# Audion VS Engine и ИИ-агент

English · Руководство · Консольная команда · Операции · Пресеты

**Содержание**


- Кто что делает
- Консольная команда
  - Какой пресет
- Обычные задачи
- Проверка результата
- Правила


Руководство для ИИ-агента (Claude Code, Codex и подобных), которого пользователь попросил работать с программой. Для человека — `USER_GUIDE_RU.md`, для командной строки — `CLI_RU.md`.

## Кто что делает

Программа работает с картинкой видео пресетами VapourSynth, одинаково при каждом прогоне: убирает шум (сила следует за шумом, который она измеряет), дебандит, возвращает зерно, даёт плёночный или ретро-образ, деинтерлейсит, снимает телесин, чистит блоки сжатия, радугу и дымку, увеличивает, а с видеокартой NVIDIA RTX — нейросетевой шумодав, увеличение и дорисовку кадров. Каждая кнопка её окна — операция консольной команды `vs_cli.cmd`, поэтому агент может делать всё это.

Агент переводит слова пользователя в пресет и его значения, смотрит кадр до и после, запускает, читает итог и проверяет файл. Свои скрипты VapourSynth и команды FFmpeg агент не пишет там, где есть пресет: пресет уже знает форматы, уровни и порядок фильтров.

## Консольная команда

`vs_cli.cmd` лежит в корне программы:

```
vs_cli.cmd describe                                    операции, поля, варианты,
машина (JSON)
vs_cli.cmd run --operation ID --set ИМЯ=ЗНАЧЕНИЕ       запустить операцию; --dry-run
— команды, которые она запустит
vs_cli.cmd run ... --source ФАЙЛ_ИЛИ_ПАПКА --target ПАПКА --result ИТОГ.json
vs_cli.cmd frame --operation ID --source ФАЙЛ --out ПАПКА     кадр из середины до и
после, в PNG
```

Источник — `input\` программы, если `--source` не называет папку или файл; Назначение — `output\`, если `--target` не называет другое. Коды выхода: 0 — готово, 1 — файл не вышел, 2 — неверный вызов или отказ проверок до начала (ничего не запускалось), 130 — остановлено. Файл итога `--result`: `outcome`, `ok`, `message`, `processed`, `skipped`, `failed`, `parameters` — каждое значение, с которым шёл прогон, — `seconds`, `log_file` и `data`. Все ключи — `CLI_RU.md`. `--result` — файл `.json` вне Источника; путь — полный, или команду зовут из папки программы: относительный путь считается от папки того, кто зовёт.

## Какой пресет

| Раздел | Группа | Пресет | Вызов |
|---|---|---|---|
| ТОЧНОСТЬ | Шумодав | Мягкий шумодав | `vs_cli.cmd run --operation mild_denoise` |
| ТОЧНОСТЬ | Шумодав | Шумодав теней SOTA | `vs_cli.cmd run --operation shadow_denoise_sota` |
| ТОЧНОСТЬ | Шумодав | Чистка цветового шума | `vs_cli.cmd run --operation chroma_cleanup` |
| ТОЧНОСТЬ | Дебанд | Мягкий дебанд | `vs_cli.cmd run --operation deband_safe` |
| ТОЧНОСТЬ | Дебанд | Дебанд с тонким зерном | `vs_cli.cmd run --operation deband_fine_grain` |
| ТОЧНОСТЬ | Мастер | Плёночная пересборка | `vs_cli.cmd run --operation filmic_rebuild` |
| ТОЧНОСТЬ | Мастер | Чистый архив | `vs_cli.cmd run --operation archive_clean` |

| Раздел | Группа | Пресет | Вызов |
|---|---|---|---|
| ТОЧНОСТЬ | Мастер | Подготовка к грейдингу | `vs_cli.cmd run --operation pregrade_prep` |
| ПЛЁНКА | Плёнка | Плёнка 35 мм | `vs_cli.cmd run --operation film_35mm` |
| ПЛЁНКА | Плёнка | Плёнка 16 мм | `vs_cli.cmd run --operation film_16mm` |
| ПЛЁНКА | Плёнка | Super 8 | `vs_cli.cmd run --operation super8` |
| ПЛЁНКА | Плёнка | IMAX 70 мм | `vs_cli.cmd run --operation imax_70mm` |
| ПЛЁНКА | Образ | Кинематографичный | `vs_cli.cmd run --operation cinematic` |
| ПЛЁНКА | Образ | Отбеливание | `vs_cli.cmd run --operation bleach_bypass` |
| ПЛЁНКА | Образ | Анаморфот | `vs_cli.cmd run --operation anamorphic_scope` |
| РЕТРО | Видео | VHS и кинескоп | `vs_cli.cmd run --operation vhs_crt` |
| РЕТРО | Видео | Камкордер 90-х | `vs_cli.cmd run --operation camcorder_90s` |
| РЕТРО | Фото | Полароид | `vs_cli.cmd run --operation polaroid` |
| РЕСТАВРАЦИЯ | Чересстрочность | Деинтерлейс QTGMC | `vs_cli.cmd run --operation qtgmc_deinterlace` |
| РЕСТАВРАЦИЯ | Чересстрочность | Обратный телесин TIVTC | `vs_cli.cmd run --operation tivtc_ivtc` |
| РЕСТАВРАЦИЯ | Шум | MVTools MCDegrain | `vs_cli.cmd run --operation mvtools_mcdegrain` |

| Раздел | Группа | Пресет | Вызов |
|---|---|---|---|
| РЕСТАВРАЦИЯ | Шум | Шумодав DPIR [1] | `vs_cli.cmd run --operation dpir_denoise` |
| РЕСТАВРАЦИЯ | Артефакты | Блоки H.264 | `vs_cli.cmd run --operation deblock_h264_artefacts` |
| РЕСТАВРАЦИЯ | Артефакты | Радуга и перекрёстные помехи | `vs_cli.cmd run --operation derainbow_decross` |
| РЕСТАВРАЦИЯ | Артефакты | Дымка и локальный контраст | `vs_cli.cmd run --operation dehaze_local_contrast` |
| РЕСТАВРАЦИЯ | Масштаб | Увеличение NNEDI3 2X | `vs_cli.cmd run --operation nnedi3_upscale_2x` |
| РЕСТАВРАЦИЯ | Масштаб | Real-ESRGAN 2x [1] | `vs_cli.cmd run --operation vsmlrt_realesrgan_2x` |
| РЕСТАВРАЦИЯ | Масштаб | Пересборка мягкого HD 2x [1] | `vs_cli.cmd run --operation soft_hd_rebuild_2x` |
| ПРОФИЛИ | | Запустить профиль | `vs_cli.cmd run --operation profile_run` |

[1] на vs-mlrt: виден при видеокарте NVIDIA RTX или MLRT.

Что делает каждый, на каком материале, с какими параметрами — `tools/VapourWiki_RU.md`; каждое поле с диапазоном — `OPERATIONS_RU.md`. `describe` → `machine` говорит, есть ли нейросетевые пресеты (`neural`), найденную карту NVIDIA RTX (`rtx`) и установленный MLRT (`mlrt`).

## Обычные задачи

- **«Убери шум с ночной съёмки».** `probe_source` по файлу; `shadow_denoise_sota` (шум в тенях) или `mild_denoise` (ровный); посмотреть `frame --max-width 0` в 100 %; запустить. Строка `[NOISE]` в журнале говорит, что пресет измерил и какую силу выбрал.
- **«Сделай как плёнку 35 мм».** `film_35mm` с `stock` 250D (дневная, нейтральная), 500T (лампы накаливания, холоднее) или 50D (самая чистая); `intensity` от 0 до 2. Покажите пользователю кадры двух-трёх плёнок до прогона.

- **«Это чересстрочное».** `qtgmc_deinterlace`; 29,97 с телесином 3:2 — `tivtc_ivtc`.
- **«Вдвое больше».** `nnedi3_upscale_2x`; с MLRT — `vsmlrt_realesrgan_2x` или `soft_hd_rebuild_2x` для мягкого HD.
- **«Сделай 60 кадров».** Нечем: RIFE убран, пока на видеокарте он не считает верно, — так и сказать.
- **«Запомни эти настройки».** `profile_save` с `--set profile_name=ИМЯ --set profile_preset=ID` и значениями в `--params ФАЙЛ` (`{"profile_values": "{\"stock\": \"500T\"}"}`); потом `profile_run --set profile=ИМЯ`.
- **«Поставь, чего не хватает».** Скажут `components` и `describe`; `plan --operation install_vapoursynth` (затем `install_vs_plugins`, `install_ffmpeg`, а при карте RTX — `install_mlrt`) — назовите пользователю файлы и размеры и ставьте только с его согласия.

## Проверка результата

- `outcome` — `done`, `processed` больше 0; `skipped` считает результаты, которые уже были (`--set overwrite=true` сделает их заново).
- Файл — в `<Назначение>\<Палитра>\<пресет>_<кодер>\`, с папками и именем исходника.
- `Tools\ffmpeg\bin\ffprobe.exe -v error -show_streams -show_format ФАЙЛ`: размер, частота и длина против исходника — вдвое больше у пресета 2x, у остальных — те же.
- Для образа: `frame` пресета результата на той же `--at` — это то, что одобрил пользователь.

## Правила

- Не перезаписывать, не переносить и не чистить файлы пользователя; результаты — в Назначение.
- Спрашивать перед установщиком (он скачивает, MLRT — около 3,5 ГБ) и перед `remove_vapoursynth` (он убирает движок в Корзину).
- Нейросетевому пресету нужен MLRT; MLRT ставится только при видеокарте NVIDIA RTX. Без них — сказать об этом, а не искать обход.
- Пресет и его поля лучше своего скрипта; поле, которое не названо, берёт своё умолчание.
- Читать файл итога, а не журнал; журнал — когда итог не `done`.