Audion VS Engine and an AI agent

Русский · User guide · Console command · Operations · Presets

Contents

- Who does what
- The console command
  - Which preset
- Typical tasks
- Checking the result
- Rules

A guide for an AI agent (Claude Code, Codex and the like) a user has asked to work with the program. For a person: `USER_GUIDE_EN.md`; for the command line: `CLI_EN.md`.

Who does what

The program works on the picture of a video by VapourSynth presets, the same way at every run: it denoises (the strength following the noise it measures), debands, puts grain back, gives a film or retro look, deinterlaces, removes telecine, cleans compression blocks, rainbows and haze, upscales, and with an NVIDIA RTX card runs neural denoising, upscaling and frame interpolation. Every button of its window is an operation of the console command `vs_cli.cmd`, so an agent can do all of it.

The agent turns the user's words into a preset and its values, looks at a frame before and after, runs it, reads the outcome and checks the file. The agent does not write VapourSynth scripts or FFmpeg commands of its own where a preset exists: the preset already knows the formats, the levels and the order of the filters.

The console command

`vs_cli.cmd` lies in the root of the program:

```
vs_cli.cmd describe                                    operations, fields, choices,
the machine (JSON)
vs_cli.cmd run --operation ID --set NAME=VALUE         run an operation; --dry-run -
the commands it would start
vs_cli.cmd run ... --source FILE_OR_FOLDER --target FOLDER --result OUTCOME.json
vs_cli.cmd frame --operation ID --source FILE --out FOLDER    the middle frame before
and after, as PNG
```

The Source is `input\` of the program unless `--source` names a folder or one file; the Target is `output\` unless `--target` names another. Exit codes: 0 done, 1 a file failed, 2 a wrong call or a run refused by its checks (nothing was run), 130 stopped. The outcome file of `--result`: `outcome`, `ok`, `message`, `processed`, `skipped`, `failed`, `parameters` — every value the run went with — `seconds`, `log_file` and `data`. All the keys: `CLI_EN.md`. `--result` is a `.json` file outside the Source; give it a full path, or call the command from the program folder: a relative path counts from the folder of the caller.

Which preset

Section	Group	Preset	Call
PRECISION	Denoise	Mild Denoise	`vs_cli.cmd run --operation mild_denoise`
PRECISION	Denoise	Shadow Denoise SOTA	`vs_cli.cmd run --operation shadow_denoise_sota`
PRECISION	Denoise	Chroma Cleanup	`vs_cli.cmd run --operation chroma_cleanup`
PRECISION	Deband	Deband Safe	`vs_cli.cmd run --operation deband_safe`
PRECISION	Deband	Deband Fine Grain	`vs_cli.cmd run --operation deband_fine_grain`
PRECISION	Master	Filmic Rebuild	`vs_cli.cmd run --operation filmic_rebuild`
PRECISION	Master	Archive Clean	`vs_cli.cmd run --operation archive_clean`
PRECISION	Master	Pregrade Prep	`vs_cli.cmd run --operation pregrade_prep`
FILM LOOKS	Film stock	Film 35mm	`vs_cli.cmd run --operation film_35mm`
FILM LOOKS	Film stock	Film 16mm	`vs_cli.cmd run --operation film_16mm`

Section	Group	Preset	Call
FILM LOOKS	Film stock	Super 8	`vs_cli.cmd run --operation super8`
FILM LOOKS	Film stock	IMAX 70mm	`vs_cli.cmd run --operation imax_70mm`
FILM LOOKS	Look	Cinematic	`vs_cli.cmd run --operation cinematic`
FILM LOOKS	Look	Bleach Bypass	`vs_cli.cmd run --operation bleach_bypass`
FILM LOOKS	Look	Anamorphic Scope	`vs_cli.cmd run --operation anamorphic_scope`
RETRO	Video	VHS CRT	`vs_cli.cmd run --operation vhs_crt`
RETRO	Video	Camcorder 90s	`vs_cli.cmd run --operation camcorder_90s`
RETRO	Photo	Polaroid	`vs_cli.cmd run --operation polaroid`
RESTORATION	Interlace	QTGMC Deinterlace	`vs_cli.cmd run --operation qtgmc_deinterlace`
RESTORATION	Interlace	TIVTC IVTC	`vs_cli.cmd run --operation tivtc_ivtc`
RESTORATION	Noise	MVTools MCDegrain	`vs_cli.cmd run --operation mvtools_mcdegrain`
RESTORATION	Noise	DPIR Denoise [1]	`vs_cli.cmd run --operation dpir_denoise`
RESTORATION	Artefacts	Deblock H.264 Artefacts	`vs_cli.cmd run --operation deblock_h264_artefacts`
RESTORATION	Artefacts	Derainbow Decross	`vs_cli.cmd run --operation derainbow_decross`
RESTORATION	Artefacts	Dehaze Local Contrast	`vs_cli.cmd run --operation dehaze_local_contrast`
RESTORATION	Scale	NNEDI3 Upscale 2X	`vs_cli.cmd run --operation nnedi3_upscale_2x`
RESTORATION	Scale	Real-ESRGAN 2x [1]	`vs_cli.cmd run --operation vsmlrt_realesrgan_2x`
RESTORATION	Scale	Soft HD Rebuild 2x [1]	`vs_cli.cmd run --operation soft_hd_rebuild_2x`
PROFILES		Run a profile	`vs_cli.cmd run --operation profile_run`

[1] on vs-mlrt: shown with an NVIDIA RTX card or MLRT.

What each does, on what material, with which parameters: `tools/VapourWiki_EN.md`; every field with its range: `OPERATIONS_EN.md`. `describe` → `machine` tells whether the neural presets are there (`neural`), the NVIDIA RTX card found (`rtx`) and the MLRT installed (`mlrt`).

Typical tasks

- **"Clean the noise of this night shot."** `probe_source` on the file; `shadow_denoise_sota` (noise in the shadows) or `mild_denoise` (uniform); look at `frame --max-width 0` at 100 %; run. The `[NOISE]` line of the journal says what the preset measured and the strength it chose.
- **"Make it look like 35 mm film."** `film_35mm` with `stock` 250D (daylight, neutral), 500T (tungsten, cooler) or 50D (cleanest); `intensity` 0 – 2. Show the user the frames of two or three stocks before running.
- **"This is interlaced."** `qtgmc_deinterlace`; 29.97 with 3:2 pulldown — `tivtc_ivtc`.
- **"Twice the size."** `nnedi3_upscale_2x`; with MLRT — `vsmlrt_realesrgan_2x` or `soft_hd_rebuild_2x` for soft HD.
- **"Make it 60 fps."** Nothing does it now: RIFE is out until it runs right on the video card — say so.
- **"Keep these settings."** `profile_save` with `--set profile_name=NAME --set profile_preset=ID` and the values in `--params FILE` (`{"profile_values": "{\"stock\": \"500T\"}"}`); later `profile_run --set profile=NAME`.
- **"Install what is missing."** `components` and `describe` tell; `plan --operation install_vapoursynth` (then `install_vs_plugins`, `install_ffmpeg`, and `install_mlrt` with an RTX card) — tell the user the files and sizes, and install only with their consent.

Checking the result

- `outcome` is `done` and `processed` above 0; `skipped` counts results that were already there (`--set overwrite=true` makes them again).
- The file is under `<Target>\<Palette>\<preset>_<encoder>\`, the folders and the name of the source kept.
- `Tools\ffmpeg\bin\ffprobe.exe -v error -show_streams -show_format FILE`: size, frame rate and length against the source — twice the size for a 2x preset, the same for the rest.
- For a look: `frame` of the result's preset at the same `--at` is what the user approved.

Rules

- Never overwrite, move or clear the user's files; results go into the Target.
- Ask before an installer (it downloads, MLRT about 3.5 GB) and before `remove_vapoursynth` (it moves the engine to the Recycle Bin).
- A neural preset needs MLRT; MLRT is installed only with an NVIDIA RTX card. Without them, say so rather than looking for another way.
- Prefer a preset and its fields to a script of your own; a field that is not named keeps its default.
- Read the outcome file rather than the journal; read the journal when the outcome is not `done`.