Animated Explainer Videos in Claude Code

Install the 66-style film skill, run your first film, and build a style around your own brand.

Why I use this instead of a video model

Every AI video tool I had tried before this one bills by the second and forgets what it did in the last shot. You pay per generation, the second clip never quite matches the first, and the only fix is to reroll and pay again. For an explainer, where the whole point is that frame two looks like frame one, that is a bad trade.

Lemo-Opuscar takes a different route. It is an open source Claude Code plugin that renders films in code. Canvas and WebGL draw the picture, a text-to-speech engine speaks the narration, a score is written for the piece, and FFmpeg puts the three together. No video generation model is called at any point, so the render costs you nothing in credits. A 45 second film and a six minute film cost the same.

The library ships with 66 or more styles, each one a written specification rather than a prompt you hope lands twice. You pick a style, describe your idea in plain English, and the agent handles the story, the voiceover, the storyboards, the music and the sound effects. Your hands-on time is a few minutes. Then you walk away while it renders.

01

How it actually works

1. You name a style and a topic. The agent reads three guides to decide what to make: DIRECTOR.md for story, sound, rhythm and camera, TECHNIQUE.md for frame rendering, voice synthesis and music mixing, and styles/<style>/STYLE.md for the look. Your story supplies the content, the style file supplies the aesthetic.
2. It writes and scores the piece. The agent drafts the script, synthesizes the narration through a text-to-speech engine, writes an original score, and lays out the shot list with timings and camera moves. It will ask whether you want to see a storyboard before it commits.
3. It renders frame by frame in code. The visuals are drawn with canvas and WebGL rather than pulled from stock or generated by a model, then FFmpeg composites picture, score and voice into the final file. Default output is 1920x1080 at 24 frames per second.

02

The one thing almost everyone gets wrong

People run it on a cheaper model to save tokens, get a film that looks nothing like the style demo, and decide the styles are oversold. The styles are written for Claude Opus 5.5 specifically. The repository says

plainly that other models may not reproduce the styles accurately, and that is the single biggest cause of a disappointing first render.

> **WATCH OUT**
>
> Check your model before you start the run, not after. A film takes 30 to 60 minutes of agent time and a significant number of tokens, so a wrong-model run is an expensive way to learn this. Stay on Opus 5.5 for the render.

---

03 The repo and the prompts

> **SOURCE CODE**
>
> The full repo, MIT licensed, with all 66 styles and their demo films:
>
> **https://github.com/lemomo-ai/lemo-opuscar**

INSTALL.SH

```
# Recommended: install as a Claude Code plugin
claude plugin marketplace add lemomo-ai/lemo-opuscar
claude plugin install lemo-opuscar@lemolab

# Or clone the repo and work inside it
git clone https://github.com/lemomo-ai/lemo-opuscar.git
cd lemo-opuscar
claude
```

> **YOUR FIRST FILM**
>
> Make a 45-second film in the [STYLE NAME] style about [YOUR TOPIC]. [WARM FEMALE / CALM MALE] narrator.

> **START A CUSTOM STYLE FROM A LIBRARY STYLE**
>
> I want to change the [LIBRARY STYLE NAME] style into my own version. My brand colours are [HEX, HEX], my logo is at [PATH], and my font is [FONT NAME]. It is for [PLATFORM] at [FRAME SIZE].

CUSTOM-STYLE-FOLDER

```
~/lemo-my-styles/<your-slug>/
  style.json       # slug, names, base style, creation date
  STYLE.md         # the full style specification
  CHANGES.md       # what changed from the base, and why
  DEMO.md          # how the demo film was made
  demo/            # rebuildable source, including build.sh
  styleframe.jpg   # the frame you approved
  test.mp4         # the demo film
  test.jpg         # its poster frame
```

SITES SHOWN IN THE VIDEO

- OPUSCAR 98: 98 Best Picture winners, every frame coded
  https://lemomo-ai.github.io/lemo-opuscar/opuscar98/
- The interactive style gallery, all 66 styles with demo films
  https://lemomo-ai.github.io/lemo-opuscar/
- Source repository on GitHub
  https://github.com/lemomo-ai/lemo-opuscar

Set it up in Claude Code or Codex

1. Check the three local dependencies first, because all of them fail late and loudly if they are wrong. You need Node.js 20 or newer, FFmpeg 7 or newer, and Python 3.11 or newer (uv works too). Run node -v, ffmpeg -version and python3 --version in a terminal and read the numbers.
2. If FFmpeg reports 6.x, replace it before you go further. Ubuntu 24.04 ships 6.1 and that version is too old for this. Grab a static FFmpeg 7 build and put it ahead of the old one on your PATH. On macOS, brew upgrade ffmpeg is usually enough.
3. Install the plugin in Claude Code: claude plugin marketplace add lemomo-ai/lemo-opuscar, then claude plugin install lemo-opuscar@lemolab. The skill is lemo-opuscar@lemolab. Both commands have to run, in that order, or the install command will not find the package.
4. If you would rather read the source as you go, clone it instead: git clone https://github.com/lemomo-ai/lemo-opuscar.git, then cd lemo-opuscar and start claude inside that folder. The agent picks up DIRECTOR.md, TECHNIQUE.md and the styles/ folder from the working directory.
5. For Codex, use the clone route. The plugin marketplace is a Claude Code feature, so clone the repo into your project and add a line to your AGENTS.md pointing at DIRECTOR.md, TECHNIQUE.md and styles/<style>/STYLE.md so the agent reads them before it starts. If you keep skills in .codex/skills/<name>/, the folder name has to match the name field in the skill file, or the agent will not load it.
6. Decide on a voice. Kokoro is the default text-to-speech engine and it ships with the repo, so there is no API key to set for a first run. Microsoft Edge TTS and Doubao (Volcengine) are the alternatives if you want a different voice. If you pick one that needs a key, ask Claude to add it to your shell profile, then open a brand new terminal window so the value is actually loaded.
7. Set your model to Claude Opus 5.5 before you start. The styles are tuned for it.
8. Run the first test. Open the style gallery, pick a style whose demo you like, then type: Make a 30-second film in the watercolor style about [something you already know well]. Warm female narrator. Say yes when it offers the storyboard preview, so you catch a wrong direction before the render instead of after.
9. A successful first run looks like this: the agent asks a few clarifying questions, shows you a shot list, then works for 30 to 60 minutes and leaves an MP4 at 1920x1080 and 24 frames per second, with narration and a score already mixed in. If it finishes in two minutes, something rendered empty and you should check the FFmpeg version first.

💡 QUICK TIPS

- Watch the demo film for a style before you spend a run on it. The gallery has one for every style, and five minutes of browsing saves an hour of rendering the wrong look.
- Start a run when you are about to do something else. The agent needs 30 to 60 minutes and sitting there watching it is a waste of your afternoon.
- Custom styles live in ~/lemo-my-styles/, which sits outside the library on purpose. Plugin updates replace library styles and leave yours alone, so build your brand style once and stop worrying about updates.

04

The commands I actually run

OPEN THE SKILL DIRECTLY

```
/lemo-opuscar:lemo-opuscar
```

AN EXPLAINER IN A NAMED STYLE

```
Make a [45]-second film in the [Data Storytelling] style explaining [WHAT YOU ARE EXPLAINING] to [WHO IT IS FOR]. Show me the storyboard before you render. [WARM FEMALE] narrator.
```

BUILD A BRANDED STYLE FROM REFERENCES

```
Start a new style from these references: [PATHS TO 2-3 IMAGES OR A FILM I LIKE]. Name the two or three things in them that make them work and map each one to a style section. Brand colour [HEX] as the accent, logo at [PATH], font [FONT NAME]. Keep the base light and dark structure so my text stays readable. Target [PLATFORM] at [FRAME SIZE].
```

CHANGE ONE THING WITHOUT A FULL REROLL

```
In the finished film, change [WHAT IS WRONG: a colour, a shot length, a line of narration] and rebuild from the demo source with build.sh. Leave everything else as it is.
```

UPDATE THE PLUGIN

```
claude plugin marketplace update lemolab && claude plugin update lemo-opuscar@lemolab
```

When it breaks

ERROR	FIX
The film looks nothing like the style demo	You are on the wrong model. The styles are built for Claude Opus 5.5 and the repo says other models may not reproduce them accurately. Switch model and run it again.
FFmpeg errors, or the render produces a broken or empty file	Run ffmpeg -version. If it reports 6.x you are too old. Ubuntu 24.04 ships 6.1; install a static FFmpeg 7+ build and put it first on your PATH.
claude plugin install cannot find lemo-opuscar@lemolab	The marketplace was never added. Run claude plugin marketplace add lemomo-ai/lemo-opuscar first, then the install command.
Node or Python errors as soon as the render starts	Check node -v for 20 or newer and python3 --version for 3.11 or newer. uv works in place of a system Python.
The run eats a huge number of tokens and feels stuck	That is normal. A film typically needs 30 to 60 minutes of agent runtime and a significant number of tokens. Leave it alone and come back.
The narration voice is wrong for the piece	Kokoro is the default. Switch to Microsoft Edge TTS or Doubao, and say the voice you want in the prompt (for example, warm female narrator or dry documentary male).
A custom style disappeared after an update	Look in ~/lemo-my-styles/<slug>/, not in the library. Updates replace library styles and leave that folder untouched, so your style is almost certainly still there.
You want to reuse a demo film or a still in client work	Check the licence first. Code and docs are MIT, demo films and stills are CC BY 4.0 with credit to LemoLab, and three voice tracks (Game Show Flat, Paper-cut Lightbox, Broadcast News Package) keep their original licences and should not be reused.

Ready to go deeper?

Want the videos made for you? Paste any link or idea into Content Engine and get a published video in under 3 minutes, with DM automation armed to capture leads.

Meet Content Engine →

https://builds.digicuratoragency.com/content-engine/