The 5 Claude Video Styles Breakdown

What HyperFrames, Onetake, Collage B-roll, Claude Animation Base, and Lemo Opuscar each need as input and how to install them.

Why I broke these five apart

Claude Code can make a video for you in a dozen different ways, and most people only ever try the first one that works. That's how every Claude-made video ends up looking like the same motion graphics template. I pulled apart five Claude Skills that each produce a video a different way, different input, different renderer, different look, so you can pick the one that fits your content instead of defaulting to whatever worked first.

This isn't a ranking. HyperFrames, Onetake, Collage B-roll, Claude Animation Base, and Lemo Opuscar solve different problems. Read the input and output columns below and match the skill to the video you're actually trying to make.

01

How it actually works

1. Each skill is a Claude Skill: a folder with a SKILL.md file that tells Claude Code or Antigravity how to do one job, plus whatever scripts or templates it needs to do it.
2. You install a skill by putting its folder where your agent looks for skills, then you ask for a video in plain language and the skill handles the actual rendering.
3. The five skills covered here differ in what they take as input, HTML, a single camera prompt, a set of still images, a character brief, or plain code, which is what gives each one its own visual identity instead of all of them converging on the same look.

02

The one thing almost everyone gets wrong

People try to make one skill do every job instead of matching the skill to the video.

> **WATCH OUT**
>
> If you run HyperFrames when you actually want an illustrated character video, you'll get clean motion graphics that don't fit the tone you wanted. Pick the skill based on the output style you need first, then let that decide your input, not the other way around.

03

The five skills, side by side

SKILL	INPUT	OUTPUT STYLE	LICENSE / COST
HyperFrames by HeyGen	HTML	Motion graphics	Open source on GitHub
Onetake	A single camera prompt	One continuous shot, no cuts	Free for non-commercial use
Collage B-roll	Still images	Halftone paper cutouts, animated by Google's Gemini	Open source on GitHub
Claude Animation Base	A character brief	Hand-painted cartoons starring Claude's own mascot	Open source on GitHub
Lemo Opuscar	Code, no video model	43 distinct film styles	Open source on GitHub

Set it up in Claude Code or Codex

1. In Claude Code, skills live in ~/.claude/skills//. The folder name has to match the name: field inside that skill's SKILL.md, or Claude won't resolve it correctly.
2. In Codex, put the skill in your project folder, or under .codex/skills//, and add a one-line pointer to it in your AGENTS.md so Codex knows to load it.
3. Find the skill's GitHub repo (search the skill name plus "GitHub", since each one is maintained separately) and clone it: git clone , then move the resulting folder into your skills directory under the exact name the skill expects.
4. If the skill calls an external API, HeyGen for HyperFrames, Gemini for Collage B-roll, ask Claude Code to add that key to your shell profile (for example export HEYGEN_API_KEY=... in ~/.zshrc), then open a new terminal window so the key loads before you run anything.
5. Run a first test with a small, low-stakes input, one short HTML snippet for HyperFrames, one still image for Collage B-roll, one short character brief for Claude Animation Base, so you can see the skill's output style before you commit real content to it.
6. A successful first run produces a short rendered clip in the skill's output folder (check its SKILL.md for the exact path) that matches the style in the table above. If you get a text response instead of a video file, the skill didn't actually render, check your API key first.

> 💡 **QUICK TIPS**
>
> - Install one skill at a time. Running all five before testing any of them makes it hard to tell which one is misconfigured when something fails.
> - Keep a short note of which skill you used for which video so your content stays visually consistent instead of drifting style to style.

04

The commands I actually run

MOTION GRAPHICS FROM HTML

```
Using the HyperFrames skill, turn this HTML into a 16:9 motion graphics video: [PASTE HTML OR DESCRIBE THE LAYOUT]
```

ONE CONTINUOUS SHOT

```
Using the Onetake skill, render a single continuous camera shot of: [DESCRIBE THE SCENE AND CAMERA MOVE], no cuts, non-commercial use.
```

COLLAGE B-ROLL FROM STILLS

```
Using the Collage B-roll skill, animate these images into halftone paper cutouts that assemble on screen: [LIST IMAGE FILES OR PATHS]
```

HAND-PAINTED CARTOON

```
Using the Claude Animation Base skill, create a hand-painted cartoon scene starring Claude's mascot where: [DESCRIBE THE SCENE OR SCRIPT]
```

CODE-BUILT FILM STYLE

```
Using the Lemo Opuscar skill, build this video in code using the [FILM STYLE NAME, e.g. "16mm grain" or "VHS"] style, no video model: [DESCRIBE THE SHOT LIST OR SCRIPT]
```

When it breaks

ERROR	FIX
Claude says it can't find the skill	Check the folder name under ~/.claude/skills/ matches the name: field in that skill's SKILL.md exactly, including case.
Skill runs but returns text instead of a video	An API key is missing or wasn't loaded. Add it to your shell profile and open a new terminal window before retrying.
HyperFrames output looks like a static image, not motion	Your HTML likely has no animation or transition defined. Add explicit motion (CSS transitions, keyframes) to the HTML you feed it.
Onetake output has visible cuts	You described multiple scenes in one prompt. Onetake is built for a single continuous shot, narrow the prompt to one camera move.
Collage B-roll images don't animate	Check the image files are in a supported format and sized reasonably. Oversized files can cause the skill to silently skip the animation step.
Lemo Opuscar renders the wrong film style	Double check the exact style name against its list of 43 styles. A close but incorrect name will fall back to a default look instead of erroring.
Claude Animation Base mascot looks inconsistent between scenes	Reuse the same character brief text across every prompt in a project instead of rewriting the description each time.

Ready to go deeper?

This playbook is the short version. Inside the hub you get the full build walkthroughs, the prompt library, and the systems I use to ship this in a weekend.

Join the Build →

https://hub.digicuratoragency.com/join