H.264码流第一个 NALU 是 SPS（序列参数集Sequence Parameter Set）

H.264码流第二个 NALU 是 PPS（图像参数集Picture Parameter Set）

H.264码流第三个 NALU 是 IDR（即时解码器刷新）

NAL网络提取层，里面放一些与网络相关的信息

VCL 视频编码层

Slice是片的意思，H264中把图像分成一帧（frame）或两场（field），而帧又可以分成一个或几个片（Slilce）；片由宏块（MB）组成。宏块是编码处理的基本单元

<http://blog.sina.com.cn/s/blog_8fb8cd4801018yyo.html>

1. 参照一段时间内图像的统计结果表明，在相邻几幅图像画面中，一般有差别的像素只有10%以内的点,亮度差值变化不超过2%，而色度差值的变化只有1%以内

2. 当某个图像与之前的图像变化很大，无法参考前面的帧来生成，那我们就结束上一个序列，开始下一段序列，也就是对这个图像生成一个完整帧

H264协议 三种帧：

a. 完整编码的帧叫I帧

b. 参考之前的I帧生成的只包含差异部分编码的帧叫P帧

c. 参考前后的帧编码的帧叫B帧

 H264采用的核心算法是帧内压缩和帧间压缩，帧内压缩是生成I帧的算法，帧间压缩是生成B帧和P帧的算法

a. I、B、P各帧是根据压缩算法的需要,是人为定义的,它们都是实实在在的物理帧。

b. 一般来说，I帧的压缩率是7（跟JPG差不多），P帧是20，B帧可以达到50。可见使用B帧能节省大量空间，节省出来的空间可以用来保存多一些I帧，这样在相同码率下，可以提供更好的画质。

三种帧说明：

I帧:帧内编码帧 ，I帧表示关键帧 这一帧画面的完整保留 解码时只需要本帧

I帧特点:

1.它是一个全帧压缩编码帧。它将全帧图像信息进行JPEG压缩编码及传输;

2.解码时仅用I帧的数据就可重构完整图像;

3.I帧描述了图像背景和运动主体的详情;

4.I帧不需要参考其他画面而生成;

5.I帧是P帧和B帧的参考帧(其质量直接影响到同组中以后各帧的质量);

6.I帧是帧组GOP的基础帧(第一帧),在一组中只有一个I帧;

7.I帧不需要考虑运动矢量;

8.I帧所占数据的信息量比较大。

P帧:前向预测编码帧。P帧表示的是这一帧跟之前的一个关键帧（或P帧）的差别，解码时需要用之前缓存的画面叠加上本帧定义的差别，生成最终画面。（也就是差别帧，P帧没有完整画面数据，只有与前一帧的画面差别的数据）

P帧的预测与重构:P帧是以I帧为参考帧,在I帧中找出P帧“某点”的预测值和运动矢量,取预测差值和运动矢量一起传送。在接收端根据运动矢量从I帧中找出P帧“某点”的预测值并与差值相加以得到P帧“某点”样值,从而可得到完整的P帧。

P帧特点:

1.P帧是I帧后面相隔1~2帧的编码帧;

2.P帧采用运动补偿的方法传送它与前面的I或P帧的差值及运动矢量(预测误差);

3.解码时必须将I帧中的预测值与预测误差求和后才能重构完整的P帧图像;

4.P帧属于前向预测的帧间编码。它只参考前面最靠近它的I帧或P帧;

5.P帧可以是其后面P帧的参考帧,也可以是其前后的B帧的参考帧;

6.由于P帧是参考帧,它可能造成解码错误的扩散;

7.由于是差值传送,P帧的压缩比较高。

B帧:双向预测内插编码帧。B帧是双向差别帧，也就是B帧记录的是本帧与前后帧的差别（具体比较复杂，有4种情况，但我这样说简单些），换言之，要解码B帧，不仅要取得之前的缓存画面，还要解码之后的画面，通过前后画面的与本帧数据的叠加取得最终的画面。B帧压缩率高，但是解码时CPU会比较累。

B帧的预测与重构：B帧以前面的I或P帧和后面的P帧为参考帧,“找出”B帧“某点”的预测值和两个运动矢量,并取预测差值和运动矢量传送。接收端根据运动矢量在两个参考帧中“找出(算出)”预测值并与差值求和,得到B帧“某点”样值,从而可得到完整的B帧。

B帧特点

1.B帧是由前面的I或P帧和后面的P帧来进行预测的;

2.B帧传送的是它与前面的I或P帧和后面的P帧之间的预测误差及运动矢量;

3.B帧是双向预测编码帧;

4.B帧压缩比最高,因为它只反映丙参考帧间运动主体的变化情况,预测比较准确;

5.B帧不是参考帧,不会造成解码错误的扩散。

--------------------------------

压缩算法的说明

--------------------------------

h264的压缩方法:

1.分组:把几帧图像分为一组(GOP，也就是一个序列，帧组),为防止运动变化,帧数不宜取多。

2.定义帧:将每组内各帧图像定义为三种类型,即I帧、B帧和P帧;

3.预测帧:以I帧做为基础帧,以I帧预测P帧,再由I帧和P帧预测B帧;

4.数据传输:最后将I帧数据与预测的差值信息进行存储和传输。

帧内（Intraframe）压缩也称为空间压缩（Spatial compression）。当压缩一帧图像时，仅考虑本帧的数据而不考虑相邻帧之间的冗余信息，这实际上与静态图像压缩类似。帧内一般采用有损压缩算法，由于帧内压缩是编码一个完整的图像，所以可以独立的解码、显示。帧内压缩一般达不到很高的压缩，跟编码jpeg差不多。

帧间（Interframe）压缩的原理是：相邻几帧的数据有很大的相关性，或者说前后两帧信息变化很小的特点。也即连续的视频其相邻帧之间具有冗余信息,根据这一特性，压缩相邻帧之间的冗余量就可以进一步提高压缩量，减小压缩比。帧间压缩也称为时间压缩（Temporal compression），它通过比较时间轴上不同帧之间的数据进行压缩。帧间压缩一般是无损的。帧差值（Frame differencing）算法是一种典型的时间压缩法，它通过比较本帧与相邻帧之间的差异，仅记录本帧与其相邻帧的差值，这样可以大大减少数据量。

顺便说下有损（Lossy ）压缩和无损（Lossy less）压缩。无损压缩也即压缩前和解压缩后的数据完全一致。多数的无损压缩都采用RLE行程编码算法。有损压缩意味着解压缩后的数据与压缩前的数据不一致。在压缩的过程中要丢失一些人眼和人耳所不敏感的图像或音频信息,而且丢失的信息不可恢复。几乎所有高压缩的算法都采用有损压缩,这样才能达到低数据率的目标。丢失的数据率与压缩比有关,压缩比越小，丢失的数据越多,解压缩后的效果一般越差。此外,某些有损压缩算法采用多次重复压缩的方式,这样还会引起额外的数据丢失。

序列/IDR/GOP：

1. 在H264中图像以序列为单位进行组织，一个序列是一段图像编码后的数据流，以I帧开始，到下一个I帧结束。

2. (??GOP 以IDR帧开始 中间可能有I帧，但是只有IDR帧才会丢掉之前的参考帧队列，IDR帧后面的P/B帧不会参考IDR前面的帧来解码，两个IDR帧之间的P/B帧会参照之前的N个I/P/B帧来解码??)

3. 一个序列就是一段内容差异不太大的图像编码后生成的一串数据流。当运动变化比较少时，一个序列可以很长，因为运动变化少就代表图像画面的内容变动很小，所以就可以编一个I帧，然后一直P帧、B帧了。当运动变化多时，可能一个序列就比较短了，比如就包含一个I帧和3、4个P帧。

4. 一个序列的第一个图像叫做 IDR 图像（立即刷新图像），IDR 图像都是 I 帧图像。

5. H.264 引入 IDR 图像是为了解码的重同步，当解码器解码到 IDR 图像时，立即将参考帧队列清空，将已解码的数据全部输出或抛弃，重新查找参数集，开始一个新的序列。这样，如果前一个序列出现重大错误，在这里可以获得重新同步的机会。IDR图像之后的图像永远不会使用IDR之前的图像的数据来解码。

这个问题要说清楚还是有点复杂：

NALU 类型= 5 IDR 帧（一种特殊的 I 帧）；

NALU类型!=5， slice\_type = 7 的 slice 就属于 I 帧；

如果 slice\_type = 2，那么就要判断与当前 slice 同属一帧的 slice 是否都是 I slice，如果都是，那么这些 slice 就属于一个 I 帧。

（1）解析函数（获取信息）：

ff\_h264\_decode\_nal()：解析NALU Header。  
ff\_h264\_decode\_seq\_parameter\_set()：解析SPS。  
ff\_h264\_decode\_picture\_parameter\_set()：解析PPS。  
ff\_h264\_decode\_sei()：解析SEI。  
ff\_h264\_decode\_slice\_header()：解析Slice Header。

[I 帧和 IDR 帧的区别](http://blog.csdn.net/chenchong_219/article/details/44870903)

<http://blog.csdn.net/chenchong_219/article/details/44870903>

有如下帧序列： IPPPP **I P**PPP ……（我们程序没有 B 帧，所以帧序列简单些，但道理是一样的）。按照 3 个参考帧编码。

因为“按照 3 个参考帧编码”，所以参考帧队列长度为 3 。

遇到绿色的 I 时，并不清空参考帧队列，把这个 I 帧加入参考帧队列（当然 I 编码时不用参考帧。）。再检测到红色的 P 帧时，用到的就是 PPI 三帧做参考了。

不怕自己罗嗦（好记性不如烂笔头），再强调一个： 一个参考帧，就是参考当前帧的前面的那帧（因为没涉及到 B 帧，所以“前面的那帧”既是播放顺序的，也是编码顺序的）。多个参考帧是一个道理 。 （ 我以前一直误解为从前面的几帧中找到最合适的一个参考帧）

最后，“ 但是收到 IDR 帧时，解码器另外需要做的工作就是：把所有的 PPS 和 SPS 参数进行更新。由此可见，在编码器端，每发一个 IDR ，就相应地发一个 PPS&SPS\_nal\_unit ”应该是对的吧。先这样认为：

IDR frame：I和IDR帧都使用帧内预测，在编码解码中为了方便，首个I帧要和其他I帧区别开，把第一个I帧叫IDR，这样方便控制编码和解码流程，所以IDR帧一定是I帧，但I帧不一定是IDR帧；IDR帧的作用是立刻刷新,使错误不致传播,从IDR帧开始算新的序列开始编码。I帧有被跨帧参考的可能,IDR不会。

RTP协议的H.264视频传输

NALU H.264 的基本流（elementary stream,ES）的结构分为两层，包括视频编码层（VCL）和网络适配层（NAL）。

视频编码层负责高效的视频内容表示，而网络适配层负责以网络所要 求的恰当的方式对数据进行打包和传送。

H.264 的基本流由一系列NALU （Network Abstraction Layer Unit ）组成，不同的NALU数据量各不相同。

在每个NALU 前添加起始码：0x000001(起始码不属于NALU，但在NALU header之前，用来隔开每个NALU)，用来指示一个 NALU的起始和终止位置。

每个 NALU单元由一个字节的 NALU头（NALU Header）和若干个字节的载荷数据（RBSP）组成

(上面是一个字节 0 对应的是byteBuffer的最高位 header[4]&0x80 )

( spyDroid中，会根据 MediaCodec::dequeueOutputBuffer buffer[0,1,2,3] 4个字节 确认是否一个NAL开始 00 00 00 01

然后根据 第5个字节 buffer[4] && 0x1F 取出 NALU的类型 ，如果是7/8 代表负载数据是 sps pps 等 )

F：forbidden\_zero\_bit.1 位，如果有语法冲突，则为 1。当网络识别此单元存在比特错误时，可将其设为 1，以便接收方丢掉该单元。

NRI：nal\_ref\_idc.2 位，用来指示该NALU 的重要性等级。值越大，表示当前NALU越重要。具体大于0 时取何值，没有具体规定。

Type：5 位，指出NALU 的类型。具体如表1 所示：



NRI 值为 7 和 8 的NALU 分别为序列参数集（sps）和图像参数集（pps）。参数集是一组很少改变的，为大量VCL NALU 提供解码信息的数据。

其中序列参数集作用于一系列连续的编码图像，

而图像参数集作用于编码视频序列中一个或多个独立的图像。

如果\*没能正确接收到这两个参 数集，那么其他NALU 也是无法解码的。因此它们一般在发送其它 NALU 之前发送(SpyDroid中使用RTSP先发送)，并且使用不同的信道或者更加可靠的传输协议（如TCP）进行传输，也可以重复传输。

H.264/AVC视频编码标准中，整个系统框架被分为了两个层面：视频编码层面（VCL）和网络抽象层面（NAL）

其中，前者负责有效表示视频数据的内容，而后者则负责格式化数据并提供头信息，以保证数据适合各种信道和存储介质上的传输。

平时的每帧数据就是一个NAL单元（SPS与PPS除外）。

在实际的H264数据帧中，往往帧前面带有00 00 00 01 或 00 00 01分隔符，一般来说编码器编出的首帧数据为PPS与SPS，接着为I帧…



  NALU类型是我们判断帧类型的利器

以00 00 00 01分割之后的下一个字节就是NALU类型，将其转为二进制数据后，解读顺序为从左往右算，如下:

（1）第1位禁止位，值为1表示语法出错

（2）第2~3位为参考级别

（3）第4~8为是nal单元类型

pts/dts怎样实现音视频同步