真实事故 · 修法可抄

AI 视频生产
防坑百科

80 个真实翻车

症状 → 真因 → 修法 → 防再犯

篇幅	8.7 万字
章节	8 章
附赠	技能包 + 模板

不是教程，是「你的报错我们先炸过一遍」。

灵犀造物 Lingxi Creator

一念成片

前言 · 这本书怎么用

这不是一本教你「怎么用」的教程，而是一本「炸了怎么救」的急救手册。每一条都来自真实生产事故，按四段式写：症状 → 真因 → 修法 → 防再犯。

- 遇到问题：按症状查目录，翻到对应章节，照抄修法。
- 想系统学：从第一章读到第七章，是一条完整的生产链。
- 想让 AI 帮你：附赠的技能包直接喂给你的 AI 助手。

目 录

第一章 · 环境与装机（跑不起来）

第二章 · ComfyUI 与工作流（跑起来但不是你要的）

第三章 · 提示词（画面出来了但崩）

第四章 · 声音与口型（人对了但声音不对）

第五章 · 长片与连续性（一段行，连起来就废）

第六章 · AI 智能体运维（上）

第六章 · AI 智能体运维（下）

第七章 · 质检与交付（交出去之前最后一关）

样章 · 说话人没有口型 / 镜头错位 / 脸上斑驳

第一章·环境与装机（跑不起来）

第一章·环境与装机（跑不起来）

导语：环境没装对、工具被劫持、进程杀不掉——这一章收的都是"活还没开始干就卡死"的坑，按症状对号入座，修法照抄即可。

目录

- 1-1 Win+R 敲 pwsh 报 0xC0000142，真因是 PowerShell 7 压根没装
- 1-2 RTX 4090 魔改 48G 跑 ComfyUI 满显存就整机断电死机
- 1-3 llama-server 聊到一半进程无声消失（TDR 两秒杀 GPU 任务）
- 1-4 原生程序不转换 /e/ 路径，文件悄悄落到 <本地路径>
- 1-5 Hugging Face 下模型只有 400KB/s，终端 curl 全超时
- 1-6 curl 查本机服务永远 0 字节，误判服务卡死
- 1-7 Hermes 桌面升级卡死在 get-windows：npm 12 吞镜像变量＋GitHub 包被墙
- 1-8 wsl --install 卡死"正在下载"：WSL2 装 U 盘的四个连环坑
- 1-9 Python/ComfyUI 干完大活攥着内存不还，账面 76% 不降
- 1-10 C 盘无故膨胀 27G：AMD 遥测日志 sdkusage.csv 无限长
- 1-11 taskkill 双斜杠不生效，旧进程偷跑 25 分钟用旧工作流出片
- 1-12 IDM 插件被 ego 屏蔽，报错窗口每次还原多弹一个

1-1 Win+R 敲 pwsh 报 0xC0000142，真因是 PowerShell 7 压根没装

症状

Win+R 敲 pwsh 打不开，报错被转述成 0xC0000142"DLL 初始化失败"。AI
或老手一看就判"PowerShell / 系统组件坏了"，开出退安全软件、重装 .NET、sfc/DISM
三件套——全跑完还是打不开。各家转述的错误码不一致，但"打不开＋像系统损坏"是共同点。

真因

pwsh（PowerShell 7）这台机器从来没装过。ShellExecute
复现拿到的真实报错是"系统找不到指定的文件"，0xC0000142 是转述走样。PowerShell 5.1

本体完好：CreateProcess 四路发射（新控制台/隐藏窗/SysWOW64 32 位/正常窗）全 rc=0，事件日志近 7 天无 0xC0000142 崩溃记录。机器上有金山毒霸（服务 kxescore/kxewsc＋内核驱动全在跑），但 pwsh 装好后带着它照样跑通——安全软件是这类错误的常见嫌疑人，排查该查，但别查到它就停。（2026-11-01 实测定案）

修法

装 PowerShell 7 并当场验收，缺运行库再补：

```
msiexec /i "<下载目录>\PowerShell-7.6.9-win-x64.msi" /qn ADD_PATH=1 REGISTER_MANIFEST=1<br/>pwsh -NoProfile -Command "Write-Output ok" # 三路实测全绿才算装好
```

还要 .NET 就补：`dotnet-sdk-8.0.427-win-x64.exe /install /quiet /norestart`，装完 `dotnet --list-runtimes` 确认 Desktop Runtime 8.0.33 落位。保险兜底再跑 sfc /scannow ＋ DISM /Online /Cleanup-Image /RestoreHealth。注意 winget 走 GitHub 直链必挂（0x80072efd）——浏览器下好 MSI 用本地路径装是正解。装完三处对账：Win+R 能打开、命令行出 ok、注册表 Machine PATH 里有 `<系统盘>\Program Files\PowerShell\7\`。

防再犯

- [ ] 先 `where pwsh`、看 `<系统盘>\Program Files\PowerShell`，确认装没装，再谈坏没坏
- [ ] 不接受转述的错误码：ShellExecute 复现拿真实报错文案再下结论
- [ ] 查国产安全软件要看服务＋内核驱动三层，光按进程名关键词查会漏（kxescore 是服务名，360/火绒/腾讯关键词都搜不到）
- [ ] 装完必须验收：命令实测跑通＋注册表 Machine PATH 有对应条目

关联坑

- 见 1-4：bash 里跑 DISM 长选项会被 MSYS 吃参数，同类转义陷阱
- 见 1-5：终端下载 GitHub 直链必挂的通用解法

1-2 RTX 4090 魔改 48G 跑 ComfyUI 满显存就整机断电死机

症状

跑 H3 视频工作流显存冲到 44.7G/49G，nvlddmkm 崩、整机直接断电（Kernel-Power 41）。2026-09-08 一天死机 8 次，8/31 两次、9/3 一次，恶化趋势明显；不跑活时驱动也报 nvlddmkm 错误（当天 64 条）。三天前（9/5）同样的工作流还能跑通。

真因

nvlddmkm 崩溃＋Kernel-Power 41 硬断电＝驱动/硬件层问题，不是 ComfyUI
或工作流的锅。逐项做减法：U 盘秋叶版 ComfyUI 环境正常（0.34.0＋torch
2.9.1+cu130）、extra_model_paths.yaml 挂盘正常、轻负载 SDXL 1024□ 20 步 12
秒零报错、40G 分块读写校验（20
块×2GB）显存无坏块。轻负载稳＋显存完好➡指向高负载下的供电或驱动兼容性。当时驱动
616.64，魔改卡（24 颗 2GB GDDR6X 焊 3090 PCB＋魔改 VBIOS）社区公认最稳的是 572.83
Studio。已知故障模式：GSP 固件超时、PCIe 降 Gen1、显存坏块（本机已排除）。

修法

- 换 572.83 Studio 驱动（nvidia.cn 下载，清洁安装），重跑原工作流验证；
- 仍崩就满载烤机测供电/散热，准备找商家（魔改卡无官方质保，报修带上卡 UUID
  `GPU-21ce20fe-baaf-60ee-f9fa-c7ca0c97e2a9`——换插槽不变，是唯一可靠标识）。

排查期显存完整性测试可照抄：分块读写校验（脚本
<本地路径>），别一上来就怪工作流，也别轻信"驱动降级就好"（用户自行降级试过，无效）。

防再犯

- [ ] 死机先看事件日志定层：nvlddmkm/Kernel-Power 41＝硬件驱动层，改工作流是白费劲
- [ ] 魔改卡第一时间记录 UUID，换机、换插槽、报修都靠它
- [ ] 驱动用社区公认版（魔改 4090 48G＝572.83 Studio），别追新
- [ ] 排查按"环境➡模型➡工作流➡轻负载➡显存校验"顺序做减法，每步留证据

关联坑

- 见 1-3：同是 GPU 层崩溃，另一种形态是进程无声消失

1-3 llama-server 聊到一半进程无声消失（TDR 两秒杀 GPU 任务）

症状

llama-server 模型加载正常、对话开头正常，聊到一半突然没反应，进程直接消失。GPU
显存才用 22GB/48GB，不是 OOM，日志里干干净净没有任何错误信息。有时是回答吐到一半断掉，
有时下一条提问直接连接失败。没有蓝屏、没有弹窗，一切发生得悄无声息。

真因

Windows WDDM 的 TDR（Timeout Detection and Recovery）默认超时只有 2 秒。27B 模型处理长对话时，单次 GPU 前向传播可能超 2 秒，Windows 判定"GPU 卡死"，自动重置显卡驱动，llama-server 进程跟着被杀，全程无提示。TDR 的本意是防 GPU 挂死拖垮系统：驱动无响应超过阈值就整卡重置，只杀 GPU 任务、不蓝屏，所以事后什么错都不留。显存账也证明不是显存问题：即便 131K 满上下文，27B＋Q4_0 的 KV 缓存才 2.0GB，模型加 KV 共 22GB，48GB 还剩 26GB。（2026-09-16 实测）

修法

管理员 PowerShell 放宽 TDR 超时到 60 秒，改完必须重启生效：

```
# 先查：查不到 TdrDelay/TdrLevel 两个值＝在用默认 2 秒<br/>reg query "HKLM\SYSTEM\CurrentControlSet\Control\GraphicsDrivers" /v TdrDelay<br/>reg add "HKLM\SYSTEM\CurrentControlSet\Control\GraphicsDrivers" /v TdrDelay /t REG_DWORD /d 60 /f<br/>reg add "HKLM\SYSTEM\CurrentControlSet\Control\GraphicsDrivers" /v TdrLevel /t REG_DWORD /d 3 /f
```

重启后重跑原先必崩的长对话，撑过老轮次才算修好。这套改法对所有 GPU 推理服务（llama-server、ComfyUI、本地视觉）同理。

防再犯

- [ ] 推理服务"进程无声消失＋日志干净"，先查 TDR，别怀疑模型文件损坏
- [ ] 改 TDR 前 reg query 留底，改完记下重启时间，验证生效
- [ ] 大模型部署前先算 KV 缓存账排除 OOM，再往驱动层查
- [ ] 换更大模型后重算单次前向耗时，长上下文是 TDR 超时高发区

关联坑

- 见 1-2：GPU 层崩溃的另一种形态——整机断电死机

1-4 原生程序不转换 /e/ 路径，文件悄悄落到 <本地路径>

症状

`git clone https://github.com/xxx /e/target` 显示 "Cloning into '/e/target'..." 退出码 0，但 E 盘找不到文件；再 clone 同一路径报 "destination path already exists and is not an empty directory"，而 `ls /e/target` 又报 "No such file or directory"——两个报错互相矛盾，像文件系统玄学。轻则找不到文件，重则往 C 盘写了几 GB 还以为落盘成功。

真因

bash/MSYS 环境里路径转换对原生 Windows 程序是关闭的：ls、cp、mv、mkdir 等 MSYS 程序正确理解 `/e/xxx`＝E 盘；git.exe、curl.exe、python.exe、schtasks.exe 等原生程序把 `/e/xxx` 当成"当前盘符根目录下的 e 文件夹"，即 `<本地路径>`。根子是历史包袱：MSYS 为兼容 POSIX 把 /e/xxx 翻译成 <本地路径>，但只对链接 MSYS 运行时的程序做翻译，原生 exe 拿到的是 argv 原文。于是 git clone 实际建在 `<本地路径>`；第一次 clone 超时被杀留下的半成品目录，又制造出那组"已存在/不存在"的矛盾报错。（2026-09-25 实测）

修法

给原生程序一律传 Windows 风格路径：

```
git clone https://... "E:/target" # 或 <本地路径>，别写 /e/target<br/>mv /c/e/xxx /e/xxx # 已落错的搬回去（mv 两种写法都认）<br/>rm -rf /c/e # 清残留目录
```

同族写法坑一并记：schtasks 参数用单斜杠 `/create`，写成 `//create` 报"无效参数"；PowerShell 里设变量用 `$env:VAR = "value"`，没有 export，bash 命令别直接贴去 PowerShell 跑。落错的文件多时批量搬回（`for d in /c/e/*; do mv "$d" /e/; done`），搬完 ls 对账再收工。记住：退出码 0 只代表命令没报错，不代表文件落对了地方。

防再犯

- [ ] 路径写成 /x/... 形式前，先确认目标程序是 MSYS 的还是原生 Windows 的
- [ ] clone/下载类命令跑完，到目标盘实际 ls 一眼才算完成
- [ ] 看到"已存在"和"不存在"两个报错打架，第一反应查当前盘符根下有没有同名目录

关联坑

- 见 1-11：taskkill 双斜杠不生效，同一族转义陷阱
- 见 1-8：WSL 安装实录里 curl 犯过同一个路径错

1-5 Hugging Face 下模型只有 400KB/s，终端 curl 全超时

症状

IDM 下 Hugging Face 模型只有 400KB/s，浏览器也就 1MB/s 左右，24GB 模型要下一整天；更气人的是浏览器能打开 HF，终端里 hf download、curl 全部超时。模型越大越绝望。

真因

三层叠加：HF 服务器在国外直连慢；IDM 不支持 HF 的 Token
认证，用户名密码也没用；盘古加速器这类 VPN 只对浏览器生效（系统代理
ProxyEnable=0），终端一律直连。还有一个隐藏杀手：`HF_HUB_OFFLINE=1`
被某脚本设置后，huggingface_hub 直接进离线模式，所有 HTTP
请求静默失败——连镜像一起挂，最难查。（2026-08-28
起多轮实测）浏览器快是因为代理只认浏览器流量；终端工具各自为政，要么逐条加
--proxy，要么统一吃 HTTP_PROXY 环境变量。

修法

按速度选通道（2026-09-15 实测）：curl 直链＋代理约 60MB/s（16GB 约 4
分钟）；ModelScope 免代理 47MB/s；hf CLI＋hf-mirror 约 20MB/s 自动续传。

```
export HTTP_PROXY=http://localhost:51081 HTTPS_PROXY=http://localhost:51081 # 代理端口看 PAC 文件<br/>curl -L -C - --proxy http://localhost:51081 \<br/> -o "E:/model/模型名/短文件名.gguf" \<br/> "https://hf-mirror.com/用户名/仓库名/resolve/main/完整文件名.gguf"
```

无 VPN 时用 Token 直连（Token 从 huggingface.co/settings/tokens 取，写
[REDACTED]）：`curl -L -C - -H "Authorization: Bearer <token>" ...`，约
15MB/s。ModelScope 备选：`pip install modelscope` 后用 snapshot_download 按
allow_file_pattern 只拉指定文件，别整个仓库下一遍。卡住先诊断：`echo
$HF_HUB_OFFLINE`，是 1 就 `unset HF_HUB_OFFLINE`，永久清掉用 `reg delete
"HKCU\Environment" /v HF_HUB_OFFLINE /f`。

防再犯

- [ ] 开下前先 curl -sI 验通道：代理通、直连通各测一次
- [ ] -o 的本地文件名要短：Windows 全路径超 260 字符直接写入失败
- [ ] 大安装器先 --plan 干跑、--reuse-models 复用已有模型，失败同命令重跑即续传
- [ ] 代理变量用完 unset，别污染后续命令（见 1-6）
- [ ] 优先国内镜像直连，代理只用来过 GitHub 这道关

关联坑

- 见 1-6：HTTP_PROXY 劫持本机地址的连环误判
- 见 1-7：GitHub 预编译包被墙的另一套绕法

1-6 curl 查本机服务永远 0 字节，误判服务卡死

症状

ComfyUI :8188 日志明明写着 "To see the GUI" 已就绪，`curl http://localhost:8188/` 却一直 0 字节/超时（HTTP 000）；netstat 显示 LISTENING＋ESTABLISHED 齐全，进程 CPU 正常。极易误判服务僵死、端口被占、插件卡启动，甚至误杀好进程反复重启。8765、7863 这些本地端口中招时症状一模一样。误判之后的反复重启、反复重装更是雪上加霜。

真因

之前为下载 GitHub 而 `export HTTP_PROXY=http://localhost:51081`，export 跨命令持续生效；curl 默认把 localhost/localhost 的请求也送进代理（没设 NO_PROXY 时），代理出不去回环地址，连接挂死 0 字节。export 的变量活到会话结束，中间隔多少条命令都还在；curl 在 no_proxy 为空时对 localhost 一视同仁走代理。服务本身一直是好的。2026-10-23 FreeVideo 升级 v0.4.19 实锤：`--noproxy "*"` 后 HTTP 200＋版本接口全通，期间误杀过一次健康进程。

修法

查本机服务三选一：

```
curl --noproxy "*" http://localhost:8188/ # 首选<br/>unset HTTP_PROXY HTTPS_PROXY http_proxy https_proxy # 或当场清变量（unset 同样跨命令持久）<br/>curl --proxy http://localhost:51081 ... # 代理只给单条下载命令临时用，别 export
```

判别口诀：连得上（ESTABLISHED）＋0 字节＝先查代理再查服务；`curl -v` 里出现 `HTTP/1.1 200 Connection established`（代理 CONNECT 应答）＝已走代理；再用浏览器或 PowerShell Invoke-WebRequest 开第二通道复核，两条通道结果矛盾＝代理问题。

防再犯

- [ ] 代理只在单条命令上用 --proxy，不用 export
- [ ] 判"服务死了"之前必须第二通道验证；杀进程前再验一次
- [ ] 收工前 unset 全部代理变量，防下一条命令被污染
- [ ] 长会话排查网络问题第一步 `env | grep -i proxy`，看有没有残留变量

关联坑

- 见 1-5：代理变量的源头是下载环节
- 见 1-11：杀进程前验证真死的完整流程

1-7 Hermes 桌面升级卡死在 get-windows：npm 12 吞镜像变量 + GitHub 包被墙

症状

`hermes update` 只更了引擎源码，桌面 exe 不重建（还是旧包）；手动 `hermes desktop --build-only --force-build` 卡死在 get-windows 原生模块：node-pre-gyp 下载预编译包失败，回落源码编译又报 "Could not find any Visual Studio installation"——本机根本没装 VS。排查时看日志像网络问题，实际是链路劫持和包管理器两层作祟，少查一层都修不好。

真因

两层。① GitHub releases 在终端被劫持：curl 拿 302 跳到 release-assets.githubusercontent.com 后返回 200 空包（0 字节），git-bash、PowerShell、node 全中招，gh-proxy 系镜像全挂。② npm 12 会规范化配置键名：node-pre-gyp 认的镜像键是混合形式 `npm_config_node-get-windows_binary_host_mirror`（它 replace('-','_') 只替换第一个横杠），npm 12 把 env/npmrc 键统一改成全横杠再回传，node-pre-gyp 永远收不到镜像地址；npm 11 只警告，npm 12 直接改写（"Unknown env config"警告即死因），.npmrc 写同名键也一样被改。换句话说：镜像地址没错，是 npm 把钥匙改了形状，锁孔对不上。（2026-10-27 实测）

修法

用浏览器拉预编译包（浏览器网络栈不受劫持）：`https://github.com/sindresorhus/get-windows/releases/download/v9.3.0/napi-9-win32-unknown-x64.tar.gz`（76KB），按 `v9.3.0/napi-9-win32-unknown-x64.tar.gz` 相对路径放好，`python -m http.server 8765 --bind localhost` 起本地镜像，绕开 npm 直接调 node（env 不被改写）：

```
cd hermes-agent/node_modules/get-windows<br/>env
'npm_config_node-get-windows_binary_host_mirror=http://localhost:8765/' node
../@mapbox/node-pre-gyp/bin/node-pre-gyp install
```

再 `--ignore-scripts` 跑一遍 node-deps 写收据，手工补 postinstall 模块（node-pty 自带 prebuilds/，跑 `node scripts/prebuild.js` 纯本地拷贝）；带安装脚本的包就这几个，esbuild 随包二进制不用管，lefthook 等打包无关的可跳过。最后 `ELECTRON_MIRROR=https://npmmirror.com/mirrors/electron/ hermes desktop --build-only --force-build` 重建。验证：`release\win-unpacked\Hermes.exe` 的 LastWriteTime＝构建当天即成功。

防再犯

- [ ] 构建命令 `> log 2>&1` 落盘，别接 `| tail`——管道会吞真实退出码
- [ ] 下载到 0 字节先怀疑链路劫持，换浏览器/换镜像源，别反复重试

- [ ] npm 12 环境别赌 node-pre-gyp 镜像键，绕开 npm 直接 node 调用

关联坑

- 见 1-5：终端下载通道的通用解法（代理/镜像/浏览器三路分工）

1-8 wsl --install 卡死"正在下载"：WSL2 装 U 盘的四个连环坑

症状

想把 WSL2＋Ubuntu 全装 U 盘（U:），`wsl --install` 报 WININET_E_TIMEOUT，一直"正在下载"没有任何进度。照网上常规教程走，卡在第一步，进度条永远是零。四个坑环环相扣，躲开第一个还会撞上下一个。

真因

wsl --install 要连 raw.githubusercontent.com
拉发行版清单，该域名在终端网络里被劫持，WinINet
走系统网络栈必卡。官方默认路线全程依赖被劫持域名：清单在
raw.githubusercontent.com，包在微软 CDN 与 GitHub
之间跳转，任何一环被拦都表现为"正在下载"。连带还有三个坑：curl 是原生程序不吃 /u/
路径（`curl -o /u/wsl_src/x` 报 exit 23 write error）；WSL
里网络是另一片天——github.com 直连是通的（劫持只在 Windows 栈），但 Windows 的
localhost:51081 代理在 NAT 模式 WSL 里不可达；rm 删不动是自己内核占着句柄（ZipFile 没
close），别怪杀毒软件。（2026-10-28 实录）

修法

不走 wsl --install 装发行版，直接拿 AppxBundle 解出 rootfs 导入：

```
# 1) DISM 启用组件后重启（PowerShell 管理员）<br/>dism /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart<br/>dism /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart<br/># 2) winget install Microsoft.WSL<br/># 3) 浏览器/IDM 下 AppxBundle（24.04 直链）：<br/># https://publicwsldistros.blob.core.windows.net/wsldistrostorage/Ubuntu2404-240425.AppxBundle<br/># 4) AppxBundle＝zip：解一层拿 Ubuntu_*_x64.appx（还是 zip），再解出 install.tar.gz＝rootfs<br/>wsl --import Ubuntu <导出盘>\WSLUbuntu <install.tar.gz> --version 2<br/>wsl -d Ubuntu -u root
```

路径一律 Windows 写法（U:/...）；clone/下载优先放 WSL 里做，pypi 走 tuna
镜像正常。装完整套落位：vhdx 在 `<导出盘>\WSLUbuntu\`、程序在 /root，随 U
盘走；换机器只需 DISM 开两个功能＋一条 wsl --import。

防再犯

- [ ] 装发行版不赌 wsl --install，直接 AppxBundle+--import 走离线路
- [ ] 给原生程序的路径只写 U:/...、C:/...，连 /dev/null 也别递过去
- [ ] 文件删不掉先想自己进程占句柄，close() 后再删
- [ ] cloud-images.ubuntu.com/wsl/ 只剩 manifest 没有 rootfs，别在那空找
- [ ] 安装产物落哪个盘，事后 ls+`wsl -l -v` 对账，别只信命令回显
- [ ] WSL 里外的网络分开看，别拿 Windows 的结论套 WSL

关联坑

- 见 1-4：原生程序不认 MSYS 路径的完整机理

1-9 Python/ComfyUI 干完大活攥着内存不还，账面 76% 不降

症状

电脑内存 76% 甚至更高，感觉"应该没跑什么才对"；查任务管理器，单个 python.exe 工作集 70GB+——2026-10-29 实测 ComfyUI main.py 一个进程攥了
72GB。重启机器当然能好，可下一次干完大活又涨回去。

真因

不是内存泄漏，是 CPython 通病：干完大活（H3 视频渲染、大工作流）后 free 掉的堆内存不还给
Windows，工作集虚高。空闲页其实随时可用，只是系统账面显示"已用"。CPython 的分配器拿到堆后倾向持有不放，空闲页留在进程工作集里；EmptyWorkingSet 让系统把这些页强行换出到待命列表，随时可取回，所以对业务无损。这时杀进程重启 ComfyUI 是错法——正在渲染的活当场报废，别的智能体提交的任务也一起死。

修法

不杀进程、不打断渲染，对目标 PID 跑 EmptyWorkingSet 强制吐页：

```
Add-Type -TypeDefinition @"<br/>using System;using System.Runtime.InteropServices;<br/>public class WS {<br/> [DllImport("kernel32.dll")] public static extern IntPtr OpenProcess(int a,bool b,int pid);<br/> [DllImport("kernel32.dll")] public static extern bool CloseHandle(IntPtr h);<br/> [DllImport("psapi.dll")] public static extern bool EmptyWorkingSet(IntPtr h);<br/>}<br/>"@<br/>$h = [WS]::OpenProcess(0x500, $false, <PID>) # PROCESS_SET_QUOTA |
PROCESS_QUERY_INFORMATION<br/>[WS]::EmptyWorkingSet($h) | Out-Null<br/>[WS]::CloseHandle($h) |
```

```
Out-Null
```

存成 trim_ws.ps1，`powershell -ExecutionPolicy Bypass -File trim_ws.ps1`
执行。实测战果：工作集 72GB→0.4GB，系统空闲 33GB→95.5GB，一口气放
62.5GB，正在渲染的任务照常跑完。找 PID：`Get-Process | Sort-Object WorkingSet64
-Descending | Select-Object -First 10`。OpenProcess 权限 0x500
是"设置配额＋查询信息"两个权限的组合，拿不到句柄先查权限。脚本无副作用，可反复执行。

防再犯

- [ ] 用户说"清内存"→ 先看是不是 Python 工作集虚高，是就 EmptyWorkingSet 吐页，别杀进程
- [ ] 渲染中途永远不重启 ComfyUI，队列里可能有别人的任务
- [ ] 吐页后复查工作集降了多少，确认真的生效
- [ ] trim_ws.ps1 常备一份存好，别等内存爆了现写

关联坑

- 见 1-10：另一个"资源被无名进程默默吃掉"的坑（C 盘）

1-10 C 盘无故膨胀 27G：AMD 遥测日志 sdkusage.csv 无限长

症状

C 盘无故变大，第一反应是"模型下载缓存没清"，实际真凶是
`<本地路径><用户>\AppData\Local\AMD\PPC\sdkusage.csv`——单个 CSV 能长到
27G+，而且还在实时增长。清回收站、清 pip 缓存都清不回来，空间还是每天悄悄少一截。

真因

这是 AMD ADL API 调用的遥测流水：每次驱动查询追加一行（GUID,ADL,进程名,ADL2_xxx），无上限、无轮转。判定方法：stat 看 mtime 是刚刚＝还在写；head
看内容是流水行。纯遥测垃圾，与模型、驱动功能都无关，删了安全，AMD 服务会自动重建（然后重新慢慢长）。追加写是驱动遥测的通病：只管记满数据，不做大小上限和轮转，用户的盘成了它的日志仓库。（2026-11-01 实测）

修法

直接删，同目录小件一并清：

```
rm -f <本地路径><用户名>\AppData\Local\AMD\PPC\sdkusage.csv # 27G<br/># 同目录 apprecord.csv / driverworkloadstats.csv 同族一并删；删后 AMD 目录 27G→38M
```

找大户：`du -sh <目录>/* | sort -rh | head`；找近期新增大文件：`find <目录> -type f -size +100M -mtime -7`。注意别误删在用的东西：hermes\state.db＋sessions\（聊天记录本体）、.cache\huggingface\hub 下 whisper 大模型（口型闸在用，5.8G 是模型不是缓存）都勿动。删完立刻核对可用空间，27G 秒回。

防再犯

- [ ] C 盘无故变大先查 AppData\Local 下各家 SDK/驱动的遥测目录，再怀疑缓存
- [ ] 删任何"缓存"前先 stat＋head 确认内容，别把在用模型当缓存清
- [ ] sdkusage.csv 会复发：几周后 C 盘又大，先查它；要根治加定期清理计划任务或限写入权限
- [ ] 大文件判定用 mtime＋内容双重确认，别只看文件名猜用途
- [ ] 清完过一周再看一眼，确认没有复发

关联坑

- 见 1-9：内存侧的同类"资源被默默吃掉"问题

1-11 taskkill 双斜杠不生效，旧进程偷跑 25 分钟用旧工作流出片

症状

暂停跑片时执行 `taskkill //PID $PID //F //T`，报"错误: 无效参数/选项 - '//PID'"。当时队列显示"运行中 0"就当杀成功了。结果旧跑片器一直活着：下午自行推进到 [6/60]，用 v4 旧工作流提交生成，25 分钟后才被发现，差点污染 60 片批次。

真因

这台机器 bash 环境的 MSYS 路径转换被禁用。`//PID` 本意是让 MSYS 把双斜杠还原成单斜杠 `/PID` 传给 taskkill；转换关掉后 `//PID` 原样传给 taskkill.exe，Windows 原生程序不认，报无效参数——taskkill 其实一条都没执行。（2026-09-29 事故复盘，事故等级：中）

修法

bash 里调 taskkill 用单斜杠，杀完必须验证真死：

```
Get-CimInstance Win32_Process -Filter "Name='python.exe'" |<br/> Where-Object { $_.CommandLine -match '顺序跑片' } |<br/> ForEach-Object { taskkill /PID $_.ProcessId /F /T }
```

ps1 跑完再查一遍：还能列出进程＝没杀干净。跑片器有单实例守卫（跑片器.pid），旧进程活着时新进程会默默退出——重启后必须看日志出现 `□ 跑片开始`＋`□ [1/60] …_v5_…` 才算接手成功。进程树用 `/T` 连子进程一起杀，杀完 `rm 跑片器.pid` 防守卫误判。误跑产物一律进废片隔离区，绝不混入成片。

防再犯

- [ ] 杀进程命令先小范围试跑确认参数被接受，别看队列清零就当成功
- [ ] 杀完必须验证真死：按命令行认人复查进程列表
- [ ] 重启替代进程后看日志确认新实例真的起来了
- [ ] 原生程序参数一律单斜杠（双斜杠是 MSYS 转义写法，转换关了就是毒药）

关联坑

- 见 1-4：同一族 MSYS 转义/路径陷阱
- 见 1-6：误判服务死亡前先第二通道验证

1-12 IDM 插件被 ego 屏蔽，报错窗口每次还原多弹一个

症状

ego lite 每次窗口最小化再还原，就多弹一个错误窗口：`chrome-extension://ngpampappnmepgilojfohadhhmbhlaek/welcome.html?auth=[REDACTED]&current=6.43.1&error=1&incognito=1`，页面显示"已被屏蔽 / 此页面已被 ego 屏蔽 / ERR_BLOCKED_BY_CLIENT"，循环不止。弹窗越攒越多，一次还原叠好几个，关不完。

真因

两层。那个 32 位扩展 ID＝IDM Integration Module（IDM 下载器的浏览器插件，版本 6.43.x 与 IDM 对得上），IDM 会往它发现的每个 Chromium 系浏览器自动塞插件：先经 `HKLM\SOFTWARE\Clients\StartMenuInternet\ego lite` 发现 ego 是浏览器（ego 自我注册过），再用外部安装注册表键把 `<本地路径>` 注册给所有 Chrome 系浏览器。ego 的安全策略屏蔽该插件页，IDM 判定"插件没工作"就反复打开自己的 welcome 页（error=1）。ego 的 Preferences 里 external_uninstalls 已含该 ID（卸载过），但注册表注册还在，循环不止。本质是 IDM 自动投放与 ego 安全策略顶牛：一边认定装上就该生效，一边认定这个扩展不许跑，报错全砸在用户头上。（2026-10-25 实测）

修法

先备份 4 个 .reg（StartMenuInternet_ego_lite / HKLM_ChromeExt / HKLM_WOW64_ChromeExt / HKCU_ChromeExt），然后：

```
powershell Get-Process ego | Stop-Process -Force<br/># 删 4 个注册表键：<br/>#
HKLM\SOFTWARE\Clients\StartMenuInternet\ego_lite<br/>#
HKLM\SOFTWARE\Google\Chrome\Extensions\ngpampappnmepgilojfohadhhmbhlaek<br/>#
HKLM\SOFTWARE\WOW6432Node\Google\Chrome\Extensions\ngpampappnmepgilojfohadhhmbhlaek<br/>#
HKCU\SOFTWARE\Google\Chrome\Extensions\ngpampappnmepgilojfohadhhmbhlaek
```

重启 ego，实测键不复活（ego 不重建 StartMenuInternet 注册），最小化还原循环不再弹窗。四个键是四条独立的发现/投放通道，少删一条都会复发。回滚：双击导入备份的 4 个 .reg 即可复原。

防再犯

- [ ] 报错扩展 ID 先搜 32 位 ID 查身份，再谈修谁
- [ ] "ERR_BLOCKED_BY_CLIENT＋已被 XX 屏蔽"＝客户端策略拦截不是网络问题，去查谁在反复投这个 URL
- [ ] 删外部安装注册表键前，先确认目标浏览器 profile 里没有活配置
- [ ] 动注册表前必先导出 .reg 备份，留好回滚路

关联坑

- 见 1-5：IDM 下 Hugging Face 模型慢的坑（IDM 不支持 HF 的 Token 认证，也是 IDM 自身行为埋的雷）

第二章 · ComfyUI 与工作流（跑起来但不是你要的）

第二章 · ComfyUI 与工作流（跑起来但不是你想要的）

导语：这一章的坑有个共同点——ComfyUI 不报错、照常出片，但出的不是你要的东西：格式双轨、存储多份、控件错位、素材放错层，全靠事后验尸才揪得出来。逐条对照，能替你省几个通宵。

目录

- 2-1 改了提示词等于没改：节点内容存了三份
- 2-2 py 能跑、浏览器连线断：API 格式和 UI 格式是两套文件
- 2-3 step3 提交 H3 必 500：UI 格式工作流喂了 /prompt API
- 2-4 steps 超范围报错：seed 伴生槽让整排控件错位一格
- 2-5 界面跑起英文示例：widgets_values 错位整节点退回默认值
- 2-6 怪病查不出先怪底版：装配必须用实测成功的模板
- 2-7 人物全变不跟参考图：LoadImage 主图节点被当死节点剪掉
- 2-8 分段图不加载：内层素材库少一层 input 就是废层
- 2-9 报 "outside a model execution"：wrapper 链排序坑
- 2-10 出片名字串台、宽景碎裂斑驳：位置数组才是真身
- 2-11 视频生成时间暴涨：先查分辨率，不是提示词
- 2-12 重跑同名输出必炸：sidecar 不清光就 FileExistsError
- 2-13 UI 只显示 5 段旧提示词：timeline 三源必须同步

2-1 改了提示词等于没改：节点内容存了三份

症状

改完提示词自检「通过」，用户点开 ComfyUI 主节点下一层面板，看到的还是老格式；参考图也漏改过一次。以为改完就开跑，实际跑的是老内容，白跑返工。

真因

MiniMaxH3MotionDirector
这类自定义节点把同一份数据存了好几处：`node.widgets_values[i]`（数组，timeline_data 是其中一项的嵌套 JSON 字符串，段 prompt
藏在里面）、`node.widgets_values_named.<name>`（命名副本，节点 UI「下一层」面板读的就是它，漏改用户就看到老内容）、`node.properties.mmx_director_widget_state`（节点状态缓存），加上 timeline_data 字符串内部的嵌套 JSON。副本之间没有任何同步机制，谁被读到全看调用方——只改任何一处都等于没改。2026-09-28
碳硅问道装配器第三次踩同一坑：只改了前两处、漏了 properties 副本，靠脚本断言拦下 60 片，零损失。

修法

用通用修复器递归遍历 JSON 的一切字符串值，字符串本身是 JSON（timeline_data 这类）就解析进去继续改，再原样序列化回去，任何副本都逃不掉。脚本 `patches\fix_all_formats.py` 可改造成任意内容的全库替换器。改完对旧内容特征串做全库 grep/递归扫描（排除备份目录），残留为 0 才算完成。浏览器陷阱：ComfyUI 里开着的标签页持有旧内存副本，可能自动保存把老内容写回文件——改完必须让用户 F5 刷新或重开标签，验证时确认文件未被回写。

防再犯

- [ ] widgets_values、widgets_values_named、properties 三处副本都改了吗
- [ ] 嵌套 JSON（timeline_data 等）内部也改了吗
- [ ] 全库 grep 旧特征串 = 0
  残留，且写成装配脚本的阻断式断言（fail-closed），不靠人眼
- [ ] 批量改动前整目录备份；脚本按「目录+文件名」定位目标，防同名串改

关联坑

- 2-13 timeline 三源同步是同族问题；2-10 位置数组与命名镜像之争也是它。

2-2 py 能跑、浏览器连线断：API 格式和 UI 格式是两套文件

症状

同一个 ComfyUI 工作流，Python/API
执行完全正常；浏览器界面打开却部分连线呈断开状态，Queue 提交后后端报某输入为 None（案例：`MiniMaxH3MotionDirector: segment_audios is None`）。反复装插件、重连线、拷文件都治不好。

真因

ComfyUI 工作流 JSON 有两套互不兼容的格式。API 格式：顶层键是数字字符串节点 ID，值为 `{class_type, inputs}`，连线内嵌在 inputs 里（`输入名: [源节点, 槽位]`），供 `POST /prompt` 和 py 自动化用。UI 格式：有 `nodes`/`links` 数组（version 0.4），供浏览器画布用。API 格式文件放进 `ComfyUI\user\default\workflows\` 或直接喂浏览器，按 UI 格式解析、links 元数据根本不存在，线就全丢或半丢；py 按名字传值不受影响——「py 通、浏览器断」由此造成。**不是缺插件。**判别方法：JSON 第一层键是数字字符串就是 API 格式，有 `"nodes": [...]` 就是 UI 格式。

修法

把 API 格式转换成 UI 格式再给浏览器。现成工具：Hermes 技能 `comfyui-api-to-ui`（scripts/api2ui.py），以同图 UI 母版为模板 1:1 转换，自带「重建 prompt 与原 API 图逐字段比对」的等效性校验，非 0 错误拒绝落盘。转换后浏览器 F5 刷新，再从侧栏打开新文件。备选：把 API JSON 直接拖到画布（前端自带转换）再另存为 UI 格式。转换要点：widgets_values 顺序 = widgets_values_named 键序；seed widget 后必有 control_after_generate，转换时设 `fixed` 才与 py 执行行为一致，否则浏览器每跑一次随机一次种子；JS 自定义 widget（upload、audioUI 等）不进 prompt，校验白名单放行。

防再犯

- [ ] 自动化产出的 `_pilot_*.json`、`*_fixed.json` 等 API 格式文件，绝不直接放进浏览器 workflows 目录
- [ ] 给浏览器用的工作流先看第一层键，数字键必先转换
- [ ] 转换后跑等效性校验，0 错误才落盘

关联坑

- 2-3 是反方向的同坑——UI 格式喂 API；2-4、2-5 是转换与手拼时的控件错位。

2-3 step3 提交 H3 必 500：UI 格式工作流喂了 /prompt API

症状

金来视频自动化系统 `脚本/run_all.py` 的步骤3（提交 H3 生成）每次全部失败：`[ERROR] 提交失败: HTTP Error 500: Internal Server Error`。2026-09-29 的日志里就有 1572 条同样的 500，成功的 prompt_id 为 0——该步骤从未成功过。`生成视频/视频XXX/`

文件夹全是空的，「已生成视频:
22」只是数文件夹，不是真视频。坏流程长期无人发现，就是因为进度数字骗人。

真因

`脚本/step3_提交H3.py` 的 `prepare_workflow()` 直接加载
`F:/金来提示词高级智能系统/H3-60秒防死机最优工作流.json`——这是 UI 导出格式（顶层 `nodes`/`links`/`last_node_id`），改完 `widgets_values_named` 后整个 JSON 原样 POST 到 `http://localhost:8188/prompt`。但 `/prompt` API 只接受 API 格式（`{节点ID: {class_type, inputs}}`），UI 格式直接提交就是 500——服务端解析不了这种结构，500 只是未捕获异常的通用外衣，看错误文本没用，要看提交物的第一层结构。2026-10-19
复现实锤：把原始模板不做任何修改直接 POST 到 `/prompt`，得到一模一样的 500；同一时刻 `/system_stats` 返回 200——ComfyUI 活着，就是格式不对。

修法

二选一，推荐第一种：

- 在 ComfyUI 界面打开该工作流，用 Dev 模式 `Save (API Format)` 导出 `H3-60秒防死机最优工作流_api.json`，把 `WORKFLOW_TEMPLATE` 指向它，并把 `prepare_workflow()` 里改 `nodes[].widgets_values_named` 的逻辑改成改 API 格式对应节点的 `inputs["prompt" / "seconds" / "segment_seconds"]`。
- 写 UI→API 转换器（要把 widgets_values 按节点定义映射回 input 名字，含自定义节点 MiniMaxH3EasyContextSegments，工作量大易错，不推荐）。

防再犯

- [ ] 「已生成视频: N」这类进度数字必须抽查产物文件（大小>0 的 mp4），空文件夹不作数
- [ ] 提交 API 前确认工作流 JSON 第一层是数字键（API 格式）
- [ ] 跑批前先起本地视觉服务（`一键启动Molmo反推服务.bat`），防反推降级纯文本烧 token

关联坑

- 2-2 同一格式双轨的反向案例。

2-4 steps 超范围报错：seed 伴生槽让整排控件错位一格

症状

H3-Multishot UI 件打开报「部分输入值超出允许范围 - H3 Multishot Sampler - steps」，但 steps=8 明明在声明范围 1~50 内，改 steps 数值也没用，查哪都查不出哪超了。

真因

ComfyUI 前端把控件值按位置存进 `widgets_values` 数组，恢复时按位置对号入座，任何「计划外」的控件都会让后面全体平移。seed 控件在 `seed`/`noise_seed` 槽后会自动附加一个伴生下拉（fixed/randomize/increment/decrement），序列化进 `widgets_values`（作者 CORE 件 idx5=seed、idx6='randomize'）。API 格式里没有这个字段，`api转ui工作流.py` 按 INPUT_TYPES 顺序组装位置数组时漏算它——seed 之后全体错位一格，steps 槽装进了后面的布尔值，于是「超范围」。这是 2026-10-06 实锤的转换器系统性偏移，不是数据错。

修法

- 位置数组在 seed 后手工插入 ``'fixed'``（要复现同种子必须 fixed，否则前端每跑换一次种子）。
- `api转ui工作流.py` 已打 SEED_NAMES 规则补丁（三处：连线 widget 位/值位/widgets_values_named），此后自动补 `control_after_generate`（也可用 API 输入显式指定，缺省 randomize）。

姊妹坑：COMFY_DYNAMICCOMBO_V3（SaveVideo 的 format/codec）不是标准 COMBO，转换器会当非 widget 丢值——补 widget 时按输入顺序写 `['filename_prefix','format','codec']`，format 必须用 ``'auto'``，写 ``'video/h264-mp4'`` 会炸 execute 缺参。

防再犯

- [ ] API→UI 转换后必对照「作者原版 UI 件」的 widgets_values 数量与顺序核一遍再交付
- [ ] widgets_values 数量 ≠ object_info widgetable 数时，先想伴生槽（seed、COMFY_DYNAMICCOMBO_V3）
- [ ] 要复现同种子的场合，control_after_generate 一律设 fixed

关联坑

- 2-5 控件错位的另一形态；2-2、2-3 同属格式双轨家族。

2-5 界面跑起英文示例：widgets_values 错位整节点退回默认值

症状

手写/拼装的 UI 格式工作流在 ComfyUI 打开，脚本控件显示插件英文示例文本（"A static camera sits at the end of the table…"），不是文件里写的中文脚本。点运行的后果：画面静止、念英文、时长缩水。查 JSON 明明 widgets_values[0] 是对的，界面就是不认。

真因

ComfyUI 按**索引前缀**恢复 widgets_values：数量对不齐或顺序变化，只要第一处对不上，整个节点恢复就失败或错位，落到类默认值——这就是磁盘对、界面不认的原因。新版 ComfyUI-H3-Multishot v2.6.1 的 H3MultishotSampler 实际 14 个控件（script, shot_count, width, height, frames_per_shot, seed, control_after_generate, steps, seed_per_shot, sampler_name, scheduler, self_anchor_voice, output_scale, save_every_shot），旧文件只存了 1~8 个值。其中 control_after_generate 是 ComfyUI 自动插入的控件，手拼 JSON 极易漏。「静止+英文+时长缩水」三连怪就是同一个根因（2026-10-14）。

修法

首选在 ComfyUI 画布里改：按控件名设值，走官方保存通道写回——`POST /userdata/ + encodeURIComponent('workflows/文件名.json')`（整路径一起编码，分开编码会 405/404），序列化用 `app.graph.serialize()`，格式天然正确。手写 JSON 后必须校验：widgets_values 长度 == 实际控件数且顺序一致，控件名/顺序以浏览器实测为准：

```
app.graph._nodes.find(x=>x.id==N).widgets.map(w=>w.name)
```

改完在界面重载一次读回控件值确认——磁盘对 ≠ 界面认。注意 `/prompt` API 按控件名传参不受此坑影响，但用户在 UI 点运行就中招，两边都要过。

防再犯

- [ ] 手拼 UI JSON 后校验每个节点 widgets_values 长度与控件数一致
- [ ] 控件清单以浏览器实测为准，警惕自动插入控件
- [ ] 改完界面重载验证一轮再交付

关联坑

- 2-4 种子伴生槽同族；2-10 位置数组才是真身。

2-6 怪病查不出先怪底版：装配必须用实测成功的模板

症状

自建/改造的工作流反复出怪病：语音杂音、台词串到别的片子、蓝眼、画面漂移。逐处查提示词查不出问题，越查越迷。

真因

2026-10-01 碳硅问道·梦境串台 v8 体检战例，三个病根全是「底版不干净/越界改动」。一，CR Text 多份存储打架：`widgets_values[0]` 是本片台词，`widgets_values_named['text']` 却是另一部旧片台词，渲染读错份，台词声音直接错乱。二，TTS 成功配方被改坏：模板成功配方 `speed=slightly slower` 被改成 `normal`，语速全变。三，多份存储漏改（全局提示词三份、timeline_data 三份含 `properties.mmx_director_widget_state`），只改两份，第三份留旧值，就是「内层图与提示词经常遗忘加载」的机关。共同规律：怪病查不出提示词问题时，问题多半在底版和存储副本上。

修法

装配四条铁律照办：1. **底版只选实测成功的模板**（用户实测通过的那版 JSON），不自己重建、不用未实测版本当底。2. **只改提示词类字段**：CR Text 台词、timeline 段 prompt、global_prompt、genImage 图名；参数（steps/speed/emotion/采样）、连线、节点顺序一律不动。3. **多份存储全改**：改任何提示词＝同时写 `widgets_values[i]`＋`widgets_values_named`＋`mmx_director_widget_state`＋嵌套 JSON 内部，改完逐处验尸一致。4. **交付前必跑** `配方体检.py`（模板库\碳硅问道_自由发言路由版\），绿灯才许交。

防再犯

- [ ] 新模板先实测成功才算「好模板」入档复用
- [ ] 改提示词不碰参数/连线/节点顺序
- [ ] 交付前配方体检绿灯
- [ ] 成功一次永久存档，后续自动复用，不再重造

关联坑

- 2-1 多份存储机制；2-7 只改提示词不重建工作流。

2-7 人物全变不跟参考图：LoadImage 主图节点被当死节点剪掉

症状

新版测试片生成内容完全不按参考图走、人物全变，同日旧片全部正常。查提示词、查素材都对，就是图不生效。

真因

两条都已实锤（2026-10-01）。一，LoadImage（加载图像）节点无连线但是 UI 主图锚——可达性剪枝只认连线可达的节点，它没有出边，就被装配器当死节点清走了。同日成功的片（老版样式/四集成片）都有它，唯独重建的测试片没有。二，插件源码 gen_timeline.py
里还有硬兜底：genImage 空时偷摸扫 input 目录拿「新参考图_老者美女.png」或第一张图顶上——人物正好全变。剪掉一个「没用的」节点，反而触发了最坏的静默兜底。重建工作流必引入新自由度（丢节点/丢存储/丢路由），每多一步自由度就多一个「这版好那版坏」的坑。

修法

剪枝豁免一行解决：`n['id'] in alive or n['type'] == 'LoadImage'`。配方体检新增 R15：LoadImage 主图节点必在，红灯拦截。若一时查不清兜底行为，先断掉偷扫 input 目录的分支，让它报错而不是静默换图。工作流结构问题只准打补丁，不准重造。

防再犯

- [ ] 改提示词＝拿成功的工作流当底版只改提示词字段，绝不重新搭工作流
- [ ] 剪枝逻辑对 UI 锚节点（无连线但被引用的）一律豁免
- [ ] 体检规则覆盖「关键节点必须存在」类断言

关联坑

- 2-6 底版铁律的实锤案例；2-8 换参考图三处同步。

2-8 分段图不加载：内层素材库少一层 input 就是废层

症状

分段生成时参考图没加载上——人物/场景跟参考图对不上，或报找不到图——但 LoadImage 外层图看着是好的。图明明放进去了，分段就是不认，很容易误判成提示词问题。

真因

工作流里每段的 `genImage.imageFile` 写的是
`minimax_material_library/mat_<md5>.png`，它的真实解析根目录是：

```
<本地路径> ← 正库（老图都在这层）
```

ComfyUI 对这类相对路径的解析根是 input 目录——工作流里写相对路径、引擎拼 input 根，所以素材必须落在 input 层。放成 `<本地路径>` 就是废层，少一个 `input\` 放多少都不加载；也不是 `user\minimax_h3_motion_director\material_library\files\`，那是 UI 素材库的另一套存储。三处都能放文件、都不报错，只有正库那层会被读到——路径看着只差一层，效果是零。

修法

- 新参考图算 md5，命名 `mat_<md5>.png`（与库里老图命名惯例一致）。
- 拷进 `<本地路径>`（目录不存在先建好再拷，别让拷贝静默落进别的层）。
- 工作流各段 `genImage.imageFile` 指向 `minimax_material_library/mat_<md5>.png`。
- 拷完用 md5 复核＋确认与老图同层共存；错层副本要清掉，免得下次又认错层。

防再犯

- [ ] 换参考图三处同步：LoadImage（input\投料_参考图.png）＋素材库正库（input\minimax_material_library\）＋工作流 genImage 文件名
- [ ] 放图后 md5 复核并确认与老图同层共存
- [ ] 错层副本即时清理

关联坑

- 2-7 主图锚与 genImage 兜底；2-13 同属「改一处 ≠ 全改」家族。

2-9 报 "outside a model execution"：wrapper 链排序坑

症状

运行 H3 FirstBlockCache 工作流报 `RuntimeError: MiniMax H3 FirstBlockCache was called outside a model execution`，日志刚打出 `MiniMax H3 FBCache enabled: H3 Fast — 0.10 / max 2` 就崩。插件装了、参数也对，重装也没用。

真因

ComfyUI 的 WrapperExecutor 链式执行 wrapper，但 SolAttn 的 wrapper（`minimax_h3_patch.py:165`）调的是 `executor.original(...)`——直接跳到原始函数，跳过所有后续 wrapper，而不是 `executor(...)` 链到下一个。节点链 A→B→C 中 C 的 apply() 最后执行、wrapper 先注册；当 SolAttn 排在 FirstBlockCache 之前，SolAttn 的 wrapper 先注册并直接跳到底，FBCache 的 `begin_call` 永远不执行，`cache.current` 始终 None，于

是报「在模型执行之外被调用」（2026-09-08）。这个报错名字有误导性：不是调用时机错，是链条被截断、初始化根本没跑到。通病：跳链节点必须排链末，排在它后面的 wrapper 全成死代码。同文件第二个坑：节点更新新增 mode 下拉和 temporal_guard 开关后，旧工作流 widgets_values 还是 4 个值 `[0.1, 0.1, 0.95, 2]`，按位置映射全体错位，报 `Value 2.0 bigger than max of 1.0: end_percent`。

修法

调换节点连接顺序为 TeaCache → FirstBlockCache → SolAttn → Guider/Scheduler：FBCache 先执行、wrapper 先注册（idx=0），其 wrapper 用 `executor()` 链到 SolAttn，SolAttn 再 `executor.original()` 跳到 _forward，begin_call 已在 FBCache wrapper 里执行过。顺序对了，FBCache 日志能看到缓存正常生效。widgets_values 错位则对照节点源码 INPUT_TYPES 重排（该节点正确顺序 6 值）：

```
node['widgets_values'] = ["H3 Fast — 0.10 / max 2", 0.1, 0.1, 0.95, 2, False]
```

涉及文件：`<ComfyUI目录>\user\default\workflows\精选工作流\H3_720P_FirstBlockCache_高质量版.json`。

防再犯

- [ ] 多个 wrapper 节点串联先想注册顺序：调 executor.original() 跳链的节点必须排在链末
- [ ] 插件升级后对照 INPUT_TYPES 重核 widgets_values 顺序与数量
- [ ] 报错文本与日志最后一行对照，直接定位被跳过的 wrapper

关联坑

- 2-5 widgets_values 错位的通用机制。

2-10 出片名字串台、宽景碎裂斑驳：位置数组才是真身

症状

两个坑（2026-10-04）。一，工作流叫【成片】第32集…，出的视频却叫 `碳硅问道_r2v身份锁测试_3段_00014`——视频与工作流对不上号，报问题查不回源。二，360 环拍大景碎裂斑驳：31 集 1:36 后大景全斑驳（片尾环拍最重），32 集同款环拍中段轻度，单人特写恒干净。

真因

坑一：ComfyUI 加载/出图真正吃的是节点 **widgets_values
位置数组**，`widgets_values_named` 只是命名镜像。装配器只改了 named，数组里旧值照旧生效——改了等于没改。坑二：段模板用了强光影动态类措辞（「光与影相交」「流动推移」类毒词已全库封杀），强光影措辞在宽景里被渲染成碎块；360 环拍要模型全角度补环境，逆光侧光下光逻辑崩；长视频越到片尾画质衰减越重。三因叠加，宽景必斑驳。

修法

坑一：两处都改——`wvn['filename_prefix']` ＋ `widgets_values[0]`。改任何节点 widget
同理：位置数组才是真身。坑二（用正向语）：光影措辞改「光线柔和稳定、方向全程一致、色彩连贯统一」；360 环拍带恒定锚「环拍全程光线方向与色调始终一致」。

防再犯

- [ ] 改输出名/控件值后核对位置数组与命名镜像两份一致
- [ ] 交付前抽样比对出片名与工作流名
- [ ] 宽景/环拍段禁用强光影动态措辞，必带光线恒定锚

关联坑

- 2-1 多份存储、2-13 三源同步，同族。

2-11 视频生成时间暴涨：先查分辨率，不是提示词

症状

同样 5 段 75 秒视频，原来 66 分钟变 88~90 分钟，甚至比 8 段 120 秒（87
分钟）还慢——内容更短、时间一样，第一直觉全指向提示词变长，翻日志也解释不了。

真因

排查走了 8 小时弯路：怀疑提示词变长、字符串节点太多（20 个 StringReplace + 6 CR
Text）、TTS 慢、显存，全不是。逐项 diff 后真凶是**渲染分辨率**：8 段是 480×864（每帧 41.5 万像素），5 段是 576×1056（每帧 60.8 万，+47%）。视频渲染耗时 ∝
像素总量，5×60.8 万 ≈ 304 万几乎等于 8×41.5 万 ≈ 332 万，时间自然一样。提示词长短、字符串节点数量对视频渲染时间几乎无影响——文本只编码一遍、不进逐帧渲染循环，只占几秒（2026-10-11）。

修法

生成时间异常，第一个查
width/height，再查步数/帧数/段数，最后才查提示词。估时看「像素总量 = 段数 × 宽 × 高」，不是看视频秒数。排查手法：diff 同类工作流的 MotionDirector
参数（width/height/steps/total_frames），一眼定位。5 段 362 帧 28 步实测对照：480×864 约 54 分钟；512×912 约 65 分钟；576×1056 约 88
分钟。想提速优先降分辨率，其次降步数；降分辨率的代价是清晰度，换之前先想清交付规格。

防再犯

- [ ] 生成时间异常先 diff 分辨率参数，再想别的
- [ ] 估时按「段数×宽×高」算像素总量
- [ ] 提速优先降分辨率，其次降步数

2-12 重跑同名输出必炸：sidecar 不清光就 FileExistsError

症状

段5 分身修复时把旧 mp4 改名 `_分身废弃`，重跑同名 `--out`，引擎 5
秒即退，日志：`FileExistsError: [WinError 183] 当文件已存在时，无法创建该文件。: '...20261030_B版_段5_seed42.artifacts'`。随后 whisper QC 对不存在的 mp4
二次报错，ffmpeg concat 又用残缺清单拼出 37
秒坏整片（缺段）——一环炸三环（2026-10-31）。

真因

FreeVideo generate 会在 `--out` 同名下生成一整套 sidecar：`.artifacts/`
目录（`mkdir(exist_ok=False)`）＋ `.debug.json`、`.encoding.*`、`.engine.*`、`.request.json`、`.sampling-memory.json`。这些是引擎的中间产物与审计记录，设计上假设输出名唯一，`exist_ok=False` 是防覆盖不是 bug。用户只改名/删 mp4，sidecar
原地不动，重跑必撞名拒跑。指纹好认：日志 traceback 末行 `FileExistsError ... .artifacts`，exit=1 且耗时 <10 秒。下游还会连环踩：QC 读不到 mp4、concat
拿残缺清单拼坏整片。

修法

- 重跑同名输出前，删光同名 sidecar（`.artifacts` 目录＋所有同名 `.json/.log/.csv`），幂等清场再跑——清到与「全新输出名」等价。
- 修坏片重拼时，先 `ffprobe` 验产物存在且时长对，再 concat。
- 批量脚本的「已存在跳过」判断只看 mp4 会漏掉半成品状态——补跑前一律清场。

防再犯

- [ ] 重跑同名输出先清 sidecar 全家
- [ ] concat 前 ffprobe 逐段验存在与时长
- [ ] 批量跳过判断不能只看 mp4，要看整套产物状态

关联坑

- FreeVideo 产物管理系列（引擎审计哈希、manifest 分隔符）同族。

2-13 UI 只显示 5 段旧提示词：timeline 三源必须同步

症状

API 装配改了 `widgets_values[11]`（timeline JSON）且渲染正常，但 UI
主面板「分层」里只显示 5 段旧提示词——用户看 UI
一眼揪出：「主节点里分层中的分段落只写了 5 段，应当是 8
段，还需要加载后面的三个分段。」（2026-10-16 实测）

真因

MotionDirector 节点的 timeline 存在三个独立副本，改一处 ≠ 全改：`widgets_values[11]`
/ `widgets_values_named['timeline_data']` 是**渲染真源**（MD_MAP 位 11
提交用它）；`widgets_values_named['timeline_data']` 是 **UI
面板显示源**（分层列表读它，实测底版残留 5
段旧快照）；`properties.mmx_director_widget_state['timeline_data']` 是 UI
状态快照。三份各存各的、互不联动：API 装配只写了提交链路读的那份，UI
面板读的是命名镜像，状态快照更是独立持久化。另有 `widgets_values[1]` /
`widgets_values_named['global_prompt']`
全局提示词两个副本。只改渲染源，机器读的是新的、人读的还是旧的——UI 会当场打脸。

修法

装配后三源同步＋fail-closed 验证：

```
good = n46['widgets_values'][11] # 渲染源（最新 JSON
字符串）<br/>n46['widgets_values_named']['timeline_data'] = good # 灌进 UI
显示源<br/>n46['properties']['mmx_director_widget_state']['timeline_data'] = good<br/>#
global_prompt 同理三处：widgets_values[1]、named['global_prompt']、state['global_prompt']<br/>v2 =
json.loads(n46['widgets_values_named']['timeline_data'])<br/>assert len(v2['segments']) ==
期望段数, "UI 源段数不对"<br/>assert 期望首段关键词 in v2['segments'][0]['prompt'], "UI
源内容不是新的"
```

装配完必须打印「三源段数 N N N」全等＋首末段开头抽查——只验渲染源会被 UI 打脸。

防再犯

- [ ] 装配完打印三源段数全等才算过
- [ ] 关键改动让 UI 复核一轮——UI 是第二验收人
- [ ] global_prompt 同步三处

关联坑

- 2-1 多份存储总纲；2-8 管图片三处同步、本条管 timeline 文本，同族不同物。