个人资料

读好

微博

加好友 发纸条

写留言 加关注

博客等级：
博客积分：**0**
博客访问：**31,113**
关注人气：**17**
获赠金笔：**0**支
赠出金笔：**0**支
荣誉徽章：

相关博文

更多>>

推荐博文

人人都喜欢看到林妙可失败？

《朗读者》比《见字如面》如何？

康熙爷的黄马褂为什么落到了民间

你有多久没清洗梳子

没事，林妙可，被群嘲的锅不该你

从林妙可艺考落榜反思“童星热”

20170221【早评】中级行

正文

字体大小：大 中 小

ffmpeg转码参数 (2011-11-10 17:51:09)

转 载 ▼

标签： ffmpeg 转码 it 分类： 技术

1．视频音频格式转换

Ffmpeg能使用任何支持的格式和协议作为输入：

*比如你可以输入YUV文件：ffmpeg -i /tmp/test%d.Y /tmp/out.mpg

它将要使用如下文件：

/tmp/test0.Y, /tmp/test0.U, /tmp/test0.V,

/tmp/test1.Y, /tmp/test1.U, /tmp/test1.V,等等…

*你能输入原始的YUV420P文件：ffmpeg -i /tmp/test.yuv /tmp/out.avi

原始的YUV420P文件包含原始的YUV极性，每帧以Y平面开始，跟随U和V平面，它们是Y平面水平垂直的一半分辨率

*你能输出原始的YUV420P文件

ffmpeg -i mydivx.avi -o hugefile.yuv

*你能设置几个输入文件和输出文件

ffmpeg -i /tmp/a.wav -s 640x480 -i /tmp/a.yuv /tmp/a.mpg 上面的命令行转换音频文件a.wav和原始的YUV 视频文件 a.yuv到mpeg文件a.mpeg

*你也能同时转换音频和视频

ffmpeg -i /tmp/a.wav -ar 22050 /tmp/a.mp2

上面的命令行转换a.wav的采样率到22050HZ并编码为mpeg音频

*你也能同时编码到几种格式并且在输入流和输出流之间建立映射

ffmpeg -i /tmp/a.wav -ab 64 /tmp/a.mp2 -ab 128 /tmp/b.mp2 -map 0:0 -map 0:0

上面的命令行转换一个64Kbits 的a.wav到128kbits的a.mp2 ‘-map file:index’在输出流的顺序上定义了那一路输入流是用于每一个输出流的，

转码解密的VOB:

ffmpeg -i snatCh_1.vob -f avi -vCOdeC mpeg4 -b 800 -g 300 -bf 2 -aCOdeC mp3 -ab 128 snatCh.avi

上面的命令行将vob的文件转化成avi文件，mpeg4的视频和mp3的音频。注意命令中使用了B帧，所以mpeg4流是divx5兼容的。GOP大小是300意味着29.97帧频下每10秒就有INTRA帧。该映射在音频语言的DVD转码时候尤其有用

2． Ffmpeg使用语法

ffmpeg [[options][`-i’ input_file]]... {[options] output_file}...

如果没有输入文件，那么视音频捕捉就会起作用。

作为通用的规则，选项一般用于下一个特定的文件。如果你给 -b 64选项，改选会设置下一个视频速率。对于原始输入文件，格式选项可能是需要的。

缺省情况下，ffmpeg试图尽可能的无损转换，采用与输入同样的音频视频参数来输出。

3．选项

a) 通用选项

-L liCense

-h 帮助

-fromats 显示可用的格式，编解码的，协议的。。。

-f fmt 强迫采用格式fmt

-I filename 输入文件

-y 覆盖输出文件

-t duration 设置纪录时间 hh:mm:ss[.xxx]格式的记录时间也支持

-ss position 搜索到指定的时间 [-]hh:mm:ss[.xxx]的格式也支持

-title string 设置标题

-author string 设置作者

-copyright string 设置版权

-COmment string 设置评论

-target type 设置目标文件类型(vCd,svCd,dvd) 所有的格式选项（比特率，编解码以及缓冲区大小）自动设置 ，只需要输入如下的就可以了：

ffmpeg -i myfile.avi -target vCd /tmp/vCd.mpg

-hq 激活高质量设置

-itsoffset offset 设置以秒为基准的时间偏移，该选项影响所有后面的输入文件。该偏移被加到输入文件的时戳，定义一个正偏移意味着相应的流被延迟了 offset秒。 [-]hh:mm:ss[.xxx]的格式也支持

b) 视频选项

-b bitrate 设置比特率，缺省200kb/s

-r fps 设置帧频 缺省25

-s size 设置帧大小 格式为WXH 缺省160X128.下面的简写也可以直接使用：

SqCif 128X96 qCif 176X144 Cif 252X288 4Cif 704X576

-aspeCt aspeCt 设置横纵比 4:3 16:9 或 1.3333 1.7777

-Croptop size 设置顶部切除带大小 像素单位

-Cropbottom size -Cropleft size -Cropright size

-padtop size 设置顶部补齐的大小 像素单位

-paom size -padleft size -padright size -padCOlor COlor 设置补齐条颜色(hex,6个16进制的数，红:绿:兰排列，比如 000000代表黑色)

-vn 不做视频记录

-bt toleranCe 设置视频码率容忍度kbit/s

-maxrate bitrate设置最大视频码率容忍度

-minrate bitreate 设置最小视频码率容忍度

-bufsize size 设置码率控制缓冲区大小

-vCOdeC COdeC 强制使用COdeC编解码方式。 如果用copy表示原始编解码数据必须被拷贝。

-sameq 使用同样视频质量作为源（VBR）

-pass n 选择处理遍数（1或者2）。两遍编码非常有用。第一遍生成统计信息，第二遍生成精确的请求的码率

-passlogfile file 选择两遍的纪录文件名为file

-

C)高级视频选项

-g gop_size 设置图像组大小

-intra 仅适用帧内编码

-qsCale q 使用固定的视频量化标度(VBR)

-qmin q 最小视频量化标度(VBR)

-qmax q 最大视频量化标度(VBR)

-qdiff q 量化标度间最大偏差 (VBR)

-qblur blur 视频量化标度柔化(VBR)

-qCOmp COmpression 视频量化标度压缩(VBR)

-rC_init_Cplx COmplexity 一遍编码的初始复杂度

-b_qfaCtor faCtor 在p和b帧间的qp因子

-i_qfaCtor faCtor 在p和i帧间的qp因子

-b_qoffset offset 在p和b帧间的qp偏差

-i_qoffset offset 在p和i帧间的qp偏差

-rC_eq equation 设置码率控制方程 默认texˆqCOmp

-rC_override override 特定间隔下的速率控制重载

-me method 设置运动估计的方法 可用方法有 zero phods log x1 epzs(缺省) full

-dCt_algo algo 设置dCt的算法 可用的有 0 FF_DCT_AUTO 缺省的DCT 1 FF_DCT_FASTINT 2 FF_DCT_INT 3 FF_DCT_MMX 4 FF_DCT_MLIB 5 FF_DCT_ALTIVEC

-idCt_algo algo 设置idCt算法。可用的有 0 FF_IDCT_AUTO 缺省的IDCT 1 FF_IDCT_INT 2 FF_IDCT_SIMPLE 3 FF_IDCT_SIMPLEMMX 4 FF_IDCT_LIBMPEG2MMX 5 FF_IDCT_PS2 6 FF_IDCT_MLIB 7 FF_IDCT_ARM 8 FF_IDCT_ALTIVEC 9 FF_IDCT_SH4 10 FF_IDCT_SIMPLEARM

-er n 设置错误残留为n 1 FF_ER_CAREFULL 缺省 2 FF_ER_COMPLIANT 3 FF_ER_AGGRESSIVE 4 FF_ER_VERY_AGGRESSIVE

-eC bit_mask 设置错误掩蔽为bit_mask,该值为如下值的位掩码 1 FF_EC_GUESS_MVS (default=enabled) 2 FF_EC_DEBLOCK (default=enabled)

-bf frames 使用frames B 帧，支持mpeg1,mpeg2,mpeg4

-mbd mode 宏块决策 0 FF_MB_DECISION_SIMPLE 使用mb_Cmp 1 FF_MB_DECISION_BITS 2 FF_MB_DECISION_RD

-4mv 使用4个运动矢量 仅用于mpeg4

-part 使用数据划分 仅用于mpeg4

-bug param 绕过没有被自动监测到编码器的问题

-striCt striCtness 跟标准的严格性

-aiC 使能高级帧内编码 h263+

-umv 使能无限运动矢量 h263+

-deinterlaCe 不采用交织方法

-interlaCe 强迫交织法编码 仅对mpeg2和mpeg4有效。当你的输入是交织的并且你想要保持交织以最小图像损失的时候采用该选项。可选的方法是不交织，但是损失更大

-psnr 计算压缩帧的psnr

-vstats 输出视频编码统计到vstats_hhmmss.log

-vhook module 插入视频处理模块 module 包括了模块名和参数，用空格分开

D)音频选项

-ab bitrate 设置音频码率

-ar freq 设置音频采样率

-aC Channels 设置通道 缺省为1

-an 不使能音频纪录

-aCOdeC COdeC 使用COdeC编解码

E)音频/视频捕获选项

-vd deviCe 设置视频捕获设备。比如/dev/video0

-vC Channel 设置视频捕获通道 DV1394专用

-tvstd standard 设置电视标准 NTSC PAL(SECAM)

-dv1394 设置DV1394捕获

-av deviCe 设置音频设备 比如/dev/dsp

F)高级选项

-map file:stream 设置输入流映射

-debug 打印特定调试信息

-benChmark 为基准测试加入时间

-hex 倾倒每一个输入包

-bitexaCt 仅使用位精确算法 用于编解码测试

-ps size 设置包大小，以bits为单位

-re 以本地帧频读数据，主要用于模拟捕获设备

-loop 循环输入流。只工作于图像流，用于ffserver测试

先从ffmpeg开始。

http://ffmpeg.sourCeforge.net上有说明，音视频的分离，转换，解码的完全解决方案。

其中最重要的就是libavCOdeC库。它被mplayer或者xine使用作为解码器。还有，国内比较流行的播放器影音风暴或MyMPC的后端ffdshow也是使用ffmpeg的解码库的。

ffmpeg包括一组软件，ffmpeg用于对媒体文件进行处理，ffserver是一个http的流媒体服务器，ffplay是一个基于SDL的简单播放器。两个库文件libavCOdeC和libavformat。

ffmpeg作为媒体文件处理软件，基本用法如下：
ffmpeg -i INPUTfile [OPTIONS] OUTPUTfile
输入输出文件通常就是待处理的多媒体文件了。可以是纯粹的音频文件，纯粹的视频文件，或者混合的。
大部分常见的格式都能够“通杀”。象常见的各种mpeg，AVI封装的DIVX和Xvid等等
具体的格式支持列表可以使用ffmpeg -formats查看或直接查阅文档。

另：由于Linux把设备视为文件，因此－i选项后可以跟设备名。比如DV，视频卡，光驱或者其它的各类设备。
输出的内容通过Options调整。列出几个主要的选项

-vCOdeC 视频流编码方式

-b 视频流码率（默认只有200k，一般都需要手动设置，具体的数值视COdeC选择而定）

-r 视频流帧数（一般说来PAL制式通常用25，NTSC制式通常用29）

-s 视频解析度（分辨率，也要视COdeC和你的需要而定。通常改变某个视频流的解析度是很耗费CPU的事情。另：具体写法使用“数字x数字”的形式。中间是小写字母“x”，这个用过mplayer的应该都知道）

-t 处理持续时间。

-aCOdeC 音频流编码方式

-ab 音频流码率（默认是同源文件码率，也需要视COdeC而定）

-ar 音频流采样率（大多数情况下使用44100和48000，分别对应PAL制式和NTSC制式，根据需要选择）

还有些可能需要用到的选项如
-vn和-an分别是屏蔽视频流和屏蔽音频流，分别对源文件处理一次即可得到分离的音频和视频
-author -title分别是设置媒体文件的作者和title
-f选项是强制使用某种格式

－target type是使用预置的格式转换（可以转成dvd，vCd或svCd）

除此之外还有些更高级的选项，如设定vbr，或设定high quality，或者设定vbr的buff和max/min码率，象一般我们自用的dvd抓轨啦，DV转vCd或dvd啦，网上下载的电影转成vCd或dvd都不一定需要用到它们。
具体的使用方法在man里面有介绍。简单明了。

少许使用经验：
1：ffmpeg对于rm的处理能力实在不敢恭维。也许是因为我主要使用二进制包安装的缘故，对于Real媒体格式只能处理老式的RV8编码的格式。而且效果不佳。

2：格式转换是一件很耗费CPU资源的事情。虽说ffmpeg已经比WinAVI啦，TmpgEnC这些win下的非专业级视频处理软件做的好些了。毕竟我们可以把ffmpeg运行的时候放到后台。

3：ffmpeg不是万能的，虽说支持的格式很多，但是如果你不是用的最新CVS出来的版本，可能碰上某些古怪的媒体文件就要郁闷。
4：ffmpeg全部是命令行操作。哪位达人写个GUI前端出来就可以让不少菜鸟脱离苦海了。还有就是不能批量处理，但是这个可以用shell帮忙解决。

使用ffmpeg进行.264编码的相关文章比较少，google了一下，特总结如下：

qscale的取值可以是0.01-255但实际使用超过50就很糟糕了

ffmpeg的cbr模式可以把码率控制的不错，但是vbr无法限制最高码率（虽然有max的设置，但是程序没有实现）

x264标准的封装是x264+aac in flv或者x264+aac in MP4

接下来说明下ffmpeg命令行的语法规则（本块内容来自2009-03-02官方文档）：

语法规则结构：

ffmpeg [[infile options][`-i' infile]]... {[outfile options]outfile}...

一个最简单的命令形式：

ffmpeg -i input.avi -b 64k output.avi

这个命令把视频以64k的码率重编码。

显然，输入文件前面要加一个-i选项下面介绍一些有用的全局参数：

-formats 参数。会显示你机器当前支持的封装、编码、解码器的信息 -y参数,会指示ffmpeg覆盖输出文件 -t 指定视频流持续的时常,支持以秒为单位的数字或"时:分:秒[.毫秒]" -fs 指定输出文件大小的限制 -ss 指定开始的时间,和-t的单位一样 -target 直接设定你想要转成的目标格式,所有的相关设置都会采用内设值,当然也你也可以加上自己要修改的参数。可用的选择有:
"vcd", "svcd", "dvd", "dv", "dv50", "pal-vcd", "ntsc-svcd", …
这个例子把视频转换成vcd的格式

ffmpeg -i myfile.avi -target vcd /tmp/vcd.mpg

接下来介绍视频选项：

-b 指定码率注意单位是bit/s,所以我们一般要加k,比如 -b 1000k 就是1000kb/s

-g 设置组的大小

-vframes 指定要编码的帧数,比如-vframes 1 就是编码1帧,截图的时候就这样写.

-r 指定帧率,默认是25

-s 指定图像分辨率,用wxh的格式,比如320×240

-aspect 指定宽高比 可以些16:9这种,也可以写小数比如1.3333

-croptop 指定顶部裁减多少像素,类似的还有

-cropleft -cropright -cropbuttom

-bt 设置比特率容许的误差,默认4000k,在第一阶段时使用这个参数会告诉码率控制器能够偏移平均码率多远,这个选项和最大最小码率无关.设太小了不利于质量

-maxrate 和-minrate 指定允许的最大和最小码率,一般如果要用cbr模式编码的话会用这个:
ffmpeg -i myfile.avi -b 4000k -minrate 4000k -maxrate 4000k -bufsize 1835k out.m2v
否则用处不大

-vcodec 强制使用某种编码器

-sameq 使用和源文件相同的质量,这个选项是动态码率的

-pass 指定编码阶段,这个只有1和2,第一阶段的时候应该不处理音频,并且把输出导向空,比如: ffmpeg -i foo.mov -vcodec libxvid -pass 1 -an -f rawvideo -y NULffmpeg -i foo.mov -vcodec libxvid -pass 1 -an -f rawvideo -y /dev/null

-qscale 使用固定量化因子来量化视频这个是在vbr模式的,前面有提到,越小质量越好,不要超过50,相关的参数还有

-qmin -qmax用来设定最大最小可使用的量化值

-qdiff 指定固定量化器因子允许的最大偏差

-qblur 指定量化器模糊系数，可用范围是0.0-1.0越大使得码率在时间上分配的越平均

-qcomp 指定视频量化器压缩系数，默认0.5

-me_method 指定运动估计方法（motion estimation method），可用的选择从垃圾到好排列如下：
zero （0向量）
phods
log

x1
hex
umh
epzs （默认）
full （完全搜索，很慢又好不到哪里去）

-mbd 设定宏模块决策，可用的值：
0 使用mb_cmp，不知道是什么东西，所以这些参数我参考一下mencoder里面的
1 选用需要比特最少的宏块模式
2 选用码率失真最优的宏块模式

-4mv 使用宏块的4个运动向量，只支持mpeg4 -part 使用数据划分，只支持mpeg4

-ilme 强制允许交错的编码，只支持mpeg2和mpeg4，你可以选择用-deinterlace来去交错

音频部分：

-ar 设置采样频率,默认44100hz

-ab 设置比特率,默认64k

-an 禁用音频录制

-acodec 指定音频编码器

下面举几个x264编码的例子：

我使用mencoder调用x264编码一个psp用的视频：

x264+aac in mp4（我修改过的，原作者的不能使用）

```
mencoder test_video.vob -oac lavc -lavcopts acodec=libfaac:abitrate=94 -ovc x264 -x264encopts\
cabac=1:ref=1:deblock=1,0,0:analyse=0x1,0x111:me=hex:subme=6:psy_rd=1.0,0.0:mixed_refs=0:me_range=3
8dct=0:no-chroma-me=0:chroma_qp_offset=-\
2:nr=0:dct_decimate=1:bframes=3:b_pyramid=0:b_adapt=1:b_bias=0:direct=3:keyint=250:keyint_min=25:sc
0 -lavdopts er=2 -of lavf -lavfopts format=mp4 -vf scale=720:480 -o men.mp4
```

对应的ffmpeg编码参数是：

ffmpeg

```
ffmpeg -i inputfile.avi -f psp -acodec libfaac -ab 94k -vcodec libx264 -cqp 28 -coder 1 -refs 3 -
deblockalpha 1 -deblockbeta -1 -me_method umh -subq 9 -me_range 32 -trellis 2 -chromaoffset -2 -nr
0 -bf 2 -b_strategy 1 -bframebias 0 -directpred 3 -g 250 -i_qfactor 1.3 -b_qfactor 1.4 -flags2
+bpyramid+wpred+mixed_refs+8x8dct -er 2 -s 480x320
```

需要注意的是,flags2里面那块,似乎要按照一定顺序才能正常工作,其他地方都差不多,详细情况可以从下面两篇文章得出：

第一篇是这个在网上被传了很多,但有些问题的对照表格,我修改了有问题的部分：

-g	-keyint
-b	-bitrate
-bufsize	-vbv-bufsize
-maxrate	-vbv-maxrate
-pass <1,2,3>	-pass
-crf	-crf
-cqp	-qp
-bf	-bframes
-coder <0,1>	-no-cabac
-bframebias	-b-bias
-keyint_min	-min-keyint
-sc_threshold	-scenecut
-deblockalpha -deblockbeta	-deblock
-qmin	-qpmin
-qmax	-qpmax
-qdiff	-qpstep
-qcomp	-qcomp
-qblur	-qblur
-complexityblur	-cplxblur
-refs	-ref
-directpred	-direct
-me_method	-me
-me_range	-merange
-subq	-subme

-bidir_refine <0,1>	-bime
-trellis <0,1,2>	-trellis
-nr	-nr
-level	-level
-bt	-ratetol = -bt / -b
-rc_init_occupancy	-vbv-init = -rc_init_occupancy / -bufsize
-i_qfactor	-ipratio = 1 / -i_qfactor
-b_qfactor	-pbratio
-chromaoffset	-chroma-qp-offset
-rc_eq	-rc_eq
-threads	-threads
-cmp <-chroma/+chroma>	-no-chroma-me
-partitions	-partitions
+parti8×8	i8×8
+parti4×4	i4×4
+partp8×8	p8×8
+partp4×4	p4×4
+partb8×8	b8×8
-flags	
-loop/+loop	-no-deblock/-deblock
-psnr/+psnr	-no-psnr/nothing
-flags2	
+bpyramid	-b-pyramid
+wpred	-weightb
+brdo	-b-rdo 我这里的**ffmpeg**已经不能用这个了
+mixed_refs	-mixed-refs
+dct8×8	-8×8dct
-fastpskip/+fastpskip	-no-fast-pskip
+aud	-aud

一下是这篇文章在2008年11月19日更新的版本:

Frame-type options: - keyint (x264)
-g (FFmpeg)
Keyframe interval, also known as GOP length. This determines the maximum distance between I-frames. Very high GOP lengths will result in slightly more efficient compression, but will make seeking in the video somewhat more difficult. Recommended default: 250

- min-keyint (x264)
-keyint_min (FFmpeg)
Minimum GOP length, the minimum distance between I-frames. Recommended default: 25

- scenecut (x264)
-sc_threshold (FFmpeg)
Adjusts the sensitivity of x264′s scenecut detection. Rarely needs to be adjusted. Recommended default: 40

- pre-scenecut (x264)
UNKNOWN (FFmpeg)
Slightly faster (but less precise) scenecut detection. Normal scenecut detection decides whether a frame is a scenecut after the frame is encoded, and if so then re-encodes the frame as an I-frame. This is not compatible with threading, however, and so - pre-scenecut is automatically activated when multiple encoding threads are used.

- bframes (x264)
-bf (FFmpeg)
B-frames are a core element of H.264 and are more efficient in H.264 than any previous standard. Some specific targets, such as HD-DVD and Blu-Ray, have limitations on the number of consecutive B-frames. Most, however, do not; as a result, there is rarely any negative effect to setting this to the maximum (16) since x264 will, if B-adapt is used, automatically choose the best number of B-frames anyways. This parameter simply serves to limit the max number of B-frames. Note that Baseline Profile, such as that used by iPods, does not support B-frames. Recommended default: 16

- b-adapt (x264)
-b_strategy (FFmpeg)
x264, by default, adaptively decides through a low-resolution lookahead the best number of B-frames to use. It is possible to disable this adaptivity; this is not recommended. Recommended default: 1

0: Very fast, but not recommended. Does not work with pre-scenecut (scenecut must be off to force off b-adapt).

1: Fast, default mode in x264. A good balance between speed and quality.

2: A much slower but more accurate B-frame decision mode that correctly detects fades and generally gives considerably better quality. Its speed gets considerably slower at high bframes values, so its recommended to keep bframes relatively low (perhaps around 3) when using this option. It also may slow down the first pass of x264 when in threaded mode.

- b-bias 0 (x264)

-bframebias 0 (FFmpeg)

Make x264 more likely to choose higher numbers of B-frames during the adaptive lookahead. Not generally recommended. Recommended default: 0

- b-pyramid (x264)

-flags2 +bpyramid (FFmpeg)

Allows B-frames to be kept as references. The name is technically misleading, as x264 does not actually use pyramid coding; it simply adds B-references to the normal reference list. B-references get a quantizer halfway between that of a B-frame and P-frame. This setting is generally beneficial, but it increases the DPB (decoding picture buffer) size required for playback, so when encoding for hardware, disabling it may help compatibility.

- no-cabac (x264)

-coder 0,1 (FFmpeg)

CABAC is the default entropy encoder used by x264. Though somewhat slower on both the decoding and encoding end, it offers 10-15% improved compression on live-action sources and considerably higher improvements on animated sources, especially at low bitrates. It is also required for the use of trellis quantization. Disabling CABAC may somewhat improve decoding performance, especially at high bitrates. CABAC is not allowed in Baseline Profile. Recommended default: -coder 1 (CABAC enabled)

- ref (x264)

-refs (FFmpeg)

One of H.264′s most useful features is the abillity to reference frames other than the one immediately prior to the current frame. This parameter lets one specify how many references can be used, through a maximum of 16. Increasing the number of refs increases the DPB (Decoded Picture Buffer) requirement, which means hardware playback devices will often have strict limits to the number of refs they can handle. In live-action sources, more reference have limited use beyond 4-8, but in cartoon sources up to the maximum value of 16 is often useful. More reference frames require more processing power because every frame is searched by the motion search (except when an early skip decision is made). The slowdown is especially apparent with slower motion estimation methods. Recommended default: -refs 6

- no-deblock (x264)

-flags -loop (FFmpeg)

Disable loop filter. Recommended default: -flags +loop (Enabled)

- deblock (x264)

-deblockalpha (FFmpeg)

-deblockbeta (FFmpeg)

One of H.264′s main features is the in-loop deblocker, which avoids the problem of blocking artifacts disrupting motion estimation. This requires a small amount of decoding CPU, but considerably increases quality in nearly all cases. Its strength may be raised or lowered in order to avoid more artifacts or keep more detail, respectively. Deblock has two parameters: alpha (strength) and beta (threshold). Recommended defaults:-deblockalpha 0 -deblockbeta 0 (Must have ‘-flags +loop’)

- interlaced (x264)

UNKNOWN (FFmpeg)

Enables interlaced encoding. x264′s interlaced encoding is not as efficient as its progressive encoding; consider deinterlacing for maximum effectiveness. Ratecontrol:- qp (x264)

-cqp (FFmpeg)

Constant quantizer mode. Not exactly constant completely - B-frames and I-frames have different quantizers from P-frames. Generally should not be used, since CRF gives better quality at the same bitrate.

- bitrate (x264)

-b (FFmpeg)

Enables target bitrate mode. Attempts to reach a specific bitrate. Should be used in 2-pass mode whenever possible; 1-pass bitrate mode is generally the worst ratecontrol mode x264 has.

- crf (x264)

-crf (FFmpeg)

Constant quality mode (also known as constant ratefactor). Bitrate corresponds approximately to that of constant quantizer, but gives better quality overall at little speed cost. The best one-pass option in x264.

-vbv-maxrate (x264)
-maxrate (FFmpeg)
Specifies the maximum bitrate at any point in the video. Requires the VBV buffersize to be set. This option is generally used when encoding for a piece of hardware with bitrate limitations.

-vbv-bufsize (x264)
-bufsize (FFmpeg)
Depends on the profile level of the video being encoded. Set only if you’re encoding for a hardware device.

-vbv-init (x264)
-rc_init_occupancy (FFmpeg)
Initial VBV buffer occupancy. Note: Don’t mess with this.

-qpmin (x264)
-qmin (FFmpeg)
Minimum quantizer. Doesn’t need to be changed. Recommended default: -qmin 10

-qpmax (x264)
-qmax (FFmpeg)
Maximum quantizer. Doesn’t need to be changed. Recommended default: -qmax 51

-qpstep (x264)
-qdiff (FFmpeg)
Set max QP step. Recommended default: -qdiff 4

-ratetol (x264)
-bt (FFmpeg)
Allowed variance of average bitrate

-ipratio (x264)
-i_qfactor (FFmpeg)
Qscale difference between I-frames and P-frames.

-pbratio (x264)
-b_qfactor (FFmpeg)
Qscale difference between P-frames and B-frames.

-chroma-qp-offset (x264)
-chromaoffset (FFmpeg)
QP difference between chroma and luma.

-aq-strength (x264)
UNKNOWN (FFmpeg)
Adjusts the strength of adaptive quantization. Higher values take more bits away from complex areas and edges and move them towards simpler, flatter areas to maintain fine detail. Default: 1.0

-pass <1,2,3> (x264)
-pass <1,2,3>(FFmpeg)
Used with -bitrate. Pass 1 writes the stats file, pass 2 reads it, and 3 both reads and writes it. If you want to use three pass, this means you will have to use -pass 1 for the first pass, -pass 3 for the second, and -pass 2 or 3 for the third.

-stats (x264)
UNKNOWN (FFmpeg)
Allows setting a specific filename for the firstpass stats file.

-rceq (x264)
-rc_eq (FFmpeg)
Ratecontrol equation. Recommended default: -rc_eq ‘blurCplx^(1-qComp)’

-qcomp (x264)
-qcomp (FFmpeg)
QP curve compression: 0.0 => CBR, 1.0 => CQP. Recommended default: -qcomp 0.60

-cplxblur (x264)
-complexityblur (FFmpeg)
Reduce fluctuations in QP (before curve compression) [20.0]

-qblur (x264)
-qblur (FFmpeg)
Reduce fluctuations in QP (after curve compression) [0.5]

-zones /(x264)
UNKNOWN (FFmpeg)

Allows setting a specific quantizer for a specific region of video.

-qpfile (x264)
UNKNOWN (FFmpeg)
Allows one to read in a set of frametypes and quantizers from a file. Useful for testing various encoding options while ensuring the exact same quantizer distribution. Analysis:-partitions (x264)
-partitions (FFmpeg)

p8x8 (x264) /+partp8x8 (FFmpeg)

p4x4 (x264) /+partp4x4 (FFmpeg)

b8x8 (x264) /+partb8x8 (FFmpeg)

i8x8 (x264) /+parti8x8 (FFmpeg)

i4x4 (x264) /+parti4x4 (FFmpeg)

One of H.264's most useful features is the ability to choose among many combinations of inter and intra partitions. P-macroblocks can be subdivided into 16×8, 8×16, 8×8, 4×8, 8×4, and 4×4 partitions. B-macroblocks can be divided into 16×8, 8×16, and 8×8 partitions. I-macroblocks can be divided into 4×4 or 8×8 partitions. Analyzing more partition options improves quality at the cost of speed. The default is to analyze all partitions except p4x4 (p8x8, i8x8, i4x4, b8x8), since p4x4 is not particularly useful except at high bitrates and lower resolutions. Note that i8x8 requires 8x8dct, and is therefore a High Profile-only partition. p8x8 is the most costly, speed-wise, of the partitions, but also gives the most benefit. Generally, whenever possible, all partition types except p4x4 should be used.

-direct (x264)
-directpred (FFmpeg)
B-frames in H.264 can choose between spatial and temporal prediction mode. Auto allows x264 to pick the best of these; the heuristic used is whichever mode allows more skip macroblocks. Auto should generally be used.

-direct-8×8 (x264)
UNKONWN (FFmpeg)
This should be left at the default (-1).

-weightb (x264)
-flags2 +wpred(FFmpeg)
This allows B-frames to use weighted prediction options other than the default. There is no real speed cost for this, so it should always be enabled.

-me

-merange (x264)
-me_range (FFmpeg)
MErange controls the max range of the motion search. For HEX and DIA, this is clamped to between 4 and 16, with a default of 16. For UMH and ESA, it can be increased beyond the default 16 to allow for a wider-range motion search, which is useful on HD footage and for high-motion footage. Note that for UMH and ESA, increasing MErange will significantly slow down encoding.

-mvrange(x264)
UNKNOWN (FFmpeg)
Limits the maximum motion vector range. Since x264 by default limits this to 511.75 for standards compliance, this should not be changed.

-subme 6(x264)
-subq 6(FFmpeg)

1: Fastest, but extremely low quality. Should be avoided except on first pass encoding.

2-5: Progressively better and slower, 5 serves as a good medium for higher speed encoding.
6-7: 6 is the default. Activates rate-distortion optimization for partition decision. This can considerably improve efficiency, though it has a notable speed cost. 6 activates it in I/P frames, and subme7 activates it in B frames.

8-9: Activates rate-distortion refinement, which uses RDO to refine both motion vectors and intra prediction modes. Slower than subme 6, but again, more efficient.

An extremely important encoding parameter which determines what algorithms are used for both subpixel motion searching and partition decision.

-psy-rd :(x264)
UNKNOWN (FFmpeg)
First value represents the amount that x264 biases in favor of detail retention instead of max PSNR in mode decision. Requires subme >= 6. Second value is psy-trellis, an experimental algorithm that tries to improve sharpness and detail retention at the expense of more artifacting.
Recommended starting values are 0.1-0.2. Requires trellis >= 1. Recommended default: 1.0:0.0

-mixed-refs(x264)
-flags2 +mixed_refs(FFmpeg)
H.264 allows p8x8 blocks to select different references for each p8x8 block. This option allows this analysis to be done, and boosts quality with little speed impact. It should generally be used, though it obviously has no effect with only one reference frame.

-no-chroma-me(x264)
UNKNOWN (FFmpeg)
Chroma is used in the last steps of the subpixel refinement by default. For a slight speed increase, this can be disabled (at the cost of quality).

-8x8dct (x264)
-flags2 +dct8x8(FFmpeg)
Gives a notable quality boost by allowing x264 to choose between 8×8 and 4×4 frequency transform size. Required for i8x8 partitions. Speed cost for this option is near-zero both for encoding and decoding; the only reason to disable it is when one needs support on a device not compatible with High Profile.

-trellis <0,1,2>(x264)
-trellis <0,1,2>(FFmpeg)

0: disabled

1: enabled only on the final encode of a MB

2: enabled on all mode decisions

The main decision made in quantization is which coefficients to round up and which to round down. Trellis chooses the optimal rounding choices for the maximum rate-distortion score, to maximize PSNR relative to bitrate. This generally increases quality relative to bitrate by about 5% for a somewhat small speed cost. It should generally be enabled. Note that trellis requires CABAC.

-no-fast-pskip(x264)
-flags2 -fastpskip(FFmpeg)
By default, x264 will skip macroblocks in P-frames that don’t appear to have changed enough between two frames to justify encoding the difference. This considerably speeds up encoding. However, for a slight quality boost, P-skip can be disabled. In this case, the full analysis will be done on all P-blocks, and the only skips in the output stream will be the blocks whose motion vectors happen to match that of the skip vector and motion vectors happen to match that of the skip vector and which have no residual. The speed cost of enabling no-fast-pskip is relatively high, especially with many reference frames. There is a similar B-skip internal to x264, which is why B-frames generally encode much faster than P-frames, but it cannot be disabled on the commandline.

-no-dct-decimate(x264)
UNKNOWN (FFmpeg)
By default, x264 will decimate (remove all coefficients from) P-blocks that are extremely close to empty of coefficents. This can improve overall efficiency with little visual cost, but may work against an attempt to retain grain or similar. DCT decimation should be left on unless there’s a good reason to disable it.

-nr(x264)
UNKNOWN (FFmpeg)
a fast, built-in noise reduction routine. Not as effective as external filters such as hqdn3d, but faster. Since x264 already naturally reduces noise through its quantization process, this parameter is not usually necessary.

-deadzone-inter(264)
-deadzone-intra (x264)
UNKNOWN (FFmpeg)
UNKNOWN (FFmpeg)
When trellis isn’t activated, deadzone parameters determine how many DCT coefficients are rounded up or down. Rounding up results in higher quality and more detail retention, but costs more bits - so rounding is a balance between quality and bit cost. Lowering these settings will result in more coefficients being rounded up, and raising the settings will result in more coefficients being rounded down. Recommended: keep them at the defaults.

-cqm (264)
-cqpfile(x264)UNKNOWN (FFmpeg)
UNKNOWN (FFmpeg)
Allows the use of a custom quantization matrix to weight frequencies differently in the quantization process. The presets quant matrices are "jvt" and "flat". -cqpfile reads a custom quant matrices from a JM-compatible file. Recommended only if you know what you’re doing.

转自：http://hi.baidu.com/zorru/blog/item/0c64863651e512c1a3cc2bb1.html

前一篇：教练不一定教你的开车技术，请珍藏
后一篇：用FFMPEG SDK进行视频转码压缩时解决音视频不同步问题的方法

< 前一篇
教练不一定教你的开车技术，请珍藏

后一篇 >
用FFMPEG SDK进行视频转码压缩时解决音视频不同步问题的方法