Episode Book

EPISODE #6595

How Terminal Video Editors Give AI Agents a Timeline

PUBLISHED	14 November 2026 at 15:07 UTC
DURATION	21:44
LISTEN	myweirdprompts.com/episode/terminal-video-editing-agents-16336/
HOSTS	Corn, Herman, Dorothy, Hilbert
SCRIPT MODEL	z-ai/glm-5.3
VOICES	omnivoice
PIPELINE	V5.26
SOURCES	20 cited by the research (listed at the end)
BOOK GENERATED	15 November 2026 at 11:16 UTC

SYNOPSIS

Command-line video editors are no longer just for sysadmins—they are the only editing surface an AI agent can reliably operate. This look at terminal video workflows covers melt for multitrack timelines, auto-editor for silence cuts and Premiere Pro transitions, and OpenTimelineIO as the interchange handshake into Resolve or Final Cut. Plus, findings from the Cassette-Editor paper show how a typed timeline document and rulebook boosted agent editing success from 47% to 89%.

TRANSCRIPT



Corn

The terminal quietly became a video editing suite, and nobody sent out a press release, because none of these tools has a splash screen. Daniel's latest wants us to walk past ffmpeg entirely and look at what lives beyond it — melt, auto-editor, the real editorial stuff — what each one is best at, what you can cut and composite from a shell. And he's pulling a thread through all of it: an agent can't drag clips around a GUI timeline, but it can drive a command perfectly. So how far can an agent actually get, and where does the human with the timeline still earn their keep?



Herman

And the timing on this one is almost comically good, because auto-editor shipped a release yesterday. We are not late to this. Let me get into the tools themselves, because the surprise isn't that they exist — it's how much they've grown up.



Corn

Grow up. That's the word I'd underline.



Herman

The real question hiding in Daniel's prompt isn't "what CLI tools exist" — it's why an ordinary person should care about terminal video editing at all. Twenty years ago the answer was "you're a Linux sysadmin with strong opinions." The answer now is that these tools are the only editing surface an AI agent can actually operate. Their power suddenly matters to everyone.



Corn

Because the agent can't click.



Herman

It can click, technically — there are agents that drive a mouse — but the research on that is brutal. Agents that operate a real editor through its graphical interface keep the project intact and complete very few tasks. Whereas a command line is text in, text out. That's the agent's native habitat.



Corn

So give me the map. Three tools, three jobs?



Herman

Three tools, one workflow, I'd say. melt is the multitrack timeline in a terminal — it's the engine underneath Kdenlive and Shotcut, both real GUI editors, so you're not getting a toy, you're getting their actual core exposed as commands. auto-editor is the first-pass specialist that has quietly become a small editor in its own right. And OpenTimelineIO is the interchange format — the file format that lets an edit made in one place survive into Resolve or Premiere without melting down into a render.



Corn

The handshake.



Herman

The handshake, exactly. And here's the tension I want to plant now and we can pay off later: the oldest and most powerful of the three is the least agent-friendly, and the youngest ships an installable skill for coding agents. Agent-readiness turns out to be a design choice, not a byproduct of power.



Corn

Nobody designs for the future by accident twice in a row. Start with melt.



Herman

melt. The MLT framework describes it as "a powerful, if somewhat obscure, multitrack command line oriented video editor," which might be the most honest self-description in open source. The architecture is beautifully clean if you come at it from ffmpeg: producers make frames from files or devices, filters modify frames, transitions collect from two tracks, consumers sink the result. And the filter guarantee is the part I love — for every frame a filter receives, it outputs precisely one frame. The docs say "never more, never less, ever."



Corn

They put "ever" on it.



Herman

With a period after "ever," yes. Somebody got into an argument in a bug tracker once, and we are living in the aftermath. But the editorial verbs are the real story — this is a genuine NLE command language. Minus-track for a new track, minus-blank for spacer, minus-split, minus-swap, minus-join clips, minus-mix for a crossfade region, minus-transition, minus-filter, minus-attach-cut to hang an effect on a specific cut. You build a whole multitrack timeline positionally, then minus-serialise writes the entire thing out as a text file you can replay.



Corn

A timeline as a text file. That's the sentence an agent wants tattooed somewhere.



Herman

It is, and then the docs immediately ruin it. Two warnings, verbatim: "Order is incredibly important" and "Error checking on command line parsing is weak." Every argument is positional. You move one flag and the whole thing reassembles differently, and nothing tells you.



Corn

The most powerful tool gives the worst feedback.



Herman

Hold that thought, because it comes back with a number attached. For now — auto-editor, which is the youngest and, honestly, the one I'd point a normal person at first. It started as one job: the silence cut. The default is edit audio threshold zero point zero four, all streams, with a fifth of a second of margin padding so it doesn't chop breaths. One command, dead space gone. The README calls that the "first pass," and it's very blunt that this is boring work, especially on long video.



Corn

Which is precisely the work a person should never have been doing.



Herman

Precisely the work you hand to something that doesn't get bored. But it's grown a programming model on top of that. Every moment in the source gets an integer label — zero means cut, one means keep, and it goes up to two hundred fifty-five classes, higher label wins. So you can say edit label two is audio below minus twelve dB, and then attach actions to label two — speed one point five. One command: cut silence, keep speech, speed up the loud parts.



Corn

That's not a filter, that's a rulebook.



Herman

And the action vocabulary has gone wild — speed with pitch preservation, varispeed if you want the tape-style pitch change, pitch shifts in semitones, volume fades, a de-esser, zoom, rotate, drawbox, and as of the Halloween release, burn and text. Burn draws an SRT or ASS subtitle file over the section — add burn talk dot srt — and the subtitles stay on their words through cuts and speed changes, which is the part that would take a human an afternoon.



Corn

The subtitle that chases its own word across a speed ramp. Someone solved a real problem there.



Herman

And yesterday's release, thirty-one point nine, renders every Premiere Pro transition natively. Push, clock-wipe, glitch, page-peel, the Legacy folder, audio crossfades — transition push half a second, done. Plus Final Cut XML import with connected clips, and caption tracks that land as proper subtitle streams depending on the container.



Corn

Okay. So what's the keystone, the thing that makes this workflow instead of a trick?



Herman

The export. Auto-editor can export to premiere, resolve, final-cut-pro, shotcut, kdenlive, plain EDL. The agent does the tedious pass in the terminal and hands a timeline to a human. And underneath that sits OpenTimelineIO — which is not a media container, that's the first thing to understand. It holds the cut information: order, length, references to external media. It's been around since twenty sixteen, it's Apache licensed, it's natively supported in most NLEs, and it has adapters for AAF, EDL, both flavors of Final Cut XML, even HLS playlists and SVG.



Corn

You said "handshake" as a joke earlier. That's the actual handshake document.



Herman

That's the notarized copy. So to answer Daniel's what-can-you-actually-edit question in one pass: cuts and pacing, auto-editor's bulk strength, better than a GUI for long material. Multitrack timelines, melt. Transitions, both — melt's transition system, and auto-editor now rendering Premiere's whole set. Silence removal, the signature. Subtitles that track their words. Compositing is partial but real — auto-editor has add layers, polygon masks, even convolve for impulse-response audio, and melt's filter graph is a genuine compositor. Speed ramps with easing.



Corn

Ramp with a caveat.



Herman

Several caveats. FCP and Resolve keep a ramp through the round trip; Premiere, Shotcut and Kdenlive flatten it to the mean speed. Compound clips get dropped on FCPXML import, with a warning, at least. Fidelity on the handshake is where the polish is still thin.



Corn

Well, before we get properly started on the agent half, I think we have a caller on the line. Dorothy, are you there?

D

Dorothy

Hello dear, it's me, Dorothy, with Dorothy's History of Toast.

D

Dorothy

In nineteen forty-two Britain brought in something called the National Loaf, to save wheat and shipping space, and it was a coarse grey wholemeal loaf that most people disliked.

D

Dorothy

They didn't do away with it until nineteen fifty-six, so that's fourteen years of grey bread nobody wanted, which rather sums up the history of toast if you ask me.

D

Dorothy

Anyway sweetheart, I've got a loaf proving on the radiator and it's gone a bit strange, so I'll let you get on.



Herman

Mum, always a pleasure. Fourteen years of grey bread. Put that in an adapter, why doesn't someone.



Corn

She'd get a warning on import. So — that's what the tools can do. Now here's why any of this matters to someone who isn't a video editor, and it's the agent thread Daniel was pulling.



Herman

There's a paper that landed last month and got revised a week ago — Cassette-Editor, Zeng and Zhong — and it's the cleanest test of the whole thesis I've seen. Their starting point is a comparison I found clarifying: coding agents work in an environment built for them. A compiler or a test checks each change and points at the fault. Version control keeps every change reversible. Video editing agents have none of that — the only place an edit's correctness shows up is the rendered video, which is expensive to produce and doesn't point at the decision that was wrong.



Corn

The compiler tells you the line. The render just stares at you.



Herman

And their diagnosis of why existing agentic editors fail is threefold. One, operation catalogs make editing unreliable, because edits have consequences — a trim moves later clips, moves attached titles, changes the source frames a transition blends. Operations that apply those consequences differently combine into states nobody anticipated. Two, validation is costly — nobody checks relationships between timeline items, like which clip a title is attached to. Three, pipeline output is hard to use, because a pipeline delivers a rendered video in which you can't undo one of the agent's changes.



Corn

That third one's the one a person feels. You get a finished video with one wrong cut and no way to reach it.



Herman

Their fix is the part I want your opinion on, actually. They give the agent the timeline document, not a filter graph — the agent writes typed patches to the document, a schema checks them like a type checker, and a rulebook checks them like a semantic checker. A rejected patch leaves the project unchanged and comes back with each problem located, with repair options.



Corn

Type theory applied to a video timeline. You're going to tell me the numbers now, and I'm going to enjoy them.



Herman

Same model, held fixed. Full harness passed eighty-nine percent of trials. The same model with ten basic editing operations and no rulebook — forty-seven percent. That's a forty-two point gap, p equals zero point zero zero zero four, across four hundred twenty-nine trials.



Corn

Forty-two points from the scaffolding, not the model.



Herman

Same model, and the gap isn't in finding content — both conditions passed eighty-three percent of the find trials. It's all consequences. Rule boundaries: ninety-four percent versus six. Composite edits: seventy-two versus eleven. Undo: ninety-two versus twenty-five. The basic-tools route left overlapping clips in five final documents. The full harness had no blocking violations in any of them.



Corn

Overlapping clips nobody caught, because the render doesn't file an error report.



Herman

And because basic tools failed more often, a successful task cost two point four times the agent time and two point two times the money. Being badly equipped is expensive, not just unreliable. There's a detail I love buried in it — the paper's phrase "editing parity." Manual gestures are writes in the same language, folded by the same executor and rulebook, so the agent can express every manual content change by construction. The human and the agent are editing the same document with the same rules.



Corn

That's the sentence I'd take home from the paper. Same language, same referee.



Herman

And it explains the melt paradox we parked earlier. Why does the richest editorial vocabulary produce the worst agent outcomes? Because melt's positional syntax with weak error checking is exactly the property Cassette-Editor shows kills agent success — no checker, no located errors, no clean rejection. Meanwhile auto-editor, the youngest tool on the shelf, ships an installable skill for coding agents and moved all its progress output to stderr so stdout carries only data. Someone over there is designing for agents on purpose.



Corn

Quietly, too. A stderr migration isn't a launch event.



Herman

It's plumbing, which is how you know it's sincere. We did an episode ages back on splitting an edit across surfaces — local edit, remote render — and I said then the interesting systems are the ones designed around the seam. This is the same shape of argument, one layer up: the seam is now between the agent's pass and the human's.



Corn

I remember, and you were right, which I'll only admit once. So the GUI route — agents clicking through a real editor. You said the research is brutal on it.



Herman

Brutal and brief. They keep the project but complete few tasks — that's the finding from two separate groups cited in the paper. Clicking through an NLE is the worst of both worlds: all the fragility of an operation catalog, plus a UI that resists automation at every step.



Corn

What's your read on the render problem, though? The paper says correctness only shows in the render — but an agent can render and then look at it.



Herman

That's where it gets hard. Vision-language models detect brief temporal glitches near chance. They struggle to align sounds with visual events. And even if a VLM spots something wrong, a render doesn't locate the faulty decision — you'd have to re-analyze the lot after every change. There's a cited result that just giving the agent oracle defect location raises repair scores by thirteen points. Location is most of the battle.



Corn

So the agent is a good editor and a bad audience.



Herman

That's... yes, actually, that's the shape of it. The timeline document fixes the writing side. The reading side — did this edit actually work — is still human eyes. And the paper is honest about the remaining gaps, which I appreciate. Even the full harness only got seventy-two percent on composite edits. Retimed and nested material is untested — almost everything in their test projects plays at normal speed, so the source-time mapping stays simple. Nobody's numbers cover the hard case yet.



Corn

And the broader field? There's more than one of these systems.



Herman

A whole crop, at varying stages. EditDuet from SIGGRAPH pairs an Editor agent with a Critic agent that gives natural language feedback — and tellingly, its most common failures included critic feedback its tools couldn't carry out, like adding transitions. Crayotter is an open-source multi-agent workbench for long-form that treats every intermediate artifact as first-class rather than hidden state, and it scores three point four out of five on human evaluation across twenty-three themes. Good, not magic.



Corn

The honest negative, though — is anyone shipping an agent-native CLI editor you can install today?



Herman

No. That's the gap. The closest things are auto-editor's skill, Cassette-Editor's research harness, and Crayotter's workbench. Nobody has wrapped all of it into a product where the agent holds the timeline, patches it under a rulebook, and hands you the file.



Corn

Which is odd, because the pieces are all lying on the table.



Hilbert

They're on the table, but the agent doesn't do the editing. I've just spent three evenings proving it. Ninety minutes of footage from the community hall quiz night, forty-two table shots, and I ran the whole first pass from the terminal — audio threshold, margins, the lot. The agent cut it down to eleven minutes and got one cut wrong, the bit where the raffle drum starts before anyone claps. Took me an hour in the GUI to fix that one cut. That's the correct division of labor, not a failure. And you keep the log — I still have the printout of every command it ran, folded in the manual for the mixer.



Corn

You printed the command log.



Hilbert

Paper, Herman. The render lies to you and the terminal doesn't. Also your levels were clipping around the twenty-minute mark of this episode — I'll pull it down in the mix. Where was I. The drum. No agent of mine is trusted with a drum roll.



Herman

The one-cut-an-hour ratio — that's roughly what the paper's undo numbers would predict, actually. Ninety-two percent with the full harness means the human is reaching in for the remainder.



Hilbert

Then the paper and I agree, which I'll hold against it. Though I'll say the printout was the useful part. When the drum cut went wrong I didn't scrub anything, I read back what it had done and knew in a minute which number to change.



Corn

The log as the undo button. Cheaper than a render and it already exists.



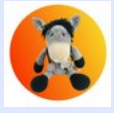
Hilbert

It was cheaper. It was free, it came out of the machine. Fine, that's a point.



Corn

So if you take one thing from this episode, take the forty-two points. Same model, same task, forty-two points of pass rate bought entirely by giving the agent a document with a referee instead of ten operations and a prayer. Power was never the bottleneck — the feedback loop was.



Herman

And the second half of that: the realistic workflow was never CLI or GUI. It's agent first pass, human finishing, with OpenTimelineIO or an FCPXML as the handshake between them. The tools that win will be the ones designed for that handshake, not the ones with the longest feature list.



Corn

Which leaves the real frontier sitting right where we found it: verification. If vision models can't see a two-frame glitch better than a coin flip, an agent can't check its own work — and every other problem here is solvable before that one is.



Herman

So the open question I'd leave with Daniel: what would an agent need in order to know its edit worked without a human scrubbing the timeline? Not more operations. A way to see.



Corn

Thanks as always to our producer Hilbert Flumingtop, who apparently also runs quiz night post-production. If this was your kind of episode, go back for episode seventeen, Cloud Render Superpowers; episode twelve, The AI Breakthrough; and episode two, Local STT For AMD GPU Owners. This has been My Weird Prompts, the human-AI collaboration podcast. If you've been enjoying the show, a review on Spotify helps more people find it, and everything lives at my weird prompts dot com.



Herman

See you soon, everyone.



Corn

See you tomorrow.

SOURCES

What the research for this episode read before the script was written. Primary sources first.

1. **GitHub, , retrieved 2026-11-14; releases 31.9.0 (2026-11-13), 31.8.0 (2026-10-30), 31.7.2 (2026-10-03). [primary]**
   https://github.com/WyattBlue/auto-editor

2. **auto-editor docs, , retrieved 2026-11-14. [primary]**
   https://auto-editor.com/docs/actions

3. **MLT Framework docs, , retrieved 2026-11-14. [primary]**
   https://www.mltframework.org/docs/melt/

4. **MLT home page (v7.42.0, 2026-10-03), , retrieved 2026-11-14. [primary]**
https://www.mltframework.org/

5. **MLT projects list (Kdenlive, Shotcut, GPLv3), , retrieved 2026-11-14. [primary]**
https://www.mltframework.org/projects/

6. **GitHub, , retrieved 2026-11-14; v0.18.1 (2025-11-09), v0.18.0 (2025-11-06). [primary]**
https://github.com/AcademySoftwareFoundation/OpenTimelineIO

7. **Zeng & Zhong, Cassette-Editor: An Agent Harness for Editing Video the Way You Do, v3 revised 2026-11-03, retrieved 2026-11-14 (full text read, pp. 1**
6).
https://arxiv.org/abs/2610.37690v3

8. **Video Generation Meets Agent Harness, submitted 2026-10-29, retrieved 2026-11-14.**
https://arxiv.org/abs/2610.40191v1

9. **Sandoval-Castaneda et al., EditDuet: A Multi-Agent System for Video Non-Linear Editing, SIGGRAPH 2025, retrieved 2026-11-14.**
https://arxiv.org/abs/2509.10761v1

10. **CutClaw: Agentic Hours-Long Video Editing via Music Synchronization, 2026-03-31, retrieved 2026-11-14.**
https://arxiv.org/abs/2603.29664v1

11. **Crayotter: Traceable Multi-Agent Workflows for Long-Form Video Editing, revised 2026-07-17, retrieved 2026-11-14.**
https://arxiv.org/abs/2606.07636v2

12. **Unified Agentic Video Editing Across Levels of Complexity and Creativity, revised 2026-10-29, retrieved 2026-11-14.**
https://arxiv.org/abs/2609.12769v3

13. **Aurora: Unified Video Editing with a Tool-Using Agent, 2026-05-18, retrieved 2026-11-14.**
https://arxiv.org/abs/2605.18748v1

14. **Agentic Visual Generation: From Generative Models to Agentic Control, 2026-09-06, retrieved 2026-11-14.**
https://arxiv.org/abs/2609.06758v1

15. **GitHub (dormant, last push 2022-02-20), retrieved 2026-11-14.**
https://github.com/wkentaro/video-cli

16. **GitHub (created 2026-06-03), retrieved 2026-11-14.**
https://github.com/swissmarley/gflow-cli

17. **GitHub (last push 2025-03-24), retrieved 2026-11-14.**
https://github.com/makalin/Videoflux

18. **GitHub (last push 2026-10-19), retrieved 2026-11-14.**
https://github.com/minimaster4734/favencoder

19. **GitHub, retrieved 2026-11-14.**
https://github.com/talwrii/ffmpeg-cookbook

20. **Wikipedia, retrieved 2026-11-14.**
https://en.wikipedia.org/wiki/Media_Lovin%27_Toolkit