, 2022. [2](#)
- [16] Yitong Li, Zhe Gan, Yelong Shen, Jingjing Liu, Yu Cheng, Yuexin Wu, Lawrence Carin, David Carlson, and Jianfeng Gao. Storygan: A sequential conditional gan for story visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6329–6338, 2019. [1](#), [2](#)
- [17] Zhen Li, Mingdeng Cao, Xintao Wang, Zhongang Qi, Ming-Ming Cheng, and Ying Shan. Photomaker: Customizing realistic human photos via stacked id embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8640–8650, 2024. [2](#)
- [18] Chang Liu, Haoning Wu, Yujie Zhong, Xiaoyun Zhang, Yan-feng Wang, and Weidi Xie. Intelligent grimm — open-ended visual storytelling via latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6190–6200, 2024. [1](#), [2](#), [5](#)
- [19] Tao Liu, Kai Wang, Senmao Li, Joost van de Weijer, Fahad Shahbaz Khan, Shiqi Yang, Yaxing Wang, Jian Yang, and Ming-Ming Cheng. One-prompt-one-story: Free-lunch consistent text-to-image generation using a single prompt. In *International Conference on Learning Representations (ICLR)*, 2025. [5](#), [8](#)
- [20] Adyasha Maharana and Mohit Bansal. Integrating visuospatial, linguistic and commonsense structure into story visualization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021. [2](#)
- [21] Jiawei Mao, Xiaoke Huang, Yunfei Xie, Yuanqi Chang, Mude Hui, Bingjie Xu, Zeyu Zheng, Zirui Wang, Cihang Xie, and Yuyin Zhou. Story-iter: A training-free iterative paradigm for long story visualization. In *International Conference on Learning Representations (ICLR)*, 2026. [2](#), [5](#), [8](#)
- [22] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations, 2022. [7](#)
- [23] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaa El-Nouby, et al. Di-

- nov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. 3, 5
- [24] Xichen Pan, Pengda Qin, Yuhong Li, Hui Xue, and Wenhu Chen. Synthesizing coherent story with auto-regressive latent diffusion models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2920–2930, 2024. 1, 2, 5
- [25] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations (ICLR)*, 2024. 3, 5
- [26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, pages 8748–8763, 2021. 3
- [27] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22500–22510, 2023. 1, 2
- [28] Ming Tao, Bing-Kun Bao, Hao Tang, Yaowei Wang, and Changsheng Xu. Storyimager: A unified and efficient framework for coherent story visualization and completion. In *European Conference on Computer Vision (ECCV)*, pages 479–495. Springer, 2024. 1, 2
- [29] Yoad Tewel, Omri Kaduri, Rinon Gal, Yoni Kasten, Lior Wolf, Gal Chechik, and Yuval Atzmon. Training-free consistent text-to-image generation. *ACM Transactions on Graphics*, 43(4):1–18, 2024. 1, 2, 5
- [30] Ao Wang, Lihao Liu, Hui Chen, Zijia Lin, Jungong Han, and Guiguang Ding. Yoloe: Real-time seeing anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 24591–24602, 2025. 3, 5
- [31] Yuxiang Wei, Yabo Zhang, Zhilong Ji, Jinfeng Bai, Lei Zhang, and Wangmeng Zuo. Elite: Encoding visual concepts into textual embeddings for customized text-to-image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15943–15953, 2023. 2
- [32] Jianzong Wu, Chao Tang, Jingbo Wang, Yanhong Zeng, Xiangtai Li, and Yunhai Tong. Diffsensei: Bridging multimodal llms and diffusion models for customized manga generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 28684–28693, 2025. 2
- [33] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. 3
- [34] Shuai Yang, Yuying Ge, Yang Li, Yukang Chen, Yixiao Ge, Ying Shan, and Yingcong Chen. Seed-story: Multimodal long story generation with large language model. *arXiv preprint arXiv:2407.08683*, 2024. 3
- [35] Hu Ye, Jun Zhang, Sibio Liu, Xiao Han, and Wei Yang. Ip-adapt: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023. 2, 3, 4, 5
- [36] Jinlu Zhang, Jiji Tang, Rongsheng Zhang, Tangjie Lv, and Xiaoshuai Sun. Storyweaver: A unified world model for knowledge-enhanced story character customization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 9951–9959, 2025. 2
- [37] Canyu Zhao, Mingyu Liu, Wen Wang, Weihua Chen, Fan Wang, Hao Chen, Bo Zhang, and Chunhua Shen. Moviedreamer: Hierarchical generation for coherent long visual sequences. In *The Thirteenth International Conference on Learning Representations*, 2025. 3
- [38] Yupeng Zhou, Daquan Zhou, Ming-Ming Cheng, Jiashi Feng, and Qibin Hou. Storydiffusion: Consistent self-attention for long-range image and video generation. *Advances in Neural Information Processing Systems*, 37:110315–110340, 2024. 1, 2, 5, 8
- [39] Cailin Zhuang, Ailin Huang, Yaoqi Hu, Jingwei Wu, Wei Cheng, Jiaqi Liao, Hongyuan Wang, Xinyao Liao, Weiwei Cai, Hengyuan Xu, Xuanyang Zhang, Xianfang Zeng, Zhewei Huang, Gang Yu, and Chi Zhang. Vistorybench: Comprehensive benchmark suite for story visualization, 2026. 3