Repair the Shot, Preserve the Story

Dependency-aware revision for multi-shot video with persistent object state

Abstract

Multi-shot video becomes difficult to revise once a plot detail spans several independently generated clips. We propose to make shot revisions transactional. A shared multimodal controller inspects generated frames, patches a persistent scene program, and regenerates the smallest supported set of affected shots. The study tests whether explicit dependencies reduce the compute and collateral changes needed to make a revision while preserving narrative continuity.

Keywords / agentic vision, multi-shot video, scene state, localized regeneration, Blender

Problem and prior work

A cup that breaks off camera must remain broken when it reappears, even after shots that never show it. A2RD uses retrieval and iterative generation to maintain long-video consistency [1]. VideoCoCo links executable Blender code to video synthesis [2]. Our target is revision locality after clips have already been accepted. We ask whether explicit shot dependencies predict where an edit must propagate and allow unrelated footage to be retained. Persistent appearance references alone do not specify which shots depend on an object's state change. This revision setting tests a specific form of feedback control within the broader agent-harness agenda [3].

Unified understanding and generation loop

Use one multimodal controller for script interpretation, visual inspection, and tool calls. It compiles a story into a scene program and shot graph. Each shot declares the object IDs, state fields and time intervals it reads, and any state changes it writes. A Python tool computes the affected dependency closure after a user edit or a mismatch detected in final footage. Blender renders updated cameras, object trajectories, masks and proxy views. A frozen video editor generates affected clips using proxy controls and accepted neighboring boundary frames.

Transactional repair

The controller inspects temporal windows with segmentation, tracking, object crops and cross-shot matching, then updates its diagnosis across rounds. Validate changed shot boundaries and retained constraints before committing a new version. Cache unaffected clips and check their hashes, while separately auditing whether they became semantically inconsistent. If declared dependencies miss an observed downstream effect, expand the affected set. Log what evidence triggered each regeneration.

Validation plan

Experimental design

Produce 80 short stories of four to six shots. Include object transfer, persistent damage or fill state, and reappearance after occlusion or a camera reversal. Cross requested edits with the number and position of affected shots. Hold out story skeletons and assets. Use separately authored gold dependency sets and inject a subset of localized errors to measure diagnosis. Compare global regeneration, rolling suffix repair, and a reference-retrieval repair loop without explicit dependencies, holding the initial clips, editor, and call budget fixed.

Outcomes and rejection criteria

Measure accepted revisions per generated frame, with acceptance requiring both the requested change and all original unaffected story constraints. Report number of re-rendered shots, preserved clip hashes, collateral semantic errors, and human-rated continuity. Include equal-budget curves and a separate audit of false dependency omissions. The hypothesis fails if restricting regeneration materially reduces edit success or causes continuity regressions on held-out stories.

Feasibility and deliverables

Weeks 1-2 create a shot graph and pilot scripts. Weeks 3-4 integrate Blender and the visual repair loop. Weeks 5-7 run evaluation. Keep early scenes to a small inventory of manipulable props and simple cameras. If a cut makes automatic identity tracking unreliable, use multi-view asset renders as references and report the difference from human diagnoses. Release story manifests, revision traces, and independently labeled shot dependencies.

References

[1] A2RD: Agentic Autoregressive Diffusion for Long Video Consistency. arXiv:2605.06924.

[2] VideoCoCo: Code-as-CoT for Physically-Consistent Video Generation via an Agentic Dual-Engine System. arXiv:2607.27380.

[3] Video Generation Meets Agent Harness: Agentic Video Generation, Agentic World Modeling and Recursive Self-Improvement. arXiv:2610.40191.