Ffmpeg支持GPU编码

Ffmpeg 通常是使用CPU进行转码，但CPU的效率较慢，随着GPU的普及，以及对于效率的要求，需要引入GPU进行转码；ffmpeg的新版本已经可以与GPU结合进行转码；本文介绍ffmpeg利用GPU进行转码的配置操作；

1. GPU环境的安装，目前的GPU基本都使用NVIDIA的产品，所以需要到NIVDIA网站中找到相应的驱动，并安装到linux系统中，还需要安装CUDA的kit包到系统中；具体的安装比较麻烦，可以参考网上的教程；

GPU驱动安装完毕后，通过下面的命令进行查看：

# nvidia-smi

图形用户界面

描述已自动生成

如果看到上面的内容，说明你就安装成功了；

# nvcc -V

文本

描述已自动生成

出现上面的内容，说明你的CUDA也安装成功了；

1. 重新编译ffmpeg

要让ffmpeg能够使用CUDA提供的GPU编解码器，必须重新编译ffmpeg，让其能够通过动态链接调用CUDA的能力

首先要编译安装nv-codec-headers库

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

# make PREFIX="$HOME/ffmpeg\_build" BINDDIR="$HOME/bin"

# make install PREFIX="$HOME/ffmpeg\_build" BINDDIR="$HOME/bin"

进入~/ffmepg\_sources/ffmpeg-3.3.8/目录重新执行ffmpeg的编译和安装  
注意configure命令参数和之前configure命令参数的区别

PATH="$HOME/bin:$PATH" PKG\_CONFIG\_PATH="$HOME/ffmpeg\_build/lib/pkgconfig"

./configure \

--prefix="$HOME/ffmpeg\_build" \

--pkg-config-flags="--static" \

--extra-cflags="-I$HOME/ffmpeg\_build/include -I/usr/local/cuda/include" \

--extra-ldflags="-L$HOME/ffmpeg\_build/lib -L/usr/local/cuda/lib64" \

--extra-libs=-lpthread \

--extra-libs=-lm \

--bindir="$HOME/bin" \

--enable-gpl \

--enable-libfdk\_aac \

--enable-libfreetype \

--enable-libmp3lame \

--enable-libopus \

--enable-libvorbis \

--enable-libvpx \

--enable-libx264 \

--enable-libx265 \

--enable-nonfree \

--enable-cuda \

--enable-cuvid \

--enable-nvenc \

--enable-libnpp

make

make install

1. 验证安装

重新安装完ffmpeg，使用ffmpeg -hwaccels命令查看支持的硬件加速选项

图片包含 文本

描述已自动生成

可以看到多出来一种叫做cuvid的硬件加速选项，这就是CUDA提供的GPU视频编解码加速选项

然后查看cuvid提供的GPU编解码器ffmpeg -codecs | grep cuvid

图形用户界面, 文本, 网站

中度可信度描述已自动生成

所有带有"cuvid"或"nvenc"的，都是CUDA提供的GPU编解码器  
可以看到，我们现在可以进行h264/hevc/mjpeg/mpeg1/mpeg2/mpeg4/vc1/vp8/vp9格式的GPU解码，以及h264/hevc格式的GPU编码

运行命令：

# ffmpeg -hwaccel cuvid -c:v h264\_cuvid -i 1428072328817lbjn.mp4 -c:v nvenc\_hevc -c:a copy -y 2.mp4