The Motion Playbook

4 Claude Code Skills for Remotion Motion Graphics That Don't Look Like Slop

Why AI motion graphics look cheap, and what fixes it

AI can write Remotion code today. Ask Claude Code for a scene and you get a working React component in a minute. The problem is what it looks like: purple gradients on every background, zooms that push for no reason, screen demos no real user would recognise, and text that animates in without pointing at anything.

None of that is a Remotion problem. Remotion renders real, frame accurate video from code, so it can do broadcast quality work. The gap is direction. The model reaches for whatever is most common in its training data, and the most common motion graphics on the internet are the tired ones.

So I stopped re-explaining good motion design in every prompt. I put the rules into four Claude Code skills, installed them once, and now I call them by name while I build each shot. The whole set is open source and in the repo below. This guide covers what each skill fixes, how to install them for Claude Code or Codex, and the prompts I actually run.

01

How it actually works

A skill is a folder with a SKILL.md file inside it. Claude Code (and Codex, Cursor, and the rest) read that file when the task matches, and the rules inside shape the code they write. That is the whole trick.

1. You describe the shot in plain language and name the skill you want applied, for example "build the intro shot using the purposeful motion skill".
2. Claude reads the skill, then writes the Remotion component with those design rules already baked in: what may move, what must hold still, which easing to reuse, how a demo has to look.
3. You review the shot, layer the next skill on the next shot, and render with Remotion once the sequence feels cohesive.

Because the rules live in the skills and not in your memory, the tenth video looks as sharp as the first.

02

The one thing almost everyone gets wrong

People install the skills and then keep prompting the way they always did: "make an animated intro for my app". That prompt has no idea to land, so the model animates everything a little and nothing on purpose.

WATCH OUT

Decide what the shot has to land before you animate anything. Give one element the job of carrying that idea. Animate only what supports it and hold everything else still. A frame where five things move at once reads as noise; a frame that emphasises one thing reads instantly.

03

The four skills and what each one fixes

Each skill targets one failure point. You install all four once and call whichever one the shot needs.

SOURCE CODE

The full skill set, the install script, and the docs live here:

https://github.com/nessalazne/aisocialhub-motion-graphics-skills

SKILL	WHAT IT FIXES
Purposeful Motion	Aimless zooms and drifts. Every movement now reveals the idea on screen instead of filling time. One idea per shot, one element to carry it.
Shot Cohesion	Jarring cuts and mismatched styles. Colour, pacing, and easing stay consistent across every shot so the piece feels like one deliberate video. Reuse the same easing curves, lock a small colour set.
Real Demos	Fake screen recordings, the instant tell. App and terminal demos read as genuine, with real layouts, cursors, and interaction states instead of plastic mockups.
Focus Highlight	Frames with no emphasis. The single thing the viewer needs to see gets a clean highlight, scale, or reveal. Emphasise one detail per frame, and use scale or a highlight rather than more motion.

A fifth habit is baked into all four: the skills tell Claude to use the correct brand logos instead of guessing, so a product shot shows the real mark rather than a placeholder.

In the repo the four skill folders are named **motion-graphics** (research and asset checks, reference selection, production gates), **cinematic-camera** (keyframed camera movement through a continuous scene), **terminal-inserts** (authentic CLI demo formatting), and **article-highlights** (editorial cards with rough.js highlighter strokes). Two rules run through all of them: never restate the voiceover on screen, and no neon, glow, bloom, or cyberpunk styling. Headlines stay at 64px or larger so they read on a phone.

QUICK TIPS

- Before every build the skills ask two intake questions: the format (4:3 at 1440x1080 for a half screen insert, or 9:16 at 1080x1920 for full screen mobile) and the background (warm light, clean dark, light grid, or dark grid). Answer them, do not skip them.
- Every clip goes through an independent visual critic pass before it counts as done. Let it run.

Set it up in Claude Code or Codex

About five minutes, one time. You need Node.js and a terminal. The skills themselves are plain markdown, so there is nothing to compile.

1. Open a terminal and clone the repo:
   `git clone https://github.com/nessalazne/aisocialhub-motion-graphics-skills.git`
   then `cd aisocialhub-motion-graphics-skills`.
2. **If you do not have a Remotion project yet**, let the install script create one with all four skills already in place: `bash install.sh` (or `bash install.sh my-project` to name the folder). Skip to step 5.
3. **Claude Code, this project only:** from inside your existing Remotion project run `mkdir -p .claude/skills && cp -R /path/to/aisocialhub-motion-graphics-skills/skills/* .claude/skills/`. Replace the path with wherever you cloned the repo.
4. **Claude Code, every project:** `mkdir -p ~/.claude/skills && cp -R /path/to/aisocialhub-motion-graphics-skills/skills/* ~/.claude/skills/`. **Codex, Cursor, OpenCode:** same copy, into `.agents/skills/` inside your project: `mkdir -p .agents/skills && cp -R /path/to/aisocialhub-motion-graphics-skills/skills/* .agents/skills/`.
5. Add the official Remotion skill as a companion so the agent knows the framework itself: `npx skills add remotion-dev/skills`.
6. Check the folders landed. You should see `motion-graphics`, `cinematic-camera`, `terminal-inserts`, and `article-highlights`, each with a `SKILL.md` inside. The folder name is how the agent finds the skill, so do not rename them.
7. Open a new Claude Code or Codex session inside the Remotion project (skills are read at session start). Ask: "Which motion graphics skills do you have available?" and it should list the four.
8. First real run: paste one of the prompts in the next section, answer the two intake questions, then preview with `npx remotion studio` and render with `npx remotion render <CompositionId> out/clip.mp4`.

> 💡 **QUICK TIPS**
>
> - Cloned into the wrong place? Skills only load from .claude/skills, ~/.claude/skills, or .agents/skills. Anything else is ignored silently.
> - You can keep both the project copy and the global copy. The project copy wins when the two disagree.

04

The commands I actually run

Paste one and swap the bracketed part. Naming the skill is what makes the difference, so keep that in.

> **BUILD ONE SHOT**
>
> ```
> Build a 6-second 1080x1920 clip for this script line: [PASTE THE LINE]. Use the motion-graphics and cinematic-camera skills. Before you write code, tell me the one idea this shot has to land and which single element carries it.
> ```

MAKE A SET OF SHOTS FEEL LIKE ONE VIDEO

```
Here are my three Remotion compositions: [NAMES]. Using the motion-graphics skill, audit them for cohesion: list every easing curve, colour, and font in use, then reduce them to one shared set and apply it to all three.
```

A DEMO THAT DOES NOT LOOK FAKE

```
Using the terminal-inserts skill, build a 4:3 1440x1080 insert that shows [THE COMMAND] being typed and run, with a realistic prompt, cursor, and output. Warm light background. Do not restate the voiceover on screen.
```

POINT THE EYE AT ONE THING

```
Using the article-highlights skill, make a 9:16 editorial card from this quote: [QUOTE]. Highlight only the phrase [PHRASE] with a rough highlighter stroke. Everything else holds still.
```

FIX A SHOT THAT FEELS LIKE SLOP

```
This shot looks generic: [COMPOSITION NAME]. Run it through the motion-graphics skill's critic pass. List what is moving for no reason, what is emphasised, and what the correct brand logo should be. Then rebuild it with one purposeful movement.
```

When it breaks

The failures I hit while setting this up, and the fix for each.

ERROR	FIX
The agent does not mention any of the skills	It was started before the files were copied. Close the session and open a new one inside the project folder.
Skills copied but nothing in the list	Check the path. The four folders must sit directly inside .claude/skills, ~/.claude/skills, or .agents/skills, each with its own SKILL.md.
It skipped the intake questions and picked a format	Ask it to run the intake first: "Before building, ask me the format and background per the motion-graphics skill."
Output has glow, neon, or a gradient background	Name the skill in the prompt. Without it the model falls back to its defaults. Re-run with "using the motion-graphics skill" and say which background you want.
npx remotion render cannot find the composition	Run npx remotion studio, read the composition id from the left panel, and pass exactly that id.
bash install.sh fails	You need Node.js installed and a network connection. Run node -v first; install Node from nodejs.org if it prints nothing.
Demo shot still looks like a mockup	Use terminal-inserts for CLI demos and give it the real command and real output. The skill cannot invent what your app actually shows.

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