术语

容器(Container)

容器就是一种文件格式，比如flv，mkv等。包含下面5种流以及文件头信息。

流(Stream)

是一种视频数据信息的传输方式，5种流：音频，视频，字幕，附件，数据。

帧(Frame)

帧代表一幅静止的图像，分为I帧，P帧，B帧。

编解码器(Codec)

是对视频进行压缩或者解压缩，CODEC =COde （编码） +DECode（解码）

复用/解复用(mux/demux)

把不同的流按照某种容器的规则放入容器，这种行为叫做复用（mux）

把不同的流从某种容器中解析出来，这种行为叫做解复用(demux)

组成

1、libavformat：用于各种音视频封装格式的生成和解析，包括获取解码所需信息以生成解码上下文结构和读取音视频帧等功能，包含demuxers和muxer库；

2、libavcodec：用于各种类型声音/图像编解码；

3、libavutil：包含一些公共的工具函数；

4、libswscale：用于视频场景比例缩放、色彩映射转换；

5、libpostproc：用于后期效果处理；

6、ffmpeg：是一个命令行工具，用来对视频文件转换格式，也支持对电视卡实时编码；

7、ffsever：是一个HTTP多媒体实时广播流服务器，支持时光平移；

8、ffplay：是一个简单的播放器，使用ffmpeg 库解析和解码，通过SDL显示；

 我们常说的mpg，rmvb，ts，mkv，mp4等其实就是容器格式，用于封装已经编码过的音频数据和视频数据。一个视频文件的播放第一步需要知道它的容器格式，并把音频数据和视频数据从容器中分离出来，我们称这过程是解复用。ffmpeg中在avformat.h文件中定义了很多用于解复用的数据结构，其中常用的有AVInputFormat(用于解复用)，AVOutputFormat(用于复用) 同一时间AVFormatContext内只能存

在其中一个

，AVStream(用于描述解复用后的数据流)，AVFormatContext(复用/解复用的上下文)。

FFmpeg处理流程

过滤器(Filter)

在多媒体处理中，filter的意思是被编码到输出文件之前用来修改输入文件内容的一个软件工具。如：视频翻转，旋转，缩放等。

码率、帧率和文件大小

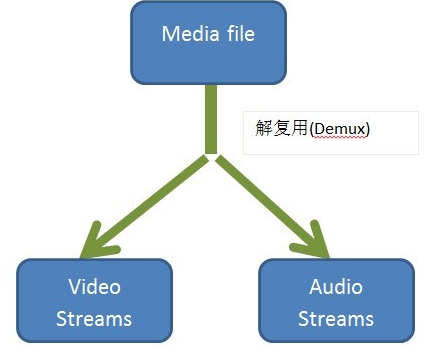
概述

        码率和帧率是视频文件的最重要的基本特征，对于他们的特有设置会决定视频质量。如果我们知道码率和时长那么可以很容易计算出输出文件的大小。

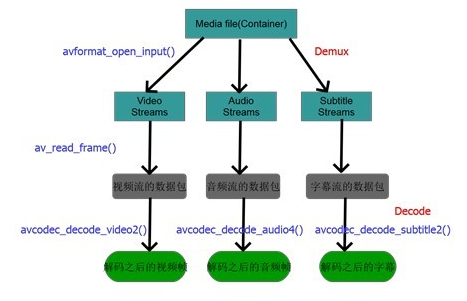
帧率：帧率也叫帧频率，帧率是视频文件中每一秒的帧数，肉眼想看到连续移动图像至少需要15帧。

码率：比特率(也叫码率，数据率)是一个确定整体视频/音频质量的参数，秒为单位处理的字节数，码率和视频质量成正比，在视频文件中中比特率用bps来表达。





**FFmpeg中解码流程对应的API函数**



FFmpeg中Demux这一步是通过avformat\_open\_input（）这个 api来做的，这个api读出文件的头部信息，并做demux，在此之后我们就可以读取媒体文件中的音频和视频流，然后通过 av\_read\_frame()从音频和视频流中读取出基本数据流packet，然后将packet送到avcodec\_decode\_video2() 和相对应的api进行解码。

<http://blog.chinaunix.net/uid-11344913-id-4282729.html>

**1.码率计算公式：**  
  
　　码率=采样率 x 位深度 x 声道  
  
　　所以，上面文件的码率= 44.1Khz x 16位 x 2声道 = 1411.2 Kbps  
  
　　2.文件大小 = 码率 x 时长 = 1411.2 Kbps x (3 x 60 + 47 )s = 1411.2Kbps x 227s=38102.4 Kb  
  
　　38102.4 Kb / 1024 Kb/M = 37.2M  
  
　　近似等于mediainfo工具显示的文件大小38.3M。  
  
　　注：此计算公式对未压缩的wav格式文件有效，不适用于mp3等被压缩的文件。  
  
  
**帧率：**  
  
　　**帧率：**  
  
　　影响画面流畅度，与画面流畅度成正比：帧率越大，画面越流畅；帧率越小，画面越有跳动感。如果码率为变量，则帧率也会影响体积，帧率越高，每秒钟经过的画面越多，需要的码率也越高，体积也越大。

音频：

这里了解一下各个参数的含义以及一些基本概念。

样本长度(sample)：样本是记录音频数据最基本的单位，常见的有8位和16位。

通道数(channel)：该参数为1表示单声道，2则是立体声。

桢(frame)：桢记录了一个声音单元，其长度为样本长度与通道数的乘积。

采样率(rate)：每秒钟采样次数，该次数是针对桢而言。

周期(period)：音频设备一次处理所需要的桢数，对于音频设备的数据访问以及音频数据的存储，都是以此为单位。

交错模式(interleaved)：是一种音频数据的记录方式，在交错模式下，数据以连续桢的形式存放，即首先记录完桢1的左声道样本和右声道样本（假设为立体声格式），再开始桢2的记录。而在非交错模式下，首先记录的是一个周期内所有桢的左声道样本，再记录右声道样本，数据是以连续通道的方式存储。不过多数情况下，我们只需要使用交错模式就可以了。

   表示YUV格式，一般用Y，U，V三者的比率来表示不同格式，比如YUV444 表示三者是比值此是 4:4:4

   即一个点数据点，Y,U,V的空间都是一样大小。目前主要有如下比例，注意所有格式中Y比值都是4,占一个字节,表示没有减少采样。不同格式中,减小只是UV的采样值.

* 4:4:4 表示色度值(UV)没有减少采样。即Y,U,V各占一个字节，加上Alpha通道一个字节，总共占4字节.这个格式其实就是24bpp的RGB格式了。
* 4:2:2 表示UV分量采样减半,比如第一个像素采样Y,U,第二个像素采样Y,V,依次类推,这样每个点占用2个字节.二个像素组成一个宏像素.
* 4:2:0  这种采样并不意味着只有Y，Cb而没有Cr分量，这里的0说的U，V分量隔行才采样一次。比如第一行采样 4:2:0 ,第二行采样 4:0:2 ,依次类推...在这种采样方式下，每一个像素占用16bits或10bits空间.
* 4:1:1 可以参考4:2:2分量，是进一步压缩，每隔四个点才采一次U和V分量。一般是第0点采Y,U,第1点采Y,第3点采YV,第四点采Y,依次类推。

   除了4:4:4采样，其余采样后信号重新还原显示后,会丢失部分UV数据，只能用相临的数据补齐，但人眼对UV不敏感，因此总体感觉损失不大。

**4:2:2示例**

   如果原始数据三个像素是 Y0 U0 V0 ,Y1 U1 V1,Y2 U2 V2,Y3 U3 V3

   经过4：2：2采样后，数据变成了 Y0 U0 ,Y1 V1 ,Y2 U2,Y3 V3

   如果还原后，因为某一些数据丢失就补成 Y0 U0 **V1**,Y1 **U0** V1,Y2 U2 **V3** ,Y3 U3 **Y2**

**4:1:1示例**

原来四个像素为: [Y0 U0 V0] [Y1 U1 V1] [Y2 U2 V2] [Y3 U3 V3]

   存放的码流为:    Y0 U0     ,Y1 , Y2  V2,   Y3

   还原出像素点为：[Y0 U0 V2] [Y1 U0 V2] [Y2 U0 V2] [Y3 U0 V2]

**4:2:0示例**

下面八个像素为：[Y0 U0 V0] [Y1 U1 V1] [Y2 U2 V2] [Y3 U3 V3] [Y5 U5 V5] [Y6 U6 V6] [Y7U7 V7] [Y8 U8 V8]

存放的码流为：  Y0 U0 ,Y1, Y2 U2, Y3 ,Y5 V5, Y6, Y7 V7, Y8

映射出的像素点为：[Y0 U0 V5] [Y1 U0 V5] [Y2 U2 V7] [Y3 U2 V7] [Y5 U0 V5] [Y6 U0 V5] [Y7U2 V7] [Y8 U2 V7]

# 1.什么是YUV格式

YUV，是一种颜色编码方法。Y表示明亮度(Luminance、Luma)，也就是灰度值。U和V则是色度、浓度(Chrominance、Chroma)，作用是描述影像色彩及饱和度，用于指定像素的颜色。与我们熟知的RGB类似，YUV也是一种颜色编码方法，主要用于电视系统以及模拟视频领域，它将亮度信息（Y）与色彩信息（UV）分离，没有UV信息一样可以显示完整的图像，只不过是黑白的，这样的设计很好地解决了彩色电视机与黑白电视的兼容问题。并且，YUV不像RGB那样要求三个独立的视频信号同时传输，所以用YUV方式传送占用极少的频宽。

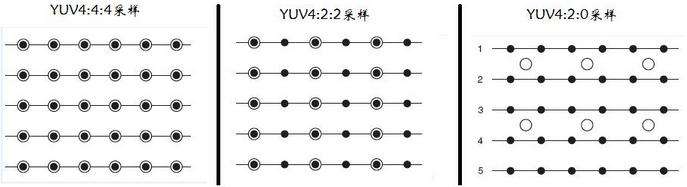
与RGB格式相比，RGB格式用红绿蓝三个颜色分量来表示一个像素，YUV则是用一个亮度分量（Y）加两个色度分量来表示一个像素。YUV也经常被称为YCbCr，Cb为蓝色色度分量，对应U；Cr为红色色度分量，对应V。

# 2.YUV格式采样方式

YUV采样方式主要描述像素Y、U、V分量采样比例，即表达每个像素时，Y、U、V分量的数目，通常有三种方式：YUV4:4:4，YUV4:2:2，YUV4:2:0。

用三个图直观地表示采集的方式，以黑点表示采样该像点的Y分量，以空心圆圈表示采用该像素点的UV分量。

**YUV4:4:4采样，每一个Y对应一组UV分量；YUV4:2:2采样，每两个Y共用一组UV分量；YUV4:2:0采样，每四个Y共用一组UV分量。**

****

4：4：4表示没有色度通道的压缩采样。

4：2：2表示2：1水平压缩采样，无垂直压缩采样。

4：2：0表示2：1水平压缩采样，2：1垂直压缩采样。

# 3.YUV格式的存储方式

YUV存储格式，主要描述像素的Y、U、V分量排列方式，分为两种格式：紧缩格式和平面格式。

**紧缩格式(packed formats)：**将Y、U、V值储存成Macro Pixels阵列，和RGB的存放方式类似。每个像素点的Y,U,V是连续交差存储的。

这种YUV经常用在网络摄像机里，且大多是4:2:2的也就是只有在水平方向下采样。例如

UYVYUYVYUYVYUYVYUYVYUYVYUYVYUYVY

**平面格式(planar formats)：**将Y、U、V的三个分量分别存放在不同的矩阵中。先连续存储所有像素点的Y，紧接着存储所有像素点的U，随后是所有像素点的V。

planar的YUV是比较合适视频编码的一种格式。像下面这样的4x4的图像

YYYYYYYYYYYYYYYY UUUU VVVV

这个例子所示的YUV常被称为YUV420P，其水平和竖直方向上均为每两个亮度分量才有一个色度分量，也就是每四个Y才需要一个U和V。除这种水平和竖直方向的色度分量都下采样的4:2:0，还有只有水平方向下采样的4:2:2格式，以及没有下采样的4:4:4格式。

上面的例子是按Y、U、V（YCbCr）的顺序存储的，如果改为Y、V、U（YCrCb），则称为YV12格式。

**Semi-planar格式**

这是Planar和Packed两种格式的混合体：Y分量单独一个plane，U和V则交叉存储。这种YUV一般都是4:2:0的。例如：

YYYYYYYYYYYYYYYY UVUVUVUV

这个示例称为NV12格式。如果交换UV的顺序，就成了NV21格式。NV21是Android系统中摄像头预览图片的默认格式。

fstream头文件定义了用于文件输入的类ifstream和文件输出的类ofstream

ofstream 对应程序来说是输出，对应文件来说是输入也就是写文件。

LED:用于发光，调节亮度。

LCD:显示图像用的。