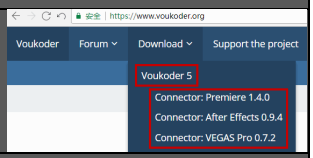
从 x264 教程综合版总结出的参数配置. 目的是把参数直接贴到软件里用, 本教程提供的参数不完美, 应根据自己情况修改

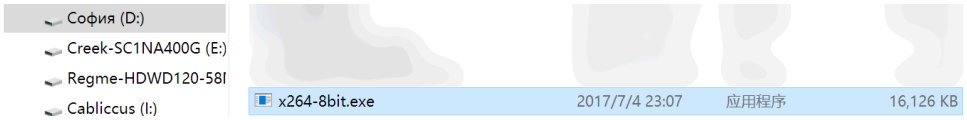
软件下载

ffmpeg 多系统兼容, 备用地址 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 
OBS 直播与录屏 支持 AVS 滤镜, 设置复杂但强大, 去 x264 教程急用版照着设置就行了
x264 by Patman, LigH √lavf 编解码, 合并 8~10bit
x264 tMod by jspdr √lavf 编解码, 支持 MCF 线程管理库(比 posix 和 win32 性能更好)
x264 7mod 谷歌盘/百度云 √lavf 编解码, √hqdn3d 降噪
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 媒体元数据/视音频格式读取器, 见网页教程, 以及搭配 Excel 的视频数据可视化(于下载 ffmpeg 的压缩包内)

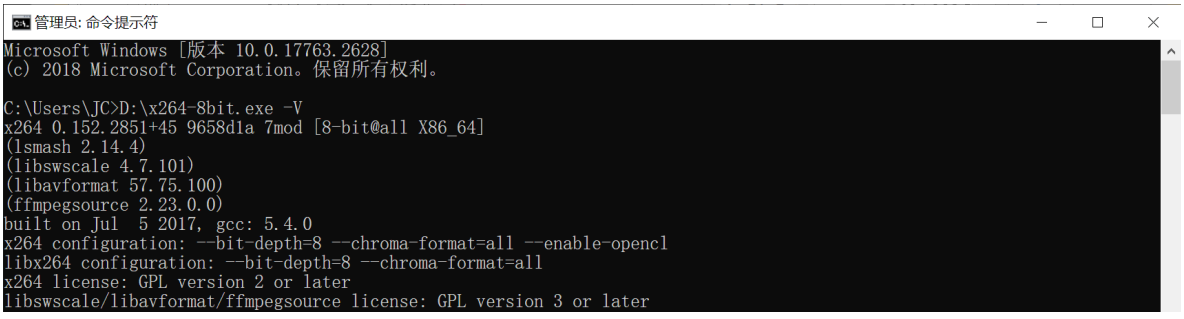
压制软件工作流程图解 (不严谨)

x264.exe 命令行参数用法教程(新人必看)

[照上表下载 x264.exe 并记住路径] 此处置于 D 盘根目录下



[打开 Windows CMD/PowerShell 或 Linux/MacOS 的 Bash/Terminal, 分别输入 ffmpeg, x264 的路径, 程序名和-V 命令并回车] 如下 D:\x264-8bit.exe -V; D:\ffmpeg.exe -V 看到程序返回值以确认程序存在



[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

[引用程序] C:\文件夹\x264.exe

[CLI 参数] --me esa --merange 48 --keyint 200 [...]

[x264/5 的导出命令和导入] --output C:\文件夹\导出.mp4 C:\文件夹\导入.mp4

[参数格式] x264.exe --me esa --merange 24 [...] --output "导出.mp4" "导入.mp4"

[参数用例] D:\x264-8bit.exe --me umh --subme 11 --merange 32 -I 270 -i 1 -b 11 --b-adapt 2 -r 3 --direct auto --crf 19 --qpmin 13 --rc-lookahead 90 --aq-mode 3 --aq-strength 1 --trellis 2 --deblock 0:0 --psy-rd 0.7:0.2 --fullrange --vf hqdn3d:1.1,1.1,1.1,1.1 --output "F:\导出.mp4" "D:\导入.mp4"

x264/5 怎么选位深

同时含 8-10-12bit 的 x264.exe (用 x264.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 x264-8bit.exe, x264-10bit.exe 则直接调用对应位深的版本

压制三角形定律 重要, 见右图

QAAC 压制音频见教程或 Github

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

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.mp4"
- ffmpeg -i ".\视频.hevc" -an -c:v copy -i ".\音轨 1.aac" -c:a copy -i ".\音轨 2.aac" -c:a copy -i ".\字幕 1.ass" -c:s copy -i ".\字幕 2.ass" -c:s copy -i ".\字体 1.ttf" -c:t 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: 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`

Pulldown 处理: 见 x264 教程完整版

x264 HDR 设置参数:

x264 --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(?,1)

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

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

-- cll <和 master-display 的 L 最大值一样>

色域标识 --colormatrix <照源, 例: gbr bt709 fcc bt470bg smpte170m YCgCo bt2020nc bt2020c smpte2085 ictcp>

色域转换 --transfer <照源, 例: gbr bt709 fcc bt470bg smpte170m YCgCo bt2020nc bt2020c smpte2085 ictcp>

通用·简单

砍了全部自定义项目，方便急用但降低了特定画面的压缩率

前瞻进程 --rc-lookahead 90 --bframes 12 --b-adapt 2

动态-帧内搜索 --me umh --subme 9 --merange 48 --no-fast-pskip --direct auto --weightb

帧控-参考 --keyint 360 --min-keyint 5--ref 3

量化 --crf 20 --qpmin 9 --chroma-qp-offset -2

自适量 --aq-mode 3 --aq-strength 0.7 --trellis 2

环滤/RDO --deblock 0:0 --psy-rd 0.77:0.22 --fgo 10

降噪 --nr 8

色彩范围 --fullrange<非 7mod x264 用，检查源视频是否使用完整色彩范围>

动-帧快速搜索 --me hex --subme 8 --merange 32 --direct auto --weightb

参考冗余优先 --sliced-threads <降低 CPU 占用，减速但时域复杂画面的压缩率可能提高，参考错误降低>

放/裁/边/降噪 --vf crop:左,上,右,下/resize:缩放后宽,缩放后高,,,,bicubic/pad:左,上,右,下,直接宽,直接高

滤镜 /hqdn3d:1,1,1,1.5

划区压制 --zones 0,<片头 OP 结束帧>,crf=30 --zones<片尾 ED 开始帧>,<片尾 ED 结束帧>,crf=30

压制范围 --seek 从第<>帧开始压 --frame 压制<>帧后停止 --fps 元数据没写多少时手动指定帧数

α——x264 CLI 命令

- x264.exe --rc-lookahead 90 --bframes 12 --b-adapt 2 --me umh --subme 9 --merange 48 --no-fast-pskip --direct auto --weightb --keyint 360 --min-keyint 5 --ref 3 --crf 20 --qpmin 9 --chroma-qp-

offset -2 --aq-mode 3 --aq-strength 0.7 --trellis 2 --deblock 0:0 --psy-rd 0.77:0.22 --fgo 10 --nr 8 --output ".\输出.mp4" ".\导入.mp4"

β——libx264 私有 CLI，兼容 libav

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -c:v libx264 -x264-params "rc-lookahead=90:bframes=12:b-adapt=2:me=umh:subme=9:merange=48:fast-pskip=0:direct=auto:weightb=1:keyint=360:min-keyint=5:ref=3:crf=20:qpmin=9:chroma-qp-offset=-2:aq-mode=3:aq-strength=0.7:trellis=2:deblock=0,0:psy-rd=0.77,0.22:nr=4" -fps_mode passthrough -c:a copy ".\输出.mp4"

γ——libx264 私有 CLI 命令 上面的 -x264-params 改成 -x264opts，但没啥意义

δ——libx264 ffmpeg CLI 命令(因版权而重写参数名；有的 ffmpeg 发行可能缺参数导致以下命令报错)

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -c:v libx264 -bf 12 -b_strategy 2 -me_method umh -subq 9 -me_range 48 -flags2 -fastpskip -directpred 3 -flags2 +wpred -g 360 -keyint_min 5 -refs 3 -crf 20 -qmin 9 -chromaoffset -2 -aq-mode 3 -aq-strength 0.7 -trellis 2 -deblockalpha 0 -deblockbeta 0 -psy-rd 0.77:0.22 -nr 4 -flags2 +bpyramid -fps_mode passthrough -c:a copy ".\输出.mp4"

通用·标准

配置起来慢些但自定义范围广。由于 x264 参数不多，所以这套参数足以涵盖高画质高压-录像-动画情形

前瞻进程 --rc-lookahead <3×帧率> --bframes 12 --b-adapt 2

动态-帧内搜索 --me umh --subme <电影 10~11，动漫 9 (11 仅 7mod)> --merange <快 20，高压 48, 4 的倍数> --no-fast-pskip --direct auto --weightb

帧控-参考 --keyint <8~10×帧率> --min-keyint <1 增加 IDR, 5 常用> --ref 3

量化 --crf 19 --qpmin 9 --chroma-qp-offset <常用-1~-2，动漫-3~-5>

自适应量化 --aq-mode 3 --aq-strength <一般 0.7，原画 1.1> --trellis 2

环滤/RDO --deblock <一般 0:0，原画-1:-1> --psy-rd<动漫 0.4~.6:0.1~.15，录像 0.7~1.3:0.12~.2> --fgo 12

CRF-VBR 压缩 --nal-hrd --vbv-bufsize <最大 kbps 每秒> --vbv-maxrate <bufsize 倍数的 kbps>

色彩范围 --fullrange<非 7mod x264 用，检查源视频是否使用完整色彩范围>

参考冗余优先 --sliced-threads <降 CPU 占用，减速但时域复杂画面的压缩率可能提高，参考错误降低>

放/裁/边/降噪 --vf crop:左,上,右,下/resize:缩放后宽,缩放后高,,,,bicubic/pad:左,上,右,下,直接宽,直接高

参数划区压制 /hqdn3d:1.1,1.1,1.1,1.1

压制范围 --zones 0,<片头 OP 结束帧>,crf=30 --zones<片尾 ED 开始帧>,<片尾 ED 结束帧>,crf=30

α——x264 CLI 命令

- x264.exe --rc-lookahead ◯ --me umh --bframes 12 --b-adapt 2 --subme ◯ --merange ◯ --no-fast-pskip --direct auto --weightb --keyint ◯ --min-keyint ◯ --ref 3 --crf 19 --qpmin 9 --

chroma-qp-offset ◯ --aq-mode 3 --aq-strength ◯ --trellis 2 --deblock ◯ --psy-rd ◯ --fgo 12 --output ".\输出.mp4" ".\导入.mp4"

β——libx264 私有 CLI，兼容 libav，不支持 fgo

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -c:v libx264 -x264-params "rc-lookahead=◯:me=umh:bframes=12:b-adapt=2:subme=◯:merange=◯:fast-pskip=0:direct=auto:weightb=1:keyint=◯:min-keyint=◯:ref=3:crf=19:qpmin=9:chroma-qp-offset=◯:aq-mode=3:aq-strength=◯:trellis=2:deblock=0,-1:psy-rd=◯,◯:nr=4" -fps_mode passthrough -c:a copy ".\输出.mp4"

γ——libx264 私有 CLI，不支持 fgo 上面的 -x264-params 改成 -x264opts，但没啥意义

δ——libx264 ffmpeg CLI 命令(因版权而重写参数名；有的 ffmpeg 发行可能缺参数导致以下命令报错)

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -c:v libx264 -me_method umh -subq ◯ -me_range ◯ -flags2 -fastpskip -directpred 3 -flags2 +wpred -g ◯ -keyint_min ◯ -bf 12 -b_strategy 2 -refs 3 -crf 19 -qmin 9 -chromaoffset ◯ -aq-mode 3 -aq-strength ◯ -trellis 2 -deblockalpha 0 -deblockbeta -1 -psy-rd ◯:◯ -nr 4 -flags2 +bpyramid -fps_mode passthrough -c:a copy ".\输出.mp4"

剪辑素材存档

加强无损压缩，降低有损压缩，增加 IDR 帧数量. 建议 YUV4:2:2 或 4:4:4 8~10bit

前瞻进程 --bframes 12 --b-adapt 2

动态-帧内搜索 --me esa --subme <电影 10~11，动漫 9 (11 仅 7mod)> --merange <快速 40，高压 48> --no-fast-pskip --direct auto --weightb

帧控-参考 --keyint <5~8×帧率> --min-keyint 1 --ref 3 --sliced-threads

量化-主控 --crf 17 --tune grain

自适-RDO --trellis 2 --fgo 15

划区压制 --zones 0,<片头/OP 结束帧>,crf=30 --zones<片尾/ED 开始帧>,<片尾/ED 结束帧>,crf=30

压制范围 --seek 从第<>帧开始压 --frame 压制<>帧后停止 --fps 元数据没写多少时手动指定帧数

α——x264 CLI 命令

- x264.exe --bframes 12 --b-adapt 2 --me esa --subme ◯ --merange ◯ --no-fast-pskip --direct auto --weightb --keyint ◯ --min-keyint 1 --ref 3 --crf 17 --tune grain --trellis 2 --fgo 15 --output ".\输出.mp4" ".\导入.mp4"

β——libx264 私有 CLI，兼容 libav，不支持 fgo

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -c:v libx264 -x264-params "me=esa:subme=◯:merange=◯:fast-pskip=0:direct=auto:weightb=1:keyint=◯:min-keyint=1:bframes=12:b-adapt=2:ref=3:crf=17:trellis=2" -fps_mode passthrough -c:a copy ".\输出.mp4"

γ——libx264 私有 CLI，不支持 fgo 上面的 -x264-params 改成 -x264opts，但没啥意义

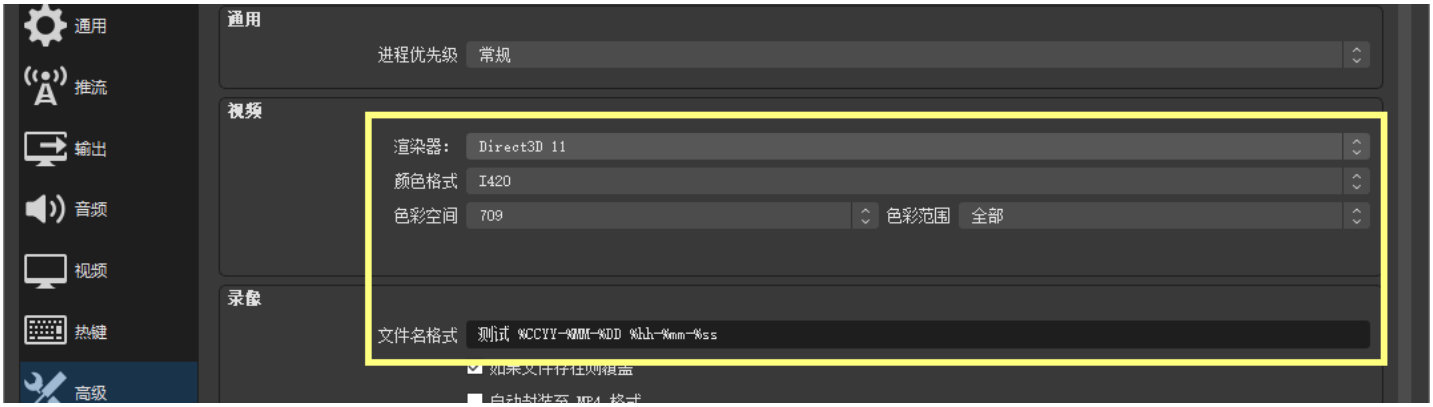
δ——libx264 ffmpeg CLI 命令(因版权而重写参数名；有的 ffmpeg 发行可能缺参数导致以下命令报错)

- ffmpeg.exe -loglevel 16 -hwaccel auto -y -hide_banner -i ".\导入.mp4" -c:v libx264 -bf 12 -b_strategy 2 -me_method esa -subq ◯ -me_range ◯ -flags2 -fastpskip -directpred 3 -flags2 +wpred -g ◯ -keyint_min 1 -refs 3 -crf 17 -tune grain -trellis 2 -flags2 +bpyramid -fps_mode passthrough -c:a copy ".\输出.mp4"

OBS 录屏

打开 OBS→文件→设置，一般录屏的预设置如下：

高级：颜色格式-色彩空间-色彩范围-文件名格式，一般除文件名外和图中保持一致就行



热键：开始/停止录制，看个人习惯，如 Ctrl+Shift+X

帧率:一般设在 57 到 58，损失一点帧数换取更低 CPU 占用和一点画质

分辨率:一般设在 1920x1040，去掉一点宽，道理同上

音频:奇怪的是占用 CPU 更低的电平和采样峰值反而要更清晰...

参数:设定输出路径，然后将下面需要的参数粘贴到这里

录屏 10fps 高压 - 音乐/助眠/棋牌/B-roll

码率控制：CRF
关键帧间隔：3（这里是秒）
CPU 使用预设：slow
CRF：20
Profile：high
tune：stillimage
设定-高级-色彩空间-色度采样：i420
参数：me=hex subme=7 chroma_me=0 bframes=12 b-adapt=1 ref=2 aq-mode=0 aq-strength=0 deblock=0,-1 trellis=2 chroma_qp_offset=-4 cabac=1 psy=0 opencl=1 fgo=15

- 由于画面变化极小也没有突发情况，所以用不到 VBR 模式（VBV 参数）

录屏 28fps 高压 - 聊天/演奏/小游戏/装机/发布会

码率控制:	VBR	关键帧间隔:	8（这里是秒）
比特率:	9000kbps	CPU 使用预设:	medium
缓冲大小:	默认	Profile:	main
CRF:	19	tune:	film
设定-高级-色彩空间-色度采样:			i420

参数: me=umh subme=6 chroma_me=1 bframes=5 b-adapt=1 ref=2 aq-mode=1 aq-strength=0.7 deblock=0,0 trellis=2 direct=temporal opencl=1 fgo=15

- 画面有动态，且以上情况里的背景一定是静态的，所以动搜增强了很多
- 画面可能有突发情况，所以用了 VBR 模式
- 增加了 IDR 帧时间来给码率让位

录屏 55fps 低压 - FPS-STG-赛车-3A 游戏/演唱会

码率控制:	VBR/ABR	关键帧间隔:	5（这里是秒）
比特率:	35000kbps	CPU 使用预设:	veryfast
缓冲大小:	3500	Profile:	无
CRF:	18	tune:	film

参数: me=hex subme=4 me_range=12 chroma_me=0 bframes=3 b-adapt=1 ref=3 aq-mode=2 aq-strength=0.9 deblock=0,0 trellis=0 deadzone-inter=8 deadzone-intra=5 direct=temporal cabac=0 opencl=1 nr=10 fgo=15

- deadzone-inter=8 deadzone-intra=5 这两个参数必须在 trellis<2 时使用，否则画面会变得很脏
- 为进一步限制码率，使用了降噪 nr 功能
- 本方案没有考虑游戏的突发高占用情况，可能会在大量 AI，大量效果等情况下造成卡顿；虽然这是游戏本身就把 CPU 占满的情况

录屏 100FPS - x264 低压

码率控制:	CRF/ABR	关键帧间隔:	6（秒）
CPU 使用预设:	fast	CRF/比特率:	18/9000 kbps（或直播平台/网速上限）
Profile:	无	tune:	无

参数: me=hex me_range=12 subme=3 chroma_me=0 bframes=3 b-adapt=1 ref=3 aq-mode=0 psy=0 mbtree=0 cabac=0 mixed_refs=0 deadzone-inter=8 deadzone-intra=5 deblock=0,0 trellis=0 direct=temporal ref=3 opencl=1 fgo=15

- 如果以上参数录制的视频仍然卡顿，CPU 占用高的话，就需要限制渲染帧率，开 fps_mode，降低画质设定，升级/超频 CPU/内存，用另一台电脑压制，用采集卡，用图灵 RTX NVENC 硬件编码，用帕斯卡

GTX Nvenc 硬件编码，用 x264 ABR 模式等其它录制方案

- 以上方案列表按照录屏画质，从高到低排序
- Potplayer 播放器的视频色彩解析有问题，建议用 mpv 预览

Twitch 3A 直播压制机 VBR 1080p60<8000kbps

4:4:4 D3D11 Limited; 超频 R7 1800x+3000Mhz CL15 2x4G *已验证*:

码率控制:	VBR	关键帧间隔:	2 (秒, Twitch 标准)
比特率:	7800kbps	CPU 使用预设:	veryfast
缓冲大小:	3500	Profile:	无
CRF:	22	tune:	无

参数: me=umh subme=5 me_range=16 chroma_me=0 bframes=5 b-adapt=1 ref=3 deblock=0,0 trellis=2 aq-mode=1 aq-strength=1.1 cabac=1

4:4:4 D3D11 Limited; 4.0Ghz R7 2700x+3000Mhz CL15 2x4G *未验证*:

码率控制:	VBR	关键帧间隔:	2 (秒, Twitch 标准)
比特率:	7800kbps	CPU 使用预设:	fast
缓冲大小:	3500	Profile:	无
CRF:	22	tune:	无

参数: me=umh subme=7 me_range=16 chroma_me=0 bframes=5 b-adapt=1 ref=3 deblock=0,0 trellis=2 aq-mode=2 aq-strength=1.1 direct=auto cabac=1 nr=15 chroma_qp_offset=-2

4:4:4 D3D11 Limited;默频 R7 3800x+3000Mhz CL15 2x8G CR-2T *未验证*:

码率控制:	VBR	关键帧间隔:	2 (秒, Twitch 标准)
比特率:	7830kbps	CPU 使用预设:	medium
缓冲大小:	3500	Profile:	无
CRF:	22	tune:	无

参数: me=umh subme=9 me_range=16 chroma_me=0 bframes=6 b-adapt=2 ref=3 deblock=0,0 trellis=2 aq-mode=2 aq-strength=1.1 direct=auto cabac=1 nr=15 chroma_qp_offset=-4 b_deterministic=0

4:4:4 D3D11 Limited; 4.2Ghz R7 5800x+3000Mhz CL15 2x16G CR-1T *未验证*:

码率控制:	VBR	关键帧间隔:	2 (秒, Twitch 标准)
比特率:	7830kbps	CPU 使用预设:	medium
缓冲大小:	3500	Profile:	无
CRF:	22	tune:	无

参数: me=umh subme=9 me_range=20 chroma_me=0 bframes=7 b-adapt=2 ref=3 deblock=0,-1 trellis=2 aq-mode=3 aq-strength=1.1 direct=auto cabac=1 nr=15 chroma_qp_offset=-4 b_deterministic=0