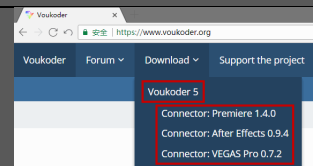


从 x264 教程综合版总结出的参数配置。目的是把参数直接贴到软件里用。要求 x265 v3.5+69 或 v3.6

<a href="#">LigH</a>	.hevc GCC10 [单文件 8-10-12bit] 附 x86, Windows XP x86 版 附 libx265.dll
<a href="#">Rigaya</a>	.hevc GCC 9.3 [8-10-12bit] 附 x86 版
<a href="#">Patman</a>	.hevc GCC 11+MSVC1925 [8-10-12bit]
<a href="#">ShortKatz</a>	arm64~64e 加 x86 版 [?] 需 macOS 运行编译命令文件 ?
<a href="#">DJATOM-aMod</a>	opt-Intel 架构与 zen1~2 优化 [10bit], opt-znver3 代表 zen3 优化 [10-12bit] GCC 10.2.1+GCC10.3
<a href="#">MeteorRain-yuuki</a>	lsmash.mkv/mp4 或 .hevc [能封装, 但传说 lavf 不如 pipe 可靠] GCC 9.3+ICC 1900+MSVC 1916 [8][10][12bit]+[8-10-12bit]
<a href="#">ffmpeg</a> 多系统兼容, 备用地址 <a href="#">ottverse.com/ffmpeg-builds</a>	
<a href="#">mpv 播放器</a> 比 Potplayer 好在没有音频滤镜, 不用手动关; 没有颜色偏差, 文件体积小	
<a href="#">x265GuiEx (Rigaya)</a> 日本語, auto-setup 安装, <a href="#">教程点此</a>	
<a href="#">Voukoder; V-Connector</a> 免费 Premiere/Vegas/AE 插件, 直接用 ffmpeg 内置编码器, 不用帧服务器/导无损再压/找破解。只要下两个压缩包, 放 Plug-Ins\Common 文件夹就行了	



## x265.exe 命令行用法教程

[照上表下载 ffmpeg, ffprobe 和 x265 并记住路径, 此处置于 D 盘根目录下]

София (D:)	ffmpeg.exe	2021/10/30 12:22	应用程序	93,660 KB
Creek-SC1NA400G (E:)				
Regme-HDWD120-581				
Cabliccus (I:)				
Hersert-HUH728080 (J)	x265-8bit.exe	2021/2/12 18:13	应用程序	20,720 KB
Cynic-HUH724040 (N:)	x265-10bit.exe	2021/3/17 17:13	应用程序	1,174 KB

[打开 Windows 的 CMD/PowerShell 或 Linux/MacOS 的 Bash/Terminal, 分别输入 ffmpeg, ffprobe, x265 的路径并回车, 确认程序存在]

[查 ffmpeg 版本信息] C:\文件夹\ffmpeg.exe; [查 x265 版本信息] C:\文件夹\x265.exe -V

```
选择管理员: 命令提示符
Microsoft Windows [版本 10.0.17763.2628]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\JC>D:\x265-10bit.exe -V
x265 [info]: HEVC encoder version 3.5+20-4c4ae0bc [DJATOM's Mod]
x265 [info]: build info [Windows][GCC 10.2.1][64 bit] 10bit
x265 [info]: using cpu capabilities: MMX2 SSE2Fast LZCNT SSSE3 SSE4.2 AVX FMA3 BMI2 AVX2

C:\Users\JC>D:\ffmpeg.exe
ffmpeg version n4.4.1-20211030 Copyright (c) 2000-2021 the FFmpeg developers
  built with gcc 10-win32 (GCC) 20210610
  configuration: --prefix=ffbuild/prefix --pkg-config-flags=--static --pkg-config=pkg-config --cross-prefix=x86_64-w64-
mingw32- --arch=x86_64 --target-os=mingw32 --enable-gpl --enable-version3 --disable-debug --disable-w32threads --enable-
pthread --enable-iconv --enable-libxml2 --enable-zlib --enable-libfreetype --enable-libfribidi --enable-gmp --enable-lz
ma --enable-fontconfig --enable-libvorbis --enable-opencl --enable-libvmaf --enable-vulkan --disable-libxcb --disable-xl
ib --enable-amf --enable-libaom --enable-avisynth --enable-libdav1d --enable-libdav2 --disable-libfdk-aac --enable-ffnv
codec --enable-cuda-llvm --disable-frei0r --enable-libglslang --enable-libgme --enable-libass --enable-libbluray --enabl
e-libmp3lame --enable-libopus --enable-libtheora --enable-libvpx --enable-libwebp --enable-lv2 --enable-libmfx --enable-
libopencore-amrnb --enable-libopencore-amrwb --enable-libopenjpeg --enable-librav1e --enable-librubberband --enable-scha
nnel --enable-sdl2 --enable-libsoxr --enable-libsrt --enable-libsvtav1 --enable-libtwolame --enable-libuavs3d --disable-
libdrm --disable-vaapi --enable-libvidstab --enable-libx264 --enable-libx265 --enable-libxavs2 --enable-libxvid --enable-
libzimg --enable-libzvi --extra-cflags=-DLIBTWOLAME_STATIC --extra-cxxflags= --extra-ldflags=-pthread --extra-ldexefla
gs= --extra-libs=lgomp --extra-version=20211030
```

[CMD 路径自动填充] 路径名写一半, 然后按[Tab]直到文件名匹配为止

[用 ffprobe 获取视频编码格式名, 宽, 编码宽, 高, 编码高, 色彩空间格式, 色彩空间范围, 逐行/分行, 帧率, 平均帧率, 总帧数] ffprobe.exe -i ".\视频.mp4" -select\_streams v:0 -v error -hide\_banner -show\_streams -show\_frames -read\_intervals "%+#1" -show\_entries frame=top\_field\_first:stream=codec\_long\_name,width,coded\_width,height,coded\_height,pix\_fmt,color\_range,field\_order,r\_frame\_rate,avg\_frame\_rate,nb\_frames -of ini

```
[frames.frame.0]
top_field_first=0          分场-是否上场优先(1/0)

[streams.stream.0]
codec_long_name=H.264      源视频格式
width=1920                 宽
height=1080                高
coded_width=1920           编码宽 - 若!=宽则代表横向长方形像素源
coded_height=1088          编码高 - 若!=高则代表纵向长方形像素源
pix_fmt=yuv420p            色彩空间
color_range=tv             色彩范围(pc=full=0~255/tv=limited=16~235)
field_order=progressive    逐行/分场(progressive/interlaced/unknown)
r_frame_rate=24000/1001    帧率
avg_frame_rate=24000/1001  编码帧率 - 若!=帧率则代表可变帧率vfr
nb_frames=20238            总帧数 - 根据压缩速度fps推测完成时间
```

[遇到分场源] 据 top\_field\_first 判断<上/下场优先>, 添加 x265 参数--interlaced<tff/bff>

[源视频为可变帧率] 手机省电用. 因兼容性问题应添加 ffmpeg 参数-vsync cfr 转换为恒定帧率 cfr

[长方形像素] 日本电视台缩宽, 旧版优酷缩高, 现今被抛弃的压缩手段. 能换源则尽可能换

[压制用时] 总帧数 ÷ 压缩速度 fps=时间(秒)

[x265 所需信息] ffmpeg 输入-pix\_fmt<源视频的色彩空间>, 类似图中结果

[参数用例] D:\ffmpeg.exe -i .\视频.mov -an -pix\_fmt yuv420p10 -f yuv4mpegpipe -strict unofficial - | D:\x265-10bit.exe --preset slow --me umh --subme 5 --merange 48 --weightb -aq-mode 4 --bframes 5 --ref 3 --hash 2 --allow-non-conformance --qg-size 16 --rd 3 --limit-modes --limit-refs 1 --rskip 1 --rd-refine --splitrd-skip --no-sao --tskip --colorprim bt2020 --colormatrix bt2020nc --transfer smpte2084 --y4m - --output F:\导出.hevc 2>D:\桌面\ffmpeg 或 x265 报错.txt

## ffmpeg, VS, avs2yuv pipe

ffmpeg -i [源] -an -f yuv4mpegpipe -strict unofficial - | x265 --y4m - --output

ffmpeg -i [源] -an -f rawvideo - | x265.exe --input-res [分辨率] --fps [] - --output

-i 导入, -f 格式, -an 关音频编码, -strict unofficial 允许自定格式, --y4m 指"YUV for MPEG"未压缩格式以便 pipe 传输, "ffmpeg - | x265 -"之间的"- "是 pipe 格式

VSpipe.exe [脚本].vpy --y4m - | x265.exe --y4m --output

VSpipe/avs2yuv [脚本].vpy - | x265.exe --input-res [分辨率] --fps [] - --output

avs2yuv.exe [脚本].avs -raw - | x265.exe --input-res [分辨率] --fps [] - --output

**ffmpeg .ass 字幕渲染滤镜:** `-filter_complex "ass='F\:/字幕.ass'"`

**中途停止压制, 并封装现有帧为视频:** `Ctrl+C`, 部分人编译的 x265.exe 自带功能

**qaac 压制音频:** 见[教程](#)或[Github 转载](#)

**ffmpeg 内置缩放:** `-sws_flags bicubic bitexact gauss bicublin lanczos spline +full_chroma_int +full_chroma_inp +accurate_rnd`

例: `-sws_flags bitexact+full_chroma_int+full_chroma_inp+accurate_rnd`)

**HDR 标识** `--master-display` <手动告知播放器拿什么色彩空间解码

**DCI-P3:**  $G(13250, 34500)B(7500, 3000)R(34000, 16000)WP(15635, 16450)L(\max\text{CLL} \times 10000, 1)$

**bt709:**  $G(15000, 30000)B(7500, 3000)R(32000, 16500)WP(15635, 16450)L(\max\text{CLL} \times 10000, 1)$

**bt2020:**  $G(8500, 39850)B(6550, 2300)R(35400, 14600)WP(15635, 16450)L(\max\text{CLL} \times 10000, 1)$

- 找到 HDR 元数据中的色彩范围, 确认用以下哪个色彩空间后填上参数
- L 的值没有标准, 每个 HDR 视频元数据里可能都不一样

**DCI-P3:**  $G(x0.265, y0.690), B(x0.150, y0.060), R(x0.680, y0.320), WP(x0.3127, y0.329)$

**bt709:**  $G(x0.30, y0.60), B(x0.150, y0.060), R(x0.640, y0.330), WP(x0.3127, y0.329)$

**bt2020:**  $G(x0.170, y0.797), B(x0.131, y0.046), R(x0.708, y0.292), WP(x0.3127, y0.329)>$

`--max-cll` <maxCLL,maxFALL>最大, 平均光强度, MediaInfo 查不出来就不用填

**色域标识** `--colormatrix` <照源, 例: gbr bt709 fcc bt470bg smpte170m YCgCo bt2020nc bt2020c smpte2084 ictcp>

**色域转换** `--transfer` <照源, 例: gbr bt709 fcc bt470bg smpte170m YCgCo bt2020nc bt2020c smpte2084 ictcp>

# 速度参考

**处理器:** R7 5800X 全 8 核负电压偏移超频 4.5Ghz, FS140 下烤机 67°C 稳, CbR23 均 15440 (PBO2 负 30 偏移超频 4.85Ghz, 86°C 仅提升 2%算力故不用)

**内存:** 海力士 MFR 2×2R×8GB/2x16GB, 3000Mhz 15-17-17-35 1T 1.44V, F-U-MCLK 等比同步

**源①:** 1920x1080 yuv420p8 24000/1001fps 312MB 低清 h.264 录像源, 高对比材质纹理, 20238 帧. **方:** 10bit crf 28 以增加像素值偏移错误差距, 用低清源来去掉画质差距, 高对比细节动态增加计算量

**preset slow:** 16 分 27, 平均~20.5fps, 压缩后 217MB, 画质损失可见 (高清源下更明显)

**通用·简单:** 24 分 48, 平均~13.6fps, 比 slow 慢 1.5x, 得 159MB, 损失可见

**高压·动漫:** 36 分 36, 平均~9.21fps, 比 slow 慢 2.2x, 得 145MB, 由于片源不匹配所以损失可见

**高压·录像:** 78 分 57, 平均~4.27fps, 比 slow 慢 4.8x, 得 189MB, 损失小但受源限制, 要仔细看

**veryslow:** 133 分 16, 平均~2.53fps, 比 slow 慢 8.1x, 得 221MB, 损失小

**源②:** 3840x2160 yuv444p12 24fps 37GB PQ ProRes4444XQ 6314 帧. **法:** 检测极高成本动漫(时间复杂度高, 空间复杂度精确到 1 像素), 4:4:4 上的压制速度, 以及稳定性

**通简-CRF16:** 75 分 21, 1.4fps 1159MB, **CRF18:** 73 分 19, 1.44fps 902MB, **CRF20:** 69 分 58, 1.5fps 698MB, **CRF22:** 中途失败, 剩一半但可后期拼接. 注: 以上开 hist-scenecut 的测试中途都失败了

**源③:** 1920x1080 yuv420p10 24000/1001fps 1.9GB h.264, 34095 帧. **方:** 检测低成本动漫上, 4:2:0 的压制速度. (空间-时间复杂度简单的源)

**高压•动漫:** 46 分 43, 平均~12fps

**源④:** 1920x1080 yuv444p10 24000/1001fps 2.9GB h.264, 40920 帧. **法:** 检测高成本动漫, 4:4:4 上的压制速度.

(空间-时间复杂度较为困难的源)

**高压•动漫:** 97 分 30, 平均~6.99fps

# 通用·简单·低清

自定义项目 all off, 方便急用且速度仅比 preset slow 慢几 fps

**预设-转场** `--preset slow --hist-scenecut`

**动态搜索** `--me umh --subme 5 --merange 48 --weightb`

**自适应量化** `--aq-mode 4`

**帧控** `--bframes 5 --ref 3`

**输入输出** `--hash 2 --allow-non-conformance`

**目标色深** `-D 8/10/12` (单程序兼容多色深时建议手动指定, 一般默认 8bit, 低勿转高, 高转低开 `--dither`)

**多处理器分配** `--pools ,,,` (举例-,+表示该电脑有两个 CPU 节点, 用第二个. 同时占用多个会造成严重的内存延迟)

**其它** [去黑边加速](#): `--display-window <整数"<←, ↑, →, ↓">像素>`, [≥16 核 cpu 优化](#): `--pme`, [分场视频](#): `--`

`field`, [抖动高质量降色深](#): `--dither`, [开始; 结束帧](#): `--seek; --frames`, [crf/abr 缓解噪点影响](#): `--rc-grain`

**目标色彩空间** `ffmpeg -pix_fmt yuv420p / yuv422p / yuv444p / yuv420p10 / yuv422p10 / yuv444p10...`

## (ffmpeg pipe) x265 CLI 命令

- `ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -an -f yuv4mpegpipe -strict unofficial -pix_fmt<ffprobe pix_fmt> - | x265.exe --preset slow --me umh --subme 5 --merange 48 --weightb --aq-mode 4 --bframes 5 --ref 3 --hash 2 --allow-non-conformance --y4m - --output ".\输出.hevc"`

## libx265 CLI, 兼容 libav

- `ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -c:v libx265 -pix_fmt<ffprobe pix_fmt> -x265params`

```
"preset=slow:me=umh:subme=5:merange=48:weightb=1:bframes=5:ref=3:hash=2:allow-non-conformance=1" -c:a copy ".\输出.hevc"
```

### libkvazaar CLI (实验性, 第三方, 暂缺 crf)模式 (libx265 ffmpeg CLI 缺 85%的命令, 无法使用)

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide\_banner -i ".\导入.mp4" -c:v libkvazaar -pix\_fmt<ffprobe pix\_fmt> -kvazaar-params "limit-tu=1:tr-depth-intra=2:pu-depth-intra=4:pu-depth-inter=3:smp=1:amp=1:bipred=1:me=tz:subme=4:merange=48:me-early-termination=off:max-merge=2:ref=3:open-gop=0:period=360:gop=16:transform-skip=1:qp=16:fast-residual-cost=1:early-skip=1:max-merge=4:rd=3:mv-rdo=1:rdoq-skip=1:intra-rdo-et=1:sao=edge:hash=checksum" -c:a copy ".\输出.hevc"

### ffmpeg 封装视音频, 字幕(后缀指定封装格式)

- ffmpeg.exe -i ".\视频流入.hevc" -an -c:v copy -i ".\音频流入.aac" -c:a copy -i ".\传统字幕.srt" -c:s copy "封装出.mkv"
- ffmpeg.exe -i ".\视频流入.hevc" -an -c:v copy -i ".\音频1入.aac" -c:a copy -i ".\音频2入.aac" -c:a copy -i ".\字幕1.ass" -c:s copy -i ".\字幕2.ass" -c:s copy "封装出.mkv"

不同封装格式的字幕格式支持: [Wikipedia - Subtitle formats support](#)

### ffmpeg 替换封装中的音频流, itoffset±秒数以对齐:

- ffmpeg.exe -i ".\封装入.mov" -i ".\新音频流入.aac" -c:v copy -map 0:v:0 -map 1:a:0 -c:a copy -itsoffset 0 ".\新封装出.mov"

ffmpeg 警告 thread\_queue\_size 太小: 加-thread\_queue\_size<压制平均码率 kbps+1000>

批处理: 完成后转换为普通命令窗(不退出): cmd /k

批处理:完成后不关闭命令窗+保留输入栏+显示 Windows 版本: cmd -k

# 通用标准

含大量自定义项目，可以配出高压或高速参数

## 分块-变换

--tu-intra-depth 3 --tu-inter-depth 3 --limit-tu 1 --rdpenalty 1

## 动搜-补偿

--me umh --subme <24fps=3, 48fps=4, 60fps=5, 100fps=6> --merange 48 --weightb

## 溯块-帧控

--ref 3 --max-merge <2 快, 3 中, 4 慢> --early-skip --no-open-gop --min-keyint 5 --

keyint <9 × 帧率> --fades --bframes 8 --b-adapt 2 --radl 3 <锐利线条: --pbratio 1.2>

## 帧内编码

--hist-scenecut <快: --fast-intra / 中: 不填 / 慢: --b-intra / 更慢: + --constrained-intra>

## 量化

--crf <18~20 超清, 19~22 高清> --crqpoffs -3 --cbqpoffs -1

## 率失优量化

--rdoq-level <1 快, 2 很慢>

## 自适应量化

<动漫源改--hevc-aq, 关 aq-mode> --aq-mode 4 --aq-strength <多面=0.8, 多线=1>

## 模式决策

--rd 3 --limit-modes --limit-refs 1 --rskip <3 快, 2 中, 1 慢> --rc-lookahead <3 × 帧率> -

-tskip-fast --rect <很慢: --amp>

## 率失真优化

--psy-rd <录像=1.6, 动画=0.6, ctu=64 加0.6, =16 减0.6> --splitrd-skip <实验性: --qp-adaptation-

range 3>

## 去块-取迁

--limit-sao --sao-non-deblock --deblock 0:-1

## 输入输出

--hash 2 --allow-non-conformance <外/内网 NAS 串流: --idr-recovery-sei>

## 目标色深

-D 8/10/12 (单程序兼容多色深时建议手动指定, 一般默认 8bit, 低勿转高, 高转低开--dither)

## 多处理器分配

--pools ,,, (举例-,+表示该电脑有两个 CPU 节点, 用第二个. 同时占用多个会造成严重的内存延迟)

## 其它

去黑边加速: --display-window <整数"←, ↑, →, ↓"像素>, ≥16 核 cpu 优化: --pme, 分场视频: --

field, 抖动高质量降色深: --dither, 开始; 结束帧: --seek; --frames, crf/abr 缓解噪点影响: --rc-grain

## 目标色彩空间

ffmpeg -pix\_fmt yuv420p / yuv422p / yuv444p / yuv420p10 / yuv422p10 / yuv444p10...

(ffmpeg pipe) x265 CLI 命令-共 12 个自定义域, 1 个危险自定义域



- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide\_banner -i ".\导入.mp4" -an -f yuv4mpegpipe -strict unofficial -pix\_fmt<ffprobe pix\_fmt> - | x265.exe --ctu ○ --min-cu-size 16 --tu-intra-depth 3 --tu-inter-depth 3 --limit-tu 1 --rdpenalty 1 --me umh --subme ○ --merange 48 --weightb --ref 3 --max-merge ○ --early-skip --no-open-gop --min-keyint 5 --fades --bframes 8 --b-adapt 2 --radl 3 --pbratio 1.2 --fast-intra --b-intra --constrained-intra --crf ○ --crqpoffs -3 --crqpoffs -1 --rdoq-level ○ --aq-mode 4 --aq-strength ○ --rd 3 --limit-modes --limit-refs 1 --rskip ○ --rc-lookahead ○ --tskip-fast --rect --amp --psy-rd ○ --splitrd-skip --qp-adaptation-range 4 --limit-sao --sao-non-deblock --deblock 0:-1 --hash 2 --allow-non-conformance --y4m --output ".\输出.hevc"

## libx265 CLI, 兼容 libav

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide\_banner -i ".\导入.mp4" -c:v libx265 -pix\_fmt<ffprobe pix\_fmt> -x265params "ctu=○:min-cu-size=16:tu-intra-depth=3:tu-inter-depth=3:limit-tu=1:rdpenalty=1:me=umh:subme=○:merange=48:weightb=1:ref=3:max-merge=○:early-skip=1:open-gop=0:min-keyint=5:fades=1:bframes=8:b-adapt=2:radl=3:pbratio=1.2:fast-intra=1:b-intra=1:constrained-intra=1:crf=○:crqpoffs=-3:cbqpoffs=-1:rdoq-level=○:aq-mode=4:aq-strength=○:rd=3:limit-modes=1:limit-refs=1:rskip=○:rc-lookahead=○:tskip-fast=1:rect=1:amp=1:psy-rd=○:splitrd-skip=1:qp-adaptation-range=4:limit-sao=1:sao-non-deblock=1:deblock=0:-1:hash=2:allow-non-conformance=1" -c:a copy ".\输出.hevc"

## ffmpeg 封装视音频, 字幕(后缀指定封装格式)

- ffmpeg.exe -i ".\视频流入.hevc" -an -c:v copy -i ".\音频流入.aac" -c:a copy -i ".\传统字幕.srt" -c:s copy "封装出.mkv"

- `ffmpeg.exe -i ".\视频流入.hevc" -an -c:v copy -i ".\音频1入.aac" -c:a copy -i ".\音频2入.aac" -c:a copy -i ".\字幕1.ass" -c:s copy -i ".\字幕2.ass" -c:s copy "封装出.mkv"`

**不同封装格式的字幕格式支持:** [Wikipedia - Subtitle formats support](#)

**ffmpeg 替换封装中的音频流, itoffset±秒数以对齐:**

- `ffmpeg.exe -i ".\封装入.mov" -i ".\新音频流入.aac" -c:v copy -map 0:v:0 -map 1:a:0 -c:a copy -itsoffset 0 ".\新封装出.mov"`

**批处理: 完成后转换为普通命令窗(不退出):** `cmd /k`

**批处理:完成后不关闭命令窗+保留输入栏+显示 Windows 版本:** `cmd -k`

# 高压·录像·要求高画质源

要求高画质源，否则画质优势比不过通用-简单，慢很多

## 分块-变换

--tu-intra-depth 4 --tu-inter-depth 4 --limit-tu 1

## 动搜-补偿

--me star --subme <24fps=3, 48fps=4, 60fps=5, 100fps=6> --merange 48 --weightb

## 溯块-帧控

--ref 3 --max-merge 4 --no-open-gop --min-keyint 3 --keyint 310 --fades --bframes  
8 --b-adapt 2 --radl 3

## 帧内编码

--hist-scenecut --constrained-intra --b-intra

## 量化

--crf 21.8 --qpmin 8 --crqpoffs -3 --ipratio 1.2 --pbratio 1.5

## 率失优量化

--rdoq-level 2

## 自适应量化

--aq-mode 4 --aq-strength <1~1.3> --qg-size 8

## 模式决策

--rd 3 --limit-refs 0 --rskip 0 --rc-lookahead <1.8 × 帧率> --rect --amp

## 率失真优化

--psy-rd <录像=1.6, 动画=0.6, ctu=64 就加 0.6, =16 就减 0.6> <实验性: --qp-adaptation-range 3>

## 去块

--deblock 0:-1

## 取样迁就偏移

--limit-sao --sao-non-deblock --selective-sao 3

## 输入输出

--hash 2 --allow-non-conformance <外/内网 NAS 串流: --idr-recovery-sei>

## 目标色深

-D 8/10/12 (单程序兼容多色深时建议手动指定，一般默认 8bit，低勿转高，高转低开--dither)

## 多处理器分配

--pools ,,, (举例-,+表示该电脑有两个 CPU 节点，用第二个。同时占用多个会造成严重的内存延迟)

## 其它

去黑边加速: --display-window <整数"←,↑,→,↓"像素>, ≥16 核 cpu 优化: --pme, 分场视频: --  
field, 抖动高质量降色深: --dither, 开始; 结束帧: --seek; --frames, crf/abr 缓解噪点影响: --rc-grain

## 目标色彩空间

ffmpeg -pix\_fmt yuv420p / yuv422p / yuv444p / yuv420p10 / yuv422p10 / yuv444p10...

## (ffmpeg pipe) x265 CLI 命令

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide\_banner -i ".\导入.mp4" -an -f yuv4mpegpipe -strict unofficial -pix\_fmt<ffprobe pix\_fmt> - | x265.exe --tu-intra-depth 4 --tu-inter-depth 4 --limit-tu 1 --me star --subme ○ --merange 48 --weightb --ref 3 --max-merge 4 --no-open-gop --min-keyint 3 --keyint 310 --fades --bframes 8 --b-adapt 2 --radl 3 --constrained-intra --b-intra --crf 21.8 --qpmin 8 --crqpoffs -3 --ipratio 1.2 --pbratio 1.5 --rdoq-level 2 --aq-mode 4 --aq-strength ○ --qg-size 8 --rd 3 --limit-refs 0 --rskip 0 --rc-lookahead ○ --rect --amp --psy-rd ○ --qp-adaptation-range 3 --deblock 0:-1 --limit-sao --sao-non-deblock --selective-sao 3 --hash 2 --allow-non-conformance --y4m - --output ".\输出.hevc"

## libx265 CLI, 兼容 libav

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide\_banner -i ".\导入.mp4" -c:v libx265 -pix\_fmt <ffprobe pix\_fmt> -x265params "tu-intra-depth=4:tu-inter-depth=4:limit-tu=1:me=star:subme=○ :merange=48:weightb=1:ref=3:max-merge=4:open-gop=0:min-keyint=3:keyint=310:fades=1:bframes=8:b-adapt=2:radl=3:constrained-intra=1:b-intra=1:crf=21.8:qpmin=8:crqpoffs=-3:ipratio=1.2:pbratio=1.5:rdoq-level=2:aq-mode=4:aq-strength=○ :qg-size=8:rd=3:limit-refs=0:rskip=0:rc-lookahead=○ :rect=1:amp=1:psy-rd=○ :qp-adaptation-range=3:deblock=0:-1:limit-sao=1:sao-non-deblock=1:selective-sao=3:hash=2:allow-non-conformance=1" -c:a copy ".\输出.hevc"

## ffmpeg 封装视音频, 字幕(后缀指定封装格式)

- ffmpeg.exe -i ".\视频流入.hevc" -an -c:v copy -i ".\音频流入.aac" -c:a copy -i ".\传统字幕.srt" -c:s copy "封装出.mkv"

- `ffmpeg.exe -i ".\视频流入.hevc" -an -c:v copy -i ".\音频 1 入.aac" -c:a copy -i ".\音频 2 入.aac" -c:a copy -i ".\字幕 1.ass" -c:s copy -i ".\字幕 2.ass" -c:s copy "封装出.mkv"`

**不同封装格式的字幕格式支持:** [Wikipedia - Subtitle formats support](#)

**批处理: 完成后转换为普通命令窗(不退出):** `cmd /k`

**批处理:完成后不关闭命令窗+保留输入栏+显示 Windows 版本:** `cmd -k`

# 剪辑素材存档

分块	<code>--ctu 32</code>
动态搜索	<code>--me star --subme &lt;24fps=3, 48fps=4, 60fps=5, 100fps=6&gt; --merange 48 --analyze-src-pics</code>
帧内搜索	<code>--max-merge 4 --early-skip --b-intra</code>
帧控制	<code>--hist-scenecut --no-open-gop --min-keyint 1 --keyint &lt;7×帧率&gt; --ref 3 --fades --bframes 7 --b-adapt 2</code>
量化	<code>--crf 17 --crqpoffs -3 --cbqpoffs -2</code>
模式决策	<code>--rd 3 --limit-modes --limit-refs 1 --rskip 1 --rc-lookahead &lt;4×帧率&gt;</code>
率失真优化	<code>--splitrd-skip</code>
环路滤波去块	<code>--deblock -1:-1</code>
输入输出	<code>--hash 2 --allow-non-conformance</code>
主控	<code>--tune grain</code>
目标色深	<code>-D 8/10/12</code> (单程序兼容多色深时建议手动指定, 一般默认 8bit, 低勿转高, 高转低开 <code>--dither</code> )
其它	去黑边加速: <code>--display-window &lt;整数"←,↑,→,↓"像素&gt;</code> , <b>≥16 核 cpu 优化</b> : <code>--pme</code> , 分场视频: <code>--field</code> , 抖动高质量降色深: <code>--dither</code> , 开始; 结束帧: <code>--seek; --frames</code> , <b>crf/abr</b> 缓解噪点影响: <code>--rc-grain</code>
目标色彩空间	<code>ffmpeg -pix_fmt yuv420p / yuv422p / yuv444p / yuv420p10 / yuv422p10 / yuv444p10...</code>

(ffmpeg pipe) x265 CLI 命令

- `ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -an -f yuv4mpegpipe -strict unofficial -pix_fmt<ffprobe pix_fmt> - | x265.exe --ctu 32 --me star --subme ○ --merange 48 --analyze-src-pics --max-merge 4 --early-skip --b-intra --no-open-gop --min-keyint 1 --keyint ○ --ref 3 --fades --bframes 7 --b-adapt 2 --crf 17 --crqpoffs -3 --cbqpoffs -2 --rd 3 --limit-modes --limit-refs 1 --rskip 1 --rc-lookahead ○ --splitrd-skip --deblock -1:-1 --hash 2 --allow-non-conformance --tune grain --y4m - --output ".\输出.hevc"`

## libx265 CLI, 兼容 libav

- `ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -c:v libx265 -pix_fmt<ffprobe pix_fmt> -x265params "ctu=32:me=star:subme=○:merange=48:analyze-src-pics=1:max-merge=4:early-skip=1:open-gop=0:min-keyint=1:keyint=○:ref=3:fades=1:bframes=7:b-adapt=2:radl=3:constrained-intra=1:b-intra=1:crf=17:crqpoffs=-3:cbqpoffs=-2:rd=3:limit-modes=1:limit-refs=1:rskip=1:rc-lookahead=○:splitrd-skip=1:deblock=-1:-1:hash=2:allow-non-conformance=1:tune=grain" -c:a copy ".\输出.hevc"`
- **目标色深-色彩空间:** `-pix_fmt yuv420p / yuv422p / yuv444p / yuv420p10 / yuv422p10 / yuv444p10...`

## ffmpeg 封装视音频, 字幕(后缀指定封装格式)

- `ffmpeg.exe -i ".\视频流入.hevc" -an -c:v copy -i ".\音频流入.aac" -c:a copy -i ".\传统字幕.srt" -c:s copy "封装出.mkv"`
- `ffmpeg.exe -i ".\视频流入.hevc" -an -c:v copy -i ".\音频1入.aac" -c:a copy -i ".\音频2入.aac" -c:a copy -i ".\字幕1.ass" -c:s copy -i ".\字幕2.ass" -c:s copy "封装出.mkv"`

不同封装格式的字幕格式支持: [Wikipedia - Subtitle formats support](https://en.wikipedia.org/wiki/Subtitle_formats_support)

# 高压·动漫·字幕组

遵守 x265 开发者定义的低到高成本动漫高压高画标准，秘诀是分块和 TU 开满+避免跳过，建议 YUV4:2:0 8~10bit

**分块·变换** `--tu-intra-depth 4 --tu-inter-depth 4 --max-tu-size 16`

**动搜·补偿** `--me umh --merange 48 --subme <24fps=3, 48fps=4, 60fps=5, 100fps=6> --weightb <80 年代动`

漫缺乏光线变化, 可略 `weightb> --max-merge 4 --early-skip`

**溯块·帧控** `--ref 3 --no-open-gop --min-keyint 5 --keyint <12×帧率> --fades --bframes 16 --b-adapt 2 --radl 3 --bframe-bias 20`

**帧内编码** `--hist-scenecut --constrained-intra --b-intra`

**量化** `--crf 22 --crqpoffs -4 --cbqpoffs -2 --ipratio 1.6 --pbratio 1.3 --cu-lossless --tskip`

**率失优量化** `--psy-rdoq 2.3 --rdoq-level 2`

**自适应量化** `--hevc-aq --aq-strength 0.9 --qg-size 8`

**模式决策** `--rd 3 --limit-modes --limit-refs 1 --rskip 1 --rc-lookahead <2.5×帧率> --rect --amp`

**率失真优化** `--psy-rd 1.5 --splitrd-skip --rdpenalty 2 <实验性: --qp-adaptation-range 4>`

**去块** `--deblock 0:-1`

**取样迁就偏移** `--limit-sao --sao-non-deblock`

**输入输出** `--hash 2 --allow-non-conformance <外/内网 NAS 串流: --single-sei --idr-recovery-sei>`

**多处理器分配** `--pools ,,,,` (举例-,+表示该电脑有两个 CPU 节点, 用第二个. 同时占用多个会造成严重的内存延迟)

**目标色深** `-D 8/10/12` (单程序兼容多色深时建议手动指定, 一般默认 8bit, 低勿转高, 高转低开 `--dither`)

**其它** **去黑边加速:** `--display-window <整数"←,↑,→,↓"像素>`, **≥16 核 cpu 优化:** `--pme`, **分场视频:** `--field`,

**抖动高质量降色深:** `--dither`, **开始; 结束帧:** `--seek; --frames`, **crf/abr 缓解噪点影响:** `--rc-grain`

**目标色彩空间** `ffmpeg -pix_fmt yuv420p / yuv422p / yuv444p / yuv420p10 / yuv422p10 / yuv444p10...`



## (ffmpeg pipe) x265 CLI 命令

- `ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -an -f yuv4mpegpipe -strict unofficial -pix_fmt<ffprobe pix_fmt> - | x265.exe --tu-intra-depth 4 --tu-inter-depth 4 --max-tu-size 16 --me umh --subme ☐ --merange 48 --weightb --max-merge 4 --early-skip --ref 3 --no-open-gop --min-keyint 5 --keyint ☐ --fades --bframes 16 --b-adapt 2 --radl 3 --bframe-bias 20 --constrained-intra --b-intra --crf 22 --crqpoffs -4 --cbqpoffs -2 --ipratio 1.6 --pbratio 1.3 --cu-lossless --tskip --psy-rdoq 2.3 --rdoq-level 2 --hevc-aq --aq-strength 0.9 --qg-size 8 --rd 3 --limit-modes --limit-refs 1 --rskip 1 --rc-lookahead ☐ --rect --amp --psy-rd 1.5 --splitrd-skip --rdpenalty 2 --qp-adaptation-range 4 --deblock -1:0 --limit-sao --sao-non-deblock --hash 2 --allow-non-conformance --y4m - --output ".\输出.hevc"`

## libx265 CLI, 兼容 libav

- `ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -c:v libx265 -pix_fmt<ffprobe pix_fmt> -x265params "tu-intra-depth=4:tu-inter-depth=4:max-tu-size=16:me=umh:subme=☐:merange=48:weightb=1:max-merge=4:early-skip=1:ref=3:open-gop=0:min-keyint=5:keyint=☐:fades=1:bframes=16:b-adapt=2:radl=3:bframe-bias=20:constrained-intra=1:b-intra=1:crf=22:crqpoffs=-4:cbqpoffs=-2:ipratio=1.6:pbratio=1.3:cu-lossless=1:tskip=1:psy-rdoq=2.3:rdoq-level=2:hevc-aq=1:aq-strength=0.9:qg-size=8:rd=3:limit-modes=1:limit-refs=1:rskip=1:rc-lookahead=☐:rect=1:amp=1:psy-rd=1.5:splitrd-skip=1:rdpenalty=2:qp-adaptation-range=4:deblock=-1:0:limit-sao=1:sao-non-deblock=1:hash=2:allow-non-conformance=1" -c:a copy ".\输出.hevc"`

# 动漫·Ripper 冷战·仅限 HEDT 工作站与原画

比字幕组参数码率更高，压制速度更慢

## 分块-变换

--tu-intra-depth 4 --tu-inter-depth 4 --max-tu-size 4 --limit-tu 1

## 动搜-补偿

--me star --subme <24fps=3, 48fps=4, 60fps=5, 100fps=6> --merange 52 --analyze-src-pics --weightb --max-merge 4

## 溯块-帧控

--ref 3 --no-open-gop --min-keyint 1 --keyint <12×帧率> --fades --bframes 16 --b-adapt 2 --radl 2

## 帧内编码

--hist-scenecut --b-intra

## 量化

--crf 16.5 --crqpoffs -5 --cbqpoffs -2 --ipratio 1.67 --pbratio 1.33

## 无损量化

--cu-lossless

## 率失真优化

--psy-rdoq 2.5 --rdoq-level 2

## 自适应量化

--hevc-aq --aq-strength 1.4 --qg-size 8

## 模式决策

--rd 5 --limit-refs 0 --rskip 2 --rskip-edge-threshold 3 --rc-lookahead <2.5×帧率> --

## 率失真优化

rect --amp --no-cutree

## 去块

--psy-rd 1.5 --rdpenalty 2 <实验性: --qp-adaptation-range 5>

## 取样迁就偏移

--deblock -2:-2

## 输入输出

--limit-sao --sao-non-deblock --selective-sao 1

## 多处理器分配

--hash 2 --allow-non-conformance <外/内网 NAS 串流: --single-sei --idr-recovery-sei>

## 目标色深

--pools ,,, (举例-,+表示该电脑有两个 CPU 节点, 用第二个. 同时占用多个会造成严重的内存延迟)

## 其它

-D 8/10/12 (单程序兼容多色深时建议手动指定, 一般默认 8bit, 低勿转高, 高转低开--dither)

去黑边加速: --display-window <整数"←, ↑, →, ↓"像素>, ≥16 核 cpu 优化: --pme, 分场视频: --field,

## (ffmpeg pipe) x265 CLI 命令

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide\_banner -i ".\导入.mp4" -an -f yuv4mpegpipe -strict unofficial -pix\_fmt<ffprobe pix\_fmt> - | x265.exe --tu-intra-depth 4 --tu-inter-depth 4 --max-tu-size 4 --limit-tu 1 --me star --subme ○ --merange 52 --analyze-src-pics --weightb --max-merge 4 --ref 3 --no-open-gop --min-keyint 1 --keyint ○ --fades --bframes 16 --b-adapt 2 --radl 2 --b-intra --crf 16.5 --crqpoffs -5 --cbqpoffs -2 --ipratio 1.67 --pbratio 1.33 --cu-lossless --psy-rdoq 2.5 --rdoq-level 2 --hevc-aq --aq-strength 1.4 --qg-size 8 --rd 5 --limit-refs 0 --rskip 2 --rskip-edge-threshold 3 --rc-lookahead ○ --rect --amp --no-cutree --psy-rd 1.5 --rdpenalty 2 --qp-adaptation-range 5 --deblock -2:-2 --limit-sao --sao-non-deblock --selective-sao 1 --hash 2 --allow-non-conformance --y4m - --output ".\输出.hevc"

## libx265 CLI, 兼容 libav

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide\_banner -i ".\导入.mp4" -c:v libx265 -pix\_fmt<ffprobe pix\_fmt> -x265params "tu-intra-depth=4:tu-inter-depth=4:max-tu-size=4:limit-tu=1:me=star:subme=○:merange=52:analyze-src-pics=1:weightb=1:max-merge=4:mcstf=1:ref=3:open-gop=0:min-keyint=1:keyint=○:fades=1:bframes=16:b-adapt=2:radl=2:b-intra=1:crf=16.5:crqpoffs=-5:cbqpoffs=-2:ipratio=1.6:pbratio=1.33:cu-lossless=1:psy-rdoq=2.5:rdoq-level=2:hevc-aq=1:aq-strength=1.4:qg-size=8:rd=5:limit-refs=0:rskip=2:rskip-edge-threshold=3:rc-lookahead=○:rect=1:amp=1:cutree=0:psy-rd=1.5:rdpenalty=2:qp-adaptation-range=5:deblock=-2:-2:limit-sao=1:sao-non-deblock=1:selective-sao=1:hash=2:allow-non-conformance=1" -c:a copy ".\输出.hevc"