Linux上的ffmpeg完全使用指南

By xishui | 2019/07/25 9 Comments

在尝试ffmpeg的时候发现一片很好的文章 https://itsfoss.com/ffmpeg/，做一个笔记。

ffmpeg 是一个处理媒体文件的命令行工具 (command line based)。它是一个拥有非常多功能的框架，并且因为他是开源的，很多知名的工具如 VLC，YouTube，iTunes 等等，都是再其之上开发出来的。也有许多为他定制UI的视频编辑器Linux video editors。

ffmpeg最吸引我的地方就是它可以用非常简练的方式(通过一两个命令)完成许多的处理任务，当然，作为一个强大的工具，他也有很多较为复杂的使用方式，有些时候甚至可以代替一个完整的视频处理流程。

在这个ffmpeg教程中，我会告诉你如何安装ffmpeg，以及各种使用方法，也会讲解一些复杂的功能。

下面是这个教程的章节：

我会详细的说明各个方面，这样即便你是linux新手也能明白。

我使用的linux是 **Ubuntu 18.04**，不过下面的命令应该可以在其他的linux发行版中同样适用。

说明: *尽管我研究并尝试了ffmepg的方方面面，但我的日常工作并不包含视频编辑。所以如果您有更好的建议、技巧或者纠正，请在留言中指出！*

让我们开始吧！

在 Ubuntu 和其他 Linux 系统上安装 ffmpeg

安装 **ffmpeg** 是非常容易的，它是个很流行的程序，所以大多数的linux发行版中您都可以通过包管理器直接安装。

在 Ubuntu 上安装 ffmpeg

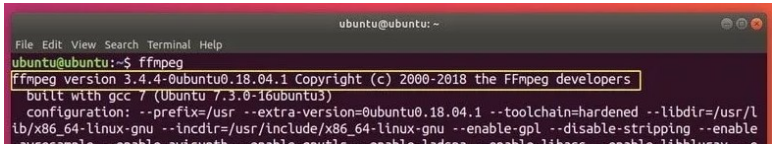
在 Ubuntu 上，ffmpeg 存在于 “Universe repository”，所以确保您开启了enable universe repository，然后更新并安装ffmpeg。下面就是您可能需要的命令。

```
sudo add-apt-repository universe
sudo apt update
sudo apt install ffmpeg
```

这就OK了，您可以通过下面的命令尝试一下有没有正确安装：

```
ffmpeg
```

他会打印出一些ffmpeg的配置和版本信息。



正如上图所示，安装的版本是 3.4.4。不过ffmpeg的最新版本应该是4.1。为了安装4.x的版本，您需要使用 ffmpeg ppa，您可以自己研究一下……

在 Arch 系的Linux上安装 ffmpeg

这个也非常简单，用下面的命令就行：

```
sudo pacman -S ffmpeg
```

在 Fedora 系的Linux上安装 ffmpeg

使用下面的命令就好了：

```
sudo dnf install ffmpeg
```

如何使用 ffmpeg: 基础

ffmpeg 安装就绪了，我来讲述一些使用这个强力工具的基本概念。

0. ffmpeg 命令

使用 **ffmpeg 命令** 的**基本形式**是：

Score发表在《一次曲折的NAS搭建小记》

xx发表在《想对女儿说的话》

Py之pygame：Python的pygame库的简介、安装、使用方法详细攻略 - 算法网发表在《用Python和Pygame写游戏-从入门到精通（目录）》

初服发表在《用Python和Pygame写游戏-从入门到精通（py2exe篇）》

fighting发表在《一次曲折的NAS搭建小记》

小伍的游乐场发表在《用Python制作游戏外挂（上）》

小伍的游乐场发表在《一次曲折的NAS搭建小记》

22发表在《用Python和Pygame写游戏-从入门到精通（1）》

缘分的穹空发表在《用Python和Pygame写游戏-从入门到精通（1）》

归档

2020年11月

2020年1月

2019年11月

2019年7月

2019年4月

2019年3月

2018年9月

2018年6月

2018年3月

2018年2月

2017年12月

2017年11月

2017年9月

2017年7月

2017年6月

2016年8月

2016年6月

2016年5月

2016年4月

2016年2月

2015年12月

2015年11月

2015年10月

2015年9月

2015年8月

2015年7月

2015年5月

2015年4月

```
ffmpeg [全局参数] {[输入文件参数] -i 输入文件地址} ... {[输出文件参数] 输出文件地址} ...
```

要注意的是，所有的参数仅仅对仅接下来的文件有效(下一个文件得把参数再写一遍)。

所有没有使用 **-i** 指定的文件都被认为是输出文件。**Ffmpeg** 可以接受多个输入文件并输出到您指定的位置。你也可以将输入输出都指定为同一个文件名，不过这个时候要在输出文件前使用用 **-y** 标记。

Note

你不应该将输入和输出混淆，先指定输入，再指定输出文件

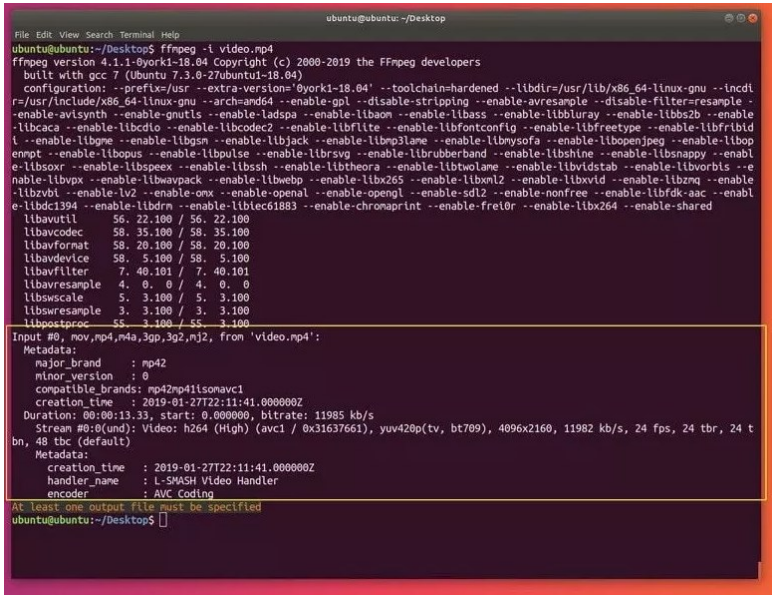
1. 获得媒体文件的信息

ffmpeg 最简单的使用就是用来 **显示文件信息**。不用给输出，只是简单的写：

```
ffmpeg -i file_name
```

视频和音频文件都可以使用：

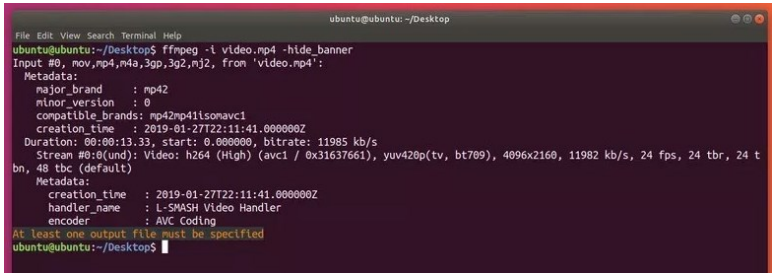
```
ffmpeg -i video_file.mp4
ffmpeg -i audio_file.mp3
```



通过ffmpeg查看文件属性

命令会输出很多与您文件无关的信息(ffmpeg本身的信息)，虽说这个蛮有用的，你可以使用 **-hide_banner** 来隐藏掉它们：

```
ffmpeg -i video_file.mp4 -hide_banner
ffmpeg -i audio_file.mp3 -hide_banner
```



如图所示，现在命令只显示你文件相关的信息了(编码器，数据流等)。

2. 转换媒体文件

ffmpeg 最让人称道常用的恐怕就是你轻而易举的在不同媒体格式之间进行自由转换了。你是要指明输入和输出文件名就行了，**ffmpeg** 会从后缀名猜测格式，这个方法同时适用于视频和音频文件

下面是一些例子：

```
ffmpeg -i video_input.mp4 video_output.avi
ffmpeg -i video_input.webm video_output.flv
ffmpeg -i audio_input.mp3 audio_output.ogg
ffmpeg -i audio_input.wav audio_output.flac
```

你也可以同时指定多个输出后缀：

```
ffmpeg -i audio_input.wav audio_output_1.mp3 audio_output_2.ogg
```

这样会同时输出多个文件.

想看支持的格式，可以用：

```
ffmpeg -formats
```

同样的，你可以使用 **-hide_banner** 来省略一些程序信息。

你可以在输出文件前使用 **-qscale 0** 来保留原始的视频质量：

```
ffmpeg -i video_input.wav -qscale 0 video_output.mp4
```

进一步，你可以指定编码器，使用 **-c:a** (音频) 和 **g-c:v** (视频) 来指定编码器名称，或者写 **copy** 来使用与源文件相同的编码器：

```
ffmpeg -i video_input.mp4 -c:v copy -c:a libvorbis video_output.avi
```

Note: *这样做会让文件后缀使人困惑，所以请避免这么做。*

3. 从视频中抽取音频

为了从视频文件中抽取音频，直接加一个 **-vn** 参数就可以了：

```
ffmpeg -i video.mp4 -vn audio.mp3
```

这会让命令复用原有文件的比特率，一般来说，使用 **-ab** (音频比特率)来指定编码比特率是比较好的：

```
ffmpeg -i video.mp4 -vn -ab 128k audio.mp3
```

一些常见的比特率有 96k, 128k, 192k, 256k, 320k (mp3也可以使用最高的比特率)。

其他的一些常用的参数比如 **-ar (采样率**: 22050, 441000, 48000), **-ac** (**声道数**), **-f** (**音频格式**, 通常会自动识别的). **-ab** 也可以使用 **-b:a** 来替代. 比如：

```
ffmpeg -i video.mov -vn -ar 44100 -ac 2 -b:a 128k -f mp3 audio.mp3
```

4. 让视频静音

和之前的要求类似，我们可以使用 **-an** 来获得纯视频(之前是 **-vn**).

```
ffmpeg -i video_input.mp4 -an -video_output.mp4
```

Note: *这个* ***-an*** *标记会让所有的音频参数无效，因为最后没有音频会产生。*

5. 从视频中提取图片

这个功能可能对很多人都挺有用，比如你可能有一些幻灯片，你想从里面提取所有的图片，那么下面这个命令就能帮你：

```
ffmpeg -i video.mp4 -r 1 -f image2 image-%3d.png
```

我们来解释一下这个命令：

-r 代表了帧率（一秒内导出多少张图像，默认25），**-f** 代表了输出格式(**image2** 实际上上 image2 序列的意思）。

最后一个参数 (输出文件) 有一个有趣的命名：它使用 **%3d** 来指示输出的图片有三位数字 (000, 001, 等等.)。你也可以用 **%2d** (两位数字) 或者 **%4d** (4位数字)，只要你愿意，你可以随便实验 一下可以怎么写！

Note: *同样也有将图片转变为视频/幻灯片的方式，下面的* ***高级应用*** *中会讲到。*

6. 更改视频分辨率或长宽比

对 **ffmpeg** 来说又是个简单的任务，你只需要使用 **-s** 参数来缩放视频就行了：

```
ffmpeg -i video_input.mov -s 1024x576 video_output.mp4
```

同时，你可能需要使用 **-c:a** 来保证音频编码是正确的:

```
ffmpeg -i video_input.h264 -s 640x480 -c:a video_output.mov
```

你也可是使用**-aspect** 来更改长宽比:

```
ffmpeg -i video_input.mp4 -aspect 4:3 video_output.mp4
```

Note: 在高级应用中还会提到更强大的方法

7. 为音频增加封面图片

有个很棒的方法把音频变成视频，全程使用一张图片（比如专辑封面）。当你想往某个网站上传音频，但那个网站又仅接受视频（比如YouTube, Facebook等）的情况下会非常有用。

下面是例子：

```
ffmpeg -loop 1 -i image.jpg -i audio.wav -c:v libx264 -c:a aac -strict experimental -
b:a 192k -shortest output.mp4
```

只要改一下编码设置 (**-c:v** 是 视频编码，**-c:a** 是音频编码) 和文件的名称就能用了。

Note: *如果你使用一个较新的ffmpeg版本（4.x），你就可以不指定* ***-strict experimental***

8. 为视频增加字幕

另一个常见又很容易实现的要求是给视频增加字母，比如一部外文电源，使用下面的命令：

```
ffmpeg -i video.mp4 -i subtitles.srt -c:v copy -c:a copy -preset veryfast -c:s
mov_text -map 0 -map 1 output.mp4
```

当然，你可以指定自己的编码器和任何其他的音频视频参数。你可以阅读这篇文章来了解字幕相关内容 editing subtitles in Linux 。

9. 压缩媒体文件

压缩文件可以极大减少文件的体积，节约存储空间，这对于文件传输尤为重要。通过ffmepg，有好几个方法来压缩文件体积。

Note: 文件压缩的太厉害会让文件质量显著降低。

首先，对于音频文件，可以通过降低比特率(使用 **-b:a** 或 **-ab**):

```
ffmpeg -i audio_input.mp3 -ab 128k audio_output.mp3
ffmpeg -i audio_input.mp3 -b:a 192k audio_output.mp3
```

再次重申，一些常用的比特率有: 96k, 112k, 128k, 160k, 192k, 256k, 320k.值越大，文件所需要的体积就越

大。

对于视频文件，选项就多了，一个简单的方法是通过降低视频比特率 (通过 **-b:v**)：

```
ffmpeg -i video_input.mp4 -b:v 1000k -bufsize 1000k video_output.mp4
```

Note: 视频的比特率和音频是不同的（一般要大得多）。

你也可以使用 **-crf** 参数 (恒定质量因子)，较小的**crf** 意味着较大的码率。同时使用 **libx264** 编码器也有助于减小文件体积。这里有个例子，压缩的不错，质量也不会显著变化：

```
ffmpeg -i video_input.mp4 -c:v libx264 -crf 28 video_output.mp4
```

crf 设置为20 到 30 是最常见的，不过您也可以尝试一些其他的值。

降低帧率在有些情况下也能有效（不过这往往让视频看起来很卡）：

```
ffmpeg -i video_input.mp4 -r 24 video_output.mp4
```

-r 指示了帧率 (这里是 **24**)。

你还可以通过压缩音频来降低视频文件的体积，比如设置为立体声或者降低比特率：

```
ffmpeg -i video_input.mp4 -c:v libx264 -ac 2 -c:a aac -strict -2 -b:a 128k -crf 28
video_output.mp4
```

Note: ***-strict -2** 和 **-ac 2** 是来处理立体声部分的。*

10. 裁剪媒体文件（基础）

想要从开头开始剪辑一部分，使用T **-t** 参数来指定一个时间：

```
ffmpeg -i input_video.mp4 -t 5 output_video.mp4
ffmpeg -i input_audio.wav -t 00:00:05 output_audio.wav
```

这个参数对音频和视频都适用，上面两个命令做了类似的事情：保存一段5s的输出文件（文件开头开始算）。上面使用了两种不同的表示时间的方式，一个单纯的数字（描述）或者 **HH:MM:SS** (小时, 分钟, 秒). 第二种方式实际上指示了结束时间。

也可以通过 **-ss** 给出一个开始时间，**-to** 给出结束时间：

```
ffmpeg -i input_audio.mp3 -ss 00:01:14 output_audio.mp3
ffmpeg -i input_audio.wav -ss 00:00:30 -t 10 output_audio.wav
ffmpeg -i input_video.h264 -ss 00:01:30 -to 00:01:40 output_video.h264
ffmpeg -i input_audio.ogg -ss 5 output_audio.ogg
```

可以看到 **开始时间** (**-ss HH:MM:SS**), **持续秒数** (**-t duration**), **结束时间** (**-to HH:MM:SS**), 和**开始秒数** (**-s duration**)的用法.

你可以在媒体文件的任何部分使用这些命令。

ffmpeg: 高级使用

现在该开始讲述一些高级的特性了(比如截屏等)，让我们开始吧。

1. 分割媒体文件

前面已经讲述了如何裁剪文件，那么如何分割媒体文件呢？只需要为每个输出文件分别指定开始时间、结束或者持续时间就可以了。

看下面这个例子：

```
ffmpeg -i video.mp4 -t 00:00:30 video_1.mp4 -ss 00:00:30 video_2.mp4
```

语法很简单，为第一个文件指定了 **-t 00:00:30** 作为持续时间(第一个部分是原始文件的前30秒内容)，然后指定接下来的所有内容作为第二个文件(从第一部分的结束时间开始，也就是 **00:00:30**)。

你可以任意指定多少个部分，尝试一下吧，这个功能真的很厉害，同时它也适用用音频文件。

2. 拼接媒体文件

ffmpeg 也可以进行相反的动作：把多个文件合在一起。

为了实现这一点，你得用自己顺手的编辑器来创建一个文本文件。

因为我喜欢使用终端，所以这里我用了 **touch** 和 **vim**. 文件名无关紧要，这里我用 **touch** 命令创建 **video_to_join.txt** 文件：

```
touch videos_to_join.txt
```

现在，使用 **vim** 编辑它：

```
vim videos_to_join.txt
```

你可以使用任何你喜欢的工具，比如nano, gedit等等。

在文件内容中，输入您想拼接的文件的完整路径(文件会按照顺序拼合在一起)，一行一个文件。确保他们拥有相同的后缀名。下面是我的例子：

```
/home/ubuntu/Desktop/video_1.mp4
/home/ubuntu/Desktop/video_2.mp4
/home/ubuntu/Desktop/video_3.mp4
```

保存这个文件，同样这个方法适用与任何音频或者视频文件。

然后使用下面的命令：

```
ffmpeg -f concat -i join.txt output.mp4
```

Note: *使用的输出文件的名称是* ***output.mp4****, 因为我的输入文件都是mp4的* 。

这样，你 **videos_to_join.txt** 里的所有文件都会被拼接成一个独立的文件了。

3. 将图片转变为视频

这会告诉你如何将图片变成幻灯片秀，同时也会告诉你如何加上音频。

首先我建议您将所有的图片放到一个文件夹下面，我把它们放到了 **my_photos** 里，同时图片的后缀名最好是 **.png** 或者 **.jpg**，不管选那个，他们应该是同一个后缀名，否则ffmpeg可能会工作的不正常，您可以很方便的把 .png 转变为 .jpg（或者倒过来也行）。

我们这次转换的格式 (**-f**) 应该被设置为 **image2pipe**. 你必须使用使用连词符(**–**)来指明输入。**image2pipe** 允许你使用管道 (在命令间使用 **|**)的结果而不是文件作为ffmpeg的输入。命令结果便是将所有图片的内容逐个输出，还要注意指明视频编码器是 copy (**-c:v copy**) 以正确使用图片输入：

```
cat my_photos/* | ffmpeg -f image2pipe -i - -c:v copy video.mkv
```

如果你播放这个文件，你可能会觉得只有一部分图片被加入了，事实上所有的图片都在，但是**ffmpeg** 播放它们的时候太快了，默认是23fps，一秒播放了23张图片。

你应该指定帧率 (**-framerate**)：

```
cat my_photos/* | ffmpeg -framerate 1 -f image2pipe -i - -c:v copy video.mkv
```

在这个例子里，把帧率设置为1，也就是每帧（每张图）会显示1秒。

为了加一些声音，可以使用音频文件作为输入 (**-i audo_file**) 并且设定copy音频编码 (**-c:a copy**). 你可以同时为音频和视频设定编码器，在输出文件前设置就可以了。你要计算一下音频文件的长度和图片张数，已确定合适的帧率。比如我的音频文件是22秒，图片有9张，那么帧率应该是 9 / 22 大约0.4，所以我这么输入命令：

```
cat my_photos/* | ffmpeg -framerate 0.40 -f image2pipe -i - -i audio.wav -c copy
video.mkv
```

4. 录制屏幕

通过 **ffmpeg** 录制屏幕同样没有困难的，将格式(**-f**) 设定为**x11grab**. 他就会抓取你的**XSERVER**. 输入的话可以这是屏幕编号（一般都是**0:0**). 抓取是从左上角开始计算的，可以指定屏幕分辨率 (**-s**). 我的屏幕是 **1920×1080**. 注意屏幕分辨率硬在输入之前指定**t**:

```
ffmpeg -f x11grab -s 1920x1080 -i :0.0 output.mp4
```

按 **q** 或者 **CTRL+C** 以结束录制屏幕。

小技巧:你可以通过命令获得真实的分辨率而不是写死一个固定的大小：

```
-s $(xdpyinfo | grep dimensions | awk '{print $2;}')
```

完整的命令这么写:

```
ffmpeg -f x11grab -s $(xdpyinfo | grep dimensions | awk '{print $2;}') -i :0.0
output.mp4
```

5. 录制摄像头

从摄像头录制就更简单了, linux上设备都是在/dev中的, 比如 **/dev/video0, /dev/video1, etc.**:

```
ffmpeg -i /dev/video0 output.mkv
```

同样, **q** 或者 **CTRL+C** 来结束录制。

6. 录制声音

Linux上同时是使用 **ALSA** 和 **pulseaudio** 来处理声音的。**ffmpeg** 可以录制两者, 不过我要特别说明 **pulseaudio**, 因为 Debian 系列的发行版默认用了它。命令如下:

在 **pulseaudio**, 你必须强制指定(**-f**) **alsa** 然后指定 **default** 作为输入**t** (**-i default**):

```
ffmpeg -f alsa -i default output.mp3
```

Note: *在你系统音频设置里, 应该能看到默认的录音设备。*

我经常玩吉他, 我平时使用一个专业音频设备才能录制声音, 当我发现ffmpeg也可以很轻松的录制的时候颇为惊讶。

录制小贴士

对于录制任务来说, 通常都需要指定编码器以及帧率, 之前讲过的参数当然也可以用到这里来！

```
ffmpeg -i /dev/video0 -f alsa -i default -c:v libx264 -c:a flac -r 30 output.mkv
```

有时候不直接录音, 而是在录屏/录像的时候给一个音频文件, 那么可以这么做:

```
ffmpeg -f x11grab -s $(xdpyinfo | grep dimensions | awk '{print $2;}') -i :0.0 -i
audio.wav -c:a copy output.mp4
```

Note: ***ffmpeg** 使用片段录取, 所有有时候非常短的录制可能不会保存文件。我建议录地可以稍微长一些(然后后期裁剪), 已保证录制的文件成功写到磁盘上。*

ffmpeg中的过滤器的基本使用

过滤器 是 **ffmpeg** 中最为强大的功能。在ffmepg中有数不甚数的过滤器存在, 可以满足各种编辑需要。因为过滤器实在太多了, 这里只会简单讲述几个常用的。

使用 过滤的基本结构是:

```
ffmpeg -i input.mp4 -vf "filter=setting_1=value_1:setting_2=value_2,etc" output.mp4
ffmpeg -i input.wav -af "filter=setting_1=value_1:setting_2=value_2,etc" output.wav
```

可以指定视频过滤器 (**-vf**, **-filter:v**的简写) 和 音频过滤器 (**-af**, **-filter:a**的简写). 过滤器的内容写到双引号里面 (“) 并且可以使用逗号(,)连接。你可以使用任意数量的过滤器（我写了个etc代表更多的，这不是做一个真实的过滤器）。

过滤器设定的通常格式是:

```
filter=setting_2=value_2:setting_2=value_2
```

过滤器不同的值使用冒号分割。

你甚至可以在值里面使用进行数学符号计算。

Note: 参考 ***ffmpeg 过滤器手册***.查看更多高级用法

这里举几个例子来说明视频和音频的过滤器。

1. 视频缩放

这是个简单过滤器，设定里只有 **width** 和 **height**:

```
ffmpeg -i input.mp4 -vf "scale=w=800:h=600" output.mp4
```

我说过你可以使用数学运算来给值:

```
ffmpeg -i input.mkv -vf "scale=w=1/2*in_w:h=1/2*in_h" output.mkv
```

很明显，这个命令让输入的尺寸变成了输入尺寸(in_w, in_h)的1/2.

2. 视频裁剪

类似缩放，这个设定也有 **width** 和 **height**，另外可以指定裁剪的原点（默认是视频的中心）

```
ffmpeg -i input.mp4 -vf "crop=w=1280:h=720:x=0:y=0" output.mp4
ffmpeg -i input.mkv -vf "crop=w=400:h=400" output.mkv
```

第二个命令裁剪原点是视频的中心点（因为我没有给x和y坐标），第一个命令会从左上角开始裁剪 (**x=0:y=0**).

这里也有一个使用数学计算的例子:

```
ffmpeg -i input.mkv -vf "crop=w=3/4*in_w:h=3/4*in_h" output.mkv
```

这会把视频裁剪剩下原大小的3/4/。

3. 视频旋转

你可以指定一个弧度，顺时针旋转视频。为了让计算简单一些，你可以给角度然后乘以 **PI/180**:

```
ffmpeg -i input.avi -vf "rotate=90*PI/180"
ffmpeg -i input.mp4 -vf "rotate=PI"
```

第一个命令将视频顺时针旋转90°，第二个则是上下颠倒了视频(翻转了180°)。

4. 音频声道重映射

有的时候，你的音频只有右耳可以听到声音，那么这个功能就很有用了。你可以让声音同时在左右声道出现：

```
ffmpeg -i input.mp3 -af "channelmap=1-0|1-1" output.mp3
```

这将右声道(1)同时映射到左(0)右(1)两个声道(左边的数字是输入，右边的数字是输出)。

5. 更改音量

你可以将音量大小乘以一个实数(可以是整数也可以不是)，你只需要给出那个数大小就行了。

```
ffmpeg -i input.wav -af "volume=1.5" output.wav
ffmpeg -i input.ogg -af "volume=0.75" output.ogg
```

第一个将音量变为1.5倍，第二个则让音量变成了原来的1/4那么安静。

技巧：更改播放速度

这里会介绍视频(不影响音频)和音频的过滤器。

1. **视频**

视频过滤器是 **setpts** (PTS = presentation time stamp). 这个参数以一种有趣的方式工作，因为我们修改的是PTS，所以较大的数值意味着较慢的播放速度，反之亦然：

```
ffmpeg -i input.mkv -vf "setpts=0.5*PTS" output.mkv
ffmpeg -i input.mp4 -vf "setpts=2*PTS" output,mp4
```

第一个命令让播放速度加倍了，第二个则是让播放速度降低了一半。

2. 音频

这里的过滤器是 **atempo**. 这里有个限制，它只接受 **0.5**(半速) 到 **2** (倍速)之间的值。为了越过这个限制，你可以链式使用这个过滤器：

```
ffmpeg -i input.wav -af "atempo=0.75" output.wav
ffmpeg -i input.mp3 -af "atempo=2.0,atempo=2.0" ouutput.mp3
```

第一个命令让音频速度慢了1/4，第二个则是加速到原来的4(2*2)倍。

Note: *如果想在同一个命令中同时修改视频和音频的速度，你得查看一下* ***filtergraphs****.*

小结

在这个手册中，我讲述了安装、基本的使用、高级的使用和一些过滤器的基础。

我希望这对于一些尝试使用ffmpeg的人，或者希望使用ffmpeg做很多工作的人来说是个有用的资源，ffmepg真的是个多功能又极其好用的工具。

如果这对您有所帮助，希望您能留言告诉我。感谢您的阅读，也许以后会讲述一些比较高级的过滤器的使用，Enjoy!

Category: 烂笔头 标签: ffmpeg

← Homestead (vagrant, virtualbox) 不自动同步时间

Linux中获取已后台运行进程的输出 →

9 thoughts on “Linux上的ffmpeg完全使用指南”

Leon Zou
2019/07/25

那么就是说，ffmpeg是无法完成adobe pr的功能吗？？

Reply ↓

xishui Post author
2019/07/25

自然，他们的定位还是有区别的，就好像ImageMagick无法完成PS的功能一样，这些工具并不对“表述的内容”本身进行处理 。

Reply ↓

gyx
2019/10/17

几乎看完了楼主写的我感兴趣的所有文章，果然是程序员才懂程序员的需求，还有我也喜欢日语，哎感觉你就像是N年后的我的样子。。。

Reply ↓

yutouwd
2019/10/21

也是从11年的pygame那篇文章找到博主的博客的，希望自己也能像博主一样一直把博客写下去吧□

Reply ↓

Pingback: RHEL 8 安装 ffmpeg，使用 you-get 下载 YouTube、bilibili 的影片 | Anderson`s blog

Yancey
2020/11/12

强

Reply ↓

Pingback: Linux上的ffmpeg完全使用指南 | All Stack

陈彰沛
2021/09/15

我是sb

Reply ↓

帅海航
2021/09/15

我是sb+1

Reply ↓

发表评论

您的电子邮箱地址不会被公开。必填项已用*标注

评论 *

显示名称 *

电子邮箱地址 *

网站地址

在此浏览器中保存我的显示名称、邮箱地址和网站地址，以便下次评论时使用。

发表评论