Contents

Using acue 3
Before anything: the checkouts 3
Make one image 4
Set the size, seed and batch 5
Make a song 5
Animate a picture 6
Run a folder of prompts 6
A comic: one cast, many panels 8
Bundle a finished story 10
Make a book of your own photos 15
Compare two models on the same prompts 17
Look at a comparison instead of reading one 18
Read this away from the screen 19
Which image model, and what each one is like 19
Draw on the Spark instead of Forge 21
Answers without starting the interface 21
A server you thought you stopped 22
When something looks wrong 23
Writing a picture story 24
Books entertain 24
Split the day before you write anything 24
The three layers, and the one question that sorts them 25
A model draws words; it does not take instructions 26
Hold a character with one object, not with a face 26
Give something to the day, and let it travel 27
Anyone you do not describe becomes the protagonist 28
Introduce, accompany, and let go — with framing 28
Turning day into night is one line 29
Deny text, and then stop asking for it 30
Panels are instants, not events 31
Motion is a line under the panel 31
Two moments on one page: --panel 31
Words are a panel too: --card 32
Words spoken over a scene: --speak and --narrate 33
How long a panel holds: --hold 33
Vary the shot, or the model will not 33
Scene length, and why six 34
How many panels is how long a video 34
Put the shared words first, the panel next, the scene last 34
A book is a city, and the file beside the stories says so 34
Seeds do not hold a story together 35
Know what is reading your prompt 35
Name the file for the shelf, the story for the reader 35
Check before you generate 36
Writing a soundtrack 37
A book is a city 37
Four rules 37
Stable Audio 3 Medium, the default 38
ACE-Step 1.5 takes direction differently from the model before it 39

Which ACE-Step model 40
The cover chain is the mechanism 40
Moods: a track without writing its prompt 41
No music, or your own 41
Under a narration 41
Songs: a short sung song a segment 41
Track length is arithmetic, not a guess 44
Re-cutting a finished bundle 45
The frame is 16:9, and the bars on an ultrawide are the player's 46
Hold one meter across the set 46
The five cities 46
What a caption can specify 46
Sampling belongs to the checkpoint, not to taste 47
A pinned key is not what breaks a track, but do not pin one anyway 47
The plan's own knobs 48
What guidance does on a turbo model: nothing audible 48
Per-city ACE-Step knobs 48
Judging a change, without fooling yourself 48
When a track is not landing 49
Auditioning 49
Check before you build 50

Using acue

Task by task: what to type, what happens, and where the images end up. This is the *guide*; acue(1) is the reference. When you want to know precisely what a flag does or what every key is bound to, read that instead — nothing here repeats it, so nothing here can contradict it.

```
man -l man/acue.1         # no installation needed
man acue                  # installed from the package
```

Wherever this guide says *beside the binary*, an installed acue (sudo apt install acue) means ~/.config/acue/ instead: acue.toml and servers.toml go there, and its own copy of models.toml if you make one. The packaged defaults are in /usr/share/acue/. Installed, what you write and what acue makes go into your own folders, as your desktop names them (~/Bilder on a German one):

What	Where
your stories, the prompts folder, and photo book drafts	<Documents>/Acue/
finished books, .acue and .sha256	<Pictures>/Acue/
their videos	<Videos>/Acue/
auditions	<Music>/Acue/auditions/
work in progress	~/.local/state/acue/out/, out of sight

[paths] in acue.toml moves any of them: books, videos, auditions, prompts, and out, which alone puts the books, videos and auditions under it too, the way a checkout keeps them. acue --print-paths prints all of them.

Before anything: the checkouts

acue contains no backend. It starts three, Forge Neo for pictures, sa3-server for soundtracks and ACE-Step for songs, and finds them in two different ways.

Forge Neo it searches for, in six places, stopping at the first hit: --repo, $ACUE_REPO, the forge server's repo in servers.toml beside the binary, up from the working directory, up from the binary, then beside the binary. It has to be found rather than only named, because everything the image half does needs it before there is any other configuration to read — and the servers.toml step is what lets the binary and the checkouts live in unrelated places.

The audio servers it is told about. servers.toml beside the binary gives each server a repo, and a server with none is one on another machine that acue does not start: the animator, the imager that can draw a book's pictures on the Spark, the agent that drafts lyrics, narration and captions, and the speaker that reads the narration. There is nothing to search for, because a server is a thing you declare rather than a thing lying around.

The usual layout is the checkouts as siblings:

```
image/
  acue/                   this
  sd-webui-forge-neo/     images
  sa3-server/             soundtracks
  ACE-Step-1.5/           songs
```

If Forge cannot be found it says so and names the three options that are answers rather than

© Andre Bremer · https://hyperquader.com

guesses. Without a servers.toml there is one built-in Forge entry and no audio or video at all. examples/checkouts/ has one to copy. Check that it found one, and what is on disk, without starting anything:

```
acue --list
```

A server that takes an API key. The servers on another machine, and Stable Audio, take a key in their servers.toml entry, sent with every request to that server as a bearer token. Write it one of three ways:

```
[[server]]
id         = "imager"
kind       = "imager"
modalities = ["image"]
model      = "zit-spark"
host       = "spark.lan"
port       = 9602
key        = "keyring"                # the desktop keyring's key for spark.lan
# key      = "$ACUE_IMAGER_SECRET"    # or an environment variable
# key      = "the key itself"         # or the key, which makes the file a credential
```

For "keyring", store the key once for the entry's host. It is read without echo, or from a pipe, and never from the command line:

```
acue --store-key spark.lan
acue --clear-key spark.lan              # and to remove it
```

The variables are named for the Spark's roles: ACUE_AGENT_SECRET, ACUE_IMAGER_SECRET, ACUE_SPEAKER_SECRET and ACUE_ANIMATOR_SECRET, and ACUE_SA3_SECRET for Stable Audio. Write $NAME, never ${NAME}. A variable that is not set, or a key the keyring does not hold, stops acue at that line. A locked keyring stops it with the keyring did not answer within 3s (locked?): unlock it and run again. The key is never shown, logged, or written into a book.

Make one image

The short version: Enter, *start the api*, wait, type, Enter.

1. Open acue. You land on the **Launch** pane with the model list.
2. j/k to a model, or / to filter by name. The mark at the left says whether it is fully on disk.
3. **Enter**, then *start the api*: Forge starts in the background for that model — or acue attaches to one already answering on its port — and the **Image** pane opens. Weight loading takes a minute or two; the server row says how far along it is. On the Image pane itself, S starts the selected model's server the same way.
4. The params row shows what will be sent and where it came from:

   ```
   params    Euler a / Automatic · 24 steps · CFG 4.5   ← from profiles/xl.json
   ```

   Those values are the model's profile. 4 is the Profile pane: j/k picks a setting, ←/→ changes it, s saves it to profiles/<profile>.json and R puts the preset's defaults back.
5. **e** to edit the prompt and type it. The box grows as you write and keeps a scrollbar once it reaches the rows the pane can spare; **alt-enter** or **ctrl-j** starts a new line inside it.
6. **Enter** generates, from inside the box as well as outside it. x interrupts.

The status line says what was asked for — generating 4 images… — and while it runs the progress line reads:

```
[███████▒▒▒] 62 %  8s
```

The percentage is Forge's, polled while the request is in flight. All of a batch is denoised

© Andre Bremer · https://hyperquader.com

together, so there is no "image 2 of 4" to report.

Forge writes the files, not acue. They land in the folder the profile gives Forge — acue's out folder, ~/.local/state/acue/out/forge unless acue.toml moves it — under Forge's own naming and numbering, the same as a browser generation. The summary line names the folder and how long it took:

```
4 image(s) — 1m08s — Forge saved them under:
  /home/you/.local/state/acue/out/forge
seed 3324972956, 3324972957, 3324972958, 3324972959
```

Quitting does **not** stop the server — discarding a 20 GB model load on exit would be rude. S stops it deliberately, from any pane, and acue prints the port and how to stop it on the way out if you leave it running.

Set the size, seed and batch

j/k moves between fields, e edits, esc leaves the field.

- **size** — 1024x1024. Anything the model can do. A prompt file's pictures take their size from the file instead: its --ratio, or landscape, at the model's size for that shape (A comic).
- **seed** — -1 rolls a new one per generation. u reuses the seed Forge just used, which is how you keep a result and change one word. U puts it back to random.
- **batch** — images denoised *together* in one pass. Fast, and every one of them is in VRAM at once, so this is the field that runs a card out of memory.
- **count** — passes, run *after* one another. Slower for the same number of images, and each pass costs only its own batch of VRAM.

Both are bounded, so ←/→ (or h/l) step them without editing; e still types one for when the number is large.

Total images is the two multiplied: batch 3, count 4 is twelve. Neither is capped at what your card can take — an over-large batch comes back as an OOM from Forge with the traceback in it, which is a better answer than a launcher quietly sending a smaller number than the one on screen. When it happens the server stays up and loaded: lower the batch, raise the count, press Enter again.

With more than one pass the progress line gains the only position Forge actually has:

```
[██████▒▒▒▒] 62 %  8s   pass 2 of 4
```

There is no position *within* a pass, because its images are made at once.

Forge writes every parameter into each PNG, so an image is its own record. m reads one back into the pane — prompt, negative prompt, size and seed are filled in, and whatever it used that this pane cannot set is shown on the image row instead.

```
acue --print-metadata some.png     # the same thing, outside the interface
acue --strip some.png              # a metadata-free copy, for sharing
```

Make a song

The same steps as a picture, on the audio model:

1. On the Launch pane (3), choose ace, press Enter and pick *start the api*. ACE-Step starts in the background and the **Audio** pane (1) opens.
2. e on style and describe the music: genre, mood, instruments.
3. e on lyrics for the words, or leave it empty for an instrumental.
4. Enter generates, from inside either box as well as outside it.

© Andre Bremer · https://hyperquader.com

```
style     warm lo-fi hip hop, soft piano, vinyl crackle, slow brushed drums
lyrics
duration  30
```

Leave bpm, key and time empty and the model chooses them; the result line says what it chose. The take is copied to acestep/<date>/HHMMSS.mp3 under the out folder (the as field picks mp3, wav or flac), and o plays it.

c makes the last take the source of the next one, a cover: rewrite the style, press Enter, and the next take is a cover in the new style. C clears the source. x stops waiting for a take, but ACE-Step has no way to stop a job once it is queued, so the server finishes it anyway.

Animate a picture

The **Animate** pane (6) turns one still into a short clip on an animator, an image-to-video server on another machine. acue never starts or stops it; it only needs an entry in servers.toml:

```
[[server]]
id         = "spark"
name       = "Spark animator"
kind       = "animator"
modalities = ["video"]
model      = "ltx25"
host       = "animator.lan"
port       = 9608
```

1. f takes the newest picture the Image pane's server saved as the start frame, or e on start to type a path.
2. e on motion and say what moves and how the camera moves.
3. Set seconds, width and height if the defaults (10, 832 and 448) do not suit. The line under them says what will be sent, because the frame count and the size are snapped to what the model accepts.
4. Enter sends it.

The clip lands in animator/<date>/HHMMSS.mp4 under the out folder, and o plays it. x stops waiting; the animator finishes the clip regardless.

Run a folder of prompts

Put text files in prompts/ beside the binary. Each --image line is one prompt, and a story's prose around them is not. A file of plain paragraphs, one picture each, says so with --illustrate on its first line: then every blank-line-separated paragraph is one prompt as well, and lines within a paragraph are joined, so a prompt can span three lines and stay one prompt. --illustrate may stand anywhere, and holds until --illustrate --end or the end of the file, so a story can be pictured for a stretch and not the rest. Subdirectories are searched, files are taken in sorted order.

A file with neither makes no picture, and the run goes on without it. At the end, one line names every such file:

```
note: 2 file(s) made no picture, having no --image: notes.txt, inner/story1.txt — --illustrate, or illustrate = true under [prompts] in acue.toml, makes every paragraph a picture
```

--illustrate on the command line reads every file as though it said --illustrate at its top, for that run; illustrate = true under [prompts] in acue.toml does the same for every run. A

© Andre Bremer · https://hyperquader.com

file's own --illustrate --end still ends it from its line. A file that is meant to be narrated rather than drawn is the reason neither is the default. --narrate and narrate = true do the same for speech: every paragraph spoken in a narrated book (below).

Check the split before spending an hour on it:

```
acue --print-prompts
```

A line starting with -- is a directive rather than a prompt: it sets the seed, size or style for the prompts around it, and is what the next section is about. Lines this program does not recognise belong to another tool and are ignored, except a name one or two letters from a real directive, which is most likely a typo and is warned about.

To check one file, or a few, before a run reads them:

```
$ acue check prompts/pond.txt
prompts/pond.txt:3:1: warning: --ratoi is not a directive; did you mean --ratio?
prompts/pond.txt:7:42: warning: --ratio takes square, landscape, portrait, wide or
tall, not huge, so every option on that line was dropped
1 file, 3 pictures, 2 warnings, 0 errors
```

It reads each file the way a run does, --illustrate included, and starts no model and uses no network. A warning is a line that does less than it looks like it does; an error is a file a run cannot read (missing, not UTF-8, over 1 MB). It exits 1 when there is an error and 0 when there are only warnings.

Then **p** on the Image pane runs the lot, one request per prompt, in file order. The pane goes read-only: x skips the current prompt, X stops the run, and q, S and the pane digits keep working. The title carries the position, and so does the progress line:

```
[██████▒▒▒▒] 62 %  8s   images 9–12 of 48   prompt 3 of 12   ~34m left
```

The estimate is the mean of the prompts already finished, times the ones left. It appears once two have finished — one is not an average, and the first also pays for the model load. It is a guess and says so with the tilde: a later prompt with its own --width costs what it costs.

As each story file finishes, a line under the bar says what it came to, and the run's own summary replaces it at the end:

```
amberlin-in-berlin-morning: 24 image(s) in 8m41s
finished: 168 image(s) from 7 file(s) in 58m12s — 20s each · slowest #31 41s · fastest
#9 12s
```

each is per picture, not per prompt — a batch of four in one request is four pictures. The slowest and fastest are named only when they differ once rounded, because a run of evenly matched panels has no range worth reporting. A file is named the way its bundle would be, so inner/story1.txt reads inner-story1 and two stories with the same filename never print the same line twice.

Unattended, with no interface at all:

```
acue --prompts ernie
```

Exit 0 when every prompt produced an image, 1 if the run stopped early — so it is usable from cron. Ctrl-C interrupts it, reports where it stopped, stops a server it started itself, and exits 130.

Nothing records which image came from which prompt, and that is not a gap: Forge writes the prompt into every PNG, so the mapping is already in the images.

© Andre Bremer · https://hyperquader.com

A comic: one cast, many panels

The problem with a folder of prompts is that every panel drifts. Write the character into each one and you will describe them slightly differently by panel six; leave them out and the model invents someone new each time.

Three layers fix it, each sticky until you change it:

```
-------  ----------------------------------------
--style  the look of the whole story, and the cast
--scene  where and when, until that changes
--image  this panel, and nothing else
```

Everything below uses examples/prompts/amberlin-in-berlin-morning.txt, which ships with the program, so every snippet can be run rather than read. *What to write* in the three layers is STORY.md; this section is only the mechanics.

Its head sets the things that hold for the whole file:

```
--seed 8101
--ratio wide
--title Amberlin in Berlin · Morning
--location Berlin
--keywords Berlin, morning
--watermark Amberlin in Berlin · Morning
--avoid "text, lettering, signage, labels, numbers, …"

--style
single full-bleed illustration, soft anime style, clean linework, muted
palette, warm painterly light, a young woman in her mid twenties, copper-red
hair in a short plait over one shoulder, a cream ribbed jumper, an olive
corduroy skirt, knee length, a mustard-yellow raincoat, dark green canvas satchel on a
long
brown strap
--style --end
```

Then a scene, and the panels under it:

```
--scene the old centre of Berlin at first light, wet grey pavement, tall stucco
facades, tram wires overhead

--image wide shot from behind, she stands at the mouth of a long empty street, satchel
held in both hands
--image low angle, she walks under tram wires, the facades leaning above her
--image close shot, she folds a paper map badly, the wind taking one corner
```

Each prompt goes out as look, panel, scene — so the style and the street are in all of them and written once. (Why that order, and why it was measured rather than assumed, is in STORY.md.)

Values run to the end of their line. A second line under an inline value is not a continuation of it. For something longer, open a block — a bare --style, the text, then --style --end, as the look layer does above. A block also ends at the next directive this program reads; another tool's line inside it is skipped, except in an --image block, which it closes.

--comment and --description take the same two forms, and --comment's text is thrown away, which is how a file carries notes. Write a note one --comment per line if it mentions

© Andre Bremer · https://hyperquader.com

directives: inside a block, a line beginning --style would close the note and set the look to the rest of the sentence.

Every pane setting can be written in the file — --seed, --width, --height, --steps, --guidance, --avoid (the negative prompt) — alone on a line to set it for what follows, or after an --image for that panel alone. acue(1) has them with their bounds. A bundled value must be quoted, because options after --image are strict pairs:

```
--image "she folds a paper map badly" --seed 4072 --avoid "text, blur"
```

--ratio sets the shape of the pictures, and the model picks the size:

```
--ratio portrait
--image "she folds a paper map badly"
--image "the whole street from a rooftop" --ratio wide
```

The words are square (1:1), landscape (3:2), portrait (2:3), wide (16:9), tall (9:16) and wallpaper, and nothing else. Each model draws each shape at the size it was trained for, which models.toml lists under sizes and you can change there. A file with neither --ratio nor a size draws landscape.

A size beside a ratio:

- --width and --height both: that size, whatever the ratio. The Michelle example, example.txt, says --width 1216 --height 832 and draws at exactly that on every model; the five city books say --ratio wide instead, for their 16:9 video.
- One of them: that side as given, the other from the shape, rounded to the model's step (64 on SDXL, 16 on the rest). --width 1600 --ratio wide is 1600×896 on Qwen-Image. With no ratio, the shape is landscape.

model	square	landscape	portrait	wide	tall
SDXL (Juggernaut, Pony)	1024×1024	1216×832	832×1216	1344×768	768×1344
Z-Image-Turbo, Z-Image	1024×1024	1248×832	832×1248	1280×720	720×1280
Anima	1024×1024	1248×832	832×1248	1376×768	768×1376
Krea	1024×1024	1248×832	832×1248	1376×768	768×1376
Flux Klein	1024×1024	1248×832	832×1248	1376×768	768×1376
Ernie-Image-Turbo	1024×1024	1264×848	848×1264	1376×768	768×1376
Qwen-Image, Qwen-Image-Edit	1328×1328	1584×1056	1056×1584	1664×928	928×1664

--ratio wallpaper makes a desktop wallpaper, 3840×2160:

```
--image "the harbour at dusk, every light on" --ratio wallpaper
```

The model draws it at the largest exact 16:9 size inside its wide one (1280×720 on SDXL, 1536×864 on Qwen-Image), and Forge's hires fix upscales it with the model's upscaler and refines it at a light denoise. Forge rounds a hires size to its Resolution Step, 64 unless you change it, and 2160 is not a multiple of 64, so acue asks for 4096×2304 and scales that down to 3840×2160 itself. Every step is 16:9, so nothing is stretched or cut. A --width and --height beside it set the wallpaper's size instead (--width 2560 --height 1440), one of them the other at 16:9; the size must be exactly 16:9.

The upscaler is under upscale in models.toml. SDXL uses the settings Juggernaut XL's card gives, 4x_NMKD-Siax_200k at 0.3 denoise and 15 steps; put 4x_NMKD-Siax_200k.pth in the models folder's ESRGAN. Every other model uses Forge's own ESRGAN, which Forge fetches itself, at 0.3. Before a run draws anything, acue asks Forge which upscalers it has, and refuses the run, naming the line, if the model's is missing. In a book a wallpaper is a page like any other; outside one, acue writes it to wallpapers/ in the output folder, since the copy Forge would save is the larger one. In a proof it is shown scaled down, and in the book's video it fills the frame.

© Andre Bremer · https://hyperquader.com

A clip follows its panel's shape: a portrait panel is animated portrait.

What the top of the file sets is its defaults — everything before the first --image or the first line of text. **--reset --image** puts the pane settings and the book directives back to those; --reset --scene and --reset --style do the same for one layer, and a bare --reset for all of them.

An empty value clears a layer rather than setting it to nothing in particular:

```
--style ""
```

Prose is not a picture unless the file says so. Each --image is a prompt and loose prose between them is ignored — which is what lets a story file written for another tool be reused here for its pictures. --illustrate at the top makes each paragraph outside an --image a picture too, and is declared rather than guessed from a missing --image, so a line added at the end of a file never changes how the lines above it read.

Check before generating. --print-prompts composes every panel and shows the settings under it, without contacting anything:

```
$ acue --print-prompts
…/prompts: 24 prompt(s)
   1  single full-bleed illustration, soft anime style, clean linework, muted
      palette, warm painterly light, a young woman in her mid twenties,
      copper-red
      hair in a short plait over one shoulder, a cream ribbed jumper, an olive
      corduroy skirt, knee length, a mustard-yellow raincoat, dark green canvas
      satchel on a long
      brown strap, wide shot from behind, she stands at the mouth of a long
      empty street, satchel held in both hands, the old centre of Berlin at
      first light, wet grey pavement, tall stucco facades, tram wires overhead
      · seed 8101 · width 1216 · height 832 · negative "text, lettering,
      letters, words, writing, captions, subtitles, signage, shop signs, street
      signs, labels, numbers, watermark, signature, logo, extra fingers,
      malformed hands, lowres, blurry" · description "Amberlin arrives in Mitte"
      · watermark "Amberlin in Berlin · Morning" · title "Amberlin in Berlin ·
      Morning" · location "Berlin" · keywords "Berlin, morning"
```

Twenty-four panels is one to five minutes of GPU depending on the model, and a whole city's four parts most of half an hour; the check is instant, and it is where the layer you left set and the panel that is still a sentence turn up.

The grammar is shared with acue studio, not this program's alone: the same files feed both, and each reads its own directives and skips the other's. A --voice line in the middle of a story is not an error here, it belongs to acue studio. The scene layer is --scene. A --milieu line is foreign to every reader of the format, so a file written with it loses its scene layer, silently, by the same rule.

Bundle a finished story

A run with --book packs each prompt file's pictures, and the file itself, into one .acue book:

```
$ acue --prompts zimage --book
story1: 28 image(s) in 9m14s
story2: 28 image(s) in 9m41s
finished: 56 image(s) from 2 file(s) in 18m55s — 20s each · slowest #31 41s · fastest
#9 12s
wrote 2 book(s) to /home/you/Pictures/Acue/zimage
```


© Andre Bremer · https://hyperquader.com

```
$ unzip -l ~/Pictures/Acue/zimage/story1.acue
      mimetype
      book.toml
      creator.toml
      story1.txt
      story1-01.png
      story1-02.png
      …
```

book.toml is the book's table of contents and its facts; creator.toml is the run's own record (the image model and its checkpoint, the animator, the software versions), which --rebuild and --proof read back and a reader never needs.

Beside every book is its checksum, story1.acue.sha256, so a book you publish can be checked after it is downloaded: sha256sum -c story1.acue.sha256 on Linux, shasum -a 256 -c story1.acue.sha256 on macOS, both printing story1.acue: OK. On Windows, PowerShell's Get-FileHash story1.acue shows the same hash in capitals.

Named after the file that made them, under a folder named after the model that drew them, so a folder holding two stories gives two books rather than one muddle. The exception is a city: an amberlin-in-berlin.book.toml beside the four amberlin-in-berlin-*.txt files makes them one bundle, amberlin-in-berlin.acue, with a folder per segment inside it and the soundtrack over them, and its video as amberlin-in-berlin.mp4 in <Videos>/Acue/zimage/ — SOUNDTRACK.md is that half. The pictures are made and staged out of sight, under the out folder, and only the finished book reaches <Pictures>/Acue/. The story's own name is separate: Amberlin in Berlin is derived from the filename, or set with --title, and it never decides what the book is called.

The number on each page is its position in the run, zero-padded so a file manager sorts it, and separated from the stem because a stem may already end in a digit — story1 and 01 run together would read as story101.

--description rides along. It is sticky like the other layers and takes the same forms, but it never joins the prompt: it says what the picture is *of*, and --book writes it into the PNG where an image viewer will show it.

```
--description Amberlin arrives in Mitte

--image low angle, she walks under tram wires, the facades leaning above her
--image "she folds a paper map badly" --description "still lost, at this point"
```

The first panel carries the standing description, the second its own. A bundled one has to be quoted: options after --image are strict pairs, so --description the moment it lands parses as several of them, and the whole line's options — the seed beside it included — are dropped in silence.

--watermark, --title, --copyright, --license, --location and --keywords bundle the same way and on the same terms. Each names one page's value and beats the standing one for that page only, which is how a single plate in a book can be licensed apart from the rest of it.

--watermark is sticky in the same way and gets burned into the picture, in the bottom-right corner:

```
--watermark Amberlin in Berlin · Morning

--image low angle, she walks under tram wires, the facades leaning above her
--image "she folds a paper map badly" --watermark "detail"
--watermark ""
--image the last panel, unstamped
```


© Andre Bremer · https://hyperquader.com

The stamp is frosted rather than painted: what is behind it is blurred and lifted toward white, and the text drawn dark on that. It is done that way so one ink reads over a night street and a white gallery wall alike — a fixed colour fails on half your panels, and a colour chosen from the average underneath fails on the ones that are half dark and half light.

The font is built in — Source Code Pro, compiled into the binary — so a stamp works on a fresh checkout with nothing configured. The [book] table of acue.toml beside the binary overrides it if you want a different face:

```
[book]
font = "/usr/share/fonts/truetype/inter/Inter-Bold.ttf"
font_size = 28
```

A font naming a file that is not there falls back to the built-in one rather than giving up, so a typo in that path costs you a typeface and not a stamp. A page whose stamp will not fit still comes out unstamped rather than not at all, and the run says how many did.

--copyright and --license ride along the same way, and go into the PNG twice over. Once as Copyright and License text chunks, which is where --strip and Forge's own tooling look. And once inside an XMP packet, which is what an image viewer's properties pane reads — a page carrying only the chunks opens with Author and Copyright blank, because a pane's fields are EXIF and XMP and a PNG text chunk is neither.

Because these are the same for every story an author writes, acue.toml's [book] table carries them and no story file has to repeat them:

```
[book]
author    = "Your Name"
copyright = "2026 Your Name"
license   = "CC-BY-NC-4.0"
location  = ""
keywords  = ""
```

author is who made the pictures and becomes the Author a viewer shows. It is not the program: that is written separately as the XMP creator tool, so the picture keeps its attribution instead of spending it on the software.

--location and --keywords are the other two, and they are usually per story rather than per author — four cities want four locations — so acue.toml leaves them empty and each city's book or each file says its own:

```
--location Berlin
--keywords Berlin, morning
```

A city's .book.toml takes title, location and keywords at its top level for every page in the city.

The location is written as both photoshop:City and the IPTC location field, because a properties pane reads one and a cataloguing tool reads the other. Keywords are split on their commas into a set, so a search for *morning* finds every morning in the folder.

The page wins, then the city's book, then acue.toml: a file's own --copyright, --license, --location or --keywords beats both where it gives one, so one story in a folder can be licensed or placed differently from its neighbours, and any of them can be bundled onto a single --image when one panel differs. Short forms travel better than prose — an SPDX identifier and a link, not a paragraph.

Two honest limits. It is a claim, not a protection: acue --strip removes exactly these chunks, and it ships with the program. And it reaches only the copies in a bundle — Forge's own output is never rewritten.

© Andre Bremer · https://hyperquader.com

The pictures in the book are a second copy. Forge still writes its own, under its own numbering, where the web UI expects them; a book is the thing you send someone — which is why the stamp goes on this copy and never on the original, where it could not be taken off again.

Put a panel into motion

In a city's story files, --animate under an --image puts that panel into motion, and --video is a clip of its own made from words alone:

```
--image she leans on the bridge rail, the river below
--animate the water ripples, the camera drifts slowly left --frames 121
--video a harbour at dawn, gulls lifting off the water --frames 97 --width 832
--height 448
```

With an animator in servers.toml, a --book run makes the clips on it while this machine stops Forge and makes the music. A panel's clip is named after it, amberlin-in-berlin-morning-07.mp4 beside amberlin-in-berlin-morning-07.png, goes into the book with it, and plays in the panel's slot in the city's video. Without --frames a clip lasts its --hold, its picture's own --hold, the one in force, or as long as a still is shown, in that order. The music is made to the clips' length, and acue --print-soundtrack already counts it in the seconds each story needs.

Without an animator, or with one whose model acue does not know, the book is drawn as stills. A clip that fails leaves its panel a still, and the run says how many did. A run stopped part-way does not make a finished clip again. ANIMATE.md has the rest: sizes, the watermark, rebuilding, and what clips do not do yet.

Words on a panel: cards

--card puts words in the book as a panel of their own: a title, a time, a quote. Under a picture's line, with no blank line between, the card is drawn over that picture, blurred; after a blank line, on the hour's paper colour, the one the video dips to after it:

```
--card Amberlin in Rome

--image she leans on the bridge rail, the river below
--card "The river does not hurry." --hold 5
```

acue draws each card itself, in Source Code Pro, no model needed, at the size of the pictures around it. In the video the picture goes from sharp to blurred at the start of the card's time and the words fade in; the book's [video] table sets how long each takes, card_blur = 1.0 and card_text = 0.8 seconds unless it says otherwise. A card holds for its --hold, or the book's panel, and the music is made to fit. The book keeps each card as a picture with its words beside it, so it prints and proofs like any page. Photo books take cards too: under a --photo line, over that photo.

Two or four pictures on one page

A --panel line under an --image adds one more picture to it, and the page shows them together: one --panel makes a pair side by side, three a grid of two by two. Two (three pictures) warn at the third, which is not drawn.

```
--image she reads the menu at a café table --hold 6
--panel the waiter sets down two coffees
```

Each picture is drawn on its own, from its own line with the look and the scene, and acue lays them out on the page with a thin gutter in the segment's paper colour, the cards'

© Andre Bremer · https://hyperquader.com

colours. The --image line is the page: its --ratio is the page's shape and size (a wide picture's size on the model), its --hold the page's time, its facts the page's, and its drawing options every picture's. A pair on a wide or square page is drawn as two portrait pictures, at the model's portrait size; on a tall page they are stacked, two landscape ones. Four are each the page's own shape. Each picture is scaled to cover its cell and cut to it, centred, never squeezed. A picture's seed is the panel's seed plus its place less one: with --seed 7 on the --image line the pictures are drawn with 7, 8, 9 and 10; without one each rolls its own.

The page is a picture like any other in the video, the proof and the print. The book keeps the pictures too, as the page's parts, under /.parts/<story>-NN-1.png, so --rebuild --images redraws only the picture whose line changed and composes the page again. A panel stays still for now: an --animate under one is refused. --ratio wallpaper is refused on a panel.

How long each panel holds

--hold SECONDS, from 2 to 15, sets how long a picture, a photo or a card is on screen: alone on its line for everything after it, on an --image, --photo or --card line for that one, and --reset --hold puts back the top of the file's. Without one, the book's panel stands. On an --animate or --video line it is the clip's length. --print-soundtrack counts every hold.

Narrate it

A book can speak a line or two over each scene, in Kokoro's voice, with the music ducked under it. Write a --speak line above each scene's first picture, or let the agent draft them into the stories first:

```
acue --narration --prompts-dir berlin/      # a --speak line for each scene that has none
acue --prompts klein --book --prompts-dir berlin/
```

Read the drafts and edit them: the build speaks what the stories say and never asks a model itself. A story told in paragraphs needs no lines: from a --narrate line each paragraph is spoken over the pictures after it, until --narrate --end or the end of the file, and a --speak directly under a paragraph says it another way. It needs a speaker in servers.toml (kind = "speech"); without one the book is made without narration, said in one line. A line too long for its scene holds the scene's stills longer, and the music is made to the new length. The book's [narration] table sets the voice, its pace and level, and the duck; NARRATION.md has every key.

Fix one panel

One panel of a finished city came out wrong. The book is the unit of work:

```
cd ~/.local/state/acue/out/krea2
acue --rebuild amberlin-in-berlin.acue            # unpacks beside itself, names the files
# edit amberlin-in-berlin/morning/amberlin-in-berlin-morning.txt — the one --image line
acue --rebuild amberlin-in-berlin.acue --images   # redraws that panel, a new book beside the original
acue --proof amberlin-in-berlin.acue amberlin-in-berlin-rebuild.acue
```

Only the panel whose line changed is drawn again, with the story's own seed, by the model that drew the rest, on the backend that drew it — the bundle's run.toml says which, Forge or the Spark's imager. A book whose model now draws on the other backend, or on another stack,

© Andre Bremer · https://hyperquader.com

is refused: the same seed would draw another picture there. Every other panel keeps its bytes, the tracks are kept, and the video is cut again. The original book is never written. If you add or remove an --image line the story is redrawn whole and the run asks for --audio too, because the tracks were generated to fit the old panel count. Editing the music is the same loop with the book file and --audition / --audio — see SOUNDTRACK.md. Cards are drawn again from the staging each time, so a card over a redrawn panel shows the new picture. A card added or removed shifts the numbers of the panels after it; --images moves their pictures and clips to the new numbers first, so they are kept, not drawn again. Only the tracks have to be made again (--audio): the card's time is new. In a --panel group, only the picture whose line changed is drawn again; the page is composed again from it and the pictures the book kept. A panel given another count of pictures, or another --ratio, is drawn whole.

Sing the soundtrack

A city's soundtrack can be one short song a segment, sung in the book's language by its main character. Put kind = "song", a language and a singer in the book's [soundtrack], and a style on each track in place of its prompt; then draft the lyrics, listen, and keep the takes you like:

```
acue --lyrics   --prompts-dir paris/     # <city>-<segment>.lyrics beside the book
acue --audition --prompts-dir paris/     # two takes of every song, into
<Music>/Acue/auditions
acue --keep
~/Music/Acue/auditions/amberlin-in-paris/amberlin-in-paris-morning-seed-1234.wav
```

The lyrics are drafted by the language model in servers.toml (kind = "chat") and are yours to edit: a file that is there is never drafted over unless you add --force. A kept take is the segment's song on every build, as it is; ACE-Step cannot make the same take twice, so the file is what keeps it. SOUNDTRACK.md has every key and the rules the lyrics are written by.

Make a book of your own photos

A folder of your own photos, from a camera or a phone, becomes a book in two steps. First a draft:

```
$ acue photos ~/Pictures/Rome
wrote /home/you/Documents/Acue/rome/rome.book.toml
wrote /home/you/Documents/Acue/rome/rome-2026-07-14.txt
wrote /home/you/Documents/Acue/rome/rome-2026-07-14-1530.txt
wrote /home/you/Documents/Acue/rome/rome-2026-07-15.txt
86 photo(s) in 3 part(s)
next: write the captions, then make the book with acue --prompts-dir
/home/you/Documents/Acue/rome --book
```

acue photos ~/Pictures/Rome --describe also drafts the captions and each part's title with a vision language model, for you to check and edit. The model is the agent in servers.toml when it also lists vision among its modalities (the example file has one): on this suite the Spark's agent role, on the local network. What leaves the machine is each photo, as a copy at most 1024 pixels on its long side with no metadata in it, to that server and nowhere else; a server that is not on this machine or the local network is refused. The captions are asked to name no one, describe no face and name no exact place. Captions are optional: without such a server, or when it does not answer, the draft is written with empty captions and says why.

acue photos reads the folder's JPEGs and PNGs, its subfolders' too, and sorts them by the

© Andre Bremer · https://hyperquader.com

time each was taken: the camera's own record of it (EXIF DateTimeOriginal), or the file's time when a photo has none. It cuts them into parts: each day is a part, and a gap of more than three hours starts another.

The draft goes into a folder of its own in your documents, <Documents>/Acue/<name>/, and nothing is written into your photo folder. Each photo is a link in the draft's photos/ folder, pointing at the photo where it is, so the stories name photos/IMG_0412.jpg and never a folder of yours. Move or delete a photo and its link breaks; acue check then names it. The name is the title's, made safe for a file name; a draft or a book of that name already there gives rome-2, and so on. --into <dir> puts the draft somewhere else.

Each part is a story file, one --photo line a photo:

```
--description 14 July 2026, 09:12 to 13:40

--photo photos/IMG_0412.jpg --description ""
--photo photos/IMG_0413.jpg --description ""
```

The --description on a photo's line is its caption: write it between the quotes. The one at the top names the part, and becomes its chapter's title in the book. rome.book.toml gives each part a track of music by mood, holiday, family, travel or calm (SOUNDTRACK.md has them), and the pace. The draft writes files and nothing else, and it never replaces one.

A few photos rather than a folder, and the draft's words given at once:

```
acue photos --title "Rome 2026" --location Rome --mood travel --files IMG_0412.jpg
IMG_0440.jpg
acue photos --no-music ~/Pictures/Rome     # model = "none": a book with no music
```

Everything after --files is a photo; --files - reads them from standard input, a path a line. --title names the book, --location writes a --location line at the top of every part, --mood gives every track that mood, and --no-music writes model = "none", which keeps the tracks in the file and makes no music until it names a model again.

A place appears in the book only where you write one. The draft names none, even when a photo carries a GPS position; --location on a line of its own names the place of the photos after it, and on a photo's line that photo's:

```
--location Rome
--photo IMG_0412.jpg --description "The Forum at noon"
--photo IMG_0440.jpg --location "Ostia Antica" --description ""
```

Then the book:

```
acue --prompts-dir ~/Documents/Acue/rome --book
```

--book without --prompts draws nothing, so it starts no image model. It makes the soundtrack on the audio server, each track as long as its part's photos are on screen, then the video and the book: rome.acue in <Pictures>/Acue/ and rome.mp4 in <Videos>/Acue/, each folder as your desktop names it (acue --print-paths shows them). acue check reads the stories first and names a photo that is missing, unreadable or not a JPEG or PNG, at its line and column.

The music is optional, and so is the AI that makes it. Where Stable Audio is not set up, the book is made all the same, without music, and one line says so:

```
acue: no music: sa3 is not set up here (its weights are not on disk), so the book is
made without it
```

model = "none" under [soundtrack], or no [soundtrack] at all, makes a silent book and starts no audio server. A music file of your own plays instead of made tracks: name it at the top of the book file, relative to it,

© Andre Bremer · https://hyperquader.com

```
title = "Rome 2026"
music = "our song.mp3"
```

and each part takes the next stretch of it, faded in and out as a made track is, and the song starts again when the book is longer than it. A [soundtrack] beside it stays in the file for later and makes nothing.

What goes into the book is not the file from the camera. Each photo is turned upright, as the camera recorded it was held, scaled to at most 2560 pixels on its long side, stamped if the story asks for a --watermark, and saved again as a JPEG holding the picture and nothing else. The GPS position, the camera and its serial number, the owner's name, the thumbnail and every other piece of metadata stay behind in your original, which is never changed. A photo stays still: an --animate below one is refused.

A story may mix photos with drawn pictures. Then the run needs its model, acue --prompts <model> --book, and each photo takes its place among the drawn panels by its line. HEIC, the iPhone's own format, is not read yet: export those photos as JPEG first.

To print the book:

```
acue --print ~/Pictures/Acue/rome.acue --out rome.pdf                 # A4
acue --print ~/Pictures/Acue/rome.acue --page square --out rome.pdf   # 210 mm square
acue --print ~/Pictures/Acue/rome.acue --for screen --out rome.pdf    # to share
```

The first page is the title, with the authors and the copyright from acue.toml. Then each photo has a page of its own: the chapter's title small at the top, the photo as large as the page allows with its shape kept, its caption under it, the place under that where the story named one, and the page number at the foot. Everything sits inside a 15 mm margin, so the file has no bleed and needs none: nothing is meant to be cut at the edge of the paper. The fonts are embedded, and the photos go in as the book holds them, about 360 dpi on A4. It prints any book, a city's panels too.

That file is for a print shop, and it is large: a city book of 97 drawn pages comes to 235 MB. --for screen makes the same pages to read and share, each picture a JPEG at quality 85 scaled to 150 dpi at its size on the page; the same book is 14 MB. --for print is the default.

Driving acue from another program

Pinax's Make a Book… runs these same commands. Two flags are for a program rather than a person. acue --capabilities prints one JSON object saying what is set up here: whether music can be made ("music", with a note saying why not), whether auditions can, and where books, videos and drafts go. --progress json on photos, --book or --audition turns standard output into one JSON object a line, and sends everything a person would read to standard error:

```
{"step":"scan","n":3,"total":12}
{"step":"photo","n":5,"total":12}
{"step":"video","n":0,"total":1}
{"done":true,"books":[{"acue":"/home/you/Pictures/Acue/rome.acue","video":"/home/you/V
ideos/Acue/rome.mp4"}]}
```

A run ends with one done line, or one {"error":"…"} line and a non-zero exit. photos says where the draft went ("folder", "book"); --audition gives a line a take, with its part, seed and file. The man page lists every field.

Compare two models on the same prompts

Name more than one model and the whole folder runs against each in turn:

© Andre Bremer · https://hyperquader.com

```
acue --prompts ernie,klein,zit
```

A server holds one checkpoint, so each model means stopping the last server and starting the next — the model is the outer loop, the prompts the inner one. One seed is pinned across all of them and printed, so the model is the only thing that differs between two images of the same prompt. --seed N repeats a previous comparison.

```
comparing 3 models at seed 356139942
── ernie — model 1 of 3 ──
starting ernie on :7860 — this loads the weights, give it a minute
  ernie: generating prompt 3 of 12 — an image…
```

It refuses before loading anything if a model is missing files, if an id is named twice, or if something is already listening on a port it needs — a comparison starts and stops its own servers, and one it did not start is not its to stop. A model that will not load is reported and the run carries on.

Afterwards, look at what came out — see the next section.

Look at a comparison instead of reading one

A list of filenames tells you which models drew a prompt. It does not tell you which one drew it *best*, and that is the question. --proof turns finished bundles into one document to look at:

```
cd ~/.local/state/acue/out
acue --proof ernie/amberlin-in-berlin.acue klein/amberlin-in-berlin.acue \
      --out berlin.pdf

berlin.pdf — 111 page(s) (15 front, 96 compared), 44.5 MiB
```

Two to four bundles, and the document runs in three parts.

What is being compared. Each bundle with the sampler, scheduler, step count, CFG and checkpoint that drew it, then the book file and the prompt scripts verbatim. The settings come from the bundle's own pages, so they describe what was made rather than what the catalog says today — and *"why is that one washed out"* is usually a step count.

An index. One contact sheet per model: every panel as a thumbnail, in story order. You can watch the day cross the page — pale morning light, the midday street, dark interiors by evening — before turning to any of it.

The comparison. One page per prompt, every model's take side by side, captioned with the bundle and the seed. The grid follows the pictures' shape: landscape and square takes two by two, portrait and tall ones in one row, each whole and in its own shape, never cropped or stretched. A contact sheet's rows are as tall as their pictures, so a portrait book's thumbnails are not shrunk into landscape boxes. The seed is pinned across models, so a tile you dislike can be rerolled from the number printed under it.

Four is the cap: a page holds four cells, and five takes side by side is a wall. Choosing between more models than that is done on a handful of prompts, not on ninety-six.

Nothing is scaled down. A landscape cell on A4 landscape is about 131mm — 1547 pixels at 300 dpi, wider than the 1216 a page is drawn at — so a defect appears at the resolution the model produced it, which is the point.

What it is for. Three models drawing one woman on a street and a fourth drawing two of her is a defect in that model, not in the prompt; three models getting the same detail wrong is a defect in the prompt. Neither is visible in a folder of files, and both are obvious on a page.

© Andre Bremer · https://hyperquader.com

As a video. Add --video and the comparison is a film instead, to show somebody rather than to decide:

```
acue --proof klein/amberlin-in-paris.acue zit/amberlin-in-paris.acue \
      krea2/amberlin-in-paris.acue ernie/amberlin-in-paris.acue \
      --video --out proof-paris.mp4

proof-paris.mp4 — 9:16, 101 frame(s) (96 compared), 208.0 MiB
```

1920×1080 at 30 frames a second. A title card names the book, the day it was made and the four models; each chapter opens on its hour card, in the hour's colour, and then runs one frame per prompt: every model's take, laid out by the same grid rules as the PDF, captioned with the model and the seed. Each frame holds as long as a panel does in the book's own video, with a short crossfade, over the first book's music for that chapter. No front matter and no contact sheets: those are for reading. The same cap of two to four, and the same books give the same file.

Read this away from the screen

These three documents are written to be read start to finish, which is not what a browser tab is for:

```
make guide
```

dist/acue-guide.pdf — this guide, STORY.md and SOUNDTRACK.md, in the house font, with the story panels placed full width and the parameter tables laid out to fit. acue --guide <file.md> [...] --out <path> renders any markdown the same way; make guide is that with the three filled in.

Headings, bold, italic, lists, code blocks, tables and images all survive. Emphasis is a real italic rather than a slant, because all four weights of the face are embedded. What does not survive is a link's target: the text stays and the URL is dropped, which suits the relative links these files mostly use.

Which image model, and what each one is like

Observed on the Amberlin stories — the same prompts, the same seeds, four models. Notes here only where a book of forty-eight panels showed something a single image would not.

model	steps	per image	text encoder	what it does
anima	32	12s	Qwen3-0.6B base	The style layer holds beautifully and the fine attributes drift. Asked for "a short plait over one shoulder" it gave a bun, a ponytail and loose hair across three cities. Softest line, flattest colour.
zit	9	4s	Qwen3-4B	More literal about what a panel says. Real surface texture — brick reads as brick. Files about 40% larger than anima's for the same frame.
klein	4	2s	Qwen3-4B	Honours the detail the others drop: the plait is a plait, and hands come out with the right number of fingers. Also honours the parts of a prompt you did not mean — a panel asking for handwriting got a page of confident gibberish where anima gave a blur. Largest files, fewest steps.
krea2	8	8s	Qwen3-VL-4B	Sold as photoreal, and it obeyed soft anime style without argument. Attribute fidelity like klein's.

© Andre Bremer · https://hyperquader.com

ernie	8	16s	Ministral-3B	Obeys the wardrobe change, not just the wardrobe: the morning's green canvas satchel and the evening's small brown leather purse are different lines in different files, and it drew each where it belonged across three cities. Six specified attributes for six in Berlin morning, five for five in two evenings. On the smallest encoder that manages it, so the 4B figure below is a ceiling and not a threshold. Slowest per image, and it wants 22.5 GB of weights on a 16 GB card, which Forge's own offloading carries.

Two things worth taking from that.

The encoder predicts attribute fidelity better than the model size does. anima is a capable model that loses a hairstyle, and the thing it has that the others do not is a 0.6B *base* encoder. Every model here with a 4B encoder kept the plait, the satchel and the jumper together across ninety-six panels — and so did ernie on a 3B, which puts the boundary somewhere below four billion rather than at it. 0.6B *base* remains the one that drifts.

The sharper test is not consistency but obedience to a deliberate change. Each daytime file specifies its own outfit, and the two differ in every layer — a cream jumper under a mustard raincoat with a green canvas satchel in the morning, a green velvet dress with a mustard silk scarf and a small brown leather purse in the evening. A model that merely held one look steady would draw the satchel at dinner. Checking a bag's colour against the file that asked for it is a cheap way to tell the two apart, and worth more than eyeballing whether a series "looks consistent". If a character will not hold still, look at what is reading the prompt before rewriting the prompt.

A look layer beats a model's default register. krea2 is a photorealistic model and it drew an illustration, because the prompt asked for one. The same four sentences produced a recognisable Amberlin on five models with four different encoders.

Which to pick for a book: ernie or klein if the character has to survive scrutiny — ernie for the closest reading of a changing wardrobe, klein for the same fidelity in a quarter of the time; krea2 for a photoreal model that still takes direction; zit if the panels matter more than the person; anima for a softer, more painted look and the willingness to re-run the panels where the hair goes wrong.

And what each one costs. Measured over every example story on every model — 344 panels apiece at 1216×832, four cities and their music, on a 16 GB card, when the soundtracks were still made on ACE-Step:

model	steps	per image	images	soundtracks	leg	bundles
klein	4	2s	~14m	~20m	~40m	2.1 GB
zit	9	4s	22m22s	~20m	~45m	1.5 GB
krea2	8	8s	46m52s	~20m	~70m	1.6 GB
anima	32	12s	1h06m	~20m	~90m	1.9 GB
ernie	8	16s	1h29m	~20m	~110m	2.0 GB
all five					~6h	9.9 GB

klein's figures are scaled from 288 panels, sixteen story files rather than seventeen. xl was not in the sweep; a separate run put it at 5s an image, between zit and krea2.

The soundtracks did not depend on the image model. On ACE-Step they were a flat twenty minutes, two thirds of the fastest model's leg: sixteen generations per model — four cities of four segments, one source track and three covers from it — at about 80 seconds each. Stable Audio, the default now, renders a track in about a second after a model load of about

© Andre Bremer · https://hyperquader.com

a minute (SOUNDTRACK.md), so the images are most of a leg. An estimate built from the finished: line alone, which reports only the images, comes out at less than half the truth for klein.

Step count does not predict time. klein samples four steps in 2s and anima samples 32 in 12s, which is the ratio you would guess; but zit samples nine in 4s, krea2 eight in 8s and ernie eight in 16s — the same step count taking twice as long. Precision and the encoder decide it, so a new model is an unknown until it has run.

Size follows neither. The fastest leg is the heaviest on disk and ernie, the slowest, is nearly as heavy, so there is no inverse to lean on either. A PNG is as big as its picture is busy, which is a property of what was drawn rather than of how long the drawing took.

Judge a model on a whole part, not on a panel. klein's Rome morning showed no faults at all; the same model's lunch and evening produced pages of invented handwriting, where nine panels asked for writing. The model was faithful and the prompt was bad. Twenty-four frames of one hour will find the prompt's weak lines, and one good frame will not.

Draw on the Spark instead of Forge

The Spark's imager — SGLang's diffusion server, as a docker-stacks stack serves it on 9602 — draws a book's pictures as well as Forge does, and the rest of the book (cards, clips, narration, music, video) is made exactly as before. Its models are catalog entries of their own, with a stack where a Forge model has a preset, because the same seed draws another picture on another pipeline:

model	stack	steps / CFG
zit-spark	qwen3.8-27b-nvfp4 (the daily stack)	9 / 0.0
zimage-spark	kat-zimage-6b-bf16	50 / 5.0
klein4-spark	kat-klein-4b-bf16	4 / 1.0
klein9-spark	kat-klein-9b-bf16	4 / 1.0
ernie-spark	kat-ernie-8b-bf16	8 / 1.0
krea2raw-spark	kat-krea2-13b-bf16	30 / 5.0
anima-spark	imager-anima	35 / 4.5

The steps and CFG are docker-stacks' measured settings, sent with every picture. servers.toml names the imager and the model the Spark is serving now:

```
[[server]]
id         = "imager"
kind       = "imager"
modalities = ["image"]
model      = "zit-spark"
host       = "spark.lan"
port       = 9602
```

```
acue --prompts zit-spark --book
```

acue never switches the Spark's stack. When the imager does not answer, the run says which stack the model needs and stops — zit-spark needs the qwen3.8-27b-nvfp4 stack on spark.lan — so switch it there, set model to match, and run again. A --ratio wallpaper picture is refused at its line on these models: wallpaper needs Forge's upscaler. Each picture is sent with a seed, rolled when the story gives none, so every one can be drawn again; the book's run.toml records backend = "imager" and the stack.

Answers without starting the interface

© Andre Bremer · https://hyperquader.com

Several flags do one thing and exit, which is what you want from a shell, a script, or a question you do not want to open a pane to answer.

```
acue --list                          # the catalog: every model, what it makes
acue --profile xl                    # write xl's profile file, and say where
acue --print-command xl              # the exact launch.py argv, argument by argument
acue --launch xl                     # run that server in the foreground, no interface
acue --print-prompts                 # split prompts/ and show every prompt, unrun
acue check story.txt                 # every warning in a prompt file, at its line and
column
acue --print-soundtrack              # the music a book run would make, unmade
acue --print-metadata shot.png       # what Forge recorded into a picture
```

Two of these save real time rather than typing.

--print-command answers "why did the server start like that". It prints what would be handed to launch.py, and make parity diffs exactly this against what the checkout's own Makefile produces — so when a launch behaves oddly, the argv is a thing you can read rather than infer.

--print-soundtrack is the dry run for the expensive half. A city's music is about twenty minutes; this says, before any of it happens, how many panels each story has, how many seconds of audio that needs, the bpm, which segment of the city's book it joined, which chain it belongs to and whether it is the take the chain derives from. Reading that first is cheaper than listening to the wrong plan afterwards.

--strip removes the metadata Forge writes into a PNG: by default into a copy beside it named <name>-clean.png, with --out DIR into that folder under the original names, or over the original with --in-place. **--catalog**, **--servers** and **--prompts-dir** point any of the above at files other than the ones beside the binary; they are modifiers rather than modes, and acue(1) has the precise wording.

A server you thought you stopped

If generations get slower, or a model will not fit that fit yesterday, check what is actually holding the card:

```
$ nvidia-smi --query-compute-apps=pid,used_memory --format=csv
pid, used_memory [MiB]
1019093, 6428 MiB
1378515, 7004 MiB
```

Two entries where you expected one means a Forge is still resident. The usual cause is stopping one by hand with curl -X POST .../sdapi/v1/server-stop: that closes Forge's web server and leaves the process alive, because Forge is built to be restartable. The port frees, so ss -ltn shows exactly what a clean shutdown looks like, and several gigabytes stay on the card.

Use S in the launcher, or let a --prompts run stop the server it started — both take the process group before the graceful stop, which is the only moment it can still be found. When the launcher neither started the server itself nor can get ss to name the process on the port, S says it cannot tell which process holds it and stops nothing, rather than reporting a stop that did not happen. To clear a stranded one:

```
$ PG=$(ps -o pgid= -p 1019093 | tr -d ' ')
$ kill -TERM -"$PG"
```

Then check nvidia-smi again, not the port. The port was never the question.

© Andre Bremer · https://hyperquader.com

When something looks wrong

0 is the Doctor pane: which acue this is and what built it, then python, torch, cuDNN, the GPU and the catalog, with live CPU/GPU/network/disk charts. It reads and changes nothing — fixing an environment is that checkout's own business, and examples/checkouts/README.md says which file decides what.

- *The pane has drifted somewhere odd* — R puts every setting on it back to its default. On Audio and Image it asks first when something is written, because it clears that too; the sampling it restores is the selected checkpoint's, not a constant, so a distilled model gets its eight steps back and a base one its fifty.
- *"no server is running for the selected model"* — press S, or the server belongs to a different model than the one selected. The model row says so.
- *A model shows as partial* — 5 is the Downloads pane; d downloads what that model is missing, D everything.
- *Forge will not start* — its output is in tmp/server-<id>.log in the checkout, not swallowed.
- *An image came out worse than in the browser* — the params row is what was sent. The API ignores the per-model defaults the make targets set up, which is why acue fills them in from the profile; compare that row against the image row after pressing m on the image in question.

? lists every key for the pane you are on.

© Andre Bremer · https://hyperquader.com

Writing a picture story

How to get a city's worth of panels that look like one book rather than a pile of unrelated pictures. The mechanics — which directive does what — are in acue(1); this is about what to write in them, and every rule below is here because breaking it produced a bad page first.

The worked examples are in examples/prompts/: one young woman in five cities, each city written as four files — morning, lunch, afternoon, evening — of twenty-four panels each in Berlin, London and Paris and twelve in Kyiv and Rome. Read one alongside this. Between them they hold everything below:

what	where
a second character arriving	amberlin-in-berlin-lunch.txt, amberlin-in-london-afternoon.txt
the same one being let go	amberlin-in-berlin-afternoon.txt
a day turning into night	any -evening.txt
a prop carried between scenes	Paris from lunch onward, Kyiv and London from the afternoon

Books entertain

The books are made to entertain people, not to depress them. Everything in one, the pictures, the music, the songs and the narration, is happy, content, balanced, honest and loving: never sad, melancholic or depressing. A city is energy and life, and a quiet moment in it is a contented one, not a lonely one. Write the descriptions that way, and the drafts follow: acue --lyrics and acue --narration ask the language model for the same mood in every request, and the soundtrack prompts ask for it in their own words (SOUNDTRACK.md).

Split the day before you write anything

Four files of twenty-four panels beats one file of ninety-six, and not only because a screensaver can then show the hour it actually is. It makes the prompts better. Four reasons, and they compound:

Each panel has less to answer. Writing an --image line is answering a question, and the other layers decide how big that question is. Across one book of a whole day it is *what else happens in Kyiv* — vast, unconstrained, and answered generically: she walks somewhere, she looks at a building. Once the scene has fixed the place and the hour,

```
--scene inside a great covered market at midday, iron roof, daylight from above,
stacked jars and produce
```

the panel is only asked *what is she doing, right now, here* — and that has concrete answers, because the place is doing the narrowing:

```
--image close shot, a woman in a headscarf holds out a spoon of honey to her
--image she buys a paper twist of dried fruit, counting coins carefully
```

Neither occurs to you when the question is "what happens in Kyiv". Both occur when it is "what happens at that stall".

The diagnostic follows: **if an --image line contains a time or a place, the question was too big.** "In the evening", "in a Roman market" — the panel is answering something the scene should have answered once, for all six.

Twenty-four panels exhausts the vague ones. Six panels can all be a woman looking at

© Andre Bremer · https://hyperquader.com

something. Twenty-four cannot, and running out of generic framings is what forces the concrete moment: the spoon of honey, the barge close enough to move the water, the bell-push worn smooth by thumbs. Length is a constraint that pays.

A part is short enough to check. Twenty-four prompts can be read end to end in one sitting and held in the head while reading. Ninety-six cannot, and the weak lines survive in the middle of it.

Four parts of one shape make an outlier obvious. Each hour here has the same skeleton — she goes out, she looks at something somebody made, she writes it down — so a part that fails stands out against its three siblings. That is how nine panels still demanding handwriting were found: not by reading the files again, but by one part looking wrong beside the others.

Twelve panels a part is thin: two scenes is a fragment, not a shape. Twenty-four — four scenes of six — is where a part starts behaving like a story.

The three layers, and the one question that sorts them

--style	the look, and the cast	never changes
--scene	the scene	changes when the story moves
--image	one drawable instant	every panel

The question that decides where a phrase goes: **does it survive a change of location?** The painting style does, so it is --style. So does the protagonist — she is the same person in the museum and on the bridge, which is why the cast lives in the look layer and not in the scene. The ivy on the brick does not survive; it is --scene.

Getting this wrong is quiet. Put the style in --scene and it vanishes the moment the story moves indoors, and you will not notice until panel twelve looks like a different book.

Any of the three can be a block

The layers and --image each take their value three ways, and the example stories use the shortest one so often that the others read as though they do not exist:

```
--image she steps out of the station arch into the street
--image "she steps out of the station arch into the street"
--image
she steps out of the station arch into the street, a tram crossing behind her,
the light hard on the wet setts
--image --end
```

Bare opens a **block**: every line that follows is the value, until --image --end. That matters most for --image itself, because a panel worth writing carefully is often longer than a comfortable line — and --style, which is nearly always a block, has trained the eye to expect that only the look layer can be one.

A block also ends at any other line starting -- **that acue itself owns**, and that line is then read as the directive it looks like. So a --comment inside an --image block is read as a comment and closes the block, and the prompt lines under it belong to no picture. Write notes as one --comment per line, outside.

A -- line acue does not recognise is skipped, and any other open block (--style, --scene, --comment and the rest) stays open: these files are read by other tools too, and closing on a stranger's directive would truncate a block on a line that means nothing here. An --image block is the exception — a stranger's line closes it, so that the spoken narration a shared story file carries under a --voice line can never end up inside a picture's prompt.

© Andre Bremer · https://hyperquader.com

A model draws words; it does not take instructions

The single most expensive mistake. An early version of a look layer read:

> --style soft anime illustration, the same woman in every panel: …

which produced an **eight-panel contact sheet** — a grid of thumbnails — because "in every panel" is not an instruction to a text encoder, it is a description of a picture with panels in it. The same trap catches "consistent character across the story", "as before", and "same as the previous image".

There is nothing in the pipeline that reads your intent. Write only what is in front of the camera. If a phrase would not appear in a caption describing the finished picture, it does not belong in the prompt.

Hold a character with one object, not with a face

Faces drift. A 3B model asked for "warm brown eyes, oval face, freckles" will give you a different woman every time, and adding more adjectives makes the prefix longer without making her more stable.

What works is **one high-contrast object that is hers alone**. Amberlin has a mustard-yellow raincoat. It is in the look layer, it is in every panel, and the eye finds her by it across a whole book — including in a wide shot where her face is thirty pixels across and could be anyone's.

Pick something that survives weather, indoors and out, and does not fight the palette.

And colour everything you name. The first version of Amberlin's look layer ended canvas satchel — a prop with a material and no colour. It came out brown leather in one panel, pale blue-grey in the next, because "canvas satchel" does not say, and something has to be drawn. The coat never wavered, because mustard-yellow left nothing to decide.

The rule is blunt: **an attribute you do not state is one the model re-rolls every panel.** Not just colour — a hairstyle too. copper-red hair in a short plait held the colour perfectly and lost the plait, which came back as a bun, a ponytail and loose hair, because a colour is one word the encoder cannot misplace and a hairstyle is a shape competing with everything else in the frame. Anchoring it to the body — a short plait over one shoulder — gives it somewhere to be.

So the look layer names, for every visible thing that belongs to her, *what it is and what colour it is*:

> copper-red hair in a short plait over one shoulder, mustard-yellow raincoat over a cream jumper, charcoal trousers, dark green canvas satchel on a long brown strap

Including the half of her the coat usually hides. The first version stopped at the jumper, and a long raincoat covered the gap in most panels — so the drift went unnoticed for three cities. Where the legs did show, they were trousers in one panel and something closer to a dark skirt in the next, because nothing had said. A silhouette is a described thing like any other.

And name the dimension that decides that silhouette. Naming the garment is not the end of it. *An olive corduroy skirt* fixes what it is and what colour it is and leaves its length free, so the same skirt falls to the knee in one panel and to the ankle in the next — inside one story, where the outfit is the thing that is supposed to hold still. It reads as the model being inconsistent, and it is the prompt: nothing was ever asked for. One phrase in the look layer settles it, *an olive corduroy skirt, knee length*, and the rule underneath is the one above it a level down — whatever the shared layer leaves unsaid, the model decides

© Andre Bremer · https://hyperquader.com

again for every panel.

Where a garment is named inside a panel instead — a second character's, which belongs in the --image line and not in --style — put the length in front of it: *a knee-length olive linen dress*. The appositive form collides with whatever verb the sentence continues with.

That is longer than it was, and this is where the cost is paid: those words ride on every panel, competing with the one line that makes each panel different. Which is why it is the character and nothing else — hair, coat, jumper, trousers, bag.

Identity is constant; the outfit need not be

The layer above does two jobs, and only one of them has to hold still for the whole series.

Identity — her age, her face, copper-red hair in a short plait over one shoulder — is the same in every file of every city. Change it and she stops being the same person.

The outfit only has to hold still *within one story*. A book of a single morning is twenty-four panels of one hour; there is no reason she should be wearing at dinner what she wore at the market, and every reason she should not. So each hour has its own:

morning	cream ribbed jumper, olive corduroy skirt (knee length), mustard raincoat, green satchel
lunch	cream linen shirt, charcoal trousers, mustard cardigan at the waist, green satchel
afternoon	rust corduroy jacket over a cream blouse, charcoal trousers, green satchel
evening	deep green velvet dress (knee length), mustard silk scarf, small brown purse

When the shared layer changes, re-read every panel that names it. Nine panels here went on saying *hands deep in the raincoat pockets*, *the satchel across her*, *one yellow coat among the stalls* after the outfits were split by the hour — an evening with no raincoat, a lunch with a cardigan, an afternoon whose jacket is rust. Each was a panel contradicting the layer above it, which is the same failure as a panel asking for text under a negative that denies it. The fix in every case was to name the colour instead of the garment: *one spot of mustard in all that white stone*, *the mustard scarf pulled close*. A garment can be changed; the thread cannot.

Keep a colour thread through the changes. Look again at the pair above. The morning is a cream jumper, an olive corduroy skirt and the mustard raincoat over a green satchel; the evening is a green velvet dress, a mustard scarf and a small brown purse. Not one garment survives — and the two colours do. The raincoat becomes a cardigan becomes a scarf; the satchel becomes the dress. She is recognisable at a glance in any hour without wearing one outfit for twenty of them, and the palette of the whole set holds together.

The rule underneath: **the anchor is a colour, not a garment.** A coat can be taken off. Amber and green cannot.

Her shoes are still not in there. The line has to be drawn somewhere, and it goes where a reader would stop noticing: a silhouette changing shape is obvious, a boot becoming a shoe is not. Every word past that point is taken from the panel.

Give something to the day, and let it travel

The coat says *same person*. It cannot say *same day* — it is in every panel of every city, so it carries no information about time passing. For that you need a second object, and a different kind of one: something she does not start with.

© Andre Bremer · https://hyperquader.com

In Paris an old bookseller wraps an atlas in brown paper at midday. It is open on her knees in the afternoon courtyard, held up against the rooftops at dusk, and lying flat beside the notebook that night. Kyiv does the same with an engraving bought on the descent — and there the engraving turns out to show the street the two of them are standing in, which is the whole afternoon's point made without a word of explanation.

What it buys, none of which costs a panel:

- **A reason for scenes to connect.** The atlas is why the courtyard follows the quay. Nothing has to say so.
- **Evidence that time passed.** She has something at four o'clock she did not have at ten. That is a day, stated in an object.
- **A reason to be alone that is not loneliness.** After the bookseller is gone the atlas is still there, so the parting reads as an exchange rather than a loss.
- **A close.** Ending on the day's object beside the notebook says *this happened* better than any final wide shot of a face.

Three things make it work:

Get it from somebody. A thing she buys or is given carries the person who gave it. A thing she already owned carries nothing.

Name it identically every time. the wrapped atlas, not "her book" and then "the map" and then "the parcel" — the model has no idea those are one object, and neither will a reader skimming the panels.

Colour it like anything else. It is a prop, so the rule that caught the satchel catches this too: *an attribute you do not state is one the model re-rolls*. a flat paper parcel, an old engraving, a battered atlas — a material and a condition, so it comes back the same each time.

One object per story is plenty. Two start competing for the same hand.

Anyone you do not describe becomes the protagonist

A panel reading

```
--image the two women lie back on the grass looking up, laughing
```

came back as **two copies of the same woman** — same haircut, same coat. The look layer describes one person, and it applies to everyone in the frame who has not been described as someone else.

So a second character has to be described **on every line she appears in**, and never in --style :

```
--image a taller woman with cropped silver hair and ink-stained hands is repairing a bicycle
--image the woman with cropped silver hair looks up from the bicycle and says something
```

Repetitive to write, and it is the price. Putting Nadia in --style would put her in every panel of the file, including the ones after they part.

Introduce, accompany, and let go — with framing

The three moves a second character needs, and none of them are stated in words:

move	how

© Andre Bremer · https://hyperquader.com

```
introduce  she appears doing something of her own, before she is spoken to
accompany  both described on every line; vary the distance between them
let go     one panel with only one person in it
```

In amberlin-in-berlin-afternoon.txt the two women say goodbye in a doorway at the end of scene three, and scene four opens:

> --image wide shot, Amberlin alone in the passage outside, the lit doorway small behind her

Nothing says "Nadia has gone". The frame says it, because she is not in it. Say it in words — "she leaves", "after saying goodbye" — and you get a picture of a woman leaving, which is a different and worse panel.

Turning day into night is one line

Time of day belongs to the scene, so it changes once and every panel after it inherits the change:

> --scene Prenzlauer Berg at dusk, low orange sun between buildings, long shadows, first lamps warming

None of the six panels under it mentions evening. They describe what she does; the light comes from above them. Write "in the evening" into each --image and you are spending the panel's own words on something the scene already said, and they will disagree with each other by the third one.

The same applies to weather and season. If it is in every panel of a stretch, it is not a panel's business.



© Andre Bremer · https://hyperquader.com



amberlin-in-berlin-morning panel 6 and -evening panel 4, both on zit. Neither --image line mentions an hour: one says she sits on a kerb to wait, the other that she crosses a junction as the streetlamps come on. The light, the emptiness and the season come from one --scene line each. The same two pictures do a second job — see the outfits below.

Deny text, and then stop asking for it

No model in the catalogue can spell. Left alone it fills shopfronts, street signs and museum labels with glyph soup — shapes that look like writing, say nothing, and pull a reader's eye straight to them because eyes go to text. A street scene is the worst case, and Berlin is all street scenes.

So each file denies it once, at the top, and every panel in it inherits it:

> --avoid "text, lettering, letters, words, writing, captions, signage, shop signs, street signs, labels, numbers, …"

The catch is that a negative and a panel can contradict each other, and the panel usually loses in a way that just looks wrong. The first draft of this story asked for *she reads a street sign*, *she reads a wall label*, and *a notebook with a word underlined twice* — three panels demanding legible text from a run that had been told to draw none. They are now *she stops where two streets meet and looks down each in turn*, *she leans close to a glass case*, and *a rough pencil drawing of a carved stone volute*: the same beat, carried by the action instead of by words nobody can read.

The rule generalises. Whatever you deny, stop asking for — the negative is not a filter applied afterwards, it is a second prompt pulling the other way.

And a better model makes a bad prompt look worse. Nine panels across these stories still asked for writing after that rule was written: *a paragraph in her handwriting*, *the day in*

© Andre Bremer · https://hyperquader.com

her handwriting, *the day's dishes chalked outside a trattoria*. On a small model they came back as a suggestion of writing, blurred enough to pass. On a model with a larger encoder they came back as a page of confident gibberish held up to the camera — and one of them abandoned the single full-bleed frame entirely for a sketchbook page of panels, the contact-sheet failure returning through a different door.

The model was not the problem and neither was the negative. A faithful model draws what you asked for, so it exposes what you actually asked for. If output gets *worse* when you change to a better model, read the prompt again before blaming the model.

The fix in every case was to keep the beat and drop the demand: a pencil drawing of the fountain rather than a paragraph about it, *she looks in at a trattoria doorway, deciding* rather than reading a chalked menu. A notebook with a drawing in it says she is working things out just as well as a notebook full of words, and it is a thing a model can actually draw.

Your watermark is unaffected. --book composites that after the picture is generated, so the one piece of real lettering in the book is the only one the model never had a chance to ruin — and being composited rather than generated is also why it can be rounded and laid on gently instead of stamped.

Panels are instants, not events

A prompt is a single frozen moment. These do not work:

- *she goes to the bakery for bread* — "goes to" has no appearance
- *discovering the city culture* — not a thing that can be drawn
- *she spends the afternoon reading* — which second of the afternoon?

Each has to become something a camera could catch: *she steps into a bakery doorway, warm interior light on her face, racks of baguettes behind the counter*. Subject, pose, setting, and where the light is.

A good test: if you cannot say where the camera is, the model cannot either.

Motion is a line under the panel

A panel can move. --animate under an --image puts that picture into motion: the clip starts on the picture, and its prompt is the picture's own composed prompt followed by the motion. So the motion line says only what moves:

```
--image she stops on the bridge, the river wide and grey below her
--animate "a barge slides under the bridge, its wake spreading" --frames 121
```

--video is a clip of its own, made from its words alone with the look and the scene around them. There is no picture to carry the frame, so its text has to say what is in it:

```
--video "the empty street at first light, a tram crossing far off" --frames 97
```

The same rule as a panel holds for both: one drawable movement, not an event. --book sends them to the animator named in servers.toml and plays each clip in its panel's place in the video. A clip that cannot be made leaves its panel as a still, and the run says how many.

A clip runs as long as its --frames say, or its --hold in seconds: --animate "the barge passes" --hold 8 is eight seconds of motion, 193 frames on ltx25. Without either, a clip takes its picture's own --hold, then the one in force, then the book's panel.

Two moments on one page: --panel

© Andre Bremer · https://hyperquader.com

Some moments belong together: a before and an after, a question and its answer, two people across a table. A --panel line under an --image adds one more picture to it, and the book shows them side by side as one page:

```
--image she reads the menu at a café table, chin on her hand
--panel the waiter sets down two coffees, steam rising
```

One --panel line makes a pair; three make a grid of four. Each picture is drawn on its own, from its own line with the look and the scene around it, so write each one as you would a panel: one instant the camera can catch. The --image line keeps the panel's options — its --ratio is the shape of the whole page, its --hold the page's time.

When to use a pair:

- **Two moments.** She lifts the cup; the cup is empty. The gap between them is the story, and the reader fills it.
- **Before and after.** The shutter going up, then the bakery open. The square at dawn, then the same square full.
- **Two people.** Her face, then the stranger's, each in a close shot, where one wide picture would put them both at a distance.

Use one a scene at most. A pair is a beat, not a habit: a book of pairs reads as a contact sheet. Leave a panel still; an --animate under it is refused. A grid of four is a montage — a market's stalls, four hands at work — and suits a scene that is a list rather than a moment.

Words are a panel too: --card

A card is words drawn as a panel of their own — a title, a date, a line of dialogue, a quote. acue draws it in the house font, sized to the panel, wrapped and centred, and no model ever sees it. Where it is drawn is the line above it:

```
--card Amberlin in Rome

--image she stops on the bridge, the river wide and grey below her
--card "The city before it is busy." --hold 5
```

- **Directly under a picture's line**, with no blank line between, the card is drawn over that picture, blurred and darkened or lightened until the words read. The picture keeps its own panel; the card follows it. In the video the picture goes from sharp to blurred at the start of the card's time and the words fade in on it.
- **After a blank line**, the card is on the segment's paper colour: the colour the video dips to after that hour, which is the colour the day already blends through — amber in the morning, orange at lunch, rust in the afternoon, dusk violet in the evening, and a dark grey for any other segment (a photo book's day, a lone story).
- **Several lines** are a block, closed by --card --end. The lines are kept as you wrote them; a long one wraps.

```
--image she stops under a stone arch where washing hangs above her
--card
Every stone here has stood
longer than the story.
--card --end
```

A card is the size of the pictures around it, so a book with --ratio wide has 16:9 cards. It is a page of the book like any picture — in the video, the .acue, --proof and --print — and the book keeps its words beside it, so a reader can show them in its own type. It is not read aloud and never reaches a language model.

© Andre Bremer · https://hyperquader.com

Words spoken over a scene: --speak and --narrate

A narrated book speaks a line or two over each scene, and the music eases down under it. Write the words above the scene's first picture, after a blank line:

```
--description Amberlin eats alone
--scene a small Berlin restaurant in the evening, misted glass, warm lamps

--speak She found warmth in the small restaurant, letting the misted glass soften the evening. Her pencil traced a tram wire, the brass letter holding the page still.
--image she sits at a corner table, the window misted beside her
```

The line covers the pictures after it until the next spoken words, and is spoken once, from the scene's first picture. Write what those pictures show: a listener hears it while looking at them. Two sentences fill a scene of six; a longer line holds the scene's stills longer rather than being cut. acue --narration drafts a line for every scene that has none, for you to edit.

A story told in paragraphs can be spoken as it is: from a --narrate line every paragraph is spoken over the pictures after it, until --narrate --end or the end of the file, and with --illustrate each paragraph is a picture too. A --speak directly under a paragraph says it another way, and --speak "" there keeps it silent. NARRATION.md has the rest: the voice, the duck, and the book's [narration] table.

How long a panel holds: --hold

--hold is how long something is on screen, in seconds, from 2 to 15. On a line of its own it holds every picture, photo and card after it until it changes; on an --image, --photo or --card line it is that one's alone and wins; without either, the book's panel stands. --reset --hold goes back to the top of the file's.

```
--hold 8
--image "the bridge at dusk"
--card "Rome, evening" --hold 4
```

The picture holds for 8 seconds and the card for 4. A picture with clips holds for its clips; its --hold is their length instead, unless the --animate line gives its own. The music is generated to the holds as it is to the pace, so --print-soundtrack counts them.

Vary the shot, or the model will not

Left alone, a run settles into one distance — usually a medium shot of a person centred in frame — and a city of those is a contact sheet with extra steps. Say the framing:

```
wide shot from behind, …        low angle, …        close shot on her hands, …
```

Eleven of the sixteen scenes in Berlin's four files end on a wide shot, and that is the pattern worth stealing: whatever happens in the middle, pulling back at the end closes a scene the way a paragraph break closes a paragraph. The close shots go where the attention is — a notebook, a face, four hands over a book.

Not every line needs a framing word. The middle scenes there mostly do without, because a line already carrying two described characters has no room for one, and an unmarked line gets a medium shot, which is a reasonable default to fall back to. State the framing when it is doing work: the opening, the closing, and wherever the subject is not the whole point of the frame.

© Andre Bremer · https://hyperquader.com

Scene length, and why six

Fewer than about four panels and a scene does not settle: the setting appears, is not explored, and is gone. More than about eight and it stops moving. Six gives room for an establishing shot, three or four beats, and a closing one, and it divides a book into a shape a reader can feel without being told.

How many panels is how long a video

A story is not only read — it becomes a segment of a video, and its panel count is what decides how long. The book's [video] table sets how long a panel is on screen and how long a pair overlaps; the arithmetic, and what it means for the music, are in SOUNDTRACK.md. At the default pace a panel holds alone for three seconds; Paris sets panel = 6.0, which gives five.

Two consequences worth having before you write:

A panel is seconds of someone's attention. General slideshow guidance puts 4–6 seconds per image as the band that lets a viewer register a picture without getting bored, and detailed or illustrated images earn the upper end. That is a long time to look at a panel that is only a small change from the one before it — which is another reason to vary the shot rather than repeat it.

The music is generated to that length, so the panel count decides the track too: adding a panel adds its seconds of music as well as of video.

Guidance on total length is 5–10 minutes for a slideshow. Four segments of twenty-four panels sit at about five minutes at the default pace and eight at Paris's; four of forty at Paris's would be thirteen, which is too long.

Put the shared words first, the panel next, the scene last

acue composes each prompt as look, panel, scene, which is worth knowing when you are deciding how long to make each layer.

The look leads because it is identical on every panel: identical leading tokens are what hold the cast still. The scene trails the panel because it is the setting clause, and a scene in front of the panel bleeds into it — measured, on a panel asking for an interior gallery under an outdoor scene, where the scene's gravel appeared on the gallery floor.

The practical consequence: **keep the look and the scene short.** They are on every panel, and every word in them is a word competing with the one thing that makes this panel different from the last.

A book is a city, and the file beside the stories says so

Four story files named for the hour are one book: one .acue, one soundtrack, one video beside it. What says so is <city>.book.toml sitting beside them — named for the city, so nothing inside it has to name one.

```
prompts/
  amberlin-in-berlin-morning.txt
  amberlin-in-berlin-lunch.txt
  amberlin-in-berlin-afternoon.txt
  amberlin-in-berlin-evening.txt
  amberlin-in-berlin.book.toml
```


© Andre Bremer · https://hyperquader.com

The join is the name: a story belongs to the longest city name it starts with, and what is left is its hour. So the segment word at the end of a filename is load-bearing — rename …-morning.txt to something else and it silently stops being the morning of that day.

A story no book file covers is still a book: its own .acue, one story, no music, which is right for a lone story. acue --print-soundtrack names any story no city claims, before anything is generated.

Seeds do not hold a story together

A fixed seed is worth having for reproducibility — you can re-run the file and get the book back — but it is not what gives a story continuity. Words are. With a long shared prefix a fixed seed makes panels *rhyme*, repeating compositions, which is the opposite of the varied framing above.

Set one so the run is repeatable. Do not expect it to do the coat's job.

The seed is also what makes one wrong panel fixable on its own. Edit its --image line in the unpacked bundle and acue --rebuild <book.acue> --images redraws that panel with the story's seed, so it lands among its neighbours rather than in a different room — see **Fix one panel** in GUIDE.md.

Know what is reading your prompt

Not all models want the same prose. Two families, and they behave differently:

- **CLIP encoders** (SDXL and its descendants) read 77 tokens at a time, weight the early ones heavily, and are native to comma-separated phrases. Front-load them.
- **Language-model encoders** (Z-Image on Qwen3, ERNIE on Mistral, anima on a small Qwen3) read ordinary language and have no 77-token window — but a small *base* model is extremely literal, which is where the contact sheet came from, and will not recover from a misspelling the way a large one might.

acue --print-metadata <png> on an image you liked tells you what was actually sent. It is the fastest way to learn what a given model rewards.

Quality and obedience are separate axes, and a good model can fail the second one. Across a sweep of every example story on five models, krea2 drew the best pictures of the set and was also the one that quietly dropped described elements in a minority of them. A prompt that produced a wrong picture on one model is therefore not evidence of a bad prompt. Generate the panel that failed on a second model before touching the words: if another model got it right, the prompt already said what it needed to, and rewriting it to satisfy the first model makes the line worse everywhere else.

Name the file for the shelf, the story for the reader

A prompt file is named for sorting and for the shell — amberlin-in-berlin-lunch.txt — and a reader wants "Amberlin in Berlin". Both, so neither has to be compromised: the title is made from the filename by turning - and _ into spaces and capitalising each word. Short joining words (a, an, and, at, by, for, from, in, of, on, or, the, to, with) stay lower case unless they come first.

```
amberlin-in-berlin.txt  ->  Amberlin in Berlin
amberlin_in_paris.txt   ->  Amberlin in Paris
the-old-city.txt        ->  The Old City
```

The rule is mechanical, so it is also occasionally wrong. A file named 01-arrival.txt for

© Andre Bremer · https://hyperquader.com

ordering titles itself "01 Arrival". And the Amberlin files, being split by the hour, would derive "Amberlin in Berlin Lunch" — which is why every one of them sets its own:

```
--title Amberlin in Berlin · Lunch
```

That is what --title is for: the derivation handles the ordinary case so most files need nothing, and the override handles the rest without a list of special cases growing inside the rule.

```
--title The Amber Road
```

Naming a series. The twenty Amberlin files share a prefix so they sort together — by city, then by hour — and their look layer is word for word identical across the cities for each hour, which is what makes them one series rather than twenty unrelated books. London, the fifth city, took four new files, the city's name changed in each --title, --watermark, --location and --keywords line, and its own .book.toml; nothing else moved. The look layer is copied from an existing file rather than retyped, because twenty hand-typed copies of one paragraph is twenty chances for the series to quietly stop matching. The bundle keeps the *filename*, not the title, so two stories can share a title without one quietly overwriting the other.

The watermark stays explicit. --watermark Amberlin in Paris · Lunch is written out even though the title says the same thing, because a stamp burned into every page of a book is not a thing to acquire by implication.

Check before you generate

acue --print-prompts composes every panel and prints it with its settings, without contacting anything. A city's panels are minutes to an hour of GPU depending on the model — GUIDE.md has the measured figures — and the check is instant, and it is where you find the layer you left set, the scene that never changed, and the panel that is still a sentence about an afternoon.

One thing --print-prompts cannot tell you, because it is about what is *missing*: read the look layer for anything you would notice changing between panels but have not described — a length, a sleeve, a silhouette. That is the gap that reads as a bad model later.

Then --book bundles the result with the file that made it, so the story and its pictures stay together. See GUIDE.md.

© Andre Bremer · https://hyperquader.com

Writing a soundtrack

How to get four tracks that sound like one piece of music at four times of day, rather than four pieces that happen to be in the same folder. The mechanics, which field does what, are in acue(1); ACE-Step's versions and traps are in COMPOSE.md; this is about what to write in the fields, and it is the audio half of STORY.md.

A soundtrack runs on Stable Audio 3 Medium by default, served by sa3-server. ACE-Step 1.5 sings songs: a book can have a short sung song a segment instead of an instrumental bed ("Songs" below), and the sections from "ACE-Step 1.5 takes direction differently" on are about it.

The worked example is the same day those stories tell: one character walking through a city from morning to evening, four prompt files, four segments. The music is the third layer telling that story, after the wardrobe and the colour grade. It should be recognisable as one tune returning — which is exactly what the character on screen is doing.

A book is a city

One city, one book, one .acue: four story files named for the hour, the four tracks that play under them, and one video of the lot beside it. The file that says so is <city>.book.toml, sitting beside those stories — so the file *is* the city and nothing inside it has to name one, which is what keeps it to thirty lines.

acue.toml beside the binary keeps what every city shares: the font, the author, the terms. The city book overrides those and adds the music. A story with no city book covering it is still a book — its own .acue, one story, no soundtrack.

acue --print-soundtrack shows the whole plan before anything is generated: every city with its model, seed and cover settings, its tracks in the order they will run, the panels each story has, the seconds of audio that needs (padding included), and which stories no city claims.

Four rules

Happy, never sad. The books are made to entertain people, not to depress them. Their music, like their songs, narration and pictures, is happy, content, balanced, honest and loving: never sad, melancholic or depressing. Calm is allowed; sombre is not. Write it into every prompt in plain words, a bright major key and a mood such as joyful, sunny, warm or content, because a model asked only for "soft background music" leans towards the sad side of calm: every Rome take of a slow adagio came back sad, and the owner rejected all of them.

A set of tracks is one theme re-voiced, not several pieces. Same melodic idea, same harmonic feel, revoiced for the hour. Four unrelated pieces make the seams between segments sound like mistakes rather than like time passing. On Stable Audio the four prompts share every word except the time of day and the tempo; on ACE-Step the covers carry the theme.

Acoustic instruments only. Name real ones — upright piano, nylon-string guitar, accordion, vibraphone, clarinet, muted trumpet, brushed drums, upright bass. No synthesiser, no drum machine, and no electric piano: a Rhodes reads as a period signature rather than as a room, and it fights illustrated artwork that has no electricity in it either. If a caption produces something obviously synthetic, name the acoustic instrument you wanted instead — that is a stronger correction than any adjective.

© Andre Bremer · https://hyperquader.com

It plays behind something. This is a background bed for a wordless picture story, and it must not demand attention: played softly, no solos that pull focus, no dramatic arrival, nothing that makes a viewer look up from the page. The mood is the city's character on a happy day (sunny in Rome, romantic in Paris), always warm and never sombre. Warmth over cleverness.

Everything below is in service of those four.

Stable Audio 3 Medium, the default

A book's [soundtrack] runs on Stable Audio unless its model names an ACE-Step checkpoint. model = "sa3" says so explicitly, as the city books do. sa3-server renders each track on its own, in one request: there is no planner and no cover chain. For each track acue sends three things:

field	from the book
prompt	the track's prompt, then " 72 BPM" from its bpm. Stable Audio has no tempo field, so the tempo goes into the words
seconds	the segment's length plus pad, the same arithmetic as below
seed	the track's own seed, else the soundtrack's; a seed of -1 rolls one per run

with the eight steps and CFG 1.0 that models.toml gives the checkpoint.

A seed is a track. A fixed seed and prompt give the same file byte for byte (measured, docs/DECISIONS.md §47), so a book keeps its music as seeds, not as WAV files. Pick them by listening:

```
acue --audition                       # four seeds of every track of every Stable Audio city
acue --audition --takes 8 --out auditions/berlin --prompts-dir berlin/
```

Each take is printed with its seed and written twice, cut as the video plays it and as generated (see "Auditioning" below). A file already in the folder is never replaced. The track lengths come from the story files, so a city can be auditioned before a single picture is drawn. Each render takes about a second: the four city books at four takes a track took 77 s with the model load. Put the seed to keep on its track:

```
[[soundtrack.track]]
segment = "morning"
bpm     = 92
seed    = 2811460337
prompt  = "Soft background soundtrack, 1980s Berlin film music in a bright major key, ..."
```

The ACE-Step keys (timesig, melody, codes, planner, cover_seed, format) do nothing on Stable Audio, and a book that sets one is refused when the run starts rather than ignored. A track's seed is refused on ACE-Step for the same reason.

Prompting Stable Audio

The city books' prompts share a shape:

- **Prose, not tags.** Two or three sentences that say what the music is.
- **Start with what it is for, and how loud.** "Soft background soundtrack, 1980s Berlin film music in a bright major key, played softly with a light step." Every city's prompt opens this way.
- **A style and an era, in a major key.** "1980s French film music", "a sunny 1960s Italian film comedy score". "In a bright major key" keeps it happy; "played quietly and slowly"

© Andre Bremer · https://hyperquader.com

made it sad.
- **Acoustic instruments, by name.** "A cheerful solo piano, a warm cello and soft strings"; "a cheerful accordion melody, soft strings, a nylon-string guitar and upright bass". "Acoustic instruments only" says it once more.
- **What to leave out, when it matters.** "No synthesizers, no electronic sounds, no drums, no sharp or high notes." Paris keeps its light, brushed rhythm, so it leaves the drums alone; a city with hand percussion says "no drum kit". "No sharp or high notes" is in every prompt since Rome's harpsichord and high violins went shrill.
- **A happy mood and a picture.** "Unobtrusive and happy, warm and content, a fresh, sunny morning over the city." Joyful, sunny, warm, content and loving; never wistful, contemplative, melancholic or sad.
- **End with what it plays behind.** "Made to play behind pictures."

Stable Audio 3 Medium makes instrumental music, so a prompt needs no "no vocals". The tempo goes in bpm, not in the prose: acue adds it.

A day is one prompt four times. The morning sets the sound; lunch, afternoon and evening change only the time-of-day phrase ("a bright, busy midday over the city", "a warm, glowing evening as the city lights come on") and the tempo, which quickens at lunch and eases after it but stays lively: 92, 100, 88 and 84 BPM in Berlin.

ACE-Step 1.5 takes direction differently from the model before it

There is a Qwen3-family language model in front of the diffusion transformer, acting as a planner: it reads the caption and reasons its way to structured metadata and an expanded caption before any audio is rendered. Four consequences, all of which contradict advice written for the 1.0-era model:

- **Write prose, not tags.** Comma-separated tags under 120 characters was advice for a model with no planner. Describe the instrument, the mood, the texture, and what the piece is *for*. Terse tag soup gives the planner less to work with, not more.
- **There is no instrumental switch.** The old [inst] marker is gone and no boolean replaced it. What produces an instrumental is an empty lyrics field and saying "no vocals" in the caption — every caption below ends that way, and that is why.
- **bpm, time signature, key and seed are fields, not caption text — and the caption should not repeat them.** Upstream's own guidance is explicit: a tempo or a key in the prose competes with the parameter rather than reinforcing it. Say what the music *is* and let the fields say how fast and in what.

  The time signature is a bare number, not a fraction: "3" is 3/4, "4" is 4/4, "6" is 6/8. acestep/inference.py:68 enumerates that mapping, which is worth citing because another docstring in the same file says e.g. "4/4" and upstream's own skill copied the looser one. A fraction where a number is expected is exactly the sort of thing that silently produces the wrong feel.

  Key is best left empty unless there is a reason for one. Where there is, C, G, D, Am and Em are reported as the most stable.
- **The caption reaches the music model as written.** The server can let the language model rewrite it first (use_cot_caption, on by default in its API and off in its own interface); acue always sends it off, so the words in the book are the words the music is made from. The planner still writes its plan and the metadata the fields leave open. A soundtrack sends its vocal language as en. The interface sends unknown for an instrumental, but by ear en made the better music: three of four English takes passed, one of four unknown.
- **lm is the planner.** On for a fresh generation. Covers skip it entirely — see below — so on a cover the caption reaches the diffusion more directly and concrete timbral words carry more weight than mood words.

© Andre Bremer · https://hyperquader.com

Which ACE-Step model

For a song, use **ace-xl**, the 4B XL turbo: model = "ace-xl" in the book's [soundtrack]. It is the best-sounding ACE-Step checkpoint in the catalog. The run stops the image server and starts that one itself.

It comes with a constraint, and the catalog states it rather than leaving it to be discovered mid-generation: the XL 4B runs out of memory on a 16 GB card paired with the 1.7B language model — 298 MiB short, mid-sample, returned as a failed job rather than a refusal. ace-xl therefore pairs it with the **0.6B** LM, which is what makes it fit. That pairing is the modules entry in models.toml; do not override it.

If the first track generates and every cover after it fails out of memory, the covers are not the problem: the allocator setting in COMPOSE.md is missing.

Switching models is a restart, the same as it is for Forge. Start ace-xl from the Launch pane and leave it up for the whole set: every track in a chain should come from one checkpoint, or the covers are varying a source the current model never made.

The cover chain is the mechanism

Two ways to make tracks resemble each other, and they are not equally strong.

A shared seed is the weak version. Four captions on one seed start from a common place, but the planner re-plans each from scratch and they drift.

A cover chain is the strong version. Generate the first track from text. Then generate each of the others as a *cover* of it: the source's latents are cloned and anchor the structure, while the diffusion moves the style toward the new caption. The planner's structural work is skipped because the source has already done it.

In the Audio pane that is the source row. Make the first track, press c to hand it to the next one, rewrite the style, generate. C clears it. What c puts in the row is a staged copy rather than the take's own path, for the reason docs/COMPOSE.md gives. A path in source is what makes a request a cover; there is no separate task type to disagree with it.

Two knobs, and their API names lie about which is which. This cost a whole soundtrack, so it is worth stating plainly.

book key	API field	what it is
melody	cover_noise_strength	how much of the source's tune survives. 0 keeps none of it — pure style transfer from the caption. 0.1–0.25 is the server's recommended range. 1 keeps the melody and starts resisting the new style.
codes	audio_cover_strength	how closely the diffusion follows the language model's own plan. Nothing to do with the source audio. 0 ignores the plan, 1 is the server's default and follows it closely.

audio_cover_strength reads like the source dial and is not one; the server's own interface labels it "LM Codes Strength" and labels the other one "Cover Strength (Melody Retention)". Sending the first and never the second gives a chain with no theme carried at all, which is what a set of covers that sounds like two songs playing at once actually is.

Full plan adherence and the source's own seed must not be combined. Measured by ear over two sweeps of the same source: codes = 1.0 together with the seed the source was made from is the one configuration that comes out rigid and doubled — the model re-plans nearly the source's structure, follows it exactly, and the source conditions against it. Either constraint on its own is fine. Rolling a cover's seed removes half of it whatever codes

© Andre Bremer · https://hyperquader.com

says, which is why covers roll by default and only the source uses the book's seed.

Moods: a track without writing its prompt

A track may name a mood instead of a prompt. A photo book's draft does, one per part of the day:

```
[[soundtrack.track]]
segment = "2026-07-14"
mood    = "holiday"
```

Each mood is a prompt written by the rules above, with a tempo, which a track's own bpm replaces:

mood	bpm	the music
holiday	90	a nylon-string guitar, ukulele, marimba and upright bass: sunny, long days by the sea
family	80	piano, cello, acoustic guitar and a music box: warm, an afternoon at home
travel	88	a fingerpicked guitar, violin, accordion and upright bass: curious, a long road
calm	68	solo piano, soft strings and a harp: still, evening light over water

All four end the same way: no synthesizers, no electronic sounds, no drums, ambient and unobtrusive, made to play behind pictures. A track takes a prompt or a mood, never both; one with neither has no music, and the book file says so at its line.

No music, or your own

A book needs no soundtrack. With no [soundtrack] table, or with model = "none" in it, the book and its video are silent and no audio server starts; "none" keeps the tracks in the file, so one word brings the music back. Where Stable Audio is not set up on the machine, a book is made without music all the same, and one line says why.

A music file of your own replaces the made tracks. Name it at the top of the book file, relative to it:

```
title = "Rome 2026"
music = "our song.mp3"
```

Each part takes the next stretch of the song, cut to the part's length and faded in and out as a made track is, and the song starts again when the book runs longer than it. The cuts go into the book as audio/.flac, so the .acue plays them under their chapters and the video is cut from them. A [soundtrack] beside music makes nothing, and the run says so once.

Under a narration

A narrated book (--speak lines or --narrate regions in its stories, NARRATION.md) plays its music under a voice. The tracks are made as they always are, to the segment's length after the narration has lengthened any scene it outruns, so nothing loops. In the mix the music is 6 dB down throughout ([narration] music) and eases down 12 dB more before each line (duck, over duck_lead), coming back up after it (release). Write the music as you would for pictures alone: it is the ducking that makes room, not a quieter prompt. A song book is not narrated.

Songs: a short sung song a segment

© Andre Bremer · https://hyperquader.com

A book's soundtrack can be one short song a segment, sung by ACE-Step in the book's language, instead of an instrumental bed. In listening tests the songs were "miles better than the instrumentals". The singer is the book's main character, singing in the first person about that part of the day.

```
[soundtrack]
kind     = "song"
language = "fr"
singer   = "Amberlin, a young woman in her mid twenties with copper-red hair in a
short plait"

  [[soundtrack.track]]
  segment = "morning"
  style   = "chanson française, café accordion, upright bass, brushed drums, a young
woman's clear vocal singing in French, romantic and light"

  [[soundtrack.track]]
  segment = "lunch"
  style   = "..."
```

key	what it is
kind	"song"; "instrumental" is the default and needs no line
language	the code ACE-Step sings in: de, fr, it, uk, … (its VALID_LANGUAGES)
singer	who sings, in the first person. The lyrics are written for this person
model	ace-xl unless named: the XL turbo with the 0.6B language model. Stable Audio refuses a song
style	per track, the caption the song is sung in, in place of prompt
file	per track, a kept take, relative to the book file. Written by acue --keep

bpm, timesig, planner, format and pad mean what they mean for ACE-Step. Each song is its own text2music, not a cover of the first, so melody, codes and cover_seed are refused.

Write the style the way the tests did. A genre with its place, the instruments by name, the voice and the language it sings in, and the mood: "chanson française, café accordion, upright bass, brushed drums, a young woman's clear vocal singing in French, romantic and light". Name the voice in every track: a style without one leaves the model to choose the singer.

The lyrics

```
acue --lyrics --prompts-dir paris/          # drafts what is missing
acue --lyrics --force --prompts-dir paris/  # drafts over what is there
```

Each segment's lyrics are a file beside the book, <city>-.lyrics (not .txt: every .txt in the folder is a story). --lyrics drafts them with the language model servers.toml names (kind = "agent", the Spark's agent), in two questions:

1. **A summary.** The segment's captions, in story order — its --description lines and each picture's own words, or a photo's --description — told back as a short story of four to six sentences, written in the book's language. That way everything the second question shows the model is in the song's language: an English summary read last was sometimes followed instead.
2. **The song.** Lyrics from that summary, in the book's language, sung in the first person by the singer, with these rules:
   - be creative, and use the story's moments as inspiration, not as a checklist: pick the few images that make a good song, leave the rest out, and invent freely around them. A

© Andre Bremer · https://hyperquader.com

song made from every element it is given is not a good song;
- simple and singable, with a four-line chorus that repeats;
- in the book's language;
- shaped to the segment's length, which the question states:

segment	sections	lines of singing
under 50 s	[intro] [verse] [chorus] [outro]	8–10
50 to 95 s	[intro] [verse] [chorus] [verse] [chorus] [outro]	12–16
longer	the same with [instrumental] in the middle	14–18

The reply is checked for its language by its common words (English, French, German, Spanish and Italian; other languages are not checked). It is a safety net: with the summary in the song's language, 32 drafts of the Paris and Berlin songs all came back in the right language (docs/DECISIONS.md §56), but a reply that does not is asked for again once, and a second reply in the wrong language stops --lyrics with that said, and nothing is written.

For a song not in English, the summary and the finished lyrics are then each translated into English, to read along with — not to sing. A translation the model does not give leaves the lyrics without one.

The intro is about 6–8 s of music and the outro 8–10 s. A longer segment gets a break rather than many more lines, because a long song drifts: a five-minute take went off at 4:10, while three minutes held.

The file starts with # lines, which are never sung: what it is, and the summary it was drafted from. The English translation comes after the lyrics, under # English translation (not sung), and every line of it starts with # too, so none of it reaches ACE-Step:

```
[chorus]
Ich atme tief die Morgenluft,
…

# English translation (not sung)
#
# The summary:
#   Amberlin reaches Mitte in Berlin and stands, uneasy, at a deserted street
#   …
#
# The song:
#   [chorus]
#   I breathe in the morning air deeply,
#   …
```

Edit the lyrics freely. **A lyrics file that is there is kept**, and nothing is asked, unless --force says otherwise; a song whose lyrics you write yourself needs no language model at all. With no agent in servers.toml, or one that does not answer, --lyrics says so and stops. An instrumental book never asks.

Audition and keep

```
acue --audition --prompts-dir paris/ --out auditions/songs   # two takes of every song
acue --keep auditions/songs/amberlin-in-paris/amberlin-in-paris-morning-seed-1234.wav
```

--audition on a song book drafts any missing lyrics first, then renders two takes of every song (--takes changes it) in one batch request a song, at the segment's length plus the pad. Each take is written three times: <city>--seed-<N>.wav cut and faded as the video plays it, …-raw.wav as generated, and …-seed-<N>.json, its seed and request. Nothing already

© Andre Bremer · https://hyperquader.com

in the folder is replaced.

A take is kept as a file, not as a seed. ACE-Step cannot make a take again: its planner samples, and its server drops the plan, so the same seed and request give another song. --keep copies the take as generated to audio/<city>-.wav beside the book, with its record, and writes the track's file:

```
[[soundtrack.track]]
segment = "morning"
file    = "audio/amberlin-in-paris-morning.wav"
style   = "chanson française, café accordion, …"
```

A kept file is used as it is on every build and never made again. A build sings only the songs that are not kept, from their lyrics, and a book whose every track is kept starts no audio server. A kept file that has gone stops the book rather than becoming a new song. Keeping another take of the segment replaces the one kept before. file works on any track, instrumental too.

The .acue plays a song like any track. Its lyrics are not in the book yet: the format has no place for them (MoSCoW).

Track length is arithmetic, not a guess

A track must be at least as long as the segment it plays under. Generate to length rather than relying on a loop: a two-minute segment hears a short loop twice and it gets tiresome.

The pace is the book's, in its [video] table:

```
[video]
panel = 4.0     # seconds a panel is on screen, crossfade included
xfade = 1.0     # how long a pair overlaps
```

Each panel is on screen for panel seconds and each pair overlaps by xfade, so a segment of *n* panels runs

```
segment = n × panel − (n − 1) × xfade
```

At the default 4.0 and 1.0 that is 3n + 1 seconds: twelve panels is **37s**, twenty-four is **73s**.

A panel with --animate clips is the exception: its slot lasts as long as its clips played one after another, each --frames over --fps (snapped to what the animator takes), or panel seconds when the line gives no frame count. A --video entry is a slot of its own, as long as its clip. The segment is then the sum of its slots less the crossfades. The length comes from the frame counts asked for, so the track can be generated to it before any clip exists, and a slot lengthens only for a clip an animator can make or one already on disk.

A picture, a photo or a card with a --hold is the other: its slot is its hold rather than panel. A card is a slot like a picture's.

Ask for more than that, and throw the extra away. pad defaults to 10, and it is there to be discarded.

The model returns precisely the duration requested, and composes a piece that **ends**: the last six seconds are a resolution, fixed rather than proportional — 5s of a 30s request and 6s of a 127s one. Measured on a real track, holding at −16 dB to 119s and then −23 at 121, −30 at 123, −58 at 125.

That cadence is where the off-notes live. Reported by ear across a whole sweep: every clip "plays a borderline off note at the very end". So the padding exists to push the entire cadence *past* the end of the segment, where book-video.sh's atrim discards it and the video's own 2.5s fade ends the music instead. Ten rather than six, because six is the cadence itself

© Andre Bremer · https://hyperquader.com

and the approach to it begins earlier.

Not 0, which would land the cadence exactly where the pictures end: a cadence nobody asked for, with a bad note in it, is worse than a fade.

ACE-Step handles ten seconds to ten minutes in one generation, so a full segment is never a problem. Do not worry about the ends — the assembly fades each track in and out so it breathes at the dips between segments.

The panel count is the number of --image entries in the story file (every paragraph, under --illustrate), which is the number of PNGs the run writes. Take it from --print-soundtrack rather than from memory: at the default pace a story that gained a panel gains three seconds.

Re-cutting a finished bundle

The pictures and the music are the expensive half — an hour and five minutes respectively for a city. The video is minutes, so anything that is only a question about the video should never cost a regeneration. Unpack the book with --rebuild, which generates nothing without a flag, and run the script over the folder it makes:

```
acue --rebuild out/<model>/<city>.acue                          # unpacks into
out/<model>/<city>/
scripts/book-video.sh --help                                    # every knob
scripts/book-video.sh --panel 6.0        out/<model>/<city> slower.mp4
scripts/book-video.sh --backdrop black   out/<model>/<city> bars.mp4
scripts/book-video.sh --width 3440 --height 1440 out/<model>/<city> ultrawide.mp4
scripts/book-video.sh --chunk 6          out/<model>/<city> lowmem.mp4
```

Flags, not environment variables, and --help lists all of them with their defaults. The launcher passes --panel and --xfade the same way — a script whose options are reachable only by exporting variables is one that nobody but its caller can drive, which is not what a program in share/acue should be.

One caveat for pace: a --panel other than the book's makes the segments outrun their music, which the script covers by looping the track. Fine for judging pictures, wrong for a finished book — put the number in the book and acue --rebuild <city>.acue --audio regenerates the tracks to fit.

What belongs in a book, and what belongs to the renderer

A setting lives in <city>.book.toml only if something other than the video script depends on it. That is one rule and it decides every case:

setting	who else depends on it	where
panel, xfade	the soundtrack — they decide how long a track is generated	the book
backdrop	nobody but the script	a book-video.sh flag
width, height	nobody but the script	a book-video.sh flag
crf, preset, dips, fades	nobody but the script	a book-video.sh flag

backdrop is not a book key: what fills the edge of a frame is the renderer's business, the same as the frame itself. A book has one identity and several possible cuts — 16:9 for publishing, 21:9 for a desk — and a key that implied otherwise would make the rule impossible to state.

© Andre Bremer · https://hyperquader.com

The frame is 16:9, and the bars on an ultrawide are the player's

1920x1080 for anything published; book-video.sh --help says why, and what a wider frame costs in picture. The bars a 16:9 video shows fullscreened on an ultrawide are outside the video, added by the player, and no setting here can reach them.

The blurred backdrop does fill the video's own frame, edge to edge, which is checkable rather than a matter of opinion: read the leftmost column of a frame and it is luma 107 on a scale where black is 0. If it looked like the blur had stopped short, it had not.

Hold one meter across the set

Do not change time signature between tracks in a chain. The cover works by cloning the source's latents to anchor structure, so a meter change fights the exact mechanism holding the theme together. Pick the meter with the first track and keep it.

Use a **different seed per chain** — see auditioning for where a rolled one that worked is printed.

The five cities

The city books in examples/prompts/ run on Stable Audio, one era and one set of instruments per city, so a listener who hears one then the next hears five pieces of music rather than one with a new coat:

city	style	instruments	tempo
Berlin	1980s Berlin film music, bright major key	cheerful solo piano, cello, soft strings; no drums	92, 100, 88, 84 BPM
Paris	1980s French film music, bright major key, a light swing	accordion, strings, nylon-string guitar, upright bass, brushed rhythm	104, 112, 100, 96 BPM
Rome	a cheerful Italian café band, bright major key	acoustic guitar, mandolin, upright bass, hand percussion; no drum kit	104, 112, 100, 96 BPM
Kyiv	1980s Ukrainian film music, bright major key	bandura, piano, warm strings; no drums	92, 100, 88, 84 BPM
London	cheerful British light orchestral music, bright major key	strings with pizzicato, clarinet, piano; no drums	100, 108, 96, 92 BPM

The tempos are morning, lunch, afternoon and evening. Every prompt is happy and says "no sharp or high notes" (STORY.md: books entertain, never sad); the first prompts, played quietly and slowly, came out wistful or sad. Each hour's track pins the first take of the happy audition, seed and all, so every model's book of a city carries the same music. Rome and London were auditioned in three styles each; the two not chosen stay in their book files as comments. London's book file also carries its song version in comments: the series' one song in English, so its lyrics need no translation. The prompts themselves are in the files and are not repeated here: a prompt copied into a document goes stale the first time the file is edited, and the file is what the run reads.

What a caption can specify

© Andre Bremer · https://hyperquader.com

Nine dimensions, and a caption naming several anchors the result far better than one naming a mood. Each caption in the worked pair's city books reaches for five or six:

dimension	examples
style	jazz, bossa nova, cafe waltz, folk-jazz, lounge
emotion	cheerful, sunny, playful, warm, contented
instruments	upright piano, nylon-string guitar, accordion, vibraphone, bass clarinet, muted trumpet, brushed drums, upright bass
timbre	warm, soft, bright, sparse, spacious
production	warm acoustic recording, intimate, clean
era or reference	gypsy-jazz, cool jazz, manouche
rhythm	unhurried, lilting, walking, at a brisker step
structure	long gaps between phrases, a closing-time reprise
what it is for	gentle background music

Two further rules worth having.

Granularity is a dial, not a target. Less detail leaves the model more room; more takes control. Neither is better — a chain's first take wants enough detail to be memorable, and its covers want enough to move.

Conflicting words degrade rather than blend. "Classical strings" and "hardcore metal" in one caption gives something worse than either. Two ways out, both better than deleting words: repeat the element you want more of, or turn the conflict into a sequence — "opens with soft strings, turns to brass in the middle" — which the planner can actually stage.

Sampling belongs to the checkpoint, not to taste

steps, cfg and shift are declared per model in models.toml, and the launcher and the Audio pane both read them from there. They are a property of the weights:

	steps	cfg	shift
ace, ace-xl — distilled turbo	8	1.0 (none)	3.0
ace-base, ace-sft — not distilled	50	7.0	1.0

A distilled turbo checkpoint is retrained so the whole sampling trajectory lands in eight steps, and it takes no classifier-free guidance at all (see below). A non-distilled one wants fifty steps and real guidance, which is how it can be steered.

Running either at the other's numbers produces bad audio. Twenty steps on a turbo model does not refine further, it desynchronises: judged by ear, "broken, two songs playing" — the same symptom a mis-wired cover chain gives, from an unrelated cause, which is worth knowing because the symptom does not identify the fault.

So on a turbo model there is no setting that trades minutes for fidelity: the eight steps *are* the model, and "highest quality, time no object" is answered by choosing a different checkpoint rather than by spending longer on this one. Selecting one in the Launch pane moves the pane's numbers with it.

A pinned key is not what breaks a track, but do not pin one anyway

A 20-step clip with key_scale = "G Major" sounds wrong, and reads as the key fighting the caption. It is the step count: the same settings without a key are just as broken. That clip varies two things at once, which is the mistake the section on judging changes warns against, and it costs a wrong conclusion until the isolating clip is played.

© Andre Bremer · https://hyperquader.com

The key is still left empty by default. The planner picks one from the caption, and nothing measured suggests overriding it helps.

The plan's own knobs

Four, all reachable in the Audio pane and all starting at the server's documented default, because a parameter the request sends and the interface hides can only be explored by writing a shell script:

field	default	what it does
lm-t	0.85	creativity against predictability. 0.0–0.5 safer and more conventional, 0.8–1.5 surprising and riskier, above 1.5 chaotic
lm-cfg	2.0	how hard the model is held to the caption. 1.0 none, 1.5–2.0 balanced, 2.5–3.0 faithful but less varied, 5.0+ introduces artifacts
top-k	0 (off)	restrict each step to the K likeliest. 10–30 focused, 50–100 diverse
top-p	0.9	nucleus sampling; 1.0 disables

All four steer the *plan* rather than its rendering, so all four are inert while the planner is off — the pane says so on the row rather than hiding them.

lm-t is the one to reach for when notes sit on the edge: the default 0.85 is inside the band upstream itself calls "surprising and riskier", and 0.4–0.6 asks for conventional harmony instead. Unmeasured by ear. A book run sends all four at these defaults; a value that earns its keep in the pane is not a book key.

What guidance does on a turbo model: nothing audible

Upstream says of guidance_scale that it is *"only supported for non-turbo model"*, and the catalog's audio models are all turbo. Two clips generated from one seed and one caption, at 7.0 and at 1.0, were reported as both sounding good and indistinguishable in character — so the parameter is inert here rather than harmful, and it is not worth a knob.

It is worth knowing, though, because it looks like a quality dial and is not one. If the harshness in a track needs chasing, guidance_scale is not where it lives. The unturned knob is lm-cfg (lm_cfg_scale, default 2.0), which steers the language model rather than the diffusion.

The same pair also ruled out the model: the 2B SFT — undistilled, fifty steps, real CFG — was judged less interesting and less warm than the 4B turbo. Being steerable did not beat being bigger.

Per-city ACE-Step knobs

melody, codes and planner are per book, so on ACE-Step a set of days can differ in how it varies rather than only in what it is made of. Kyiv once turned the planner off with planner = false, so lm-t, lm-cfg, top-k and top-p did nothing there. A book at codes = 1.00 is safe only because covers roll their own seed (cover_seed = -1). See the rule above.

Judging a change, without fooling yourself

Two sweeps here produced a wrong conclusion before they produced a right one, and both mistakes are easy to repeat.

Sweep one variable through a configuration that is not already broken. The first melody sweep ran at codes = 1.0, which turned out to be half of the one combination that sounds bad

© Andre Bremer · https://hyperquader.com

— so it measured retention through a strangled output and proved nothing about retention. If a whole row sounds equally poor, suspect the constants, not the variable.

A rolled seed makes every clip a different draw. Two clips that differ in a setting *and* in seed cannot tell you which did the work. Pin a seed for a sweep — but not the source's, since that is half of the bad combination. Include two rolled clips at identical settings as a control: if they differ from each other as much as the settings do, the settings are not what you are hearing.

Thirty seconds is enough. A cover that has lost the theme, or that sounds like two pieces at once, says so within a few bars. Full-length takes cost minutes each and tell you the same thing.

Generate one source and cover *that*, rather than regenerating a source per clip. Everything then differs in exactly what you changed.

When a track is not landing

In this order, cheapest first:

1. **Regenerate with a different seed.** Results vary a lot seed to seed and some simply capture a transformation better than others.
2. **Name the missing instrument.** Covers skip the planner, so concrete timbral wording reaches the diffusion more directly than mood description.
3. **Move melody.** Losing the tune, raise it. Too close to the source to sound like a new hour, lower it.
4. **Only then rewrite the caption.**

If the *first* track is wrong, fix it before generating anything else — which is what an audition is for.

Auditioning

On Stable Audio, acue --audition (above) renders four seeds of every track straight from the story files. For a finished book the same works through the bundle: acue --rebuild my.acue --audition renders the four seeds of every track into my/audio/, cut and raw as below, and --takes changes the count.

On ACE-Step the audition is of the first track only. A finished book is an .acue, and the book is the unit of work. It carries the story files, the city's book file and the panels, so a rebuild needs nothing else:

```
acue --rebuild my.acue               # unpacks into my/ beside it, says what to edit
  # edit my/<city>.book.toml: a caption, the seed, or both
acue --rebuild my.acue --audition    # two takes of the first track →
my/audio/<segment>-audition-seed-<N>.wav
  # listen; not right → edit again and repeat
acue --rebuild my.acue --audio       # the whole soundtrack and the video →
my-rebuild.acue, .mp4
acue --proof my.acue my-rebuild.acue
```

An audition is about a minute; the whole chain, the video and the book about eight. The first track is the one every cover hangs off, and a cover cannot rescue a source with no tune in it, so it is the one to judge on its own.

One audition asks for two takes in one request: the book's seed, and a second rolled one, each pinned and printed. Each take is written twice: -audition-seed-<N>.wav is what the video plays, cut where the segment ends and faded in and out as book-video.sh fades it,

© Andre Bremer · https://hyperquader.com

so it is judged as a viewer hears it; -audition-seed-<N>-raw.wav is the take as generated, with the pad and the model's own ending the video throws away (see "Ask for more than that" above). The first take is also the folder's .wav. With seed = -1 in the book each audition is a new roll, and the numbers to keep are on the screen; with a seed pinned, the first take is that seed. The book is the only place a seed lives.

Rewriting the caption is the stronger lever. A seed changes which piece in the caption's neighbourhood you get; a caption asking for calm gives calm on every seed.

Check before you build

Play the first track of one set against the first of the other. If you cannot immediately tell which is which, the two themes are not far enough apart, and the fastest fix is a new seed on the second rather than pushing the captions further apart.

Then play one set from first to last. Does it read as one day, or as four songs?

© Andre Bremer · https://hyperquader.com