**腾讯明眸·极速高清SDK（中心鉴权） 部署指引**

目录

[一. TSC SDK文件包内容说明 3](#_Toc105773893)

[二. TSC SDK插件链入ffmpeg 4](#_Toc105773894)

[三. TSC执行配置 8](#_Toc105773895)

[四. TSC 执行参数建议 8](#_Toc105773896)

[1. Ten265 8](#_Toc105773897)

[2. Tenav1 10](#_Toc105773898)

[3. Ten264 11](#_Toc105773899)

[五. TSC SDK 前置处理参数详细说明 13](#_Toc105773900)

[六. Ten265编码器常用参数说明 14](#_Toc105773901)

[七. Tenav1编码器常用参数说明 16](#_Toc105773902)

[八. Ten264 编码器常用参数说明 17](#_Toc105773903)

[九．机型配置建议 19](#_Toc105773904)

[十. 对比数据： 20](#_Toc105773905)

[1.Ten265, X265 ,X264对比 20](#_Toc105773906)

[2. TenAV1 , SVTAV1对比 21](#_Toc105773907)

[3. Ten264 ，X264对比 21](#_Toc105773908)

# 一. TSC SDK文件包内容说明



1. **sdk\_config 文件：**

远端鉴权配置文件

**2. ffmpeg**

已经嵌入TSC SDK插件的ffmpeg, 可以用来快速测试编码压缩效果

**3. fflib\_gpu**

TSC SDK画质处理的依赖库

**4. src lib**

TSC SDK 插件文件及插件库， 插件链接到用户自己的ffmpeg时使用

**5. tenmodel**

TSC SDK画质处理使用的AI模型

# 二. TSC SDK插件链入ffmpeg



（若使用提供的ffmpeg binary直接进行测试，只需将依赖库添加环境变量：export LD\_LIBRARY\_PATH=./fflib\_gpu/:$LD\_LIBRARY\_PATH，无需以下步骤）

（建议使用ffmpeg4.2）

**1. 将fflib\_gpu和lib拷贝到FFmpeg 目录下**

**2. 进入到FFmpeg目录下，添加环境变量：**

export LD\_LIBRARY\_PATH=./fflib\_gpu/:$LD\_LIBRARY\_PATH

**3. 按照如下步骤更改FFmpeg 对应文件**

（1）在libavcodec/Makefile 添加：

OBJS-$(CONFIG\_LIBTEN265\_ENCODER) += libten265.o

OBJS-$(CONFIG\_LIBTENAV1\_ENCODER) += libtenav1.o

OBJS-$(CONFIG\_LIBTEN264\_ENCODER) += libten264.o

如图：



（2）在libavcodec/allcodecs.c 添加：

extern AVCodec ff\_libten265\_encoder;

extern AVCodec ff\_libtenav1\_encoder;

extern AVCodec ff\_libten264\_encoder;

如图：



（3）在libavfilter/Makefile 添加：

OBJS-$(CONFIG\_TENFILTER\_FILTER) += vf\_tenfilter.o

如图：



（4）在libavfilter/allfilters.c 添加：

extern AVFilter ff\_vf\_tenfilter;

如图：



（5）如果使用FFmpeg5.0编译, 则在libavutil/version.h 添加：

 #define FFMPEG5\_SDK\_VERSION

如图：



**4. 拷贝src 目录下的libten265.c 到ffmpeg libavcodec路径下**

**5. 拷贝src 目录下的libtenav1.c 到ffmpeg libavcodec路径下**

**6. 拷贝src 目录下的libten264.c到ffmpeg libavcodec路径下**

**7. 拷贝src 目录下的vf\_tenfilter.c 到ffmpeg libavfilter路径下**

**8. 如果需要支持GPU解码**

注意：A10卡对应的软件和驱动版本号分别是：tensorrt ：8.2.5， cuda：11.4 ，driver ：470.63.01

(1)安装必要依赖库：

yum install build-essential yasm cmake libtool libc6 libc6-dev unzip wget libnuma1 libnuma-dev

(2) 编译安装nv-codec-headers库:

git clone <https://git.videolan.org/git/ffmpeg/nv-codec-headers.git>

cd nv-codec-headers

make install

**9. 编译（AV1需要FFmpeg4.2及以上）：**

1) 静态编译：./configure --enable-cross-compile --enable-gpl --enable-nonfree --enable-openssl --extra-cflags='-I./lib' --extra-ldflags='-L./fflib\_gpu/ -L./lib/' --extra-libs="-ltscsdk\_center -lstdc++ -ltvp -lm -ldl -lrt -lpthread -lz"

2）动态编译：./configure --enable-cross-compile --enable-gpl --enable-nonfree --enable-openssl --enable-shared --extra-cflags='-I./lib' --extra-ldflags='-L./fflib\_gpu/ -L./lib/' --extra-libs="-ltscsdk\_center -lstdc++ -ltvp -lm -ldl -lrt -lpthread -lz"

3）静态编译（支持GPU解码）: PKG\_CONFIG\_PATH="/usr/local/lib/pkgconfig " ./configure --enable-gpl --enable-nonfree --enable-cuda --enable-cuvid --enable-nvenc --enable-libnpp --enable-cross-compile --enable-openssl --extra-cflags='-I./lib -I/usr/local/cuda-11.4/include/' --extra-ldflags='-L./fflib\_gpu/ -L./lib/ -L/usr/local/cuda-11.4/lib64/' --extra-libs="-ltscsdk\_center -lstdc++ -ltvp -lm -ldl -lrt -lpthread -lz"

4）动态编译（支持GPU解码）: PKG\_CONFIG\_PATH="/usr/local/lib/pkgconfig " ./configure --enable-gpl --enable-nonfree --enable-cuda --enable-cuvid --enable-nvenc --enable-libnpp --enable-cross-compile --enable-openssl --enable-shared --extra-cflags='-I./lib -I/usr/local/cuda-11.4/include/' --extra-ldflags='-L./fflib\_gpu/ -L./lib/ -L/usr/local/cuda-11.4/lib64/' --extra-libs="-ltscsdk\_center -lstdc++ -ltvp -lm -ldl -lrt -lpthread -lz"

注：动态编译完后，需要将ffmpeg 生成动态库放入到环境变量

**10. make -j8**

# 三. TSC执行配置

1）切换到root模式

2）添加环境变量：export LD\_LIBRARY\_PATH=./fflib\_gpu/:$LD\_LIBRARY\_PATH

3）保证网络与中心节点互通

4) 鉴权中心配置

sdk\_config, 访问中心鉴权中心配置：



net\_card : 为本机使用的网卡

dst\_ip： 为鉴权中心IP，支持多个鉴权中心高可用配置

注意：鉴权配置文件支持指定路径（如不指定，默认路径和ffmpeg bin 放在一个目录），再命令行中通过参数-sdk\_config 指定，如图：



# 四. TSC 执行参数建议

注：1.中心模式，执行前需要按照步骤三，配置相关执行环境  
 2.使用锐化/去压缩伪影/hdr转sdr时, 每个filter只能启用一个功能，可以通过多个tenfilter串联起来

## 1. Ten265

* 1. **支持GPU的环境，可以开启AI画质增强：**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -vf tenfilter=af=weak:log=info:gpu=01:mdir=./tenmodel,tenfilter=unsharp\_size=7:unsharp\_amount=0.5 -c:v libten265 -crf 28 -preset 2 ./test-sharp.mp4

* 1. **不支持GPU的环境，可以参考参数**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -vf tenfilter=unsharp\_size=7:unsharp\_amount=0.5 -c:v libten265 -crf 28 -preset 2 ./test-sharp.mp4

* 1. **hdr转换为sdr，可以开启hdr2sdr参数：**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -vf tenfilter=hdr2sdr=1 ./test.mp4

* 1. **同时开启hdr2sdr和锐化：**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -vf tenfilter=hdr2sdr=1,tenfilter=unsharp\_size=11:unsharp\_amount=1.1 ./test.mp4

* 1. **常见码率控制模式**

**vbv+crf**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -vf tenfilter=unsharp\_size=7:unsharp\_amount=0.5 -c:v libten265 -ten265-params preset=1:crf=28:ltr=0:vbv-maxrate=2000:vbv-bufsize=4000:rc=3 ./test-sharp.mp4

**恒定质量模式**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -vf tenfilter=unsharp\_size=7:unsharp\_amount=0.5 -c:v libten265 -ten265-params preset=1:crf=28:ltr=0 ./test-sharp.mp4

**abr模式**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -vf tenfilter=unsharp\_size=7:unsharp\_amount=0.5 -c:v libten265 -ten265-params preset=1:ltr=0:bitrate=2000 ./test-sharp.mp4

## 2. Tenav1

* 1. **常见码率控制模式**

**vbv+crf**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -vf tenfilter=unsharp\_size=7:unsharp\_amount=0.5 -c:v libtenav1 -tenav1-params preset=5:crf=28:vbv-maxrate=2000:vbv-bufsize=4000 ./test-sharp.mp4

**恒定质量模式**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -vf tenfilter=unsharp\_size=7:unsharp\_amount=0.5 -c:v libtenav1 -tenav1-params preset=5:crf=28 ./test-sharp.mp4

**abr模式**

./ffmpeg -i 1080P-5M-23fps-1min.mp4  -c:v libtenav1 -tenav1-params preset=9:bitrate=900 ./test-sharp.mp4

**2-pass模式**

1）1pass: ./ffmpeg -i 1080P-5M-23fps-1min.mp4 -c:v libtenav1 -tenav1-params preset=10:pass=1:twopass\_path=sec1:crf=28 ./test.mp4  
 2) 2pass: ./ffmpeg -i 1080P-5M-23fps-1min.mp4 -c:v libtenav1 -tenav1-params preset=10:pass=2:twopass\_path=sec1:bitrate=2000:stat\_keep=1 ./test.mp4

## 3. Ten264

* 1. **常见码率控制模式**

**vbv+crf**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -c:v libten264 -preset slower -ten264opts crf=24:vbv-maxrate=2000:vbv-bufsize=4000 ./test-sharp.mp4

**恒定质量模式**

./ffmpeg -i 1080P-5M-23fps-1min.mp4 -c:v libten264 -preset slower -ten264opts crf=24 ./test-sharp.mp4

# 五. TSC SDK 前置处理参数详细说明

* 去压缩伪影：

af=weak/average/strong/auto，对应弱/中/强/自动级别的去压缩伪影

* 人脸保护：

face\_protect\_enable=0/1控制人脸保护逻辑， face\_af\_ratio=0.5人脸区域去噪弱化系数， face\_sp\_ratio人脸区域锐化强化系数

* 锐化增强：

unsharp\_size 控制大小， unsharp\_amount 控制强度

* hdr转换为sdr：

hdr2sdr 控制hdr转sdr开关，0：关闭，1开启，默认关闭

* 指定设备：

gpu=01，指定使用gpu0和gpu1双卡推理， 如未使用af，face\_protect则无需指定

* 日志级别：

log=info/warning/error，指定info打印详细日志信息，如耗时信息

* 模型路径：

mdir=./tve，模型的路径，如：mdir=./tenmodel。 如未使用af，face\_protect则无需指定

# 六. Ten265编码器常用参数说明

**--preset**

指定编码器编码参数集合的配置 -1: ripping 0: placebo; 1:veryslow; 2: slower; 3: slow; 4: universal; 5 medium; 6: fast; 7: faster; 8:veryfast; 9 superfast

**--rc**

码率控制方式 0:CQP 1:ABR\_VBV 2:ABR 3:CRF\_VBV 4:CRF

**--bitrate**

ABR模式下面的输出视频的码率

**--crf**

CRF模式下面的CRF数值，取值范围：[1,51]

**--aq-mode**

0：关闭aqmode 1：开启aqmode 2：基于方差的aqmode 3：基于方差的aqmode并且偏向于暗场景（默认是2会产生更好的ssim结果）

**--vbv-maxrate**

vbv的最大码率（默认情况下这个数值和配置的码率相同）

**--vbv-bufsize**

vbv缓冲buffer的大小（默认情况下这个数值是配置的码率的四倍）

**--rc-lookahead**

lookahead长度 （这个最小是24，在直播场景中这个数值越大，编码延迟越高）

**--scenecut**

场景切换阈值，取值范围[0,100]（0:关闭，默认是开启的，一般不建议关闭）

**--open-gop**

是否开启open gop ，0：关闭，1：开启（默认是开启，在直播场景中为了支持随机接入，建议关闭这个功能）

**--keyint**

关键帧最长间隔 （默认是256，可以根据实际业务情况配置，需要大于50且是8的倍数）

**--ltr**

是否要支持长期参考帧 0：关闭，1: 开启（默认是开启的，如果播放HEVC视频的硬件设备比较差，建议关闭这个功能）

**--pool-threads**

WPP使用的线程池的线程数（默认和CPU的核数相同，如果想减少CPU占用，可以降低这个数目）

# 七. Tenav1编码器常用参数说明

**--log\_level**

0-2,

**--gop\_size**

编码帧结构的GOP大小（默认是8，只有4，8，16有实际意义，一般不推荐修改这个数值）

**--wpp**

是否要开启WPP的功能，0：关闭，1：开启（默认是开启，一般不推荐修改这个数值）

**--preset**

指定编码器编码参数集合的配置,影响编码速度 -1: ripping 0: placebo; 1:veryslow; 2: slower; 3: slow; 4: universal; 5 medium; 6: fast; 7: faster; 8: veryfast, 9: superfast, 10: ultrafast

**--crf**

码率控制方式 crf [0-51], 码率大致可以与V265对应

**--vbv\_maxrate**

vbv的最大码率（默认情况下这个数值和配置的码率相同）

**--vbv\_bufsize**

vbv缓冲buffer的大小（默认情况下这个数值是配置的码率的四倍）

**--lag\_in\_frames**

lookahead长度，可以为任意正整数 （在直播场景中这个数值越大，编码延迟越高）

**--enable\_scenecut**

是否开启场景切换，取值范围[0,1,2]，0：关闭，1和2：开启，2适用于简单场景（默认是开启的，一般不建议关闭）

**--scenecut\_threshold**

场景切换阈值，取值范围[0,100]（0:关闭，默认是开启的，一般不建议关闭）

**--open\_gop**

是否开启open gop， 0：关闭，1：开启（默认是开启，在直播场景中为了支持随机接入，`建议`关闭这个功能）

**--keyint**

关键帧最长间隔 （默认是256，可以根据实际业务情况配置，需要大于50且是8的倍数）

**--pool\_threads**

WPP使用的线程池的线程数（默认和CPU的核数相同，如果想减少CPU占用，可以降低这个数目）

# 八. Ten264 编码器常用参数说明

**--preset**

指定编码器编码参数集合的配置 – 0："ultrafast", 1："superfast", 2："veryfast", 3:"faster", 4:"fast", 5:"medium", 6:"slow", 7:"slower", 8:"veryslow",9: "placebo", 0 };

**--bitrate**

ABR模式下面的输出视频的码率

**--crf**

CRF模式下面的CRF数值

**--aq-mode**

0：关闭aqmode 1：开启aqmode 2：基于方差的aqmode 3：基于方差的aqmode并且偏向于暗场景（默认是2会产生更好的ssim结果）

**--vbv-maxrate**

vbv的最大码率（默认情况下这个数值和配置的码率相同）

**--vbv-bufsize**

vbv缓冲buffer的大小（默认情况下这个数值是配置的码率的四倍）

**--rc-lookahead**

lookahead长度 （这个最小是24，在直播场景中这个数值越大，编码延迟越高）

**--scenecut**

是否开启场景切换（默认是开启的，一般不建议关闭）

**--keyint**

关键帧最长间隔 （默认是256，可以根据实际业务情况配置，一般配置成2~5s的时间间隔的帧数）

**--threads**

使用的线程池的线程数

**--lookahead-threads**

预分析，lookahead使用线程数

**--profile**

"baseline", "main", "high", "high422", "high444"

# 九．机型配置建议

1.如果需要GPU前处理能力，建议GPU使用Tesla T4卡，驱动版本：418.67  
2.CPU 需要支持avx2 指令集，CPU核数建议16核以上（如果同时有GPU T4卡，为了使GPU和CPU的利用率比较均衡，建议一张T4卡搭配40核），CPU基频建议2GHz以上；

3.内存建议32g以上

# 十. 对比数据：

## 1.Ten265, X265 ,X264对比



## 2. TenAV1 , SVTAV1对比



## 3. Ten264 ，X264对比



|  |  |  |
| --- | --- | --- |
| 版本 | 日期 | 修改说明 |
| V\_1.0.0 | 2021.12.13 | 添加AV1样例 |
| V\_1.0.1 | 2021.1.4 | 支持H264 |
| V\_1.0.2 | 2021.1.5 | 添加Ten264数据对比 |

|  |  |  |
| --- | --- | --- |
| 版本 | 日期 | 修改说明 |
| V\_1.0.0 | 2021.12.13 | 添加AV1样例 |