

从 x265 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

[LigH](#)

.hevc GCC10 [单文件 8-10-12bit] 附 x86, Windows XP x86 版 附 libx265.dll

[Rigaya](#)

.hevc GCC 9.3 [8-10-12bit] 附 x86 版

[Patman](#)

.hevc GCC 11+MSVC1925 [8-10-12bit]

[ShortKatz](#)

arm64~64e 加 x86 版 [?] 需 macOS 运行编译命令文件 ?

[DJATOM-aMod](#)

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

[MeteorRain-yuuki](#)

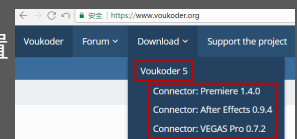
lsmash.mkv/mp4 或.hevc [能封装, 但传说 lavf 不如 pipe 可靠] GCC 9.3+ICC 1900+MSVC 1916 [8][10][12bit]+[8-10-12bit]

[ffmpeg](#) 多系统兼容, 备用地址 [ottverse.com/ffmpeg-builds](http://ottverse.com/ffmpeg-builds)

[mpv 播放器](#) 开源免费强大便携的现代软件, [安装配置教程见网页](#), 无色彩错误, 体积小

[Voukoder; V-Connector](#) 免费 Premiere/Vegas/AE/达芬奇插件, 用 ffmpeg 内置

编码器, 不用导无损再压/破解. 只要解两个压缩包, 放 Plug-Ins\Common 即可



[MediaInfo](#) 开源免费勤更新方便的 GUI 媒体元数据/视音频格式读取器, 用于配置正确的压制参数

[ffprobe](#) 和 ffmpeg 同源的 CLI 元数据/视音频格式读取器, 使用见[网页教程](#) (下载 ffmpeg 的压缩包内)

## x265.exe 命令行用法教程

[照上表下载 ffmpeg, ffprobe/MediaInfo, 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-HDWD120-58I				
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] 如此处检查 D:\x265-10bit.exe -V 和 D:\ffmpeg.exe 确认程序存在

```
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+20-4c4aee0bc [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-
```

[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  
[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 -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
```

[源视频为可变帧率] 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)

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

## -pix\_fmt 与 -strict 参数

ffmpeg 能够检测元数据并自动设定 pix\_fmt, 除非元数据没写才(与分辨率、帧率一并)手动设定。需要确认时可以使用 MediaInfo 或 ffprobe 查找, 类型有: yuv420p, yuv422p, yuv444p, yuv420p10le, yuv420p12le, yuv422p10le, yuv422p12le, yuv444p10le, yuv444p12le, yuv444p10le, yuv444p12le, gray, gray10le, gray12le, nv12, nv16

在使用管道/pipe 参数时, 超过 8bit 的 YUV for MPEG 流并不合规, 需要额外提供 -strict 参数解除合规性限制, 而在使用 ffmpeg 内置库时则不会用到管道, 因此无需指定。

```
[yuv4mpegpipe @ 0000018dde853540] 'yuv420p10le' is not an official yuv4mpegpipe pixel format. Use '-strict -1' to encode to this pixel format.
[out#0/yuv4mpegpipe @ 0000018dde8d9e40] Could not write header (incorrect codec parameters ?): Invalid argument
[vf#0:0 @ 0000018dde85e640] Error sending frames to consumers: Invalid argument
[vf#0:0 @ 0000018dde85e640] Task finished with error code: -22 (Invalid argument)
[vf#0:0 @ 0000018dde85e640] Terminating thread with return code -22 (Invalid argument)
[out#0/yuv4mpegpipe @ 0000018dde8d9e40] Nothing was written into output file, because at least one of its streams received no packets.
```

## x264/5 怎么选位深

同时含 8-10-12bit 的 x265.exe (用 x265.exe -V 检查)通过参数-D, 如-D 10 设置编码 10bit 位深; 若下载了已区分为 x265-8bit.exe, x265-10bit.exe 则直接调用对应位深的版本

## ffmpeg, VS, avs2yuv pipe

```
ffmpeg -i [源] -an -f yuv4mpegpipe - | x265 --y4m --input - --output
```

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

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

```
VSpipeline.exe [脚本].vpy --y4m - | x265.exe --y4m --input - --output
```

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

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

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

## QAAC 压制音频见[教程](#)或 [Github](#)

**ffmpeg 内置缩放:** 例: `-sws_flags bitexact+full_chroma_int+full_chroma_inp+accurate_rnd`

- `-sws_flags bicubic/bitexact/gauss/bicublin/lanczos/spline/+full_chroma_int/+full_chroma_inp/+accurate_rnd`

**ffmpeg 封装视音频, 更改导出文件后缀名以指定封装格式(.mkv 格式还支持封装字幕, 字体)**

- `ffmpeg.exe -i ".\视频输入.hevc" -an -c:v copy -i ".\音频输入.aac" -c:a copy -i ".\字幕.srt" -c:s copy "封装输出.mp4"`
- `ffmpeg -i ".\视频输入.hevc" -an -c:v copy -i ".\音轨1输入.aac" -c:a copy -i ".\音轨2输入.aac" -c:a copy -i ".\字幕1输入.ass" -c:s copy -i ".\字幕2输入.ass" -c:s copy -i ".\字体1输入.ttf" -c:t copy "封装输出.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: small thread\_queue\_size 警告:

- thread\_queue\_size<(源平均码率 kbps+1000)/可调用 CPU 核心数>

批处理: 完成后转换为普通命令窗(不退出): cmd /k+显示 Windows 版本: cmd -k

x265 HDR 设置参数:

x265

--master-display <手动告知播放器拿什么色彩空间解码

HDR 标识

DCI-P3: G(13250,34500)B(7500,3000)R(34000,16000)WP(15635,16450)L(maxCLL × 10000,1)

bt709: G(15000,30000)B(7500,3000)R(32000,16500)WP(15635,16450)L(maxCLL × 10000,1)

bt2020: G(8500,39850)B(6550,2300)R(35400,14600)WP(15635,16450)L(maxCLL × 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 ictp>

色域转换

--transfer <照源, 例: gbr bt709 fcc bt470bg smpte170m YCgCo bt2020nc bt2020c smpte2084 ictp>

杜比视界 dolby vision/DV 有 DV-MEL (BL+RPU)和 DV-FEL (BL+EL+RPU)两种带 RPU 的格式, x265

支持共 3 种样式/profile 的 DV-MEL

样式	编码	BL:EL 分辨率	x265 支持	伽马	色彩空间
4	10bit hevc	1:1/4		SDR	YCbCr
5		仅 BL (DV-MEL)	✓		ICtCp
7		4K=1:1/4; 1920x1080=1:1		UHD 蓝光	
8.1		仅 BL (DV-MEL)	✓	HDR10	YCbCr
8.2			✓	SDR	

8.4				HLG	
9	8bit avc	仅 BL (DV-MEL)		SDR	YCbCr

**--dolby-vision-profile**<选择 5/8.1 (HDR10)/8.2 (SDR)>8.1 需要写 master-display 和 hdr10-opt

**--dolby-vision-rpu**<路径>导入 rpu 二进制文件(.bin)用

目标色深

ffmpeg 有能够发送视频帧元数据的 yuv-for-mpeg pipe (管道), 和只发送视频帧的 raw pipe, 而管道下游的 x265.exe 根据版本和 mod 不同, 不一定能够识别 yuv-for-mpeg 的元数据; 同时, x265 的位深设定是仅 CLI (CLI-ONLY) 参数, 例如在 ffmpeg 的 libx265 中, 位深由 ffmpeg 自身指定。因此, 本教程中 ffmpeg pipe 的参数会要求 -D 选项指定视频色深, 而 ffmpeg libx265 则没有。

x265 管道输入参数变更

x265 v4.0 版中引入了 Multiview Encoding (多视角输入编码), 因此 ffmpeg pipe 的格式从自 x264 以来的"- "变更为"--input - "参数

选择规格 Profile, 级别 Level

根据视频位深选择规格 Profile, 分辨率和帧率选择级别 Level, 最后细分到档次 Tier (Main/High)

级别 Level	最大亮度像素流量 (Luma Samples)	最大亮度平面面积 (Luma Size)	8-10bit 最大码率	12bit 最大码率	4:4:4 12bit 最大码率	最高分辨率@帧率
1	552,960	36,864	Main: 128 Kbps High: -	Main: 192 Kbps High: -	Main: 384 Kbps High: -	176 × 144@15 fps
2	3,686,400	122,880	Main: 1500 Kbps High: -	Main: 2250 Kbps High: -	Main: 4500 Kbps High: -	352 × 288@30 fps
2.1	7,372,800	245,760	Main: 3000 Kbps High: -	Main: 4500 Kbps High: -	Main: 9000 Kbps High: -	640 × 360@30 fps
3	16,588,800	552,960	Main: 6000 Kbps High: -	Main: 9000 Kbps High: -	Main: 18 Mbps High: -	960 × 540@30 fps
3.1	33,177,600	983,040	Main: 10 Mbps High: -	Main: 15 Mbps High: -	Main: 30 Mbps High: -	1280 × 720@33.7 fps

4	66,846,720	2,228,224	<b>Main:</b> 12 Mbps <b>High:</b> 30 Mbps	<b>Main:</b> 18 Mbps <b>High:</b> 45 Mbps	<b>Main:</b> 36 Mbps <b>High:</b> 90 Mbps	1280 × 720@68 1920 × 1080@32 fps
4.1	133,693,440	2,228,224	<b>Main:</b> 20 Mbps <b>High:</b> 50 Mbps	<b>Main:</b> 30 Mbps <b>High:</b> 75 Mbps	<b>Main:</b> 60 Mbps <b>High:</b> 150 Mbps	1920 × 1080@64 fps 2048 × 1080@60 fps
5	267,386,880	8,912,896	<b>Main:</b> 25 Mbps <b>High:</b> 100 Mbps	<b>Main:</b> 37.5 Mbps <b>High:</b> 150 Mbps	<b>Main:</b> 75 Mbps <b>High:</b> 300 Mbps	3840 × 2160@32 fps 4096 × 2160@30 fps
5.1	534,773,760	8,912,896	<b>Main:</b> 40 Mbps <b>High:</b> 160 Mbps	<b>Main:</b> 60 Mbps <b>High:</b> 240 Mbps	<b>Main:</b> 120 Mbps <b>High:</b> 480 Mbps	3840 × 2160@64 fps 4096 × 2160@60 fps
5.2	1,069,547,520	8,912,896	<b>Main:</b> 60 Mbps <b>High:</b> 240 Mbps	<b>Main:</b> 90 Mbps <b>High:</b> 360 Mbps	<b>Main:</b> 180 Mbps <b>High:</b> 720 Mbps	3840 × 2160@128 fps 4096 × 2160@120 fps
6	1,069,547,520	35,651,584	<b>Main:</b> 60 Mbps <b>High:</b> 240 Mbps	<b>Main:</b> 90 Mbps <b>High:</b> 360 Mbps	<b>Main:</b> 180 Mbps <b>High:</b> 720 Mbps	7680 × 4320@32 fps 8192 × 4320@30 fps
6.1	2,139,095,040	35,651,584	<b>Main:</b> 120 Mbps <b>High:</b> 480 Mbps	<b>Main:</b> 180 Mbps <b>High:</b> 720 Mbps	<b>Main:</b> 360 Mbps <b>High:</b> 1440 Mbps	7680 × 4320@64 fps 8192 × 4320@60 fps
6.2	4,278,190,080	35,651,584	<b>Main:</b> 240 Mbps <b>High:</b> 800 Mbps	<b>Main:</b> 360 Mbps <b>High:</b> 1200 Mbps	<b>Main:</b> 720 Mbps <b>High:</b> 2400 Mbps	7680 × 4320@128 fps 8192 × 4320@120 fps

为了方便使用，本教程设定为一直打开 `--high-tier` 选项，详见 [x265 教程完整版](#)

## 有兼容性问题的参数

### ffmpeg

`-strict unofficial / -l`（允许超过 8bit 的 y4m 以及其他格式，提高管道/pipe 兼容性，但不能提早发现一些下游兼容性问题，在使用 ffmpeg 内置库时）

`-hwaccel auto`（硬件解码，由于部分硬件厂商的解码实现较差，所以可能会花屏，同时可能与 `analyze-src-pics` 有冲突）

### x265

`--allow-non-conformance`（允许不合规参数以提高压缩率和画质，但可能会遇到播放、剪辑兼容问题）

## 非必要参数

### ffmpeg

`-hide_banner`（解决命令行窗口被版权协议等信息填满的问题）

`-pix_fmt`, `-strict`（见上方：`-pix_fmt` 与 `-strict` 参数）

## x265

--hash (每帧校验, 纠错的效果和没有纠错差不多)

--radl (支持 I 帧前放置 RADL 帧, 会改动 GOP 结构, 虽然播放没问题, 但分段拼合时兼容性差)

--mcstf (不稳定, 可能会导致压制失败)

--rd 5 (大多情况下 3 就够了, 但因压力高于 Cinebench R23, 所以可用于测试 CPU 超频稳定性)

--qp-adaptation-range (有人在 x265 v4.1 遇到了编码后视频帧播放一小段后冻结的问题, 但这可能是硬解错误)

## 速度排名

通用简单 > 素材存档 > 动漫高压 > 通用标准 > 电影高压 > HEDT。最高差距为 13~15 倍 fps

# 通用·简单

去掉所有自定义项目填 Profile, Level, 方便急用且速度仅比 preset slow 慢几 fps

**兼容性** `--profile<8/10/12bit: main/main10/main12, YUV4:2:2: main422-10/main422-12, YUV4:4:4: main444-8/main444-10/main444-12> --high-tier`

**预设-转场** `--preset slow`

**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`

**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>`, **≥22 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...`

## α——(ffmpeg pipe) x265 CLI 命令

- `ffmpeg.exe -y -i ".\导入.mp4" -an -f yuv4mpegpipe -pix_fmt ○ -strict -1 - | x265.exe --profile ○ --high-tier --preset slow --me umh --subme 5 --merange 48 --weightb --aq-mode 4 --bframes 5 --ref 3 --y4m --input - --output ".\输出.hevc"`

## β——ffmpeg libx265 CLI, 拷贝音频并封装为 mp4

- `ffmpeg.exe -y -i ".\导入.mp4" -c:v libx265 -profile:v ○ -x265-params "high-tier=1:preset=slow:me=umh:subme=5:merange=48:weightb=1:bframes=5:ref=3" -fps_mode passthrough -c:a copy ".\输出.mp4"`



# 通用·标准

含大量自定义项目，可以配出高压或高速参数

## 兼容性

--profile<8/10/12bit: [main/main10/main12](#), YUV4:2:2: [main422-10/main422-12](#), YUV4:4:4: [main444-8/main444-10/main444-12](#)> --high-tier

## 分块-变换

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

## 动搜-补偿

--me umh --subme <24fps: [3](#), 48fps: [4](#), 60fps: [5](#), 100fps: [6](#)> --merange <1920:1080: [48](#), 2560:1440: [52](#), 3840:2160: [56](#)> --weightb

## 溯块-帧控

--ref 3 --max-merge <快: [2](#), 中: [3](#), 慢: [5](#)> --early-skip --no-open-gop --min-keyint 5 --fades --bframes 8 --b-adapt 2 <锐利线条: --pbratio [1.2](#)>

## 帧内编码

<快: --fast-intra / 中: 不填 / 慢: --b-intra / 极慢且有兼容性问题: --constrained-intra>

## 量化

--crf <超清: [18~20](#), 高清: [19~22](#)> --crqpoffs -3 --cbqpoffs -1

## 率失优量化

--rdoq-level <快: [1](#), 很慢: [2](#)>

## 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 <快: [2](#), 中: [1](#), 慢: [0](#)> --rc-lookahead <[3](#)×帧率, 大于 bframes> --rect <很慢: --amp>

## 率失真优化

--psy-rd <录像: [1.6](#), 动画: [0.6](#), ctu64: +0.6, ctu16: -0.6> --splitrd-skip

## 去块-取迁

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

## 目标色深

-D 8/10/12 <单程序兼容多色深时须手动指定, 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>, ≥22 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

## α——(ffmpeg pipe) x265 CLI 命令-共 11+2 个自定义

- ffmpeg.exe -y -i ".\导入.mp4" -an -f yuv4mpegpipe -strict -1 - | x265.exe --profile ☐ --high-tier  
--ctu ☐ --min-cu-size 16 --tu-intra-depth 3 --tu-inter-depth 3 --limit-tu 1 --rdpenalty 1 --me  
umh --subme ☐ --merange ☐ --weightb --ref 3 --max-merge ☐ --early-skip --no-open-gop --  
min-keyint 5 --fades --bframes 8 --b-adapt 2 --pbratio 1.2 --fast-intra --b-intra --crf ☐ --crqpoffs  
-3 --cbqpoffs -1 --rdoq-level ☐ --aq-mode 4 --aq-strength ☐ --rd 3 --limit-modes --limit-refs 1  
--rskip ☐ --rc-lookahead ☐ --rect --amp --psy-rd ☐ --splitrd-skip --limit-sao --sao-non-  
deblock --deblock 0:-1--y4m --input - --output ".\输出.hevc"

## β——ffmpeg libx265 CLI, 拷贝音频并封装为 mp4

- ffmpeg.exe -y -i ".\导入.mp4" -c:v libx265 -profile:v ☐ -x265-params "high-tier=1:ctu=☐:min-cu-  
size=16:tu-intra-depth=3:tu-inter-depth=3:limit-tu=1:rdpenalty=1:me=umh:subme=☐:merange=  
☐:weightb=1:ref=3:max-merge=☐:early-skip=1:open-gop=0:min-keyint=5:fades=1:bframes=8:b-  
adapt=2:pbratio=1.2:fast-intra=1:b-intra=1:crf=☐:crqpoffs=-3:cbqpoffs=-1:rdoq-aq-mode=4:aq-strength=  
☐:rd=3:limit-modes=1:limit-refs=1:rskip=☐:rc-lookahead=☐:rect=1:amp=1:psy-rd=☐:splitrd-skip=1:limit-  
sao=1:sao-non-deblock=1:deblock=0,-1" -fps\_mode passthrough -c:a copy ".\输出.mp4"

# 高压·录像/3D 动画

建议高清源，否则画质不如通用-简单，更慢，但一般压缩率更高

## 兼容性

--profile <8/10/12bit: main/main10/main12, YUV4:2:2: main422-10/main422-12, YUV4:4:4: main444-8/main444-10/main444-12> --high-tier

## 分块-变换

--ctu 64 --tu-intra-depth 4 --tu-inter-depth 4 --limit-tu 1 --rect --tskip --tskip-fast

## 动搜-补偿

--me star --subme <24fps: 3, 48fps: 4, 60fps: 5, 100fps: 6> --merange <1920:1080: 48, 2560:1440: 52, 3840:2160: 56> --weightb

## 溯块-帧控

--ref 4 --max-merge 5 --no-open-gop --min-keyint 3 --keyint <9×帧率> --fades --bframes 8 --b-adapt 2 --analyze-src-pics

## 帧内编码

--b-intra <极慢且可能会造成块失真，增加压缩率: --constrained-intra>

## 量化

--crf 21.8 --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 <多面: 1~多线: 1.3> --qg-size 8

## 模式决策

--rd 5 --limit-refs 0 --rskip 0 --rc-lookahead <1.8×帧率, 大于 bframes>

## 率失真优化

--psy-rd <录像: 1.6, 动画: 0.6, ctu64: +0.6, ctu16: -0.6>

## 去块

--deblock 0:-1

## 取样迁就偏移

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

## 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>, ≥22 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

## α——(ffmpeg pipe) x265 CLI 命令

- ffmpeg.exe -y -i ".\导入.mp4" -an -f yuv4mpegpipe -strict -1 - | x265.exe --profile ☐ --high-tier  
--ctu 64 --tu-intra-depth 4 --tu-inter-depth 4 --limit-tu 1 --rect --tskip --tskip-fast --me star --  
subme ☐ --merange ☐ --weightb --ref 4 --max-merge 5 --no-open-gop --min-keyint 3 --keyint  
☐ --fades --bframes 8 --b-adapt 2 --analyze-src-pics --b-intra --crf 21.8 --crqpoffs -3 --ipratio 1.2  
--pbratio 1.5 --rdoq-level 2 --aq-mode 4 --aq-strength ☐ --qg-size 8 --rd 5 --limit-refs 0 --rskip  
0 --rc-lookahead ☐ --psy-rd ☐ --deblock 0:-1 --limit-sao --sao-non-deblock --selective-sao 3--  
y4m --input - --output ".\输出.hevc"

## β——ffmpeg libx265 CLI, 拷贝音频并封装为 mp4

- ffmpeg.exe -y -i ".\导入.mp4" -c:v libx265 -profile:v ☐ -x265-params "high-tier=1:ctu=64:tu-intra-  
depth=4:tu-inter-depth=4:limit-tu=1:rect=1:tskip=1:tskip-fast=1:me=star:subme=☐ :merange=  
☐ :weightb=1:ref=4:max-merge=5:open-gop=0:min-keyint=3:keyint=☐ :fades=1:bframes=8:b-  
adapt=2:analyze-src-pics=1:b-intra=1:crf=21.8:crqpoffs=-3:ipratio=1.2:pbratio=1.5:rdoq-level=2:aq-  
mode=4:aq-strength=☐ :qg-size=8:rd=5:limit-refs=0:rskip=0:rc-lookahead=☐ :psy-rd=☐ :deblock=0,-  
1:limit-sao=1:sao-non-deblock=1:selective-sao=3" -fps\_mode passthrough -c:a copy ".\输出.mp4"

# 剪辑素材存档

通过减少 P 帧，B 帧数量来降低解码压力，从而降低剪辑软件负载；兼容 ≥ 画质+压缩

## 兼容性

--profile <8/10/12bit: main/main10/main12, YUV4:2:2: main422-10/main422-12, YUV4:4:4: main444-8/main444-10/main444-12> --high-tier

## 分块-变换

--ctu 32 --tskip

## 动态搜索

--me star --subme <24fps: 3, 48fps: 4, 60fps: 5, 100fps: 6> --merange <1920:1080: 48, 2560:1440: 52, 3840:2160: 56> --analyze-src-pics

## 帧内搜索

--max-merge 5 --early-skip --b-intra

## 帧控制

--no-open-gop --min-keyint 1 --keyint <5×帧率> --ref 3 --fades --bframes 4 --b-adapt 2

## 量化

--crf 17 --crqpoffs -3 --cbqpoffs -2

## 模式决策

--rd 3 --limit-modes --limit-refs 1 --rskip 1 --rc-lookahead <4×帧率, 大于 bframes>

## 率失真优化

--splitrd-skip

## 环路滤波去块

--deblock -1:-1

## 主控

--tune grain

## 其它

去黑边加速: --display-window <整数"←, ↑, →, ↓"像素>, ≥22 核 cpu 优化: --pme, 分场视频: --field,

## α——(ffmpeg pipe) x265 CLI 命令

- `ffmpeg.exe -y -i ".\导入.mp4" -an -f yuv4mpegpipe -strict -1 - | x265.exe --profile ☐ --high-tier --ctu 32 --tskip --me star --subme ☐ --merange ☐ --analyze-src-pics --max-merge 5 --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--tune grain --y4m --input - --output ".\输出.hevc"`

## β——ffmpeg libx265 CLI, 拷贝音频并封装为 mp4

- `ffmpeg.exe -y -i ".\导入.mp4" -c:v libx265 -profile:v ☐ -x265-params "high-tier=1:ctu=32:tskip=1:me=star:subme=☐:merange=☐:analyze-src-pics=1:max-merge=5:early-skip=1:open-gop=0:min-keyint=1:keyint=☐:ref=3:fades=1:bframes=7:b-adapt=2: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:tune=grain" -fps_mode passthrough -c:a copy ".\输出.mp4"`

# 高压·动漫·字幕组

建议 YUV4:2:0; 8~10bit

兼容性	<code>--profile&lt;8/10/12bit: main/main10/main12, YUV4:2:2: main422-10/main422-12, YUV4:4:4: main444-8/main444-10/main444-12&gt; --high-tier</code>
分块-变换	<code>--tu-intra-depth 4 --tu-inter-depth 4 --max-tu-size 16 --tskip --tskip-fast</code>
动搜-补偿	<code>--me umh --subme &lt;24fps: 3, 48fps: 4, 60fps: 5, 100fps: 6&gt; --merange &lt;1920:1080: 48, 2560:1440: 52, 3840:2160: 56&gt; --weightb --max-merge 5 --early-skip</code>
溯块-帧控	<code>--ref 3 --no-open-gop --min-keyint 5 --keyint &lt;12×帧率&gt; --fades --bframes 16 --b-adapt 2 --bframe-bias 20</code>
帧内编码	<code>--b-intra &lt;极慢且可能会造成画面问题: + --constrained-intra&gt;</code>
量化	<code>--crf 22 --crqpoffs -4 --cbqpoffs -2 --ipratio 1.6 --pbratio 1.3 --cu-lossless --psy-rdoq 2.3 --rdoq-level 2</code>
率失优量化	<code>--hevc-aq --aq-strength 0.9 --qg-size 8</code>
自适应量化	<code>--rd 3 --limit-modes --limit-refs 1 --rskip 1 --rc-lookahead &lt;2.5×帧率, 大于 bframes&gt; --</code>
模式决策	<code>rect --amp</code>
率失真优化	<code>--psy-rd 1.5 --splitrd-skip --rdpenalty 2</code>
去块	<code>--deblock 0:-1</code>
取样迁就偏移	<code>--limit-sao --sao-non-deblock</code>
其它	去黑边加速: <code>--display-window &lt;整数"←, ↑, →, ↓"像素&gt;</code> , $\geq 22$ 核 cpu 优化: <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> , crf/abr 缓解噪点影响: <code>--rc-grain</code> , 外/内网 NAS 串流: <code>--single-sei --idr-recovery-sei</code>

## α——(ffmpeg pipe) x265 CLI 命令

- ffmpeg.exe -y -i ".\导入.mp4" -an -f yuv4mpegpipe -strict -1 - | x265.exe --profile ☐ --high-tier --tu-intra-depth 4 --tu-inter-depth 4 --max-tu-size 16 --tskip --tskip-fast --me umh --subme ☐ --merange ☐ --weightb --max-merge 5 --early-skip --ref 3 --no-open-gop --min-keyint 5 --keyint ☐ --fades --bframes 16 --b-adapt 2 --bframe-bias 20 --constrained-intra --b-intra --crf 22 --crqpoffs -4 --cbqpoffs -2 --ipratio 1.6 --pbratio 1.3 --cu-lossless --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 --deblock -1:0 --limit-sao --sao-non-deblock --y4m --input - --output ".\输出.hevc"

## β——ffmpeg libx265 CLI, 拷贝音频并封装为 mp4

- ffmpeg.exe -y -i ".\导入.mp4" -c:v libx265 -profile:v ☐ -x265-params "high-tier=1:tu-intra-depth=4:tu-inter-depth=4:max-tu-size=16:tskip=1:tskip-fast=1:me=umh:subme=☐:merange=☐:weightb=1:max-merge=5:early-skip=1:ref=3:open-gop=0:min-keyint=5:keyint=☐:fades=1:bframes=16:b-adapt=2:bframe-bias=20:b-intra=1:crf=22:crqpoffs=-4:cbqpoffs=-2:ipratio=1.6:pbratio=1.3:cu-lossless=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:deblock=-1,0:limit-sao=1:sao-non-deblock=1" -fps\_mode passthrough -c:a copy ".\输出.mp4"



# 动漫/原画·高算力 HEDT 工作站

压力高，画质高，压缩率不高，不适合大部分情况

兼容性	<code>--profile&lt;8/10/12bit: main/main10/main12, YUV4:2:2: main422-10/main422-12, YUV4:4:4: main444-8/main444-10/main444-12&gt; --high-tier</code>
分块-变换	<code>--tu-intra-depth 4 --tu-inter-depth 4 --max-tu-size 4 --limit-tu 1 --rect --amp --tskip</code>
动搜-补偿	<code>--me star --subme &lt;24fps: 3, 48fps: 4, 60fps: 5, 100fps: 6&gt; --merange &lt;1920:1080: 52, 2560:1440: 56, 3840:2160: 64&gt; --analyze-src-pics --weightb --max-merge 5</code>
溯块-帧控	<code>--ref 3 --no-open-gop --min-keyint 1 --keyint &lt;12×帧率&gt; --fades --bframes 16 --b-adapt 2</code>
帧内编码	<code>--b-intra</code>
量化	<code>--crf 18.1 --crqpoffs -5 --cbqpoffs -2 --ipratio 1.67 --pbratio 1.33 --cu-lossless</code>
率失优量化	<code>--psy-rdoq 2.5 --rdoq-level 2</code>
自适应量化	<code>&lt;普通: --hevc-aq --aq-strength 1.4; Jpsdr Mod: --aq-auto 10 --aq-bias-strength 1.3 --aq-strength-edge 1.4 --aq-bias-strength 1.1&gt; --qg-size 8</code>
模式决策	<code>--rd 5 --limit-refs 0 --rskip 2 --rskip-edge-threshold 3 --rc-lookahead &lt;2.5×帧率, 大于bframes&gt; --no-cutree</code>
率失真优化	<code>--psy-rd 1.5 --rdpenalty 2 &lt;实验性: --qp-adaptation-range 5&gt;</code>
去块	<code>--deblock -2:-2</code>
取样迁就偏移	<code>--limit-sao --sao-non-deblock --selective-sao 1</code>

α——(ffmpeg pipe) 普通 x265 CLI 命令

- `ffmpeg.exe -y -i ".\导入.mp4" -an -f yuv4mpegpipe -strict -1 - | x265.exe --profile ☐ --high-tier --tu-intra-depth 4 --tu-inter-depth 4 --max-tu-size 4 --limit-tu 1 --rect --amp --tskip --me star --subme ☐ --merange ☐ --analyze-src-pics --weightb --max-merge 5 --ref 3 --no-open-gop --min-keyint 1 --keyint ☐ --fades --bframes 16 --b-adapt 2 --b-intra --crf 18.1 --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 ☐ --no-cutree --psy-rd 1.5 --rdpenalty 2 --deblock -2:-2 --limit-sao --sao-non-deblock --selective-sao 1--y4m --input - --output ".\输出.hevc"`

## β——(ffmpeg pipe) x265 jpsdr-Mod CLI 命令

- `ffmpeg.exe -y -i ".\导入.mp4" -an -f yuv4mpegpipe -strict -1 - | x265.exe --profile ☐ --high-tier --tu-intra-depth 4 --tu-inter-depth 4 --max-tu-size 4 --limit-tu 1 --rect --amp --tskip --me star --subme ☐ --merange ☐ --analyze-src-pics --weightb --max-merge 5 --ref 5 --no-open-gop --min-keyint 1 --keyint ☐ --fades --bframes 16 --b-adapt 2 --b-intra --crf 18.1 --crqpoffs -5 --cbqpoffs -2 --ipratio 1.67 --pbratio 1.33 --cu-lossless --psy-rdoq 2.5 --rdoq-level 2 --aq-auto 10 --aq-bias-strength 1.3 --aq-strength-edge 1.4 --aq-bias-strength 1.1 --qg-size 8 --rd 3 --limit-refs 0 --rskip 2 --rskip-edge-threshold 3 --rc-lookahead ☐ --no-cutree --psy-rd 1.5 --rdpenalty 2 --deblock -2:-2 --limit-sao --sao-non-deblock --selective-sao 1--y4m --input - --output ".\输出.hevc"`

## γ——普通 ffmpeg libx265 CLI, 拷贝音频并封装为 mp4

- `ffmpeg.exe -y -i ".\导入.mp4" -c:v libx265 -profile:v ☐ -x265-params "high-tier=1:tu-intra-depth=4:tu-inter-depth=4:max-tu-size=4:limit-tu=1:rect=1:amp=1:tskip=1:me=star:subme=☐:merange=☐:analyze-src-pics=1:weightb=1:max-merge=5:ref=3:open-gop=0:min-keyint=1:keyint=`

○:fades=1:bframes=16:b-adapt=2:b-intra=1:crf=18.1: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=○:cutree=0:psy-rd=1.5:rdpenalty=2:deblock=-2:-2:limit-sao=1:sao-non-deblock=1:selective-sao=1" -fps\_mode passthrough -c:a copy ".\输出.mp4"