

# Cross-Multiply Autonomous Video Spec

Hand Claude a folder of influencers and a folder of products, tell it to cross them, and let it run to delivery without you.

---

## 01 When to use this

Use this once a single hero frame and video pairing works end to end. This is the prompt that turns one working clip into a fleet. Do not run it before the unit works, because it multiplies whatever the unit does.

## 02 The prompt

Inputs:

- Influencers: **<INFLUENCER\_FOLDER>**
- Products: **<PRODUCT\_FOLDER>**

Cross them. Every influencer against every product. Influencer A on product A and product B, influencer B on product A and product B, and so on for everything in those folders.

For each pairing:

1. Generate a hero frame of that influencer with that product in a plausible setting
2. Write **<N>** scripts using this rule, apply it yourself per script:
  - under 22 words means generate an 8 second video
  - 22 to 30 words means generate a 10 second video
  - never exceed 30 words
3. Generate **<K>** video candidates per script, using the hero frame as the start frame

Run this autonomously. Do NOT stop to ask me anything until the final delivery step. Parallelize wherever you can, but cap concurrent video jobs at 3, because the platform caps around 8 and larger batches fail. Retry any failures one at a time.

Then run automated frame QA on every clip before I see it:

- Split each clip into 1 second frames and screenshot each one
- Feed the frames to a vision model and ask: is anything malformed, does the product persist across every frame, is the on-pack text legible, do the hands make sense
- Auto-delete any clip that fails
- Cut the first 3 frames of every surviving clip, bad opening frames are common

Deliver the survivors to **<OUTPUT\_FOLDER>** with a contact sheet I can scan.

### 03 What it is actually doing

Everything before this note produced one clip at a time. This prompt turns that into a fleet.

Cross-multiplying is the core idea. Two influencers and two products is not four separate jobs you set up by hand, it is one instruction: cross them. Add more folders and the count grows on its own. Ten influencers, ten settings, ten scripts gives a thousand candidates.

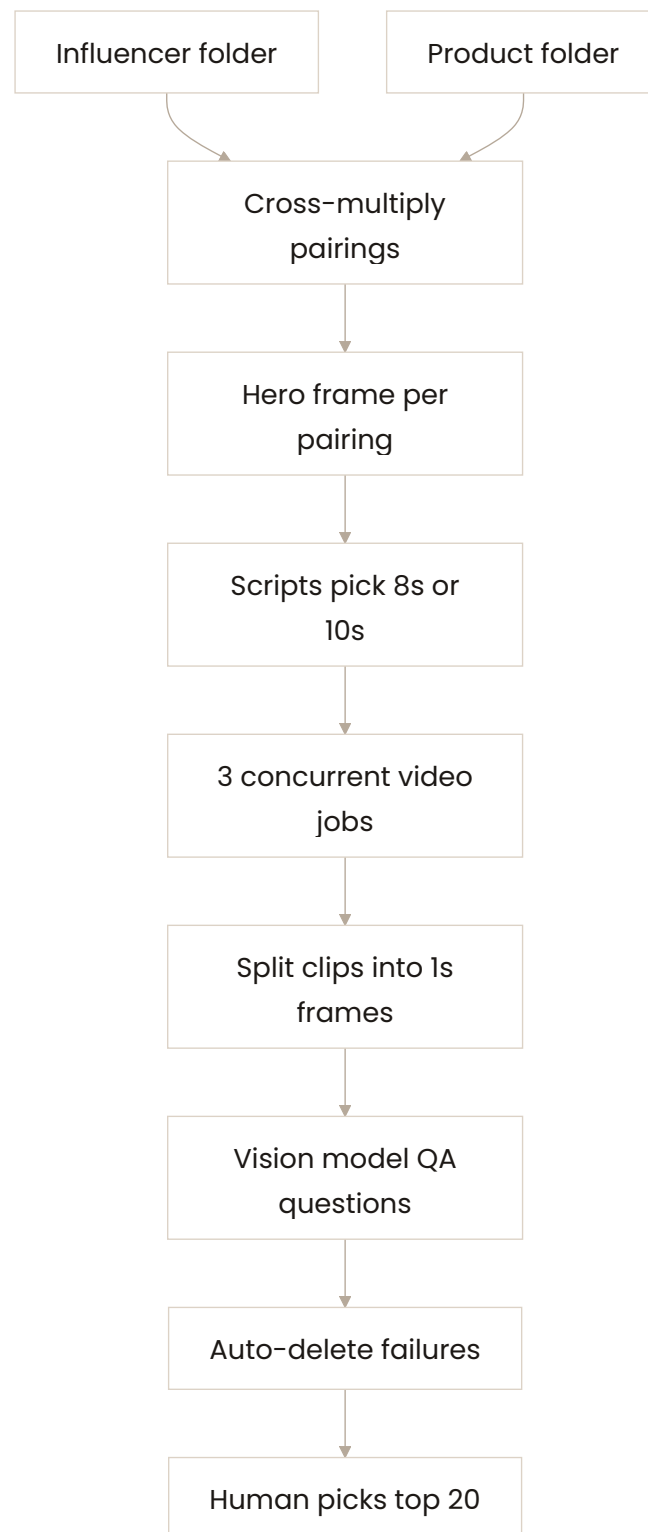
Autonomy is the second half. You tell Claude not to stop and ask you anything until delivery. That means every rule it might have asked about has to be in the prompt already, including the word count threshold from the script note, so it self-selects eight or ten seconds per script without checking in.

Nick's live run was four pairings times three candidates, twelve jobs. They hit the concurrency cap and failed. He updated the skill to cap at three concurrent and retry failures one at a time. That is the shape of most fixes at this scale: not better prompting, just respecting a limit.

Then there is the part nobody expects to need. A thousand clips is impossible to watch. Ten seconds each is nearly three hours of viewing, and most of it is broken. So you add an automated frame QA layer. Split each clip into one second frames, screenshot them, hand them to a vision model, and ask four blunt questions: is anything malformed, does the product persist across frames, is the on-pack text legible, do the hands make sense. Delete the failures automatically.

That trims a thousand down to maybe two hundred fifty to five hundred. Now you are looking at forty to eighty minutes of human review to pick a top twenty. You have replaced a creative department's throughput and you only paid for generations.

## 04 How it flows



---

## 05 Step by step

---

### 01 Prove one pairing first

Run a single influencer and product end to end before you cross anything. This prompt multiplies whatever the unit does, including its flaws.

---

### 02 Point at folders, not files

Give Claude an influencer folder and a product folder. Adding one file to a folder grows the matrix without you rewriting the prompt.

---

### 03 Say cross them explicitly

Spell out that every influencer pairs with every product. Ambiguity here produces one clip per input instead of the full matrix.

---

### 04 Encode the duration rule

Write the word count threshold into the prompt so it self-selects 8 or 10 seconds per script. It cannot ask you, so it has to decide.

---

### 05 Ban check-ins until delivery

State that it should not stop to ask anything until the final step. Anything it might have asked about belongs in the prompt already.

---

### 06 Cap concurrency at three

Nick's 12 jobs hit the platform's roughly 8 job cap and failed. Cap at 3 concurrent and retry failures serially.

---

### 07 Add the frame QA layer

Split every clip into 1 second frames, screenshot them, ask a vision model the four questions, and auto-delete failures. Then cut the first 3 frames of every survivor, since bad opening frames are common.

---

08

**Deliver a reviewable set**

Send the survivors plus a contact sheet. Two hundred fifty good clips with a review screen beats a thousand raw files in a folder.

---

---

**06 Words explained**

---

**Cross-multiply**

Pair every item on one list with every item on another. Two influencers and two products makes four pairings.

---

**Autonomous run**

The agent works from start to finish without stopping to ask you questions. Every decision has to be settled up front.

---

**Frame QA**

Checking a video by looking at its individual frames instead of watching it. A machine can do this at a scale no person can.

---

**Vision model**

A model that looks at images and answers questions about them, like whether hands look wrong or a product disappeared.

---

**Candidate**

One generated attempt. You make many because most are unusable, then keep the few that work.

---

**On-pack text**

The words printed on the product itself, like the label. Video models garble it often, which is why QA checks for it by name.

---

---

## 07 Watch out for

- 
- Nick's 12 jobs hit the platform's roughly 8 job concurrency cap and failed. Cap at 3 and retry failures one at a time.
- 
- The product vanishes or teleports mid shot. This is common and it is exactly what the frame QA layer is there to catch.
- 
- Bad opening frames are frequent. Cut the first 3 frames of every clip rather than trying to prevent them.
- 
- Male generations fail noticeably more often because there is less training data on men, so budget extra attempts for them.
- 
- Skin that is too smooth reads as AI immediately. Add grain and a light background audio bed to close the gap.
- 

---

## 08 How it connects

---

### Hero Frame and Video Generation

That note is the unit of work this one repeats, and its concurrency cap becomes a hard rule in the spec.

---

### UGC Script Generation

The word count rule from that note gets encoded here as an explicit threshold, so the run self-selects duration with nobody watching.

---

### Batch Generation and Contact Sheet

Same pattern as the image side: generate far more than you need, then hand a human one review screen rather than a folder.

---

## Compounding Failure in AI Pipelines

Volume of attempts plus a cheap selection step is the whole strategy, and the frame QA layer is how selection stays cheap at a thousand clips.

---

## UGC Video Ad Pipeline

This is steps five and six of that chain plus the QA layer that makes the scale math actually work.

---

---

**Credit.** Framework and approach by Nick Saraev, from the cross-multiply and frame QA segment of the UGC video build in his Zero to One course on Claude Code for marketing. Written up for the Kape vault by BrewedOps.